dataBASE DNA Data Storage

Mason Matich

1. Personal Section

The origins of my research project start in my 10th grade AP Biology class. My teacher was lecturing on DNA and its biological function for storing cellular information, which she described as analogous to computer storage. It was an interesting analogy that I expanded on with genes being “files” stored in the genome “hard drive.” I eventually began to wonder if it was feasible to use DNA directly for computer storage, as after all it must possess incredibly high storage density to be viable for its biological purpose. Around this time, I was taking a summer program about satellite design where I learned about the challenges of data storage in space due to its high levels of radiation. With these two insights in mind and wanting to start a new research project, I decided to develop a storage system for deep space and enterprise applications based on DNA.

It’s helpful to think of my research in two parts, as a computer and as a wet lab project. The computer side consisted of architecting and implementing custom algorithms to convert back and forth between binary and DNA with margin for error. This is no small task, and it required me to learn about existing computer storage systems and the underlying math that made them work. It could be tedious at times, but this research helped immensely when it came time to code my software. On the wet lab side, which was completed at the Pennsylvania Biotechnology Center, it was a constant struggle to create the workflow for duplicating my data in DNA form. To save cost, I essentially hacked files into bacterial cells and used a standard process from there. And while this method was slow, working from such a low level helped me gain a deeper understanding of the biology behind my storage system, which in turn helped me optimize it further. This conversation between both parts really let my project come alive because I could see how changing small variables on either side affected the overall system.

The best advice I can give to other high schoolers looking to start a research project is to come up with an idea you’re passionate about, and then to stick with it. Take my two-year project as an example. Because I started when I was a relatively new coder, the initial version of my codec had strange logic flow choices and poor syntax. And since I was a beginner, I didn’t know I should document my code or rewrite functions that were slow or buggy. At the end of year one, about half my lines of code were error handling or conversions from older lines of code, and in essence, my software was held together with virtual duct tape. It became near impossible to add new features, and I realized I couldn’t use any of it anymore. So, after a year, I essentially had to start from zero all over again. But what I did have was an idea I was passionate about and a year of experience working with DNA and coding. And using that as a base, I was able to leap forward with my project and in three months far exceed what I accomplished in a year.

Most of the paths you will go down in your research are going to be dead ends, but the act of walking down them is how you get experience and grow as a researcher. I think the best thing you can do is to focus on what you’ve learned during the journey and remember to stick your head up every once and a while to make sure you’re actually progressing toward your project’s end goal. And above all, make sure you still love your idea, because that passion is how you will find your breakthrough.

.

2. Methods

2.1. Introduction

DNA is several magnitudes more storage dense than current storage mediums, including flash and optical[1]. It is also far more reliable, being stable for hundreds or even thousands of years in a dried form[1]. This combination of density and longevity makes it attractive for both archival and space applications where data storage needs to be reliable and small.

Specifically, in space applications, DNA's relative resiliency to radiation could potentially make it a superior alternative to current methods of storage. Radiation in space is inherently dangerous to computer systems. Digital logic relies on low and high voltages, and high-energy radiation can cause these logic bits to flip, generating system instability. This is called a Single Event Upset, or an SEU. This isn't a problem on Earth since its magnetic field deflects most radiation, but in space, no such shield exists. This leaves computers exposed to cosmic rays, highly energized particles common throughout space, increasing the frequency and severity of SEUs. This problem is compounded in flash storage which uses logic gates to store information. These gates can get flipped (bit-flip) permanently by radiation, over time causing data corruption and SEUs.[2] Current methods of error correction can combat this, as well as periodic reprogramming of storage, but these do not solve the underlying issue.[3] DNA is susceptible to radiation damage like current storage mediums, but this issue can be mitigated by simply synthesizing more DNA once a strand becomes damaged, almost like having a silicon memory fab that can fit into a spacecraft. This could improve mission times for satellites and other spacecraft significantly.

2.2. Purpose

The goal of this project is to design a method of encoding digital information into a DNA strand and subsequently decoding the information with an error correction method usable up to 15% error.

2.3. dataBASE Codec

The encoding/decoding process is illustrated in Figure 1. The input file, represented as a hexadecimal, is encoded into a DNA sequence and synthesized into a DNA strand. To retrieve the data, the DNA strand is sequenced. In this process, some errors are incurred. These are fixed during error correction. Finally, the data is extracted and decoded to its original format (.jpg, .txt, etc.)

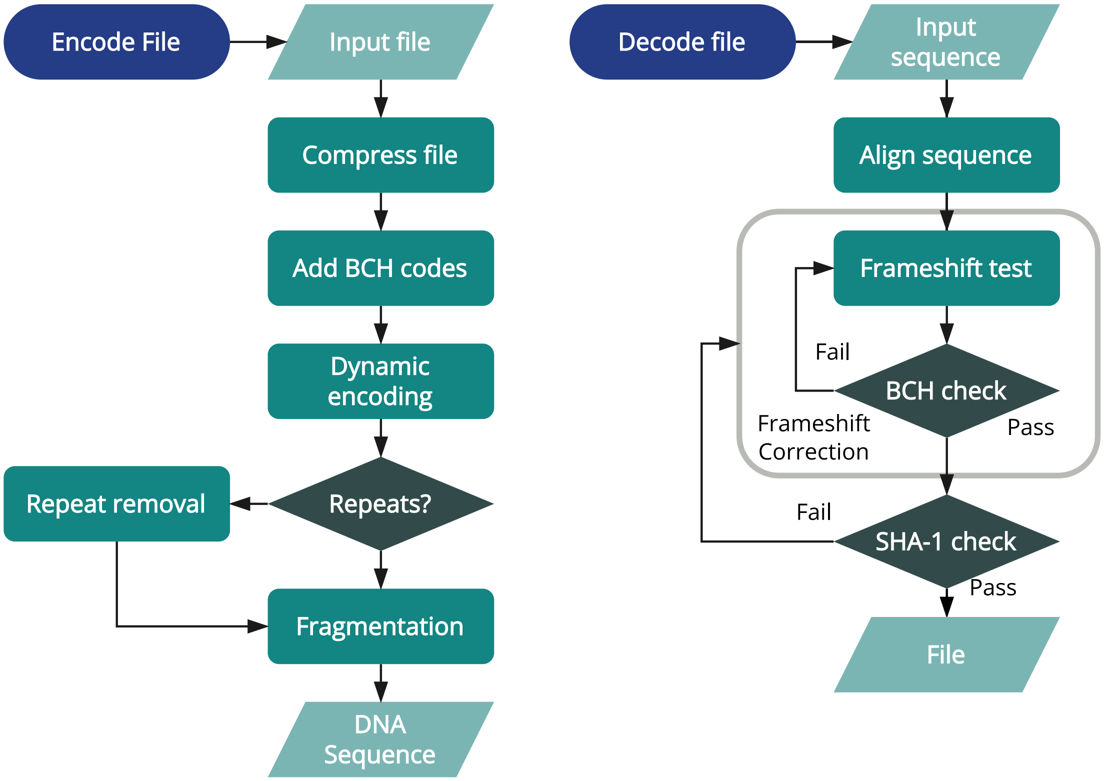
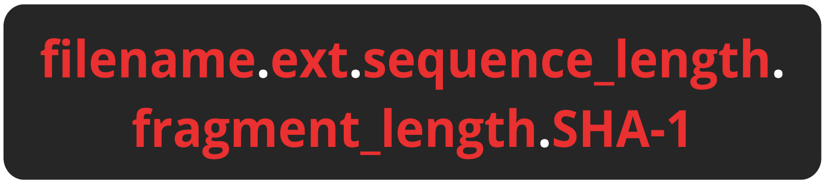


Figure 1: Flowchart of storage codec

2.3.1. Encoding Method

Repeated DNA sequences over 20 bp can cause severe errors. In synthesis, repeated bases increase the risk of bases not attaching properly to the end of a sequence. In assembly, repeated sequences can cause fragments to assemble in the wrong order, scrambling data. In sequencing, repeat sections cannot be used as primers to ensure the correct portions of the sequence are being read. This issue is partially mitigated by compressing the file, in this case with level 22 Zstandard compression, before generating a DNA sequence to remove most repeats in the source data. The file was further processed with 2-byte BCH codes appended to 40-byte blocks of data. An SHA-1 hash is then generated for the entire file.

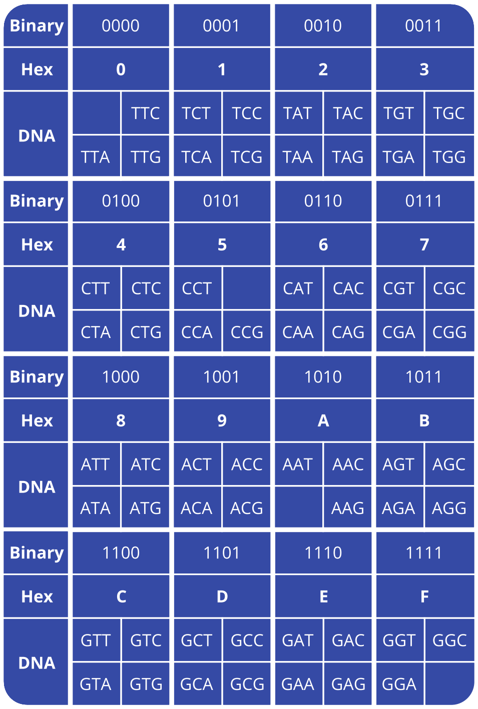
Metadata as shown in Figure 2 consists of a file name, sequence length, fragment length, and SHA-1 hash separated by "." and encoded in utf-8. The sequence length algorithm is calculated to include itself. The metadata also has a 2-byte BCH code appended. The metadata is added to the front of the processed file and is converted to hexadecimal.



filename.ext.sequence_length.
fragment_length.SHA-1

Figure 2: Metadata format

After this binary preprocessing, DNA encoding can begin. A direct binary conversion was initially tested with 2 and 4 bits per base, but this led to frequent repeat sequences. Instead, a dynamic system stores 4 bits per one 3-base word. Using 3-base words instead of 2-base words increases the possible mapping from 16 (42) to 64 (43), allowing for four possible mappings per hexadecimal as shown in Figure 3. The mappings 'AAA', 'GGG', 'CCC', and 'TTT' have been removed to prevent repetitions.



Binary	0000		0001		0010		0011	
Hex	**0**		**1**		**2**		**3**	
DNA		TTC	TCT	TCC	TAT	TAC	TGT	TGC
	TTA	TTG	TCA	TCG	TAA	TAG	TGA	TGG
Binary	0100		0101		0110		0111	
Hex	**4**		**5**		**6**		**7**	
DNA	CTT	CTC	CCT		CAT	CAC	CGT	CGC
	CTA	CTG	CCA	CCG	CAA	CAG	CGA	CGG
Binary	1000		1001		1010		1011	
Hex	**8**		**9**		**A**		**B**	
DNA	ATT	ATC	ACT	ACC	AAT	AAC	AGT	AGC
	ATA	ATG	ACA	ACG		AAG	AGA	AGG
Binary	1100		1101		1110		1111	
Hex	**C**		**D**		**E**		**F**	
DNA	GTT	GTC	GCT	GCC	GAT	GAC	GGT	GGC
	GTA	GTG	GCA	GCG	GAA	GAG	GGA	

Figure 3: Hex to DNA word conversion table

To start, the file is encoded with a random word chosen. A secondary algorithm then checks the initial sequence for repeat sequences over 20 bp, reassigning words as needed. This variability combined with data compression effectively prevents repeated sequences in DNA fragments up to 1,800 bp (longer fragments are possible but were not tested due to DNA sequencing restrictions).

For assembly, 30 bp flanking the restriction site on both sides of the plasmid are added to the beginning and end of the completed DNA sequence. This finalized DNA sequence is saved as a text file in the forward and reverse directions. For assembly, it is also split into user-defined fragment lengths with 25 bp overlaps and saved in a CSV file. This encoding system has a calculated storage density of ~87.72 exabytes per gram as shown in Equation 1. The ideal encoding data rate is ~70 kb/sec but varies depending on file size.

$$\frac{288\ bytes}{1977257.18\ Daltons} = \frac{2.88e^{-16}\ exabytes}{3.283e^{-18}\ grams} = 87.72\ \frac{exabytes}{gram}$$

Equation 1: dataBASE codec storage density equation

2.3.2. Decoding Method

The decoder software takes the raw sequence and identifies/reads the metadata, correcting it as needed. The metadata is then removed from the raw sequence. Finally, the sequence is decoded from words back into hexadecimal and error corrected as needed. The different possible words will appear as only one hexadecimal in the decoder as shown in Figure 4. The resultant file's SHA-1 hash is compared to the SHA-1 hash in the metadata to check data integrity. The ideal decoding data rate is ~72 kb/sec.

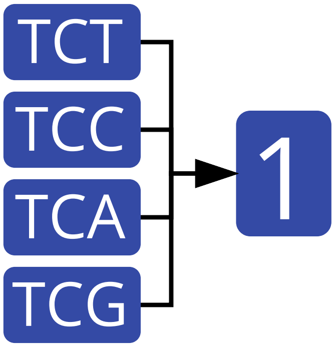


Figure 4: Example of conversion from word to hexadecimal

2.3.3. Error Correction

A certain amount of error is expected during DNA synthesis and sequencing. For next-generation sequencing, a newer type of sequencing that is much faster but more error-prone, this rate is about 10%. In a space environment, ionizing radiation can cause double-strand breaks, single-strand breaks, and abasic sites. This project's error correction is designed to correct these types of errors.

2.3.3.1. Alignment Implementation

This project uses a global alignment system (Needleman-Wunsch Algorithm), where both sides of the DNA strand are compared to each other to identify errors. Global alignment algorithms are more accurate but only work reliably on closely related DNA strands. Local alignment algorithms work better on more diverged DNA strands but are less accurate. Since each side of the DNA strand will remain closely related to the other, even with 15% error, a global alignment system was used instead of a local alignment system.

The output of the Needleman-Wunsch global alignment algorithm is an aligned sequence with mutations (in this context errors) represented as dashes. By comparing the divergence of both sequences, errors in the file can be identified.

Dashes present on both sides represent a base mismatch as shown in Figure 5. Dashes only on the data strand represent either a deletion on the data strand or an insertion on the error correction strand as shown in Figure 6. Dashes only on the error correction strand represent either a deletion on the error correction strand or an insertion on the data strand as shown in Figure 7. In the last two cases, dashes could represent errors in the data or error correction strands. The origin of the error though is irrelevant because both will cause a frameshift mutation.

```
Data | A T - G C
  EC | A T - G C
```

Figure 5: Mismatch

```
Data | A T C - C
  EC | A T C G C
```

Figure 6: Error on data strand

```
Data | A T C G C
  EC | A T C - C
```

Figure 7: Error on EC strand

2.3.3.2. Frameshift Correction

Since the file is split into blocks of known size, an algorithm can be run to remove the frameshifts as shown in Figure 9. First, the identified errors from both strands are mapped to the data strand. This identifies all errors in the data strand at the cost of introducing additional errors from the error correction strand. In the example in Figure 8, the green box shows the correct length Y for Block 1. The dotted line shows the length of the block initially and the black line represents the correct length of the block. If no errors are present in this block and the BCH code clears, the block is considered complete and is removed from the data strand.

However, if errors do exist, a loop must be run which adds or removes up to X bases at error positions, X being the number of errors present in the block. Insertion errors are removed from the sequence, shifting the sequence left. Deletion errors are removed by adding a placeholder base and shifting the sequence right. The algorithm runs through all possible combinations of insertions and deletions, starting with all errors considered insertions. Once the frameshift mutations are removed, the BCH code corrects the remaining substitution errors in binary. The corrected block is removed from the data strand and the algorithm repeats on the next block.

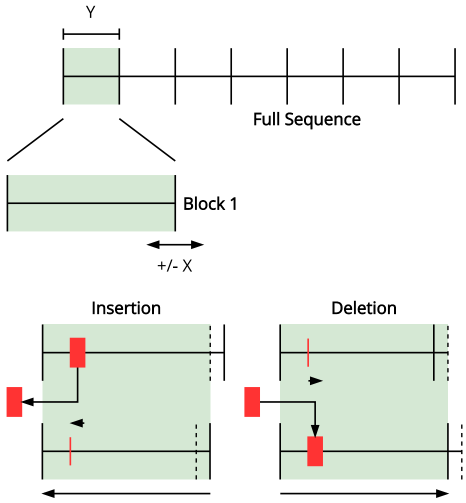
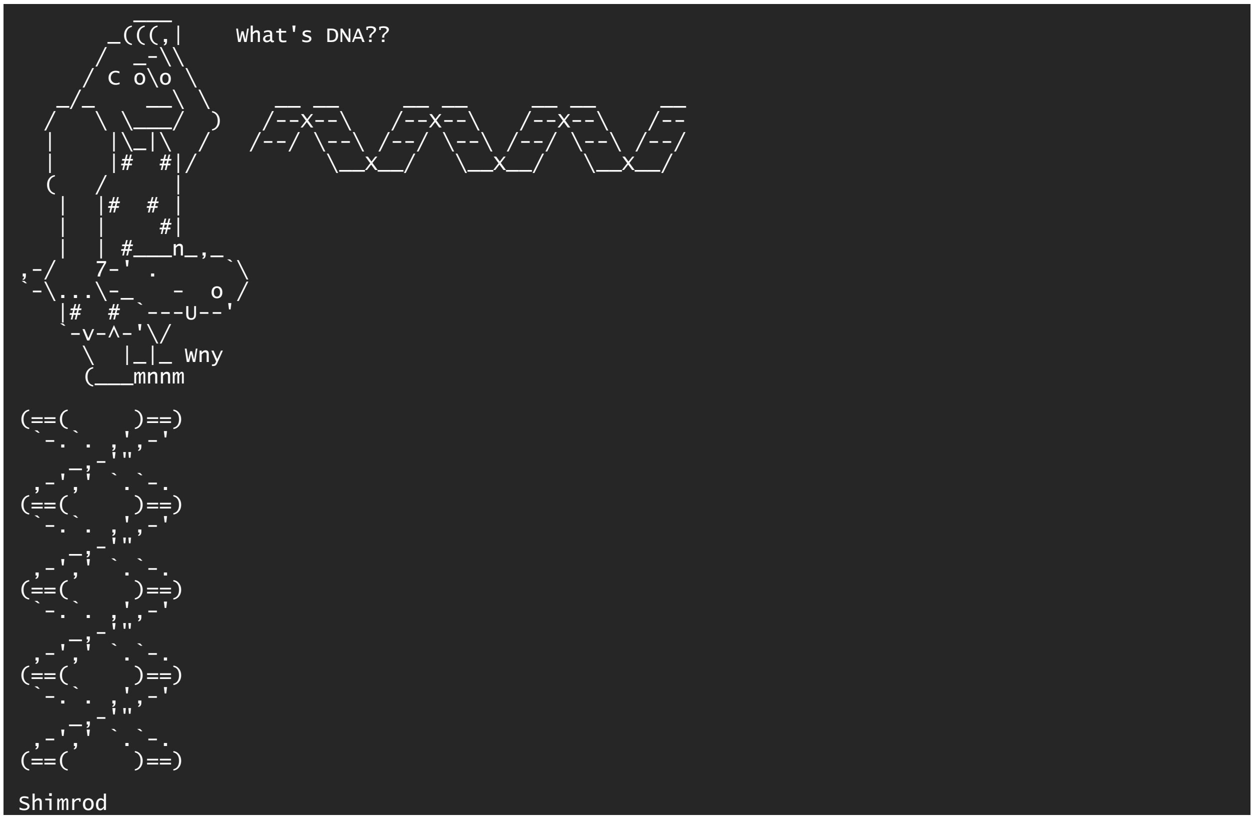


Figure 8: Frameshift algorithm diagram on sequence with one error.

2.3.4. Data Payload



```
         ___
       _(((,|    What's DNA??
      /  _-\\
     / C o\o \
   _/_    __\ \     __ __     __ __     __ __     __
  /   \ \___/  )   /--X--\   /--X--\   /--X--\   /--
  |    |\_|\  /   /--/ \--\ /--/ \--\ /--/ \--\ /--/
  |    |#  #|/          \__X__/   \__X__/   \__X__/
  (   /     |
   |  |#  # |
   |  |    #|
   |  | #___n_,_
,-/   7-' .     `\
`-\...\-_   -  o /
   |#  # `---U--'
   `-v-^-'\/
     \  |_|_ Wny
     (___mnnm

(==(     )==)
 `-.`. ,',-'
    _,-'"
 ,-',' `.`-.
(==(     )==)
 `-.`. ,',-'
    _,-'"
 ,-',' `.`-.
(==(     )==)
 `-.`. ,',-'
    _,-'"
 ,-',' `.`-.
(==(     )==)
 `-.`. ,',-'
    _,-'"
 ,-',' `.`-.
(==(     )==)

Shimrod
```

Figure 9: ASCII text art that composes the test file data.txt

2.4. Laboratory Methods

2.4.1. Strand Assembly

NEBuilder was used to assemble DNA fragments into a plasmid, in this case pUC19. The linearized plasmid (cut with restriction enzyme NdeI) and DNA fragments are mixed with the provided solution and incubated at 50°C for 15-60 minutes. During the reaction as shown in Figure 10, an exonuclease chews back the 5’ ends to create a 3’ overhang. Complementary overlaps between fragments will then base pair with each other, with DNA polymerase filling in the gaps. DNA ligase seals nicks, producing an assembly solution with the desired insert as shown in Figure 11. Assembly solution was used as needed for bacterial transformation.

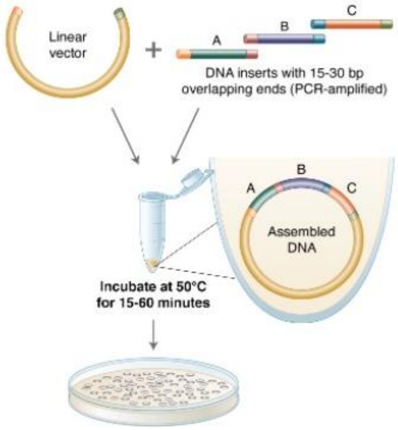

Linear vector
+
A
B
C
DNA inserts with 15-30 bp overlapping ends (PCR-amplified)
A
B
C
Assembled DNA
Incubate at 50°C for 15-60 minutes

Figure 10: NEBuilder process (NEB)

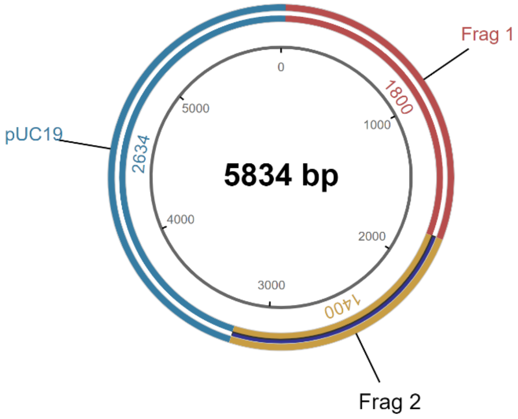


Figure 11: Assembled plasmid with data insert

2.4.2. Bacterial Transformation & Sequencing

The assembled plasmid was used to transform DH5α Calcium Competent E. coli cells according to Figure 12. After an initial inoculation plate step using Ampicillin, the transformed bacteria will be grown from a 3mL culture up to a 50mL culture for sequencing. An alkaline lysis as shown in Figure 13 was used to prep the plasmid from the transformed bacteria.

The plasmid was sequenced using Sanger sequencing. This process, while far more accurate and affordable than nanopore sequencing, can only sequence about 800 bp of DNA before results become unreliable. Consequently, primer sets must be designed to split up the plasmid into smaller sections for sequencing as shown in Figure 14. The data portion is split into five chunks with forward and reverse primers for bidirectional sequencing. They are about 700 bp apart from one another with 60 bp overlaps for decoding alignment.

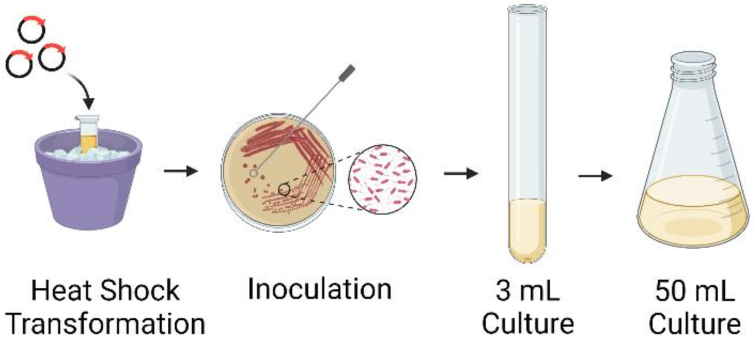


Figure 12: Transformation workflow (created with Biorender)

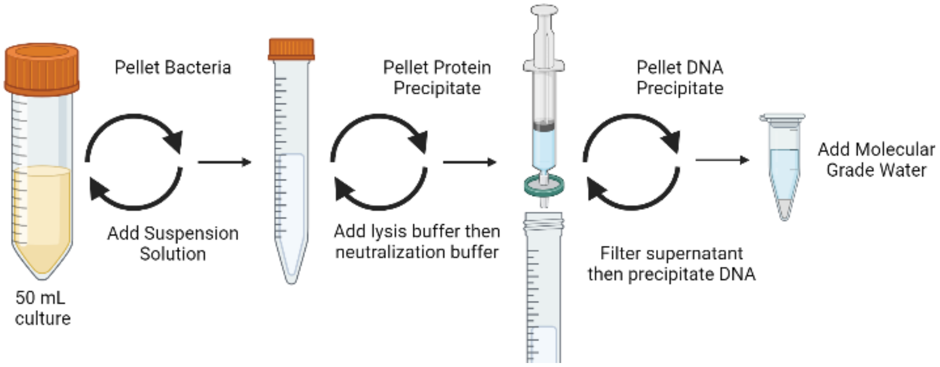

Pellet Bacteria
Add Suspension Solution
50 mL culture
Pellet Protein Precipitate
Add lysis buffer then neutralization buffer
Pellet DNA Precipitate
Filter supernatant then precipitate DNA
Add Molecular Grade Water

Figure 13: Alkaline lysis workflow (created with Biorender)

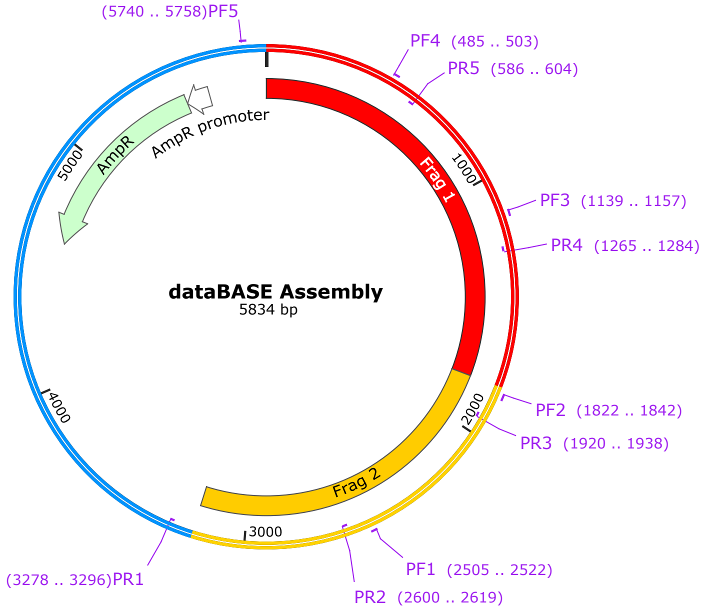


Figure 14: Assembly plasmid map with primers

2.4.3. Primer Design/Optimization Rules

Primers for sequencing must have a length of 18-24 base pairs, contain 40-60% G/C content, end with 1-2 G/C pairs, and avoid repeated sequences. These constraints must be considered when designing primers for amplification and sequencing. The primer sets used are shown in Table 1.

Table 1: Sequencing primer sets

	Sequence	bp	%GC	T_m (C)
PR1	TGGCGTAATAGCGAAGAGG	19	53	57
PF1	TACATACTGTACACGGCC	18	50	57
PR2	TCGAACTGGACAAGCCGTAC	20	55	59
PF2	GAAGCGTAGCCAGTCACAATG	21	52	59
PR3	GGTTGTATAGCCTCCAACC	19	53	53
PF3	TGTTACTTCGTTCGCGTCG	19	53	58
PR4	CCGATAAGATCTAGCCGGTG	20	55	57
PF4	ACAGCCACTAAGTTCAGCG	19	53	58
PR5	CAATCGTAACGCATTCGAG	19	47	54
PF5	TTGTCTGTAAGCGGATGCC	19	53	57

2.5. Procedure

Materials

- NEBuilder® HiFi DNA Assembly Cloning Kit (E5520S)
- NdeI Endonuclease
- KpnI Endonuclease
- PvuI Endonuclease
- HindIII Endonuclease
- Q5® High-Fidelity DNA Polymerase
- DH5α E. coli
- DNA Fragments
- TAE Buffer
- Agarose
- LB-amp broth
- Ethidium bromide
- Glycerol

Wet Lab Procedure

1. Cut pUC19 plasmid with NdeI using Restriction Enzyme Digestion protocol from NEB[4]
2. Assemble plasmid using NEBuilder HiFi DNA Assembly Reaction Protocol from NEB[5]
3. Transform DH5α cells using NEBuilder® HiFi DNA Assembly Electrocompetent Transformation Protocol from NEB[6]
4. Culture transformed bacteria up to 50 mL LB-amp culture
5. Prep assembly bacteria according to Alkaline Lysis Maxiprep Protocol[7]
6. Cut assembly plasmid with KpnI, PvuI, HindIII using Restriction Enzyme Digestion protocol from NEB[4]
7. Run Gel Electrophoresis on Step 5 digests using Agarose Gel Electrophoresis protocol from Addgene[8]
8. Prep remaining uncut assembly plasmid for DNA sequencing according to Genewiz guidelines

3. Results

3.1. Full Sequence

AGCAGATTGTACTGAGAGTGCACCACTCGAGCAGCTACACTCCCGCCTTCAGTCCTACGAACGACTTCGAATGCGGCTGTAAGAA
TGTTGCTGTTACTGTTTCTGGTTGTACGACTGCTCGTGGATCTGCTTATGCTTGTAAGAGTGCCGTTGATGATGCCCTTGACGCTGG
CAATGCCATCACTACCATCAGTGCTGGTGACCATGATACTGCTGTTGCTTGCACTCCCACCTTTGGTTCTGACGGCAGCATCATCA
GTGGATGCAACCGCACTACTGGTATCATCCTTGATTACATTATTGACTATGTCTTTGTTCACACTGCTGACATCAACCTCATCCACA
ACTCTGGTGCTGTCTGCACCTATGGATATGTTATTGCTTATAGGATGCGAACCGTACGCAGTGAGATCGGAACTAGGAGAATCCTG
CAAATTAGCTAGCGCTTCGTGCTAGTACAGCTGCCGTTGAGAGTGTCTACCGCACAGCCACTAAGTTCAGCGTGAGTCTCACCCTG
GGCAGCCCTCCATGGTACTTATTCTACTTGTACATTAGCCCGTACGGAGGCGCACAGTTGAGCGAATTCTCGAATGCGTTACGATT
GTTAACTTAGTTGGATTATATCTCCGACCAATTACTCGCTGCTTGCTTCTCGTTCTTATGCCTCGTTGTGGGCCTCCCTAATCGTCAC
AGAGCACTGATTTAATGCTCGATAAAGGTAGGCATATCTGGACTCGCATCTTGTGTATCTGATCCGGATGAAAAGGTCATTGCTGC
GTGAATTACCACAGCCTAAATTGTAGATGGACTTCTAATACATAGTTACTTGAGGAACCTAGCGTACAAGCCAAATGGAAGCTCG
CGCGACCCTGTCAATACACAAAGGGGCTAACTACTGACGCGATCACCTTATGTGAAGGCAAAGAACGTTCCATTGTTAAGTTATA
TATCTGTGAATGGAGTCTTGTTAAAGCTGAGGCTGTGCGTTTAGAGCTTGCACTTACACTTGACTCGAAATGTTACTTACAATAAG
CTGTCGGTCCATAACATACGTCAGTTATTGGTACACGAAATTCTGCCTTTGCACTGGCCAAAGACACAACTTCCACCAAGGAGCTT
CTGACTCTCGATAGACATATGTTACTTCGTTCGCGTCGAATACGAACCGTCGACACTAGATCTCGAGTCTGTATACTGTGACATGT
TCCGGTTGGCGGCGACAAACAATGCCAACACACGCGACAGCACCGGCTTACAGAGGTCCCACCGGCTAGATCTTATCGGACGCTA
TCCTGGCCATCGAATCGCACATTTCCATGGCAATGTCAGACCAATACCGGGTGGAATAACGCCTTTGTCTCTAGGAGCCGGAAAG
CACAATTCGTGCGCATGATCAATGGTTTCCTCGTTCCCTCATGTCCACTGTGACAGGTTAAAGGCACACAAGATTCGGCCTGCCCC
TAATAGGCGCAGCGCATGCTGGAATCCAACGCTTCCACAGCGACACAATACGCCATATATAGCGTTGTGCATGGTGGATGTAATC
GACAACGACGGAGTATGACGGATTGTAAGGCCAAGATATACGAATGGTAACGCAATCTGCACCGCTAATTCGTGAGGTTAGTGCT
GTCAACCACAAGATAAGACCCTTAGCTTGATCTTGGTTGGCCTCGCCTTAATTCGGTACCACGACATTTGCAGTCACGTTCAGTGG
TACGGTTACTGGCGTGTATTAGATGGCGAAATGAGCTGCGGCATTAGAGTCAGCCATATCATGGCAAGGCTCAGATGTATAATCC
ATGCGAAGGAGGATCGCCTGAAGCGTAGCCAGTCACAATGAGCAAGAAATTGCGTTCCGGGAGGCGCGACAAAGGTCGCAGCCG
CTGGCCAGGTATCCGATGCGGGATAACCAACCCGGTTGGAGGCTATACAACCGGGTAATTAGGCGTACCAATCCTAATTATAATT
ACCGACCATCAGTCTGGCTTTGTGGTGCGCGACGCTTTCATCTACACGCTAGAGTGGAGGTTTGAGTTATCAGGTTTGAAGCAGCA
AGGTCAATCAGGCCGAACTTGGCGGATCAGTCCTAATATCATAATCTCTATCCACGCAAAACGTCAAGATTAAGTTACTACGACC
GTGGTTGTTAGGTTTCACGTTCAGGATGAAGATCCCAGTGGAATGAACTCGCCAAGGAGGCTCCAGATAATGGTAATTACTCTTA
ATCATTTCGTTAAGTAACGCCAGTCACTTCCGAACTGTCAATCTACTAGATCCAATGTAAAGTCTACCTCCAGCGGCTCGTACGAG
CACAGCCGTTTGATCTTTCATTTCAGATTTGCAGTTGGCTTACTGAGGACGAGGCGTAGGCTGCTTCATTGAGGCGCAATAGGCAG
CCCTGTACACATGGGCTACATTAAGCTGCAGGGCGATGCGCTAAACACCATTGTGAATCGTGATATTGAACGCACAGCCGCGCGT
TACGTGGCTATCGTTCTTACTACATACTGTACACGGCCAGCTATACGTCTGGCCTTCCTGACGATTATTCTTCGCATCCATTTTCTC
ATCCCGCACCGATACTTGGCCAATTGACGTACGGCTTGTCCAGTTCGACGTTATGTCTCCTGCCAACATACTCTCTCATGCACCTA
CATGTCGAGCATTCCCTCAGCCATCATCAAGATCGGAAGAAACGTATTAAGTCGGATTGTATGTCGCATGATACATAAATTTGTAT
GATGATCAATCGCACATCCGCCGATGGGTCCTCCTTCATTTGAAGCCAAGTCGTCGATATCAGTGTATCCTGAGCAGTTAGTGCGT
TTATACGCGACCAATAGTTTCACAGGCGAACCCTTTACATGTTGATCACACGTCGGGATTAAGATCGAATAACGTAACCAACGGA
TCCGTTCGGTTATGGTTCTACATAGGACAGTCTACGCTTCTAATCTGTGACGATTAGACACTCTCAGTTGGCCTTACCTCATACATG
TTATCGGCCATTTCGCCTCAATTCGCGAATACGAGGCCACCGGAGGGATACTCTGTGGACTAACCATCTAGTTACTAAGTTGCTTA
TGAATGTACCGTGTGCGCCCTGCCTAATGCGTGTAACTGACACTTAGGTTGGGAGGACGGTTAGGCGTACTAGCCTCTGGAACTC
GAGTGCGGTGTGAAATACCGCACAGATG **(3,200 bp)**

3.2. Fragment #1

AGCAGATTGTACTGAGAGTGCACCACTCGAGCAGCTACACTCCCGCCTTCAGTCCTACGAACGACTTCGAATGCGGCTGTAAGAA
TGTTGCTGTTACTGTTTCTGGTTGTACGACTGCTCGTGGATCTGCTTATGCTTGTAAGAGTGCCGTTGATGATGCCCTTGACGCTGG
CAATGCCATCACTACCATCAGTGCTGGTGACCATGATACTGCTGTTGCTTGCACTCCCACCTTTGGTTCTGACGGCAGCATCATCA
GTGGATGCAACCGCACTACTGGTATCATCCTTGATTACATTATTGACTATGTCTTTGTTCACACTGCTGACATCAACCTCATCCACA
ACTCTGGTGCTGTCTGCACCTATGGATATGTTATTGCTTATAGGATGCGAACCGTACGCAGTGAGATCGGAACTAGGAGAATCCTG
CAAATTAGCTAGCGCTTCGTGCTAGTACAGCTGCCGTTGAGAGTGTCTACCGCACAGCCACTAAGTTCAGCGTGAGTCTCACCCTG
GGCAGCCCTCCATGGTACTTATTCTACTTGTACATTAGCCCGTACGGAGGCGCACAGTTGAGCGAATTCTCGAATGCGTTACGATT
GTTAACTTAGTTGGATTATATCTCCGACCAATTACTCGCTGCTTGCTTCTCGTTCTTATGCCTCGTTGTGGGCCTCCCTAATCGTCAC
AGAGCACTGATTTAATGCTCGATAAAGGTAGGCATATCTGGACTCGCATCTTGTGTATCTGATCCGGATGAAAAGGTCATTGCTGC
GTGAATTACCACAGCCTAAATTGTAGATGGACTTCTAATACATAGTTACTTGAGGAACCTAGCGTACAAGCCAAATGGAAGCTCG
CGCGACCCTGTCAATACACAAAGGGGCTAACTACTGACGCGATCACCTTATGTGAAGGCAAAGAACGTTCCATTGTTAAGTTATA
TATCTGTGAATGGAGTCTTGTTAAAGCTGAGGCTGTGCGTTTAGAGCTTGCACTTACACTTGACTCGAAATGTTACTTACAATAAG
CTGTCGGTCCATAACATACGTCAGTTATTGGTACACGAAATTCTGCCTTTGCACTGGCCAAAGACACAACTTCCACCAAGGAGCTT
CTGACTCTCGATAGACATATGTTACTTCGTTCGCGTCGAATACGAACCGTCGACACTAGATCTCGAGTCTGTATACTGTGACATGT
TCCGGTTGGCGGCGACAAACAATGCCAACACACGCGACAGCACCGGCTTACAGAGGTCCCACCGGCTAGATCTTATCGGACGCTA
TCCTGGCCATCGAATCGCACATTTCCATGGCAATGTCAGACCAATACCGGGTGGAATAACGCCTTTGTCTCTAGGAGCCGGAAAG
CACAATTCGTGCGCATGATCAATGGTTTCCTCGTTCCCTCATGTCCACTGTGACAGGTTAAAGGCACACAAGATTCGGCCTGCCCC

TAATAGGCGCAGCGCATGCTGGAATCCAACGCTTCCACAGCGACACAATACGCCATATATAGCGTTGTGCATGGTGGATGTAATC
GACAACGACGGAGTATGACGGATTGTAAGGCCAAGATATACGAATGGTAACGCAATCTGCACCGCTAATTCGTGAGGTTAGTGCT
GTCAACCACAAGATAAGACCCTTAGCTTGATCTTGGTTGGCCTCGCCTTAATTCGGTACCACGACATTTGCAGTCACGTTCAGTGG
TACGGTTACTGGCGTGTATTAGATGGCGAAATGAGCTGCGGCATTAGAGTCAGCCATATCATGGCAAGGCTCAGATGTATAAT
(1,800 bp)

3.3. Fragment #2

TCATGGCAAGGCTCAGATGTATAATCCATGCGAAGGAGGATCGCCTGAAGCGTAGCCAGTCACAATGAGCAAGAAATTGCGTTCC
GGGAGGCGCGACAAAGGTCGCAGCCGCTGGCCAGGTATCCGATGCGGGATAACCAACCCGGTTGGAGGCTATACAACCGGGTAA
TTAGGCGTACCAATCCTAATTATAATTACCGACCATCAGTCTGGCTTTGTGGTGCGCGACGCTTTCATCTACACGCTAGAGTGGAG
GTTTGAGTTATCAGGTTTGAAGCAGCAAGGTCAATCAGGCCGAACTTGGCGGATCAGTCCTAATATCATAATCTCTATCCACGCAA
AACGTCAAGATTAAGTTACTACGACCGTGGTTGTTAGGTTTCACGTTCAGGATGAAGATCCCAGTGGAATGAACTCGCCAAGGAG
GCTCCAGATAATGGTAATTACTCTTAATCATTTCGTTAAGTAACGCCAGTCACTTCCGAACTGTCAATCTACTAGATCCAATGTAA
AGTCTACCTCCAGCGGCTCGTACGAGCACAGCCGTTTGATCTTTCATTTCAGATTTGCAGTTGGCTTACTGAGGACGAGGCGTAGG
CTGCTTCATTGAGGCGCAATAGGCAGCCCTGTACACATGGGCTACATTAAGCTGCAGGGCGATGCGCTAAACACCATTGTGAATC
GTGATATTGAACGCACAGCCGCGCGTTACGTGGCTATCGTTCTTACTACATACTGTACACGGCCAGCTATACGTCTGGCCTTCCTG
ACGATTATTCTTCGCATCCATTTTCTCATCCCGCACCGATACTTGGCCAATTGACGTACGGCTTGTCCAGTTCGACGTTATGTCTCC
TGCCAACATACTCTCTCATGCACCTACATGTCGAGCATTCCCTCAGCCATCATCAAGATCGGAAGAAACGTATTAAGTCGGATTGT
ATGTCGCATGATACATAAATTTGTATGATGATCAATCGCACATCCGCCGATGGGTCCTCCTTCATTTGAAGCCAAGTCGTCGATAT
CAGTGTATCCTGAGCAGTTAGTGCGTTTATACGCGACCAATAGTTTCACAGGCGAACCCTTTACATGTTGATCACACGTCGGGATT
AAGATCGAATAACGTAACCAACGGATCCGTTCGGTTATGGTTCTACATAGGACAGTCTACGCTTCTAATCTGTGACGATTAGACAC
TCTCAGTTGGCCTTACCTCATACATGTTATCGGCCATTTCGCCTCAATTCGCGAATACGAGGCCACCGGAGGGATACTCTGTGGAC
TAACCATCTAGTTACTAAGTTGCTTATGAATGTACCGTGTGCGCCCTGCCTAATGCGTGTAACTGACACTTAGGTTGGGAGGACGG
TTAGGCGTACTAGCCTCTGGAACTCGAGTGCGGTGTGAAATACCGCACAGATG **(1,425 bp)**

4. Discussion

This was a difficult project to complete simply due to the number of moving parts involved. On the computer science side, the planning, coding, and bug testing required for a working build of the encoding/decoding method outlined in Figure 1 took time. That time was simultaneously needed for the biological side of this study, which composed of learning the requisite lab skills needed to assemble the strand and prepare it for sequencing without error. This required a careful balancing act of competing priorities: working computer code and sequencing ready DNA. Further complicating matters were designing around the biological quirks of DNA on a computer; preventing repetitions was particularly difficult.

The dataBASE codec as detailed in Figure 1 has met the requirements described in Purpose. The encoder is stable and efficient in its current build, executing an encoding operation at a rate of ~70 kb/sec, though this number depends on the file size. It also prevents most repeat sequences as shown in Figure 3, though more work is needed to fine-tune this algorithm for edge-case manufacturing issues. The decoder as shown in Figure 4 can extract and remove metadata with high accuracy, though it is currently unable to automatically assemble raw sequencing data. This feature is under active development, however, more sample data is needed to ensure reliability. The alignment-based error correction system as shown in Figure 8 is effective at removing all simulated errors up to 15%. While slow past this point, more optimizations with the encoder could push this error rate threshold even further.

Bacterial transformation and plasmid prep has been effective in producing high concentrations (~378 ng/µL) of pUC19 parent plasmid in DH5α E. coli bacteria. The addition of an RNase step in conjunction with Lithium Chloride and PEG precipitations during purification has resolved a long-running RNA contamination issue during plasmid prep. The NEBuilder assembly reaction so far has produced more mixed results, with bacterial colonies seemingly producing recircularized pUC19 plasmid without data inserts. This should be a rare occurrence, and other colonies from this reaction are currently being tested to determine if any have the correctly assembled plasmid. As a result, the assembly as shown in Figure 14 cannot yet be sequenced, thus precluding a full test of the dataBASE codec with real-world data.

5. Conclusion

The dataBASE codec has met all design goals while maintaining reliability and speed. The encoding method can successfully generate a manufacturable DNA strand, avoiding G/C balancing issues and repetitive sequences. Error correction can identify and fix errors in virtual DNA sequences with accuracy approaching 100%. On the biological side, the failure of NEBuilder to insert data fragments has precluded the sequencing of the data plasmid and a full proof of concept test. Overall, the method described shows promise as a storage system for archival and space applications.

6. Future Studies

Based on the results of this project, several improvements and expansions can be made. An additional restriction site can be added during assembly to prevent the recircularization seen in assembly V4. An adapter primer can be used in conjunction with PCR to add the required overlaps for the new restriction site into the data fragments. For future testing, a longer test strand would be ideal to stress the assembly and sequencing process, proving out larger file sizes. Using nanopore sequencing to retrieve files would be ideal as it can sequence whole DNA molecules. This method would remove the need to split up the data strand for sequencing purposes, greatly simplifying the process but at the cost of additional stress on error correction. Due to budgetary restrictions, these options were unavailable for this study. To further speed up the decoding the codec could be implemented in an FPGA to exploit the speed advantage of using direct transistor logic.

7. Applications

Future applications of this research are varied. The focus here is on space travel, where DNA's resilience, repairability, and power consumption make it an attractive alternative to flash memory and magnetic tape. A system like this could work autonomously onboard a spacecraft, with microfluidics advances such as on-chip DNA synthesis and solid-state nanopore sequencing allowing for a compact package. Such a system would have to be tested for radiation resistance and function in zero-g. Storage of excess nucleic acids and reagents would be another design challenge.

However, storing information in this way fuses both the data itself and the structure that stores it together a major advantage over flash in particular as it prevents bit-flips. This structure means that DNA also does not have a constant power requirement to store data, only to sequence and/or synthesize it. As long as these steps remain low power, a DNA system could free up a spacecraft’s power budget for other things. Its higher density means the actual storage takes up less space, though again some of these gains would be offset by sequencing and synthesis equipment. This synthesis and sequencing process would also lead to much slower write and read speeds, making DNA work better as a backup system rather than main program storage.

DNA can still be damaged by ionizing radiation, but its density would make it a much smaller target to hit compared to existing methods, reducing the rate of degradation. More storage can be synthesized onboard a spacecraft to expand capacity or replace degraded strands, giving near-infinite redundancy as long as more nucleotides are available. A DNA system therefore would be much more resilient than flash memory and much smaller and less power-hungry than a magnetic tape-based system.

For many of these reasons, DNA could also be used as an archival format since its long-term stability makes it more desirable than magnetic tape. Again, DNA data storage will be slower than current storage methods but in cold storage applications, which make up ~60% of the market, this weakness isn’t critical. Having a near-infinite medium for storage, however slow, would be an enormous advantage for data center applications as well. To realize this, however, the size of a synthesizer/sequencer would also have to be small to ensure high density.

Works Cited

[4]AddGene. (n.d.). Agarose Gel Electrophoresis. AddGene. http://www.addgene.org/protocols/gel-electrophoresis/

AddGene. (n.d.). Polymerase Chain Reaction (PCR). AddGene. http://www.addgene.org/protocols/pcr/

AddGene. (n.d.). Purifying DNA from an Agarose Gel. AddGene. http://www.addgene.org/protocols/gel-purification/

Banal, J. L., Shepherd, T. R., Berleant, J., Huang, H., Reyes, M., Ackerman, C. M., Blainey, P. C., & Bathe, M. (2021). Random access DNA memory using Boolean search in an archival file storage system. Nature Materials, 20(1272-1280). https://doi.org/10.1038/s41563-021-01021-3

Ferrell, K. (2021). Hard drive. In World Book Advanced.https://www.worldbookonline.com/advanced/article?id=ar722718

How is Data Stored on Tape? [Fact sheet]. (2021, August 24). We Buy Used Tape. Retrieved November 9, 2021, from https://webuyusedtape.net/2021/08/24/how-is-data-stored-on-tape/

Ionkov, L., & Settlemyer, B. (2021, May 28). DNA: The Ultimate Data-Storage Solution. Scientific American. https://www.scientificamerican.com/article/dna-the-ultimate-data-storage-solution/

Lee, H., Wiegand, D. J., Griswold, K., Punthambaker, S., Chun, H., Kohman, R. E., & Church, G. M. (2020). Photon-directed multiplexed enzymatic DNA synthesis for molecular digital data storage. [Abstract]. Nature, 5246. https://doi.org/10.1038/s41467-020-18681-5

Lee, S. Y. (2019, July 1). DNA Data Storage Is Closer Than You Think. Scientific American. https://www.scientificamerican.com/article/dna-data-storage-is-closer-than-you-think/

Magnetic Core Memories: What they are and how they function [Fact sheet]. (n.d.). vt100. Retrieved November 9, 2021, from https://vt100.net/docs/misc/core/

McDowell, J. C. (2023, March 1). GCAT: General Catalog of Artificial Space Objects. Jonathan's Space Report | GCAT. Retrieved March 1, 2023, from https://planet4589.org/space/gcat/web/cat/llog.html

[1]MIT News. (2021, June 10). Could all your digital photos be stored as DNA? [Press release]. https://news.mit.edu/2021/dna-data-storage-0610

[4]NEBcloner. (n.d.). Restriction Enzyme Digestion. NEBcloner. NEBcloner. https://nebcloner.neb.com/#!/protocol/re/single/NdeI

[6]New England BioLabs. (n.d.). NEBuilder® HiFi DNA Assembly Electrocompetent Transformation Protocol. New England BioLabs. New England BioLabs. https://www.neb.com/protocols/2015/02/11/nebuilder-electrocompetent-transformation-protocol-e5520

[5]New England BioLabs. (n.d.). NEBuilder HiFi DNA Assembly Reaction Protocol. New England BioLabs. https://www.neb.com/protocols/2014/11/26/nebuilder-hifi-dna-assembly-reaction-protocol

New England Biolabs. (n.d.). PCR Using Q5® High-Fidelity DNA Polymerase. New England Biolabs. https://www.neb.com/protocols/2013/12/13/pcr-using-q5-high-fidelity-dna-polymerase-m0491

[3]Nwankwo, V. U., Jibiri, N. N., & Kio, M. T. (2020). The Impact of Space Radiation Environment on Satellites Operation in Near-Earth Space. In V. Demyanov, & J. Becedas (Eds.), Satellites Missions and Technologies for Geosciences. IntechOpen. https://doi.org/10.5772/intechopen.90115

Piantanida, L., & Hughes, W. L. (2021). A PCR-free approach to random access in DNA. Nature Materials, 20(1173-1174). https://doi.org/10.1038/s41563-021-01089-x

T. R. Oldham et al., "TID and SEE Response of Advanced Samsung and Micron 4G NAND Flash Memories for the NASA MMS Mission," 2009 IEEE Radiation Effects Data Workshop, 2009, pp. 114-122, doi: 10.1109/REDW.2009.5336305.

Tanaka, T., & Letsinger, R. L. (1982). Syringe method for stepwise chemical synthesis of oligonucleotides. Nucleic Acids Research, 10(10). https://doi.org/10.1093/nar/10.10.3249

[2]Villas-Boas, A. (2019, July 2). Tech companies have been silently battling a bizarre phenomenon called 'cosmic rays' that would otherwise wreak havoc on our electronics. Insider. https://www.businessinsider.com/cosmic-rays-harm-computers-smartphones-2019-7

[7]Willett, S., Dr. (2022). Plasmid Prep Protocol, Alkaline Lysis Method, LiCl & PEG Precipitations.