AUTOMATED HTML REPORT PARSING AND EXCEL INTEGRATION SYSTEM"

At STMicroelectronics Pvt. Limited

A Thesis

Submitted in the partial fulfilment of the requirements for the award of the degree **of**

Master of Engineering *in*
Electronics & Communication Engineering

Submitted by

Divya Sharma
Roll No. 802261001

Faculty Mentor
Dr. Mayank Agarwal
Asst. Professor
ECED, TIET,
Patiala

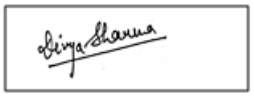
Industry Mentor
Er. Misbah Ur Rahman
Senior Staff Engineer
STMicroelectronics,
Greater Noida

ELECTRONICS AND COMMUNICATION ENGINEERING DEPARTMENT
TIET, PATIALA-147004, PUNJAB
INDIA
JULY 2024

CERTIFICATE

This is to certify that the project titled "**Automated HTML Report Parsing and Excel Integration System**," submitted by **Divya Sharma** bearing the University Registration No. **802261001**, to **the Department of Electronics and Communication Engineering at Thapar Institute of Engineering & Technology (TIET), Patiala, Punjab**, is an authentic record of the original work carried out by her under the guidance and supervision of Mr.Naveen Gupta.

This project report is submitted in partial fulfillment of the requirements for the degree of ME. It is hereby acknowledged that the content of this project report does not contain any material previously published or written by another person, except where due reference is made in the text of the report.



Divya

802261001

DECLARATION

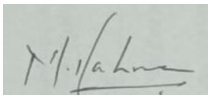
I hereby declare that the project work entitled is " **AUTOMATED HTML REPORT PARSING AND EXCEL INTEGRATION SYSTEM**" in partial fulfillment of the requirement of the award of the degree **of Master of Engineering(ECE)** submitted at **Electronics and Communication Engineering Department**, Thapar institute of Engineering & Technology(Deemed to be university),Patiala is a record of work carried out under supervision of **Dr. Mayank Agarwal**(Assistant Professor, Electronics and Communication Engineering Department, Thapar Institute of Engineering & Technology(Deemed to be university) from **JUNE 2023 TO JUNE 2024**. The Matter in this has not been submitted in part to any other university or institute for the award of any other degree.

Date:26.06.2024

Certified that the above statement made by the student is correct to the best of our knowledge and belief.

.

Faculty Mentor
Dr. Mayank Agarwal
Asst. Professor
ECED, TIET,
Patiala



Industry Mentor
Er. Misbah Ur Rahman
Senior Staff Engineer
STMicroelectronics,
Greater Noida

ACKNOWLEDGEMENT

I would like to express my sincere appreciation to the individuals who have played a pivotal role in shaping my internship journey at **STMicroelectronics**. Their unwavering guidance, support, and encouragement have been invaluable, and I am deeply grateful for their contributions towards my professional growth.

First and foremost, I extend my heartfelt thanks to my mentor, **Mr. Misbah Ur Rahman**, whose wealth of knowledge, expertise, and guidance have been instrumental in shaping my professional growth during this internship. His patience, encouragement, and willingness to share his insights and experiences have been truly inspiring. I am grateful for the time and effort he dedicated to mentoring me and for his unwavering support throughout the internship.

I also express my deep appreciation to my manager, **Mr. Naveen Gupta**, whose leadership, guidance, and trust in my abilities have been invaluable. He provided me with challenging projects, allowed me to take ownership of my work, and provided constructive feedback that greatly contributed to my personal and professional development. I am grateful for his mentorship and for creating a positive and inclusive work environment that fostered learning and growth.

I wish to extend my gratitude to my college mentor, **Dr. Mayank Agarwal**, for his continuous support and guidance throughout my internship. His advice, wisdom, and encouragement have been invaluable in helping me navigate through the challenges and make the most out of this experience. I am thankful for his unwavering support and for being a constant source of motivation.

I acknowledge the college placement cell, **CILP** (Centre for Industrial Liaison & Placement), Thapar Institute of Engineering & Technology, and the talent acquisition head**, Mrs. Shweta Butola**, for their efforts in connecting me with this internship opportunity and facilitating the entire process.

Lastly, I extend my sincere thanks to all the team members at STMicroelectronics, who made my internship experience truly enriching and memorable. Their support, collaboration, and camaraderie made me feel welcomed and encouraged me to push my boundaries and excel in my work. I am grateful for the opportunity to work alongside such talented and dedicated individuals and for the invaluable lessons learned during this internship.

Divya Sharma

ABSTRACT

The "**Automated HTML Report Parsing and Excel Integration System**" is a sophisticated tool designed to streamline the process of extracting relevant data from HTML reports and seamlessly integrating this information into Excel spreadsheets. This system is engineered to enhance efficiency, accuracy, and productivity in data analysis and reporting tasks.

The core functionality of this system revolves around an automated parsing module that meticulously scans HTML documents. It employs advanced algorithms to identify and segregate data points based on predefined criteria. Once extracted, the data is then structured into a format that is compatible with Excel, facilitating easy manipulation, analysis, and visualization.

The integration with Excel is achieved through a robust interface that allows for the automatic population of spreadsheets with the parsed data. This integration supports a variety of Excel functionalities, including but not limited to, data sorting, filtering, and the use of formulas and pivot tables for comprehensive data analysis.

One of the key benefits of this system is the significant reduction in manual data entry errors and the time savings realized by automating the data transfer process. It ensures that data is quickly and accurately reflected in Excel, enabling stakeholders to make informed decisions based on the latest information.

The system is designed with a user-friendly interface, ensuring that it is accessible to users with varying levels of technical expertise. Additionally, it is scalable and can be customized to accommodate the evolving needs of businesses and organizations.

In summary, the **"Automated HTML Report Parsing and Excel Integration System"** represents a pivotal advancement in data management, providing a seamless bridge between HTML reports and Excel's analytical capabilities. It stands as a testament to the potential for automation to revolutionize data processing and business intelligence methodologies.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1 - INTRODUCTION

1.1 About the Company (STMicroelectronics Pvt. Ltd.)

STMicroelectronics India Pvt. Ltd. is a subsidiary of STMicroelectronics, a global leader in semiconductor solutions. With a strong presence in the Indian market, STMicroelectronics India has established itself as a leading provider of innovative semiconductor products and solutions. STMicroelectronics India Pvt. Ltd. in Greater Noida is a prominent research and development centre that contributes significantly to the Indian semiconductor industry. With state-of-the-art facilities and a skilled workforce, the site specializes in designing and developing advanced semiconductor technologies. It collaborates with academic institutions, research organizations, and industry partners to drive innovation and stay at the forefront of technology. Committed to corporate social responsibility, STMicroelectronics India in Greater Noida promotes sustainable practices and community development. The site's dedication to excellence, collaboration, and social responsibility positions it as a trusted player in the semiconductor market, delivering cutting edge solutions across various industry sectors.

Figure1Image

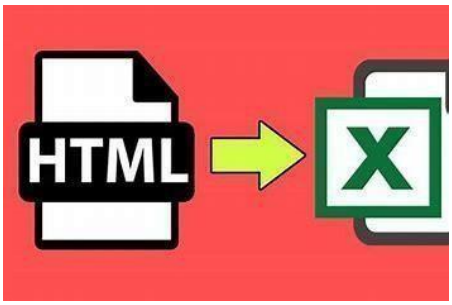
Product Portfolio:

- **Microcontrollers and Microprocessors :**Ranging from ultra-low-power devices for battery operated applications to high-performance solutions
- **Analog and Mixed-Signal Ics :** These components include amplifiers, converters, sensors, and power management ICs
- **Power Devices and Modules :** STMicroelectronics offers a wide range of power devices and modules, including MOSFETs, IGBTs, power transistors, and diodes
- **MEMS and Sensors:** portfolio includes motion sensors, environmental sensors, pressure sensors, and microphones
- **Wireless Connectivity :** STMicroelectronics offers a range of wireless connectivity solutions, including Wi-Fi, Bluetooth, NFC, and RF transceivers
- **Imaging Solutions:** Solutions for applications such as smartphones, automotive cameras, and industrial vision systems

CHAPTER 2 – LITERATURE REVIEW ON AUTOMATED HTML REPORT PARSING AND EXCEL INTEGRATION SYSYTEM

2.1 Introduction:

The Automated HTML Report Parsing and Excel Integration System is designed to automate the extraction of data from HTML reports and its subsequent integration into Excel spreadsheets. This system serves to optimize data management workflows, reduce manual intervention, and improve the accuracy of data analysis within various industries. The code provided appears to outline the practical implementation of such a system, encompassing HTML parsing, data extraction, transformation, and integration with Excel.



The Python script used here is a multi-faceted program that performs a series of operations on file directories, parses data from HTML files, and ultimately organizes this data into a formatted Excel workbook. The script leverages a combination of Python libraries, each serving a distinct purpose, to automate what would otherwise be a manual and time-consuming task.

2.1.1 Data Parsing

- Data parsing is the process of analyzing and breaking down data into smaller components for easier analysis and manipulation.
- It is commonly used in computer programming, data analysis, and data management.
- Text parsing involves breaking down text data into individual words, phrases, or sentences.

- XML parsing involves extracting data from XML documents using specialized parsers.
- JSON parsing involves extracting data from JSON objects using specialized parsers.
- Data parsing can be done manually or using automated tools and software.
- The output of data parsing can be used for various purposes such as data visualization, data mining, and machine learning.

2.1.2 Libraries Utilized

- **os**: A versatile module that provides a way to interact with the operating system. This module is used to handle file paths, directories, and other OS-level operations including file manipulation.
- **shutil**: A module for high-level file operations such as copying, moving, or deleting files and directories. In this script, it is used to remove certain directories as part of a cleanup operation.
- **BeautifulSoup**: A powerful library for parsing HTML and XML documents. It is adept at navigating the parse tree and retrieving the data needed. The script uses BeautifulSoup to extract text from HTML files.
- **openpyxl**: A library targeted at reading and writing Excel 2010+ xlsx/xlsm files. The script uses this library to create and manipulate an Excel workbook where the parsed data will be stored.
- **re**: The regular expressions module, which is used for string searching and manipulation. This module can be used to find patterns within strings, although its specific use is not demonstrated in the provided code snippet.

```
import time
import os
from bs4 import BeautifulSoup
import openpyxl
import re
from openpyxl import Workbook, load_workbook
from openpyxl.styles import Alignment, Font, PatternFill
from openpyxl.drawing.image import Image as OpenpyxlImage
from PIL import Image as PILImage
from io import BytesIO
import shutil
```

`import time`	•**Module for time-related tasks**: Provides functions for working with times, and for converting between representations •**Sleep function**: The `time.sleep()` function delays execution for a specified number of seconds. •**Time access and conversions**: Functions like `time.time()` and `time.localtime()` provide access to the current time and allow for conversions between different time formats.
`import os`	•**Operating system interfaces**: Offers a way of using operating system dependent functionality like reading or writing to a file system. •**File and directory management**: Functions like `os.remove()` for file deletion, `os.mkdir()` for directory creation, and `os.walk()` for walking through a directory tree. •**System calls**: Facilitates executing system-level operations, such as `os.system()` to run a shell command.
`from bs4 import BeautifulSoup`	•**HTML and XML parsing**: Designed for quick turnaround projects like screen-scraping. •**Navigating parse trees**: Allows you to search and navigate the parse tree that it creates from HTML and XML content. •**Data extraction**: Facilitates extracting data from HTML, which is useful for web scraping.
`from openpyxl.styles import Font, Alignment`	•**Excel file handling**: Used for reading and writing Excel 2010 xlsx/xlsm/xltx/xltm files. •**Workbook and sheet operations**: Allows you to create new workbooks or load existing ones and perform operations on the sheets. •**Cell manipulation**: Provides cell reading and writing, including handling of different data types and styles. •**Styling Excel content**: Used to style Excel cells with fonts, including bold, italic, size, color etc., and alignment like horizontal and vertical alignment. •**Cell appearance customization**: Enhances the visual presentation of spreadsheet data.
`import re`	•**Regular expressions**: Provides regular expression matching operations similar to those found in Perl. •**Pattern matching**: Allows for complex string searching and manipulation. •**String parsing and manipulation**: Useful for validating and parsing strings for patterns or extracting substrings.

2.1.3 HTML to excel sheet Conversion

- The code parses a set of HTML files and extracts specific data from them.
- It uses the os module for file system interaction, allowing the user to select specific folders and files to be parsed.
- The BeautifulSoup library is used to parse the HTML files and extract the desired data.
- The extracted data is written to a text file and an Excel workbook for further analysis.
- The code uses conditional statements to check if certain conditions are met and output a pass or fail result.

- The code has some innovations, such as file HTML parsing, file handling, string manipulation, conditional statements, Excel workbook creation, worksheet manipulation, and workbook saving.

- The code also faces some major challenges, such as handling missing files, file format changes, large input data, errors, and Excel workbook size.

2.1.4 Script Functionality Breakdown

- **Initialization and Timer Setup**: The code initializes by importing the necessary modules and setting up a timer to monitor the total runtime of the script, which can be useful for performance tracking.

- **Function Definitions**: Two functions are defined to calculate averages from a list of values. The first function appears to be a generic average calculator, while the second is tailored to handle floating-point numbers.

- **Directory Cleanup**: The script uses os and shutil to navigate to a specified directory and remove subdirectories that are deemed unnecessary. This step is crucial for preparing the environment for data processing.

- **Data Parsing and Writing**: The script opens a text file to log the results of the data parsing. It then iterates over HTML files in specified directories,
using BeautifulSoup to parse the HTML content and extract relevant data based on certain criteria (e.g., checking for 'PASS' or 'FAIL' conditions).

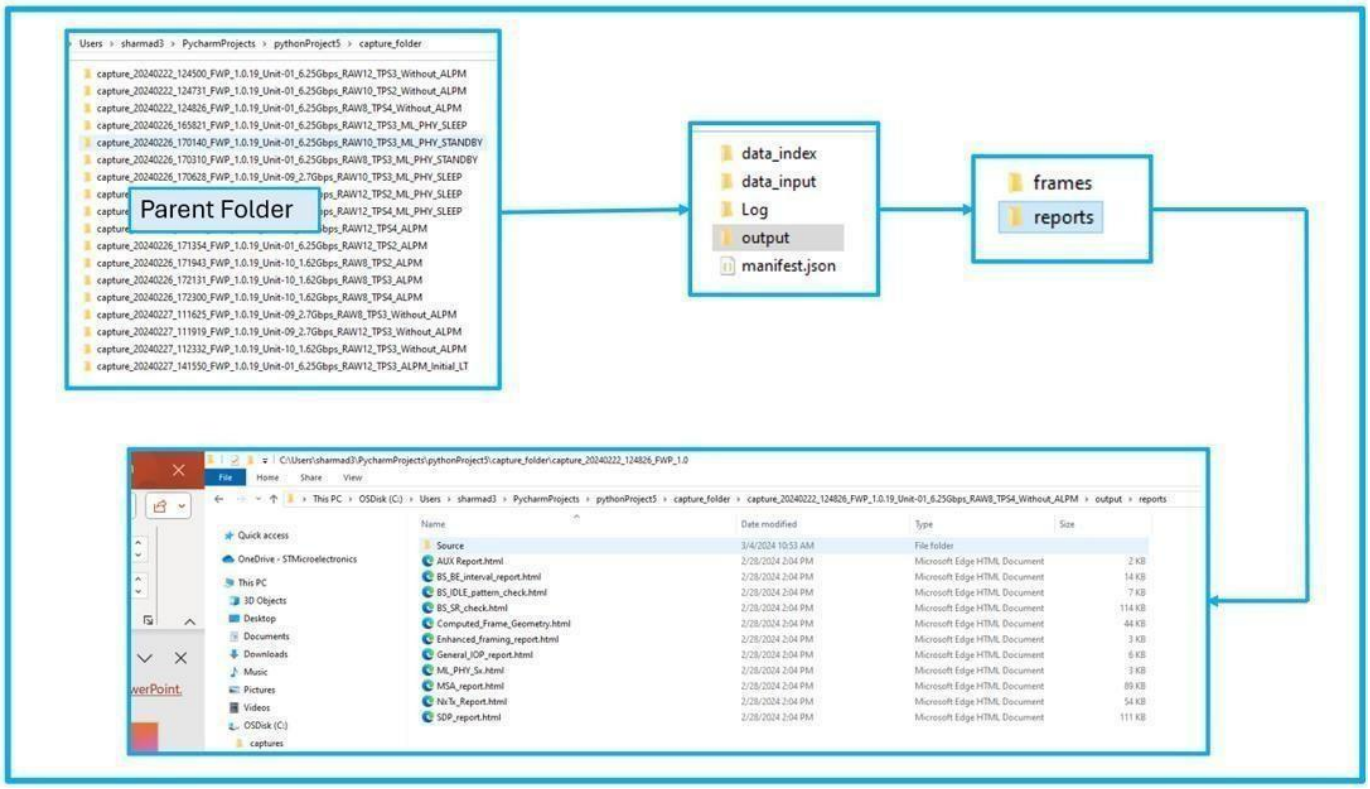
- **Data Extraction Logic**: The code contains several functions designed to extract specific pieces of information from the HTML files. These functions look for patterns like average values, frame geometry, and other parameters. The extracted data is then written to the log file.

- **Excel File Creation and Data Insertion**: The script uses openpyxl to create a new Excel workbook and populates it with headers and the parsed data. Each set of data is carefully placed in the appropriate columns of the workbook.

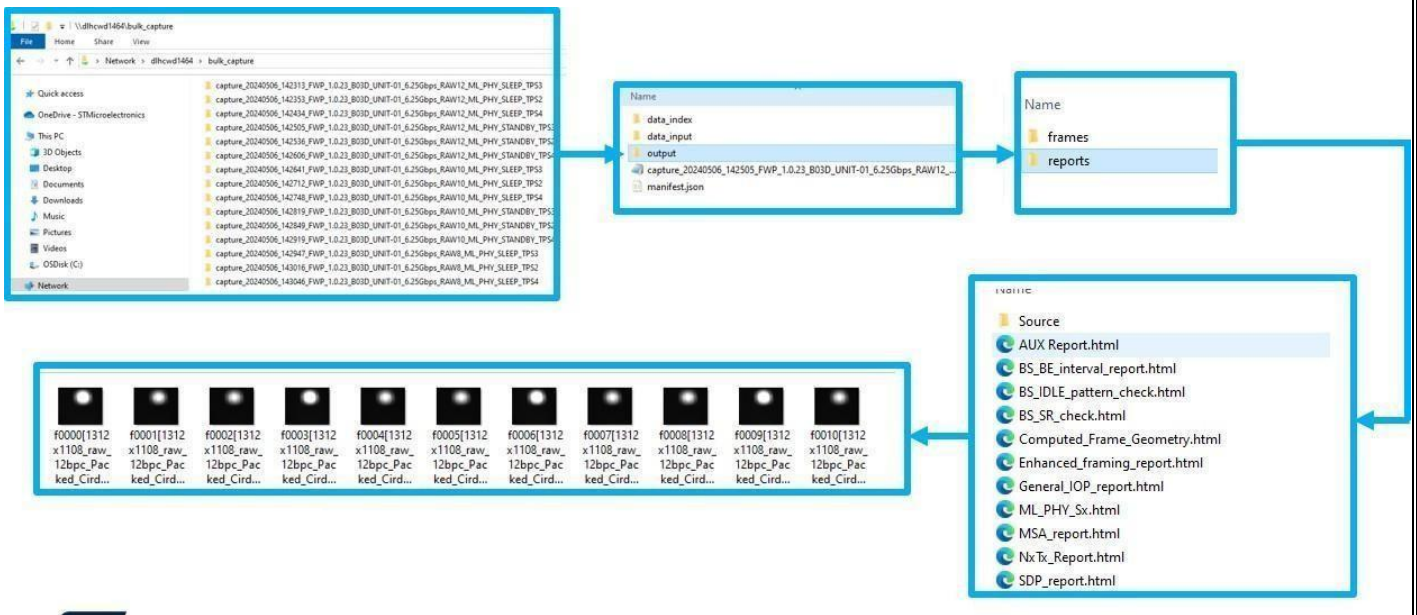
- **Excel Formatting**: The script applies various formatting options to the Excel file, such as adjusting column widths, enabling text wrapping, and setting font properties. This step is essential for enhancing the readability and presentation of the data.

- **Conditional Formatting Application**: The script applies conditional formatting to highlight specific cells in the Excel file based on their content. For example, cells containing an average value of zero are marked to draw attention to them.

2.1.5 Workflow



2.1.6 Workflow for Image Extraction



2.1.6 Script Design and Use Environment

1. **Technical or Analytical Setting**:
    - The script is tailored for environments that require regular data analysis and reporting, such as business intelligence, finance, or research departments.
    - Users in these settings often deal with large volumes of data that need to be processed and analyzed systematically.

2. **Purpose of HTML Report Parsing**:
    - HTML reports are commonly generated by various applications and systems for human-readable output.
    - The script automates the extraction of data from these HTML reports, bypassing the need for manual data entry or copy-pasting, which can be error-prone and time-consuming.

3. **Aggregation of Data**:
    - The script is capable of processing multiple HTML files across different directories, indicating its use in aggregating data from various reports into a single, centralized format.
    - This aggregation is crucial for creating comprehensive datasets that provide a holistic view of the information.

4. **Excel Integration for Review**:
    - After parsing and extracting data from HTML reports, the script integrates this data into Excel spreadsheets.
    - Excel is a widely-used tool for data analysis, and its spreadsheet format allows for easy manipulation, visualization, and review of data.

5. **Data Transformation and Clean-up**:
    - The script includes functions to calculate averages and check for consistency, which are part of data transformation and clean-up processes.
    - Clean and structured data is essential for accurate analysis and decision-making.

6. **Part of a Larger Workflow**:

    - While the script automates the parsing and integration tasks, it's likely a component of a larger data processing and analysis workflow.
    - It could be scheduled to run at regular intervals, feeding processed data into subsequent stages of analysis or reporting.

7. **Scalability and Customization**:

    - The script's ability to traverse directories and process files suggests that it is designed with scalability in mind, capable of handling an increasing number of reports or larger datasets.
    - Customization options, such as specifying particular files to parse, allow for flexibility in how the script is deployed in different scenarios.

8. **Potential for Further Enhancement**:

    - The script could be enhanced with additional features such as error handling, logging, and integration with machine learning models for more sophisticated data extraction.
    - Further automation could include notifications or triggers for human intervention when anomalies or critical insights are detected in the data.

2.1.7 Challenges and Opportunities

The code provided addresses some of the challenges mentioned in the literature. For example, it includes error handling for cases where data might not be present or in the expected format (e.g., checking if a list is empty before calculating averages). However, the code could be further enhanced by incorporating machine learning for more intelligent data extraction, especially in cases where HTML structures are inconsistent or more complex.

2.1.7 Input files

BS-BE interval is not less than 20 symbols: PASS

Frame	Lowest BS start to BE end, symbols	Line	PASS/FAIL
1	74	3	PASS
2	74	246	PASS
3	74	246	PASS
4	74	246	PASS

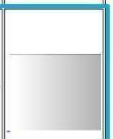
Every 512th BS is replaced with SR: PASS

SR #	SR position, bits	BS count after SR	PASS/FAIL
1	422330	511	PASS
2	34114590	511	PASS
3	45083110	511	PASS
4	56053570	511	PASS

BS-IDLE pattern check: PASS

BS-Idle #	Start position, bits	End position, bits	Symbols between BS, min	Symbols between BS, max	Symbols between SR, min	Symbols between SR, max	PASS/FAIL
1	422330	31126770	8192	8192	-	-	PASS

Computed Frames Geometry report

Index	Lines	Blank0 Lines	SF0 Width Symbols	SF0 Height Symbols	Blank1 Lines	SF1 Width Symbols	SF1 Height Lines	Blank2 Lines	SF2 Width Symbols	SF2 Height Lines	Image format	Image
1	1111	3	1312	1108							1312x1108, 8 bpc	
2	1354	246	1312	1108							1312x1108, 8 bpc	

BS-IDLE pattern check: PASS

BS-Idle #	Start position, bits	End position, bits	Symbols between BS, min	Symbols between BS, max	Symbols between SR, min	Symbols between SR, max	PASS/FAIL
1	422330	31126770	8192	8192	-	-	PASS

MSA report

Frame	Index	Line	Position, sym	Content
1	0	0	19	FIELD DEC HEX BITS --- HVid 32768 0x8000 HVid 32768 0x8000 HActive 1312 0x520 VActive 1108 0x454 HTotal 2144 0x860 VTotal 1394 0x572 HStart 820 0x334 VStart 285 0x11D VSynWidth 1 0x1 HSynWidth 4 0x4 MISC0 98 0x62 01100010 MISC1 128 0x80 10000000 MISC0.Synchronous Clock: Link clock and main video stream clock are asynchronous. MISC1.Interlaced Vertical Total Even: Number of lines per interlaced frame (consisting of two fields) is an odd number MISC1.Stereo Video Attribute: No 3D stereo video in-band signaling MISC.Colorimetry Format Value: 163 0xA3 10100011 MISC.Colorimetry Format: RAW, 8 bpc, 8bpc --- 00 80 00 08 40 08 72 00 04 00 80 00 03 34 01 1D 00 01 00 80 00 05 20 04 84 00 00 00 80 00 00 82 00 42 80 00

Nx/Tx values in VB and IFP

Frame	Line	Type (VB/IFP)	T12, ns	T3, AUX sym	T3a, AUX sym	T3b, AUX sym	T3', ns	T3elecs, ns	N1, lines	N2, sym	N3, sym	N4, lines	N5a, lines	N5b, lines	N6_IFP,lines	N7,lines
1	0	VB	-	-	-	-	-	-	0	0	0	0	0	8	0	0
2	0	VB	-	-	-	-	-	-	0	0	0	0	0	246	0	0

2.1.8 Output

	A	B	C	D	E	F	G
1	CAPTURE_NAME	BS_BE_interval_report	BS_IDLE_pattern_check	BS_SR_check	Enhanced_framing_report	General_IOP_report	Computed_Frame_Geometry
2	Capture_new\capture_20240506_142313_FWP_1.0.23_B03D_UNIT-01_6.25Gbps_RAW12_ML_PHY_SLEEP_TPS3	PASS	PASS	PASS	PASS	PASS	FAIL: Difference is of 1 lines
3	Capture_new\capture_20240506_142353_FWP_1.0.23_B03D_UNIT-01_6.25Gbps_RAW12_ML_PHY_SLEEP_TPS2	PASS	PASS	PASS	PASS	PASS	FAIL: Difference is of 2 lines
4	Capture_new\capture_20240506_142434_FWP_1.0.23_B03D_UNIT-01_6.25Gbps_RAW12_ML_PHY_SLEEP_TPS4	PASS	PASS	PASS	PASS	PASS	FAIL: Difference is of 2 lines

	A	I	J	K	L	M	N
1	CAPTURE_NAME	SDP_report	NxTx_Timings_Report	Computed_Frame_Geometry_Parameters	Frame_Geometry	bpc_value	Images
2	Capture_new\capture_20240506_142313_FWP_1.0.23_B03D_UNIT-01_6.25Gbps_RAW12_ML_PHY_SLEEP_TPS3	Average value of N1_Lines: 25	Average value of N2_Sym: 18020 Average value of T3eleos: 62009 Average value of N3_sym: 16031 Average value of N4_lines: 10 Average value of N5b_lines: 4 Average value of Blank lines: 4449	Average value of Vertical blank lines: 3341 Average value of Width Symbols: 1312 Average value of Height Symbols: 1108 Frame Geometry: 1312x1108	BPC Value: 12 bpc		
3	Capture_new\capture_20240506_142353_FWP_1.0.23_B03D_UNIT-01_6.25Gbps_RAW12_ML_PHY_SLEEP_TPS2	Average value of N1_Lines: 25	Average value of N2_Sym: 18020 Average value of T3eleos: 62010 Average value of N3_sym: 16032 Average value of N4_lines: 10 Average value of N5b_lines: 4 Average value of Blank lines: 4449	Average value of Vertical blank lines: 3341 Average value of Width Symbols: 1312 Average value of Height Symbols: 1108 Frame Geometry: 1312x1108	BPC Value: 12 bpc		
4	Capture_new\capture_20240506_142434_FWP_1.0.23_B03D_UNIT-01_6.25Gbps_RAW12_ML_PHY_SLEEP_TPS4	Average value of N1_Lines: 25	Average value of N2_Sym: 18021 Average value of T3eleos: 62009 Average value of N3_sym: 16031 Average value of N4_lines: 10 Average value of N5b_lines: 4 Average value of Blank lines: 4449	Average value of Vertical blank lines: 3341 Average value of Width Symbols: 1312 Average value of Height Symbols: 1108 Frame Geometry: 1312x1108	BPC Value: 12 bpc		

2.1.9 Conclusion

The code provided aligns with the current state of technology outlined in the literature survey. It encapsulates the core functionalities required for an Automated HTML Report Parsing and Excel Integration System, such as parsing HTML content, extracting and transforming data, and integrating with Excel for further analysis. While the code demonstrates a foundational approach, there is potential for incorporating more advanced features and technologies to address the challenges and opportunities identified in the literature.

CHAPTER 3- Software and Tools Used:

3.1. PyCharm-

PyCharm is an integrated development environment (IDE) used in computer programming, specifically for the Python language. It is developed by the Czech company JetBrains. It provides code analysis, a graphical debugger, an integrated unit tester. PyCharm is cross platform with Windows, macOS and Linux versions. The Community Edition is released under the Apache License, and there is also Professional Edition with extra features – released under a proprietary license. In this project, I have used PyCharm Version 2022.3.1.

PyCharm features includes:

- Code editing and analysis: PyCharm provides help in the code completion, syntax highlighting, and error highlighting, and hence making it easier to write clean and correct the Python code.
- Debugging: PyCharm includes a powerful debugger which allows the developers to step through the code and identify and fix various issues quickly.

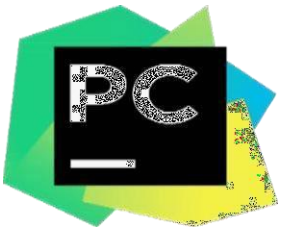


Figure 8 Logo for PyCharm

3.2. Python-

Python is the high-level, and a general-purpose programming language. Its design emphasis on the code readability. It is dynamically typed, and it supports the various programming paradigms, including structured, object-oriented and functional programming. Python codes are very simple and are easy to understand and are not too complex and have the good readability.

Python is the interpreter language which means that the source code of python program is converted into the bytecodes that is executed by the Python Virtual Machine.

And in python code runs line by line rather than running all together. Python is the Programming language with many different IDEs available. Some of which are PyCharm, and Visual Studio Code. And each IDE has the unique features and benefits.

Figure 9 Python Logo

3.3. Hex Neo Editor-

Hex Editor Neo, is a specialized tool for editing binary files, including executable programs, firmware, and other types of data. It allows users to view and edit the contents of binary files at the hexadecimal level, providing detailed insights into the structure and content of the data In the context of imaging applications, Hex Editor Neo can be used for analyzing and manipulating image data in various formats, including JPEG, PNG, and BMP.

It can also be used for analyzing and modifying the data in raw image files, which are often used in scientific imaging applications.

Figure 10 Hex Neo Editor Logo

3.4. Modules and different shortcuts used for code writing-

- **CSV** - The so-called CSV (Comma Separated Values) format is the most common import and export format for spreadsheets and databases. CSV format was used for many years prior to attempts to describe the format in a standardized way.

The csv module implements classes to read and write tabular data in CSV format. It allows programmers to say, "write this data in the format preferred by Excel," or "read data from this file which was generated by Excel," without knowing the precise details of the CSV format used by Excel.

Here is how I used it to read and write in a CSV file :-

```
#reading data from csv file

import csv
with open (filename, 'r') as f: #
file read reader = csv. reader(f)
for row in reader:
        print(row)


#writing data to csv file
new_file = open ('output.txt', 'w')
new_file.write('output:\n')
new_file.write(out)
```

- **OPEN** – It is used to open internally stored files. It returns the content of the file as python objects.

E.g.: **open ("file_name", mode)**

Mode – This parameter is a string that is used to specify the mode in which the file is to be opened. The following can be used to activate a specific mode:

1. **"r"** – This string is used to read only the file.
2. **"w"** – This is used for writing on/over the file.
3. **"a"** – It is used to append content to the existing files.
4. **"b"** – When user wants to handle the file in the binary mode.

CHAPTER 4- Work and Output

4.1. IDLE PATTERN CHECKER

PROBLEM STATEMENT 1:

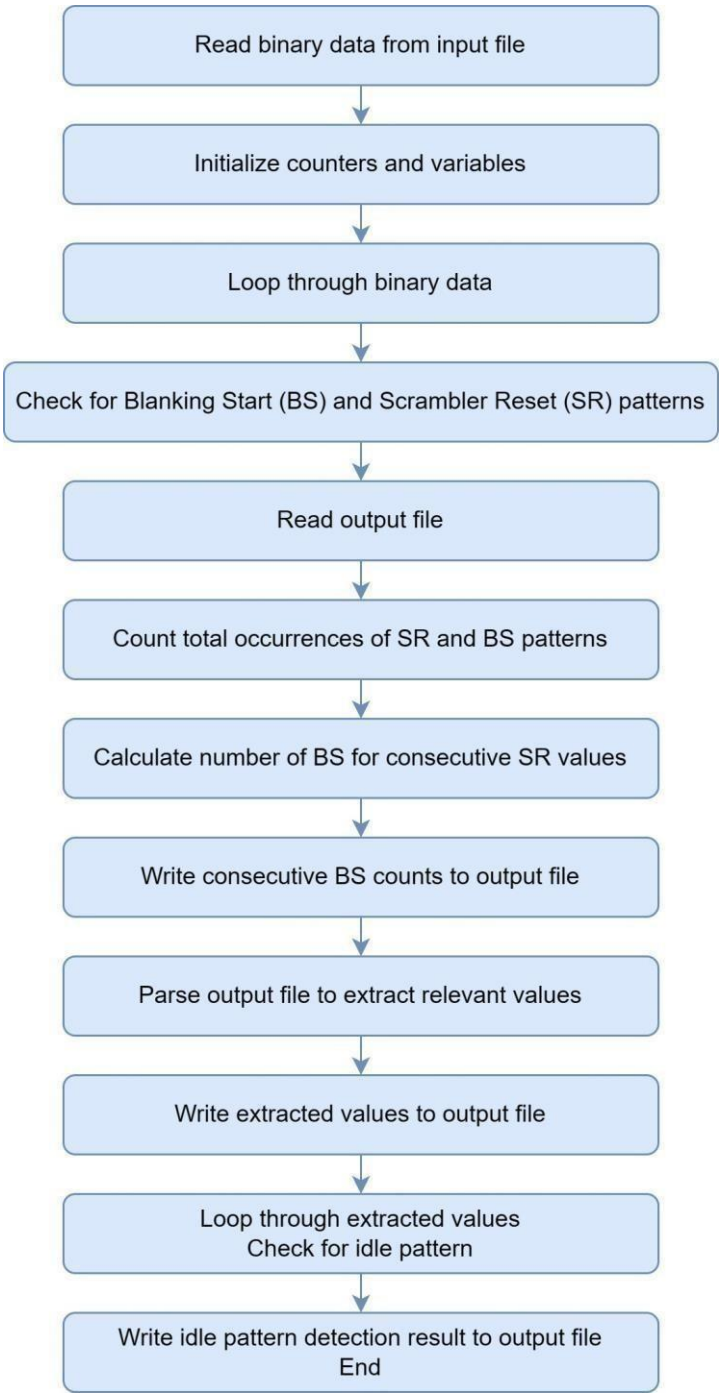
To analyze a binary data file and detect the presence of an idle pattern. The script should output the number of occurrences of the idle pattern and its location in the data file.

The mapping of these control link symbols to 8b/10b special characters is described in Section 3. When a DPTX is transmitting the Idle Pattern, the DPTX shall insert the 4-symbol sequence of BS (or SR) every 213 or 8192 symbols. In other words, there shall be 8188 symbols between the last (fourth) symbol of the 4-symbol sequence for BS (or SR) and the first symbol of the next 4-symbol sequence.

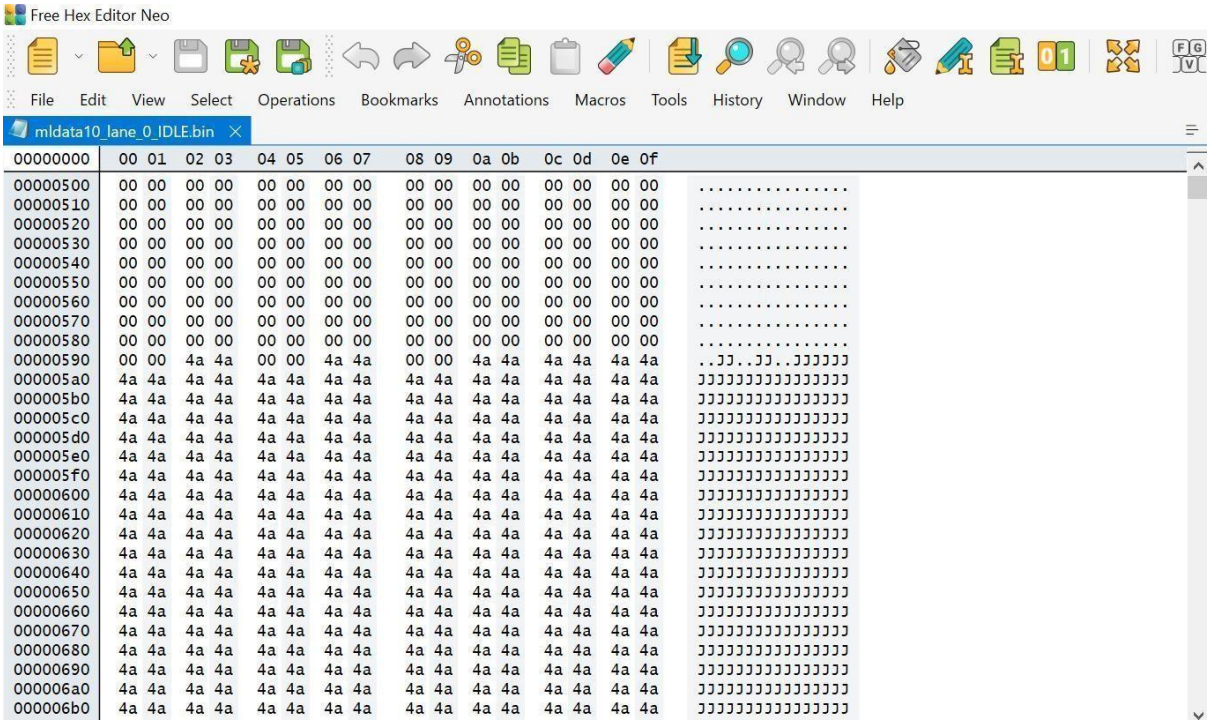
Every 512th BS symbol sequence shall be replaced with an SR symbol sequence. The last symbol of the 4-symbol sequence for SR shall be used to reset the scrambler. When switching between the Idle Pattern transmission and a stream transmission, the Source device shall avoid any overlap of the four symbols for BS and SR

- Brushing up Python Syntax and Programming concepts.
- The binary data file is read by the code.
- Specific byte patterns, including scrambler reset and blanking start, are searched for in the data.
- The number of occurrences of each pattern and the number of blanking starts between consecutive scrambler resets are produced as output.
- A function within the script extracts specific values from the output file and writes them to a new file.
- The code checks for an idle pattern by comparing the difference between consecutive values in the output file to a specific value.

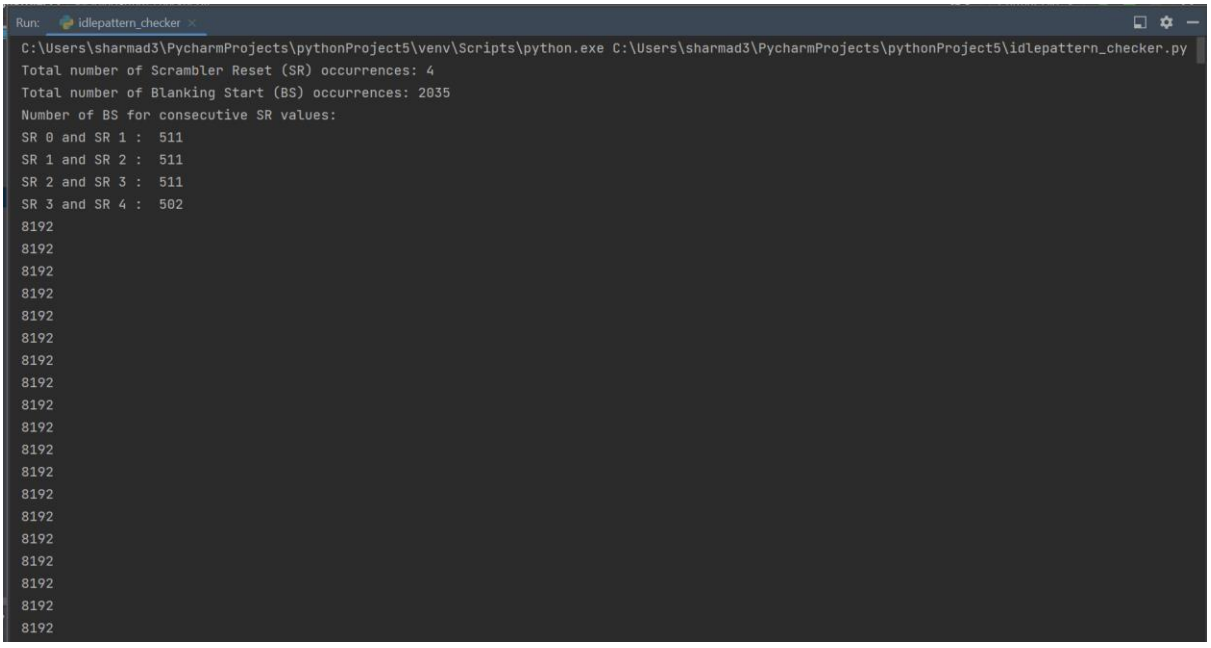
4.1.1. FLOWCHART



4.1.2. A) Input File



B) Output of the python script



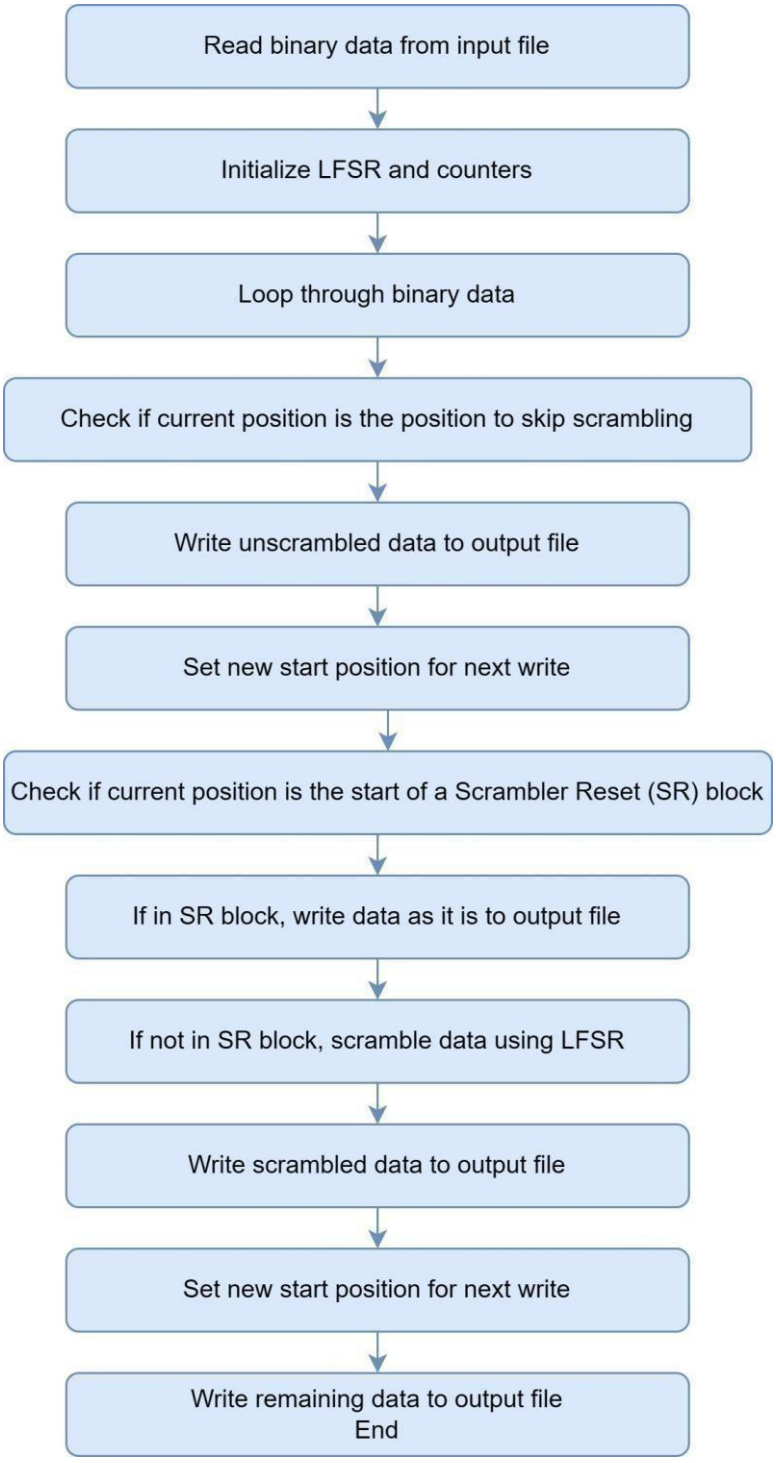
4.2. SCRAMBLER/DESCRAMBLER

PROBLEM STATEMENT 2:

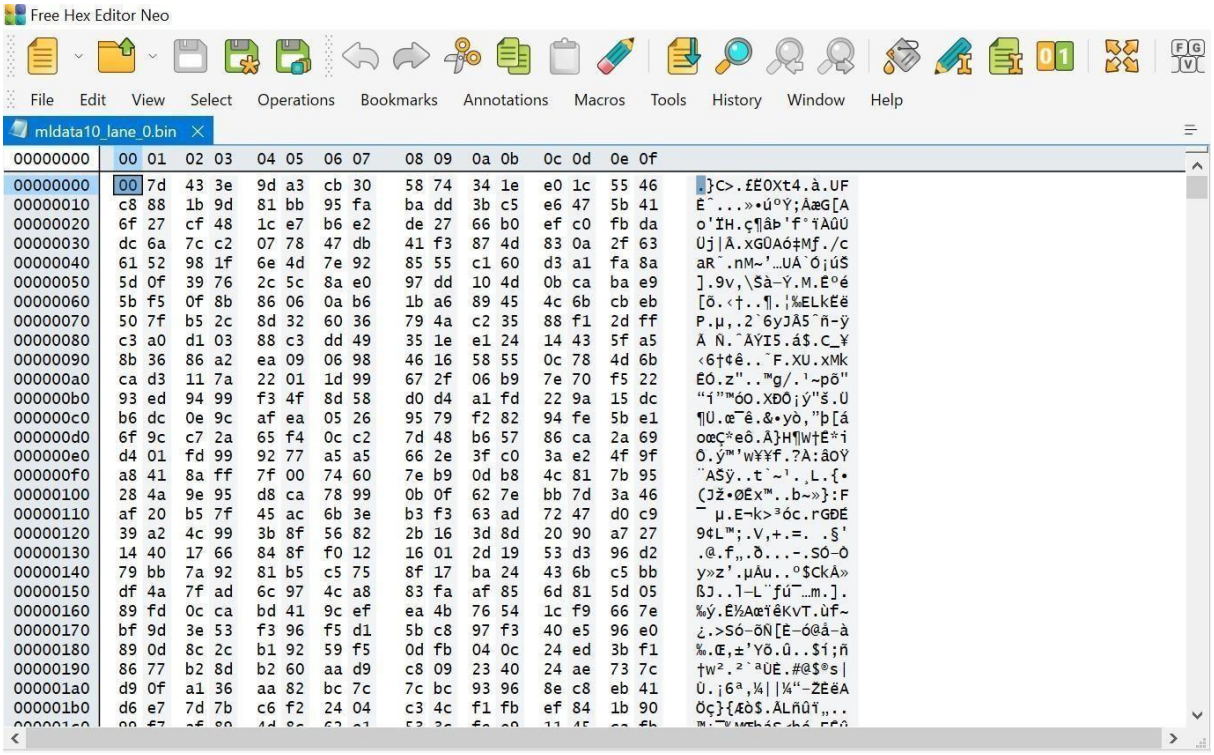
To scramble a binary data file using an LFSR and skip scrambling the data at a specific position in the file. The script should output the scrambled data to a new file.

- Read binary data from input file
- Search for SR block patterns
- Writes data to output file as is if SR block is found
- Scrambles data using LFSR if SR block is not found
- Updates LFSR value based on current state
- Skips scrambling data at specific position in input file
- Defines input and output file names at start of script

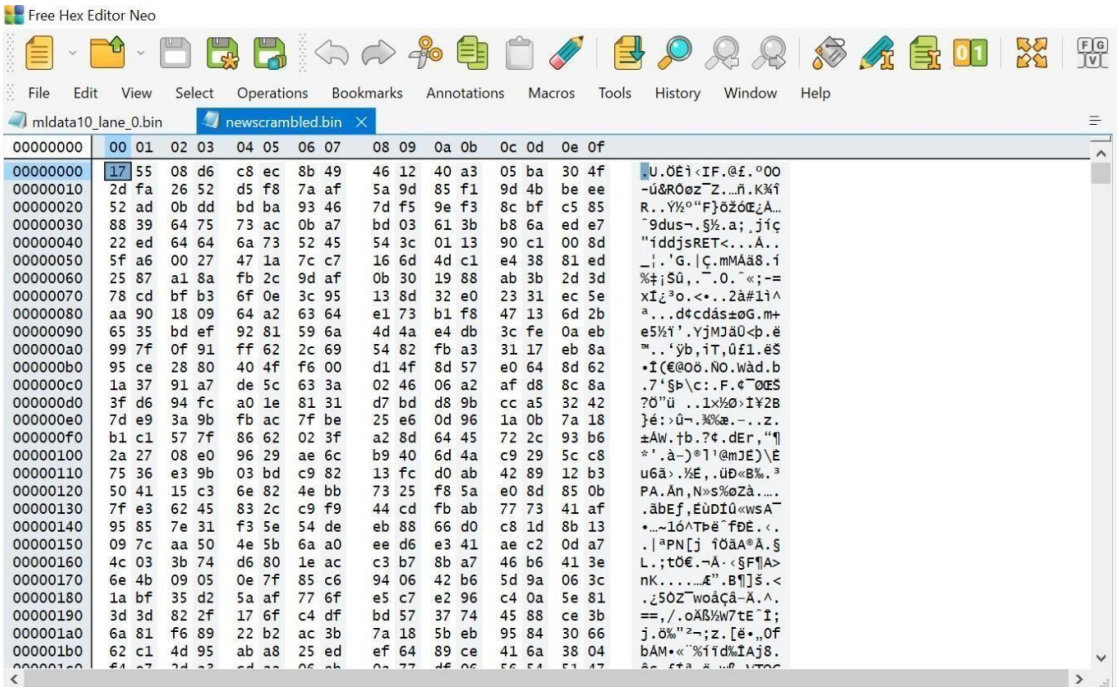
4.2.1 FLOWCHART



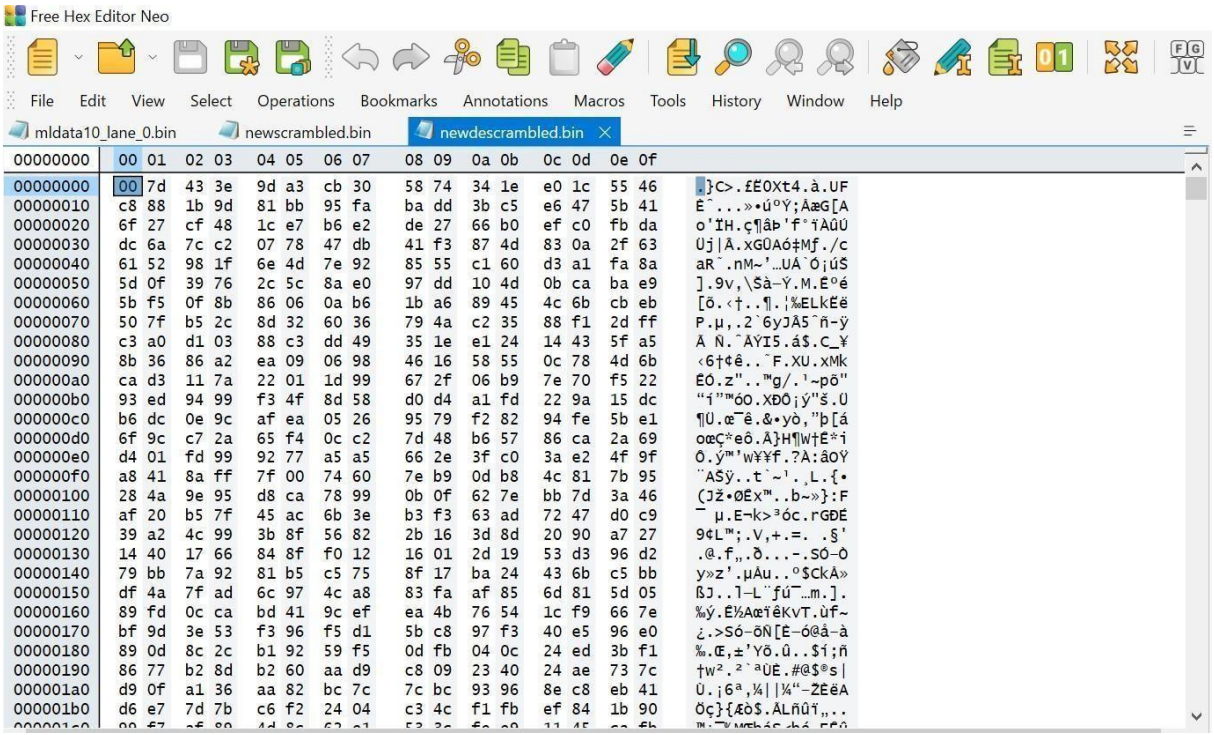
4.2.2 Input File



4.2.3 Scrambled file



4.2.4 Descrambled file



- *If a scrambled file is scrambled again using the same algorithm, the resulting file will be indistinguishable from the original scrambled file*
- *This is because the scrambling algorithm is deterministic and produces the same output for a given input every time*
- *However, if the file is scrambled again using a different algorithm, the resulting file will be different from the original scrambled file*
- *To recover the original file from the doubly-scrambled file, the descrambler must be applied twice: first to undo the second layer of scrambling, and then to undo the first layer of scrambling*
- *Scrambling and descrambling algorithms must be kept secret to maintain the security of the scrambled data*
- *If an adversary gains access to the scrambling algorithm, they can easily descramble the data and recover the original signal*

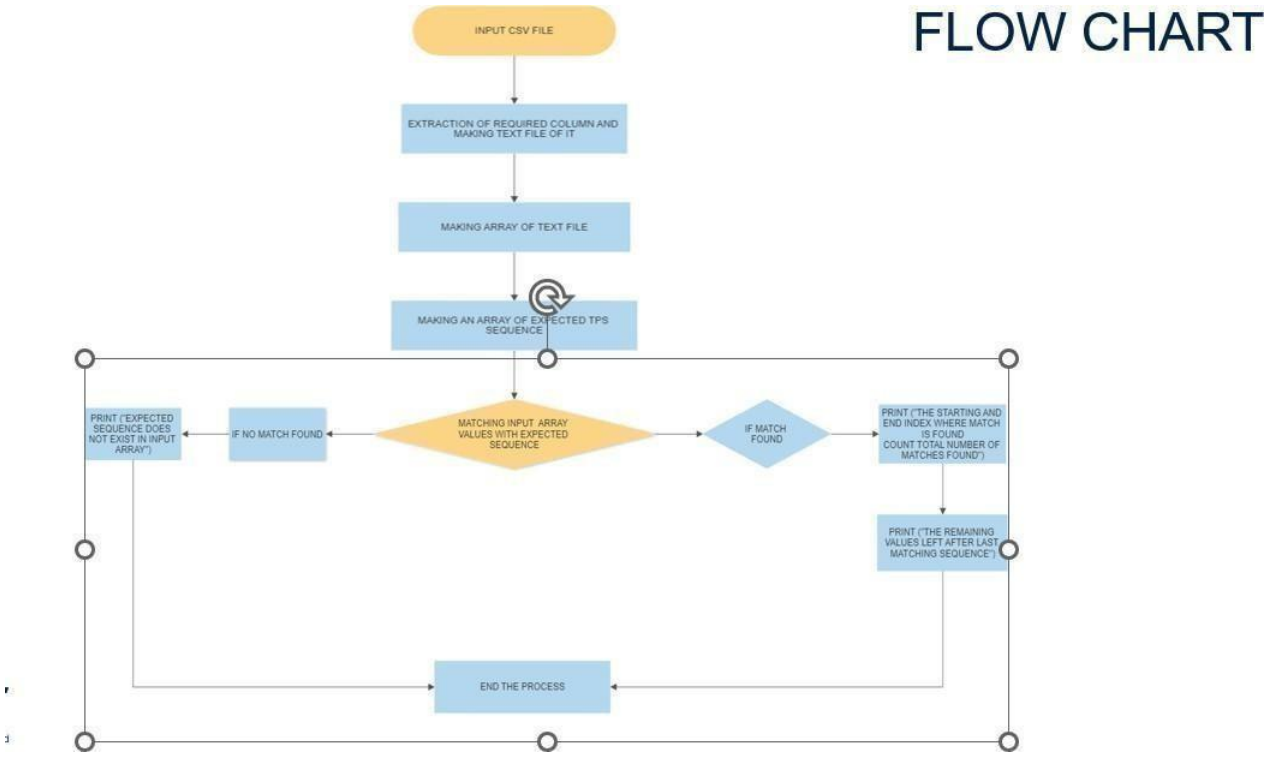
4.3. DETECTION OF DIFFERENT TRAINING PATTERN SEQUENCE

PROBLEM STATEMENT 3:

To extract a specific column from a CSV file and search for an expected sequence of values in the extracted column,then counts the number of times the expected sequence occurs in the input array and prints the indices where it is found. Additionally, prints the number of values left in the input array after the last matching sequence.

- Read CSV file and extract desired column
- Create text file with extracted data
- Read text file and create input array
- Read separate text file with expected sequence of different TPS and make expected sequence array.
- Search for expected sequence in array in input array
- Count number of occurrences of matched sequence in input array.
- Find indices of matched sequences and count of total number of occurrences
- Find number of values left in array or message if not found.

4.3.2 FLOW CHART



4.3.3 INPUT DATA

TPS2

22227	0x017C	0xBC	K28.5-
22228	0x018B	0xCB	D11.6-
22229	0x0283	0xBC	K28.5+
22230	0x018B	0xCB	D11.6-
22231	0x02AA	0x4A	D10.2-
22232	0x02AA	0x4A	D10.2-
22233	0x02AA	0x4A	D10.2-
22234	0x02AA	0x4A	D10.2-
22235	0x02AA	0x4A	D10.2-
22236	0x02AA	0x4A	D10.2-
22237	0x017C	0xBC	K28.5-
22238	0x018B	0xCB	D11.6-
22239	0x0283	0xBC	K28.5+
22240	0x018B	0xCB	D11.6-
22241	0x02AA	0x4A	D10.2-
22242	0x02AA	0x4A	D10.2-
22243	0x02AA	0x4A	D10.2-
22244	0x02AA	0x4A	D10.2-
22245	0x02AA	0x4A	D10.2-
22246	0x02AA	0x4A	D10.2-

TPS3

21915	0x017C	0xBC	K28.5-
21916	0x0283	0xBC	K28.5+
21917	0x017C	0xBC	K28.5-
21918	0x0283	0xBC	K28.5+
21919	0x02AA	0x4A	D10.2-
21920	0x02AA	0x4A	D10.2-
21921	0x02AA	0x4A	D10.2-
21922	0x02AA	0x4A	D10.2-
21923	0x02AA	0x4A	D10.2-
21924	0x02AA	0x4A	D10.2-
21925	0x02AA	0x4A	D10.2-
21926	0x02AA	0x4A	D10.2-
21927	0x017C	0xBC	K28.5-
21928	0x0283	0xBC	K28.5+
21929	0x031E	0x7E	D30.3-
21930	0x00E1	0x7E	D30.3+
21931	0x031E	0x7E	D30.3-
21932	0x00E1	0x7E	D30.3+
21933	0x031E	0x7E	D30.3-
21934	0x00E1	0x7E	D30.3+
21935	0x031E	0x7E	D30.3-
21936	0x00E1	0x7E	D30.3+
21937	0x031E	0x7E	D30.3-
21938	0x00E1	0x7E	D30.3+
21939	0x031E	0x7E	D30.3-
21940	0x00E1	0x7E	D30.3+
21941	0x031E	0x7E	D30.3-
21942	0x00E1	0x7E	D30.3+

TPS4

22816	0x00BC	0x1C	K28.0-	K28.0-
22817	0x017C	0xBC	K28.5-	K28.5-
22818	0x0283	0xBC	K28.5+	K28.5+
22819	0x00BC	0x1C	K28.0-	K28.0-
22820	0x0235	0xFF	D31.7-	0x00
22821	0x0097	0x17	D23.0-	0x00
22822	0x01B9	0xC0	D0.6-	0x00
22823	0x00B4	0x14	D20.0+	0x00
22824	0x0172	0xB2	D18.5-	0x00
22825	0x01C7	0xE7	D7.7-	0x00
22826	0x0352	0x02	D2.0+	0x00
22827	0x02D2	0x82	D2.4+	0x00
22828	0x0332	0x72	D18.3+	0x00
22829	0x030E	0x6E	D14.3+	0x00
22830	0x0258	0x28	D8.1+	0x00
22831	0x0166	0xA6	D6.5-	0x00
22832	0x015E	0xBE	D30.5-	0x00
22833	0x030D	0x6D	D13.3+	0x00
22834	0x014A	0xBF	D31.5+	0x00
22835	0x02CD	0x8D	D13.4-	0x00
22836	0x0161	0xBE	D30.5+	0x00
22837	0x02B9	0x40	D0.2-	0x00
22838	0x0178	0xA7	D7.5+	0x00
22839	0x0226	0xE6	D6.7+	0x00
22840	0x026C	0x2C	D12.1-	0x00
22841	0x0193	0xD3	D19.6-	0x00
22842	0x022D	0xE2	D2.7-	0x00
22843	0x0172	0xB2	D18.5-	0x00
22844	0x0347	0x07	D7.0-	0x00
22845	0x0352	0x02	D2.0+	0x00
22846	0x00E8	0x77	D23.3+	0x00
22847	0x026A	0x2A	D10.1-	0x00
22848	0x018D	0xCD	D13.6-	0x00
22849	0x0274	0x34	D20.1-	0x00
22850	0x015E	0xBE	D30.5-	0x00
22851	0x01C6	0xE0	D0.7+	0x00
22852	0x0178	0xA7	D7.5+	0x00
22853	0x02A2	0x5D	D29.2+	0x00
22854	0x026B	0x24	D4.1-	0x00
22855	0x0171	0xB1	D17.5-	0x00
22856	0x02E4	0x9B	D27.4+	0x00
22857	0x0151	0xA4	D1.5+	0x00

4.3.4 OUTPUT DATA

Output of different TPS sequence are:

TPS2

```
Matched sequence found at indices 49046 to 49055
Matched sequence found at indices 49056 to 49065
Matched sequence found at indices 49066 to 49075
The expected sequence occurred 2701 times in the input array.
There are 3134634 values left in the input array after the last matching sequence.

Process finished with exit code 0
```

TPS4

```
Matched sequence found at indices 48771 to 49022
Matched sequence found at indices 49023 to 49274
Matched sequence found at indices 49275 to 49526
The expected sequence occurred 106 times in the input array.
There are 999048 values left in the input array after the last matching sequence.

Process finished with exit code 0
```

TPS3

```
Matched sequence found at indices 48807 to 48840
Matched sequence found at indices 48841 to 48874
Matched sequence found at indices 48875 to 48908
The expected sequence occurred 794 times in the input array.
There are 16728307 values left in the input array after the last matching sequence.

Process finished with exit code 0
```

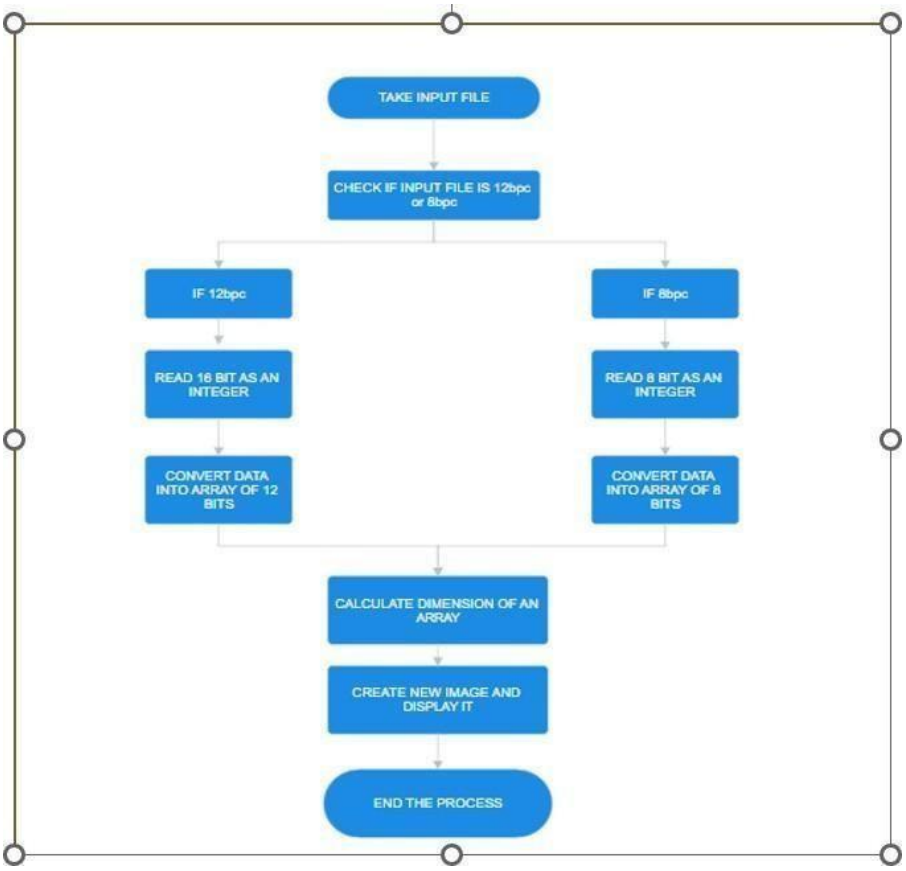
4.4. RAW FRAME DECODER

PROBLEM STATEMENT 4:

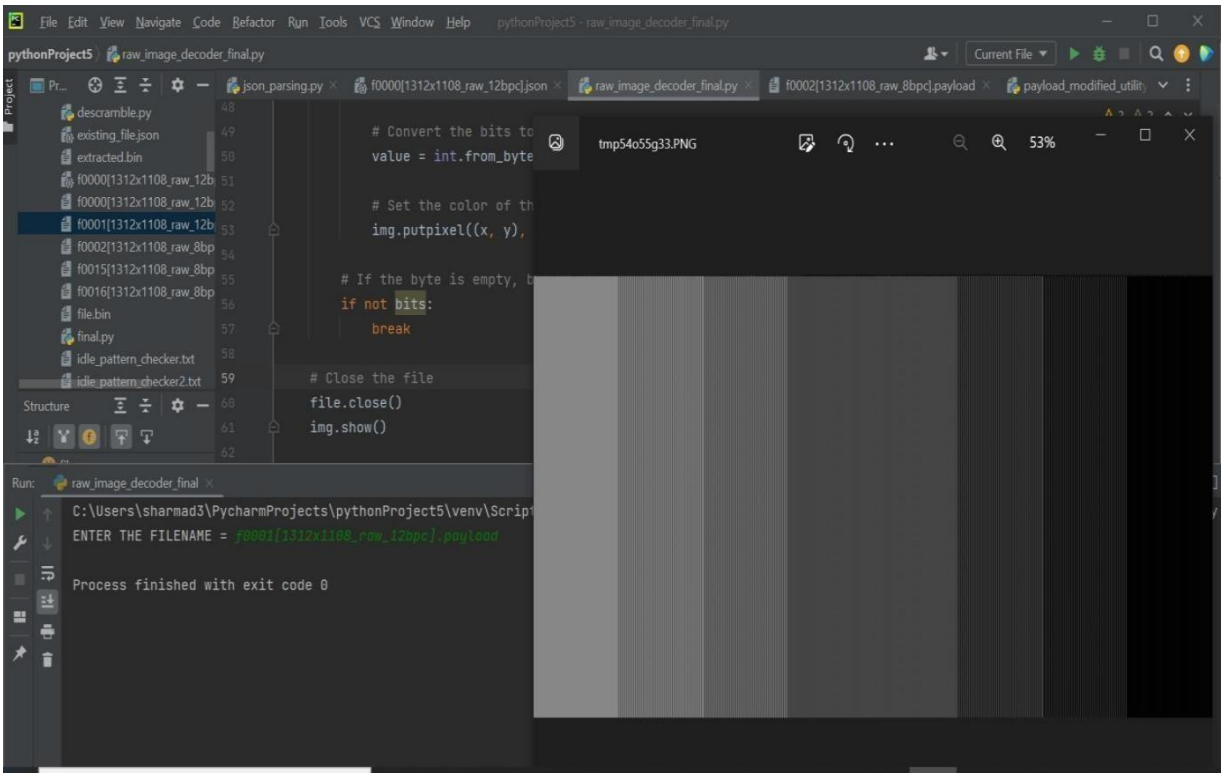
To convert a binary file containing pixel data to Visual Output • Given files were for RAW 8, 10, 12 image binary data along with their bmp. • Convert it into an image.

- Take input filename from user.
- Check if file is 12bpc or 8bpc payload file.
- Read data from file and converts it into an array of pixels.
- Calculate image dimensions for 12bpc file and creates a new image for 8bpc file.
- Set color of pixels and displays the image.

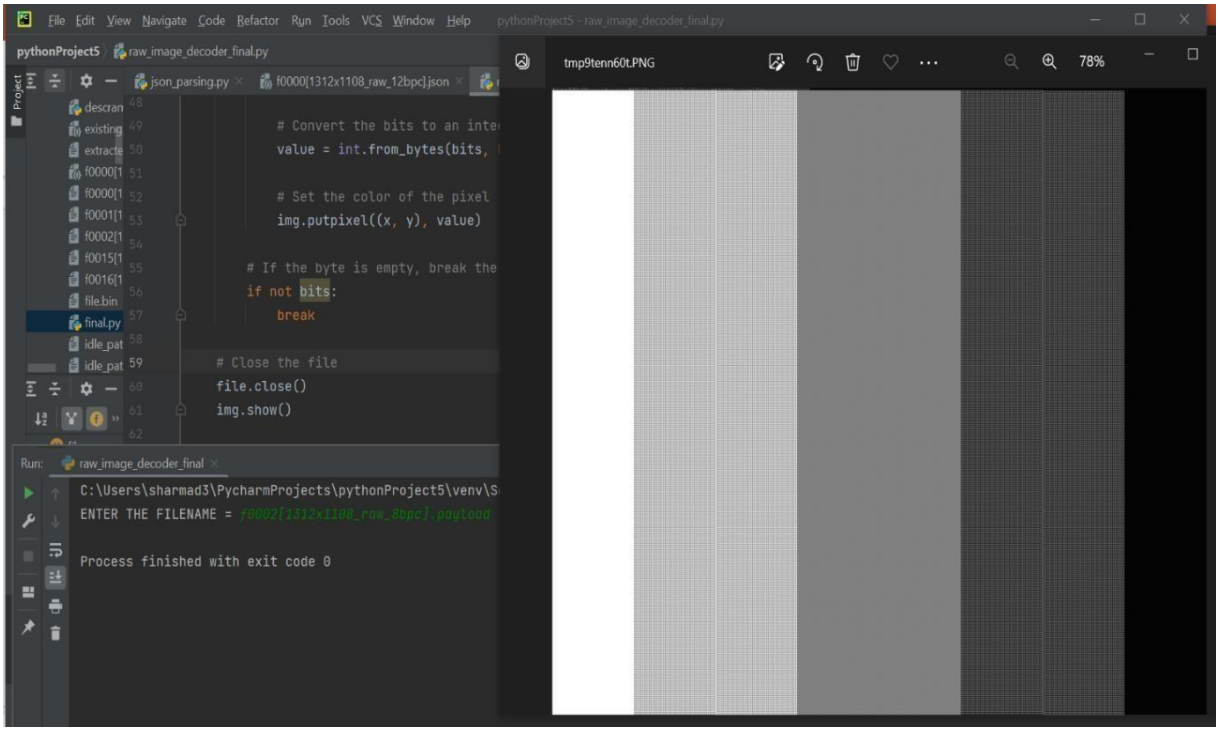
4.4.1 FLOW CHART



4.4.2 OUTPUT IMAGES



12_bpc_payloadfile



8_bpc_payloadfile

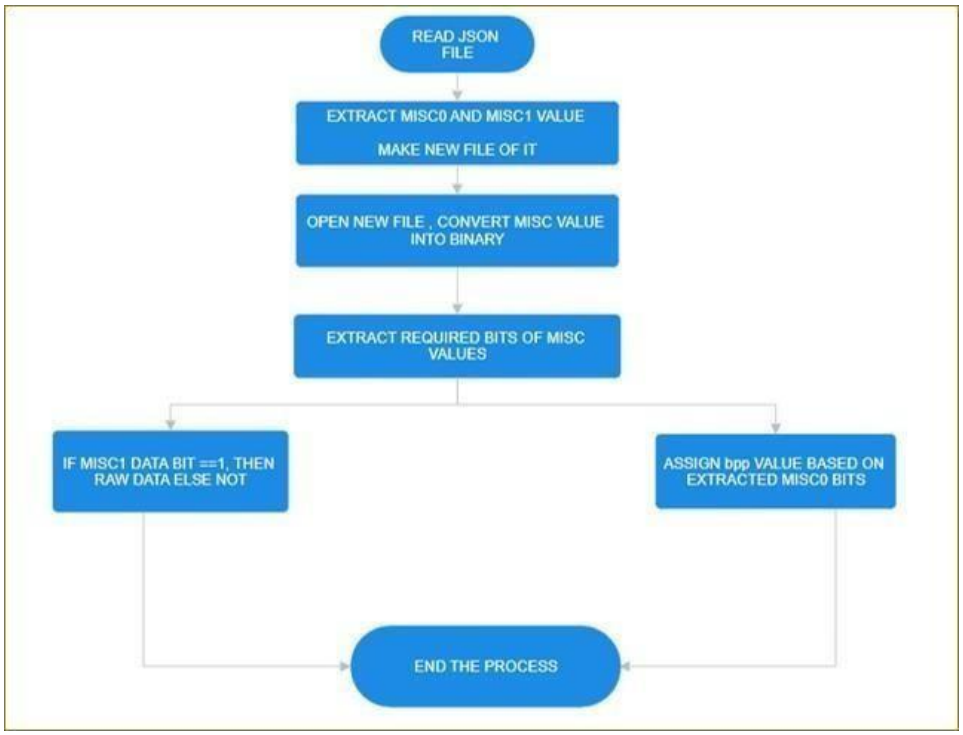
4.5. DATA PARSING OF JSON FILE

PROBLEM STATEMENT 5:

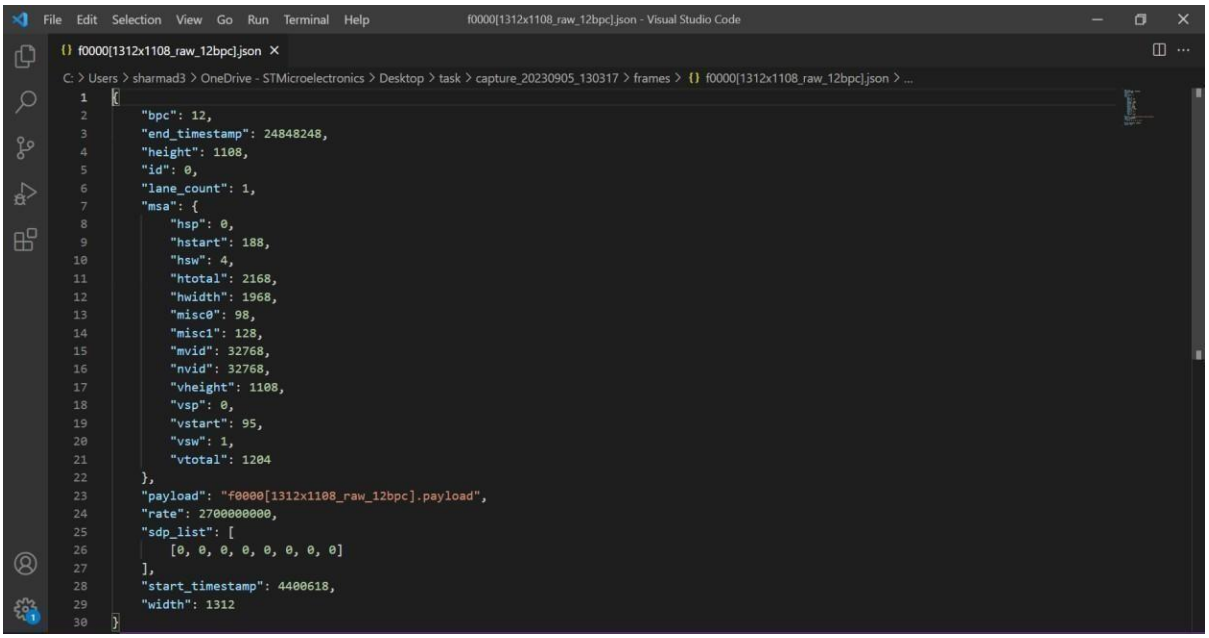
To extract specific values from a JSON file and perform binary operations on them to determine the bit depth of the data and whether it is raw or not. The code extracts the 'misc0' and 'misc1' values from the JSON file and creates a new dictionary with these values. It then writes the new dictionary to a file and reads it back in to perform binary operations on the extracted values. The code extracts specific bits from 'misc0' and 'misc1' and creates separate arrays for these bits. It then prints the extracted bits and determines the bit depth of the data and whether it is raw or not based on the extracted bits. The purpose of this code is to analyze a JSON file containing metadata about a dataset and extract important information from it.

- Load JSON data from a file and extracts misc0 and misc1 value from it.
- Create a new dictionary with the extracted values and write it to a new file.
- Open new file and convert misc0 and misc1 values into binary.
- Extract 7th bit of misc1 and 5th,6th ,7th bit of misc0.
- If 7th bit of misc0 is 1 then raw data ,otherwise not.

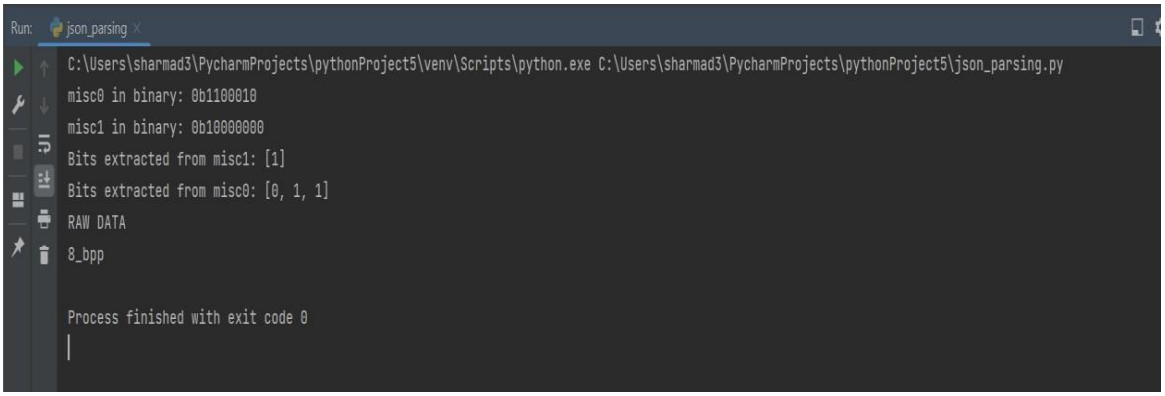
4.5.1 FLOW CHART



4.5.2 INPUT FILE



4.5.3 OUTPUT

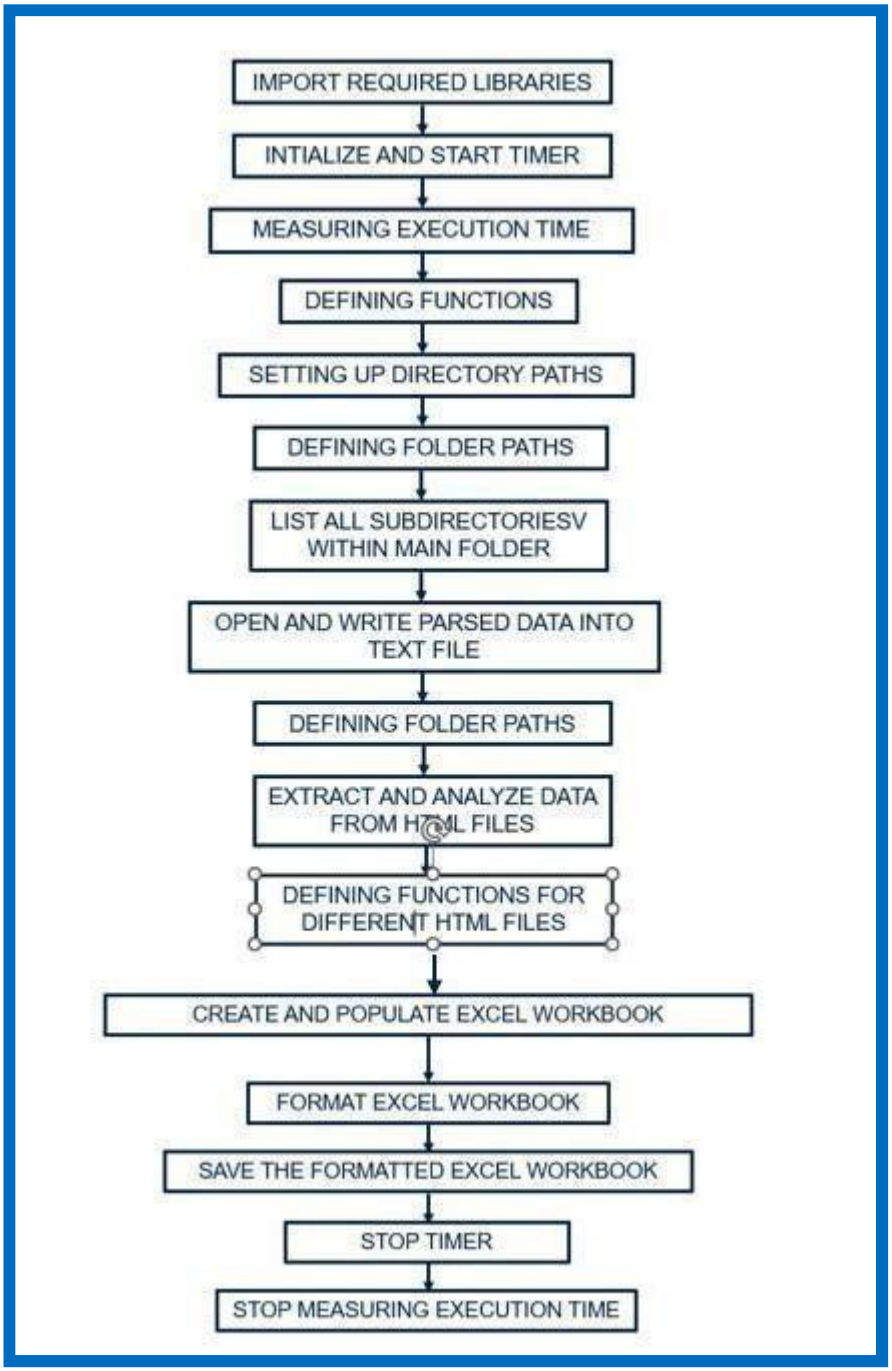

Run:
json_parsing
C:\Users\sharmad3\PycharmProjects\pythonProject5\venv\Scripts\python.exe C:\Users\sharmad3\PycharmProjects\pythonProject5\json_parsing.py
misc0 in binary: 0b1100010
misc1 in binary: 0b10000000
Bits extracted from misc1: [1]
Bits extracted from misc0: [0, 1, 1]
RAW DATA
8_bpp
Process finished with exit code 0

4.6. HTML TO EXCEL SHEET CONVERSION

PROBLEM STATEMENT 6: To extract specific information from HTML files in a directory and write the output to a text file. The extracted information includes PASS/FAIL status for specific reports, computed frame geometry, ML_PHY_Sx report, and SDP report. The code also needs to combine the extracted information for each subdirectory into an Excel file with separate sheets for each subdirectory. The Excel file should have a header row with the names of the reports and a row for each subdirectory with the corresponding PASS/FAIL status. Finally, the code needs to format the Excel file by adjusting the dimensions of cells, increasing the font size of the header row, and changing the color of PASS/FAIL text

- Begin by setting up a timer to measure the code's execution time.
- Implement the necessary helper functions, such as calculate_average for computing averages.
- Specify the main folder path and list all its subdirectories for processing.
- Write parsed data to PARSED_DATA_DUMPS.txt and process each subdirectory by selecting specific files, analyzing HTML files, and writing results.
- **Helper Functions:**
    - Check Reports: Analyze report files and write 'PASS' or 'FAIL'.
    - Check Frame Geometry: Validate frame geometry consistency.
    - Calculate ML_PHY_Sx Average: Compute average values from the report.
    - Validate SDP Report: Ensure 'Header checksum OK' is present in the SDP report.
    - NxTx Timings Report: Calculate averages and validate data.
    - Computed Frame Geometry Parameters: Calculate average values for frame geometry parameters.
    - Frame Dimensions: Extract and validate frame dimension data.
    - BPC Value: Extract and validate BPC values.
- Transfer data from the output text file to an Excel workbook.
- Adjust workbook formatting, including column dimensions, text wrapping, header font size, and conditional formatting.
- Save the formatted Excel workbook.
- End the timer and calculate the total running time.

4.6.1 FLOW CHART



4.6.1 Parsed Data

```
Capture_new\capture_20240605_172752_FWP_1.0.24_B03D_UNIT-01_6.25Gbps_RAW12_Non_ALPM_Aux
PASS
PASS
PASS
PASS
PASS
PASS
No data to calculate average.
PASS
Average value of N1_Lines: 0
Average value of N2_Sym: 0
Average value of T3eieos: NIL
Average value of N3_sym: 0
Average value of N4_lines: 0
Average value of N5b_lines: 0
Average value of Blank lines: NIL
Average value of Vertical blank lines: NIL
Average value of Width Symbols: NIL
Average value of Height Symbols: NIL
Frame Geometry: No data found
BPC Value: No data found

Capture_new\capture_20240605_180936_FWP_1.0.24_B03D_UNIT-09_2.7Gbps_RAW8_Non_ALPM_TPS4
PASS
PASS
PASS
PASS
PASS
PASS
No data to calculate average.
PASS
Average value of N1_Lines: 0
Average value of N2_Sym: 0
Average value of T3eieos: NIL
Average value of N3_sym: 0
Average value of N4_lines: 0
Average value of N5b_lines: 957
Average value of Blank lines: 2026
Average value of Vertical blank lines: 918
Average value of Width Symbols: 1312
```

CHAPTER – 5 CONCLUSION AND FUTURE SCOPE

5.1. Conclusion:

Alignment with Current Technology

- The script uses **BeautifulSoup** for HTML parsing, which is a modern and popular library in Python for this purpose. This aligns with the literature's mention of various HTML parsing techniques and tools.

- **Openpyxl** is employed for Excel integration, which is a current and widely-used library for working with Excel files in Python. This reflects the literature's discussion on methods for integrating data into Excel.

5.2. Core Functionalities

- **Parsing HTML Content**: The script reads HTML files and uses BeautifulSoup to parse the content, which is a fundamental requirement for any system that aims to automate data extraction from HTML reports.

- **Extracting Data**: Specific data points are targeted and extracted from the parsed HTML, which is necessary to convert unstructured data into a structured form.

- **Transforming Data**: The script performs basic data transformations, such as calculating averages and checking for consistency, which is crucial for preparing data for analysis.

- **Integrating with Excel**: The extracted and transformed data is then loaded into an Excel workbook, allowing for further manipulation and analysis using Excel's robust set of features.

5.3. Foundational Approach

- The script provides a **basic structure** for automating the tasks of parsing, extracting, transforming, and integrating data, which are the foundational steps in data processing.

- It demonstrates a **sequential and systematic approach** to handling multiple files and directories, which is essential for scalability and managing large datasets.

5.4. Potential Enhancements

- **Advanced Data Extraction**: Incorporating machine learning or more sophisticated pattern Recognition could improve the system's ability to handle varied and complex HTML structures.

- **Error Handling and Logging**: Adding comprehensive error handling and logging mechanisms woul the system more robust and user-friendly, providing insights into the process and allowing for troubleshooting.

- **Integration with Workflow Tools**: Connecting the script with workflow orchestration tools like Apache Airflow could automate the entire pipeline, from data extraction to analysis, and provide scheduling, monitoring, and alerting capabilities.

- **Dynamic Adaptation**: Enhancing the script to dynamically adapt to changes in HTML structures or Excel formats would increase its resilience and longevity as a tool.

- **User Interface**: Developing a user interface could make the system more accessible to users who are not proficient in scripting or programming, broadening its usability.

5.5. Conclusion

In conclusion, the provided script serves as a practical implementation of the concepts discussed in the literature survey. It captures the essence of what an Automated HTML Report Parsing and Excel Integration System is intended to do. However, the script represents a starting point, and there is ample room for incorporating additional features and technologies to create a more advanced, robust, and user-friendly system. The potential enhancements would address the challenges outlined in the literature and leverage the opportunities for innovation in this field.

5.6. Future Scope

The future development of the Automated HTML Report Parsing and Excel Integration System is poised for significant advancements across several domains. One of the primary areas of focus is the incorporation of machine learning and artificial intelligence to refine data ex

traction processes, especially from HTML reports with complex or inconsistent structures. The application of natural language processing could also be explored to extract meaningful data from reports containing natural language or unstructured text.

To ensure reliability and accuracy, robust error handling and data validation mechanisms are essential. These would include advanced error detection capabilities and the integration of stringent data validation rules. Enhancing user experience is another critical area, potentially through the development of a graphical user interface that would enable users with varying technical expertise to easily configure and manage the system. Customization features could allow users to specify their own rules for data extraction without delving into the codebase.

The system's integration with workflow orchestration tools would automate the data pipeline from extraction to analysis, complete with scheduling and real-time monitoring functionalities. As data volumes grow, scalability becomes a concern, necessitating distributed processing and cloud integration to handle larger datasets and ensure high performance.

Interoperability is also a key consideration for the future. Expanding the system to support multiple data formats and developing APIs would enhance its ability to interface with other applications and services. Security measures to protect sensitive data and adherence to compliance standards will be paramount, particularly in industries with stringent regulatory requirements.

Real-time data processing capabilities, including support for streaming data and event-driven triggers, would enable the system to provide up-to-date analysis and reporting. Moreover, the integration of built-in analytics tools and customizable reporting features would allow for immediate and tailored insights from the processed data.

Finally, the system could benefit from a continuous learning approach, where feedback loops help to improve accuracy over time, and adaptive algorithms adjust to changes in data patterns or reporting formats autonomously. Such enhancements would ensure that the Automated HTML Report Parsing and Excel Integration System remains a cutting-edge tool, capable of meeting the evolving needs of data-driven decision-making in a dynamic technological landscape.

REFERENCES

[1] Chakrabarti, S., Berg, M., & Dom, B. (2002). Focused crawling: A new approach to topic-specific web resource discovery. *Computer Networks*, 31(11-16), 1623-1640.

[2] Laender, A. H., Ribeiro-Neto, B. A., da Silva, A. S., & Teixeira, J. S. (2002). A brief survey of web data extraction tools. *ACM SIGMOD Record*, 31(2), 84-93.

[3] Miori, L., & Russo, D. (2017). Integrating Online and Traditional Involvement in Participatory Budgeting. *Electronic Journal of e-Government*, 15(2).

[4] Bruckner, R. M., List, B., & Schiefer, J. (2013). Developing requirements for data warehouse systems with use cases. *Information Systems*, 28(1), 1-14.

[5] Jones, M., & Gregorio, J. (2019). Simple Object Access Protocol (SOAP) 1.1. *W3C Note*, 08.

[6] Bao, Z., & Li, T. (2018). Smart data for mobility: Exploring the crossroad of data analytics and transportation. *IEEE Access*, 6, 13815-13829.

[7] Richardson, L., & Amundsen, M. (2013). RESTful Web APIs. *O'Reilly Media, Inc.*

[8] Lerman, K., & Minton, S. (2000). Learning the common structure of data. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence* (pp. 609-614).

[9] Tufte, E. R. (2001). The visual display of quantitative information. *Graphics press*.

[10] Codd, E. F. (2002). The relational model for database management: version 2. *Addison-Wesley Longman Publishing Co., Inc.*

[11] Kimball, R., & Ross, M. (2013). The data warehouse toolkit: The definitive guide to dimensional modeling. *John Wiley & Sons*.

[12] Han, J., Pei, J., & Kamber, M. (2011). Data mining: concepts and techniques. *Elsevier*.

[13] Zikopoulos, P., & Eaton, C. (2011). Understanding big data: Analytics for enterprise class hadoop and streaming data. *McGraw-Hill Osborne Media*.

[14] Chambers, B., & Zaharia, M. (2018). Spark: The definitive guide: Big data processing made simple. *O'Reilly Media, Inc.*

[15] Fisher, D., DeLine, R., Czerwinski, M., & Drucker, S. (2012). Interactions with big data analytics. *Interactions*, 19(3), 50-59.

[16] Loukides, M. (2010). What is data science? *O'Reilly Media, Inc.*

[17] Stonebraker, M., & Hellerstein, J. M. (1998). Content integration for e-business. *ACM SIGMOD Record*, 27(2), 552-560.

[18] Redmond, E., & Wilson, J. R. (2012). Seven databases in seven weeks: A guide to modern databases and the NoSQL movement. *Pragmatic Bookshelf*.

[17] White, T. (2012). Hadoop: The definitive guide. *O'Reilly Media, Inc.*

[18] Silberschatz, A., Korth, H. F., & Sudarshan, S. (2010). Database system concepts. *McGraw-Hill*.