Development

Problem Statements

DEV 01

Topic: Rooting for a good cause

Philanthropic organizations use their resources to improve society. However, they need to be equipped to do so. On the other side, local non-profit organizations (NGOs) rely on the funding provided by various funders. In today's world, there is a growing need for an effective platform to connect philanthropists and NGOs to help positively impact society. Currently, the process of finding and supporting the right cause can take time and effort, with limited transparency into the impact of the donations.

Objectives:

- To create a centralized platform that makes it easy for philanthropists to discover and support NGOs working in areas they are passionate about.
- To provide philanthropists with comprehensive information about each NGO, including their mission, history, and impact, to help them make informed decisions.
- To improve transparency and accountability in the donation process by providing real-time updates on the use of funds and the impact of donations.
- To provide NGOs with a way to reach a larger audience and secure the funding they need to carry out their missions.
- To facilitate better collaboration and communication between philanthropists and NGOs to help maximize the impact of their efforts.

.

Features

- Profile creation for NGOs and users/philanthropists: The NGOs can create profiles that include information about their previous works, their end goal, and their plans as to how they can achieve their goal and their overall impact on the environment and society, and their funding needs. Philanthropists can create profiles that include information about their donation preferences, such as what kind of NGOs they wish to donate to.
- Search and filter functionality: Users can search and filter NGOs based on various criteria such as location, impact area, type of NGO, etc.
- Suggestion algorithm: The platform can use an algorithm to suggest some NGOs to the users based on the type of NGOs they wish to support.
- Communication tools: The platform should include mechanisms for the users to communicate with different NGOs(text message, video meeting, etc.) to learn more about that particular NGO.

- Community features: Platform can allow users to join groups, share ideas, and connect with other NGOs/fellow philanthropists.
- News and events: Platform can also feature the latest news and events related to new NGOs and other societal/environmental welfare programs.

Brownie Points

- Progress/Activity Tracking: Users who have donated to an NGO should be able to see how much the NGO has received in donations and how these donations are put to use, as in, what they have planned for the future, and when each of their activities is scheduled.
- Crowdfunding facility: If a user wishes to conduct a philanthropic event but requires funds, the platform can provide the facility to run a crowdfunding campaign.
- Provide UPI/Payment support.

DEV02

Topic: Peeking The Emissions

Based on statistics, the download of one GB of data from the internet emits 11 grams of CO2-equivalents. An unoptimized website generally loads MBs of data on request. Considering that the website gets visited thousands and sometimes millions of times every day, major websites might have an unforeseen carbon footprint per user.

Objectives:

- Make a browser extension and calculate carbon footprint based on the total data sent/received when a user visits any website.
- Generate and display to the user the expected carbon footprint he has caused by visiting various websites.
- Create a website and display web pages based on their rank based on the overall footprint.
- Provide detailed carbon emissions caused by users on different sessions.
- To categorize websites as Green, Semi-Green, and Non-Green using parameters from emission data.

Brownie Points:

- Store all-time data showing the total emission caused by the user.
- Recommend users visit sites with a lesser footprint and provide similar functionality.
- Recommend users to upgrade their network depending upon the percentage of packets lost during transmission.

Reference:

- https://www.bbc.com/future/article/20200305-why-your-internet-habits-are-not-as-clean-as-you-think
- https://sphera.com/blog/the-carbon-footprint-of-the-internet/
- https://sustainablewebdesign.org/calculating-digital-emissions/

DEV03

Topic: All Things Green

Build a one-stop solution for all gardening aspirations and the country's farmers.
The government of India sends millions of dollars to provide schemes and educate the country's farmers. Still, these reach only a handful of them, and most still need to be used. Similarly, many nature enthusiasts want to start gardening in many metropolitan cities but need more experience. Like most farmers, they don't know what plants best suit their current geolocation and climate. We intend to provide a solution for everyone interested in getting to know plants.

Objectives:

I. **For Farmers**
- Create a central platform that connects farmers from all over India with the latest advancements in agricultural technology.
- Utilise GPS location to suggest crops for farmers to grow based on their specific field and family information.
- Provide information on available schemes and programs from the central and state governments.
- Educate farmers on various hybrid farming methods and technology.
- Include a chat system for farmers to communicate with each other and with experts in the field.

II. **For Enthusiasts**
- Build an application that helps you track all your plants and their requirements to provide regular notifications and remind you of the same.
- All users should be able to view previously asked plant-specific questions.
- Users should be able to ask plant-specific questions that the expert community can answer on the application.
- Users should be able to share their plant's progress through pictures/text with the gardening community.

Features:
- Build a unified platform for people to share their agricultural innovations.
- Users should be able to find all the specifications on how to grow any crop/plant.
- Users can browse plant-specific questions.
- Recommend the best plants to grow based on the user's geolocation and climate.
- Make farmers aware of the new policies put forward by the government.

Brownie Points:

- Add multi-language support to your website or application.
- Allow users to ask plant/crop-specific questions to experts one-on-one via chat or video call.
- Allow communities to send financial aid to farmers directly.

DEV04

Topic: Dynamic entity clustering

-contributed by BharatX

Dynamic Clustering of data can improve data processing efficiencies manifold.
Let's say there is an entity with the initial set of parameters. The system is expected to group entities based on "similar" parameters. "Similarity" could mean different things for different parameters. For example, if we consider users as the entity.

Possible parameters could be: name, date of birth, gender, age, address, etc. Similarity for gender is an exact match, for age is an exact match, whereas, for name, some partial matching algorithm would be used. The rules for clustering should also be easily configurable, we might initially want to cluster based on name only, but moving forward, we might want to cluster based on age as well.

Use case:

- Let's say we want to cluster users interacting with our app for marketing purposes. We should be able to use this system.
  The details about the user are collected over the period during which the user interacts with the app, but the clustering happens in real time as we collect more data for the user.
- This service can be used to identify users who are trying to gain access to the same resource using different digital personas. For example, suppose a user is using different email addresses to gain access to the trial period of a streaming service. In that case, they can be identified as the same user by using other parameters like time of access, IP address or browser details.

Objectives:

- To design a system that is easily configurable on the number and type of parameters and clustering rules.
- The clustering needs to occur in real time. The values for the parameters are to be fed asynchronously to the system.

Features:

- Fundamental clustering services should have APIs that would take in data from the user asynchronously and generate clusters in real time.
- Provide the user with an API interface to query all the clusters.
- Allow users to be extendable in parameters and rules
- Docker-based deployment allowing scalability

Brownie Points:

- Provide users with an interactive UI where they can see all the clusters getting formed in real-time.
- Provide an option for clubbing clusters if the user wishes to do so.

Machine Learning

Problem Statements

ML01

Topic: Generative Audio

Sustainability in language technology is crucial for supporting and preserving linguistic diversity, which is an important aspect of human culture and communication. However, many underrepresented languages and dialects are not well-supported by current speech technology due to a shortage of available speech data for training models. Developing speech recognition models for these languages and dialects can play a crucial role in promoting sustainability in language technology and preserving linguistic diversity.

Objective:

- We propose to develop a generative model that can create synthetic speech samples for underrepresented languages and dialects.
- The model should be able to generate speech samples in a variety of languages and dialects, with a focus on those that are underrepresented in existing speech datasets.
- The generated samples can then be used to train and improve speech recognition models for these languages and dialects, promoting linguistic diversity and reducing language barriers in speech technology.

Brownie Points:

- The model should be able to generate speech samples for a variety of languages and dialects, not just a few select languages.
- The model should be able to generate speech samples for both male and female speakers and for different age groups.
- If the language of the speaker is English, predict the native language of the speaker based on the accent.

Additional Features and Links & Resources:

- Evaluation metric - Mean Opinion Score (MOS)
- Mozilla Common voice Dataset
- Generative Adversarial Networks (GANs)

ML02

Topic: Observing from Space

The Greenhouse Gases observing (GOSAT) is the world's first spacecraft to measure the concentrations of carbon dioxide and methane, the two major greenhouse gases from space where, as The Orbiting Carbon Observatory-2(OCO-2)is the first National Aeronautics and Space Administration((NASA) satellite designed to measure atmospheric carbon dioxide(C02) with the accuracy, resolution, and coverage needed to quantify CO2 fluxes(sources and sinks) on regional scales

Objective:

- The main task is to use GOSAT and OCO-2 data and build/compare Machine Learning or Deep Learning Models to better predict CO2 over the surface of Earth for a smaller area.

Brownie Points:

- The model should predict accurately for even smaller regions than what oco-2 and great measure.
- The model can use both data combined for better accuracy and performance.
- Good model and code quality.

REFERENCES and Useful Links:
Dataset Links:

- https://ocov2.jpl.nasa.gov/
- https://earth.esa.int/eogateway/catalog/gosat-tanso-fts-and-cai-full-archive-and-new-products?category=Data
- https://disc.gsfc.nasa.gov/datasets?keywords=oco-2&page=1

ML03

Topic: Crop Prediction

Predicting the optimal crop that could be grown in a particular location based on the soil composition, rainfall, temperature and return of investment is essential for any farmer of the modern world. This could not only save farmers from spending extra money on fertilizers but also avoid the dangerous practice of growing the same crops on the same patch of land repeatedly, which depletes the solid from its natural resources.

Objective:

- Given the dataset, you need to predict what crop would be best for the farmer to grow based on the geolocation, season, and price.
- Make a user-friendly interface for the public to interact with your ML model.

Brownie Points:

- Suggest a variety of crops that could be cultivated for the geolocations provided in the dataset and rank them according to the sales price.
- Improve your model by factoring in the various types of soil found in various parts of India and its general nutritional composition.

REFERENCES and Useful Links:

Dataset Links:

- https://www.kaggle.com/datasets/rajanand/rainfall-in-india
- https://www.kaggle.com/datasets/vanvalkenberg/historicalweatherdataforindiancities
- https://www.kaggle.com/datasets/atharvaingle/crop-recommendation-dataset
- https://www.kaggle.com/datasets/thammuio/all-agriculture-related-datasets-for-india
- https://currentaffairs.adda247.com/major-soil-types-of-india-map/

WEB3

Problem Statements

WEB01

Topic: Carbon Taxing

One of the best ways to control carbon emissions into the environment is by taxing them. Companies are sure to take their emissions more into consideration when there's tax involved; just an environmental factor is never enough. Setting a price on greenhouse gas emissions, mainly carbon content from fossil fuels, is called Carbon Taxing.
Usually, how it works is that there's a maximum cap given to companies under which their emissions should lie. If any company exceeds this cap, heavy taxes are imposed. On the other hand, they're rewarded with points if they're below the cap. These points, in turn, can be sold to other companies, who end up polluting more but maintain the so-called equilibrium. These transactions are usually made via the government. Hence it's not very transparent to the public.

Objective:
The objective here is to create a Web 3.0 application that can help individuals and companies monitor and check their carbon emissions by keeping track of their carbon taxes in correspondence with the limits set by government bodies. The platform should be user-friendly and allow users to identify areas where they can make changes to be more environmentally friendly.

Features:

- Government bodies should set carbon caps for the corresponding industries, which will be visible to everyone.
- Companies have to submit their total value of carbon emissions. The company should be able to log in/have some credibility for themselves. Emissions need to be taken in various forms, and the total tax needs to be calculated.
- The platform should have a reward system (in the form of crypto-tokens) for individuals and companies who are below the cap. The reward should be proportional to the amount of CO2 that they could have emitted by still staying under the threshold set by the government bodies.
- Integrate the smart contract with the front-end user interface and back-end infrastructure for your dApp. You will need to write code to interact with the smart contract through an API to retrieve data and perform transactions. Make sure users/companies with their credentials can view all the data.

- The application should be user-friendly and able to provide detailed and accurate data on the carbon tax of users. The platform should also allow users/companies to share their progress with others and promote sustainability easily.

Brownie Points:

- Allow government bodies to set limits in slabs. For example, if the total emission is below 10 Tons, let the charges be zero. If it's between 10-15, let it be $1.5 Million. If it's 15 - 25 tons, let it be $3 Million, and above 25 Tons, let it be an additional $2 Million for every ton.
- Allow government bodies to set varied limits for different industrial sectors like manufacturing, textile, or agriculture and calculate the tax that's needed to be paid by the company depending upon the sector they fall in.
- Allow companies to use their crypto token to lower the amount of tax that's needed to be paid for the current month/year.
- Allow companies to share crypto tokens to lower the overall tax.

Resources:

- https://www.footprintnetwork.org/
- https://www.worldbank.org/en/programs/pricing-carbon#:~:text=A%20carbon%20tax%20directly%20sets,but%20the%20carbon%20price%20is.
- https://www.c2es.org/content/carbon-tax-basics/
- https://www.bdc.ca/en/articles-tools/sustainability/environment/how-measure-your-carbon-footprint

WEB02

Topic: Countering fake certificates

Fake certificates in the healthcare industry can have serious consequences, including misdiagnosis, inappropriate treatment, potential legal issues, and the spreading of diseases. Additionally, using fake certificates can undermine the credibility and trust in the healthcare system and put patients' health and safety at risk. To address this issue, it's crucial to have a secure and reliable system for verifying the authenticity of certificates and preventing the use of fake certificates.
Develop a blockchain-based platform to prevent fake certificates in the healthcare industry.

Objective:

- To create a secure and transparent platform using blockchain technology to prevent the issue of fake certificates in the healthcare industry.
- Secure and tamper-proof storage of healthcare certification records
- Verification of the authenticity of medical credentials and certificates
- Creation of a decentralized repository of certified healthcare professionals
- Implementation of a secure and efficient system for maintaining and updating certification records.

Brownie points:

- Facilitate self-sovereign identity for each person
- Integrate the existing healthcare records into this platform
- Issue certificates cost-effectively
- Encrypt sensitive medical information
- Transparency in the process of granting certificates

Resources:

- https://polygon.technology/polygon-id
- https://ipfs.tech/
- https://filecoin.io/
- https://www.storj.io/

Electronics (IOT)

Problem Statements

IOT01

Topic: ECO-DRIVING DRIVER ASSISTANCE SYSTEM

Vehicles have a major contribution to the total energy demand as well as to harmful emissions. In light of this, an important factor in creating sustainable cities is the reduction of energy required to power automobiles.
A method of achieving the same is eco-driving. Eco-driving is a term used to describe the energy-efficient driving of vehicles. By this principle, a driver follows certain principles while driving that help him reduce the total energy consumed.
In this problem statement, you are required to design a driver assistance system that will remind a driver to follow eco-driving principles when a certain principle is violated.

The driver assistance system must look out for the following principles.

- The Cruising Speed of the vehicle must be around 90 km/h
- Acceleration shouldn't exceed 1.4705 m/s2
- If the vehicle is idling (Idling is a working condition in which the engine is idling but does not output power, resulting in fuel consumption) for more than half a minute, recommend that the engine be switched off.
- The number of braking must be kept minimal.
- Detect obstacles ahead and suggest an optimal deceleration
- Track the location of the vehicle.

The driver assistance system must, at fixed intervals of time, transfer the vehicle's above-mentioned parameters to a remote server. After processing the parameters, the remote server must alert the user when a principle is being violated on an application or make the suggestions mentioned above. (The application can be a simple python client application). The more useful suggestions the driver assistance system can make, the better.

BROWNIE POINTS:

- The remote server can be based on any cloud platform such as Thingspeak or AWS.
- Collect user data and display his driving trends and recommend what factors could be improved upon.

Software recommended to use:

- https://wokwi.com/ for simulating any sensors needed or any other software of preference
- Use python to send and receive data from the remote server/Cloud platform
- Use python to create the program in the server processing the data
- Thingspeak or AWS can be used as a cloud platform if a bonus problem is being solved

Submission Guidelines

- Link of circuit if some sensors were simulated on wokwi or tinkercad
- Upload all scripts in GitHub and the links of circuits
- A pdf explaining the solution in detail
- A video demonstrating the working of the project
- Partial solutions will be accepted

IOT02

Topic: Drift

Smart driving systems with V2X communication are an important milestone in the field of wireless communication. It is extensively used in semi-autonomous and autonomous vehicles in today's world. This allows you to communicate via the LTE network with the vehicles and infrastructure in the vicinity. The problem statement demands you to design an IoT system which helps in traffic control and management to reduce time and pollution by proposing recommended commands to the driver.

Objective:

- To design an IoT edge module for each vehicle which replaces the need for traffic lights at intersections so that vehicles communicate with each other and smoothen the traffic flow.
- Implementing V2X communication that should help solve whenever there is a traffic hold-up.

RULES AND REGULATIONS:

- The circuit must be created in any simulator, including all the sensors and modules for processing.
- Use of a microcontroller is allowed, in which case the code must also be submitted.
- It should be edge computation and takes place inside the device.
- The participants can add more features if they want, but the above-mentioned are important.

BROWNIE POINTS:

- An efficient algorithm that takes the vehicle in light traffic and the shortest path.
- Simulate the solution in Veins software.

SUBMISSION GUIDELINES:

Submission should include (Creating a drive folder with the following files):

- Wokwi: circuit simulation
- Arduino IDE: code transmitted to the microcontroller
- Cisco Packet Tracer: v2v and v2I communication network
- Python script: working of algorithms

The submission must include the following:

- The link to the project (easyeda) or the .json file (Wokwi), Arduino IDE, Cisco Packet Tracer, python script, screen recording of veins simulation(optional).
- A pdf: Explaining the overall working of the circuit with necessary screenshots Explaining the circuit logic for every feature implemented with necessary screenshots

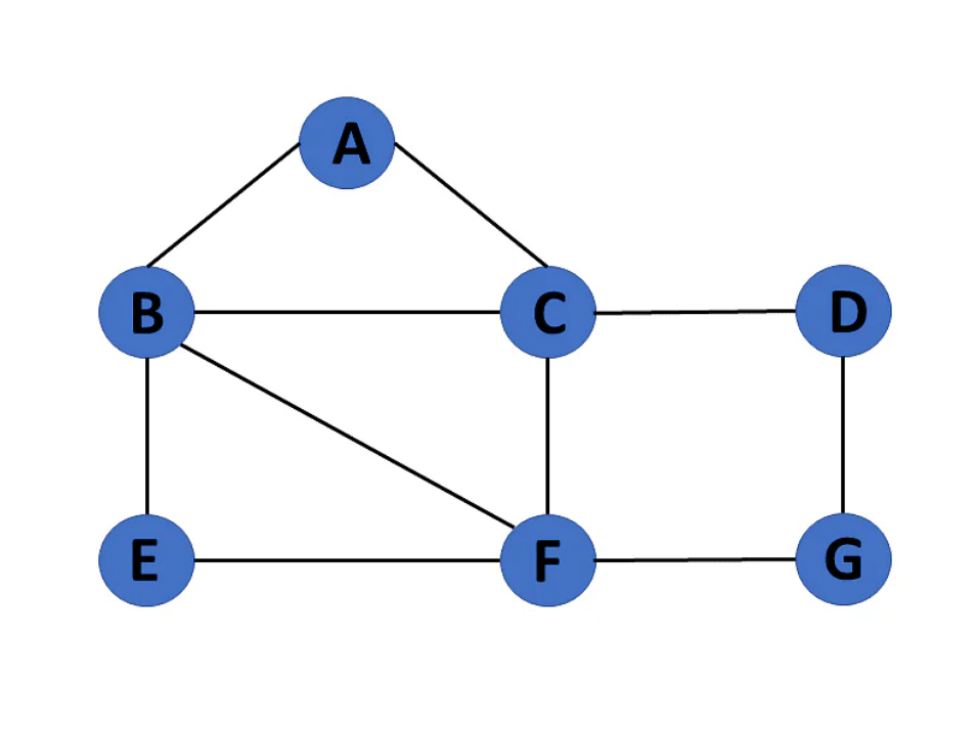
Resources:

1. Cisco Packet Tracer: https://www.netacad.com/courses/packet-tracer
2. ArduinoIDE:https://docs.arduino.cc/software/ide-v1/tutorials/Windows
3. Veins: https://veins.car2x.org/
4. Wokwi: https://wokwi.com/

IOT03

Topic: Smart Grid

Electricity consumption of households can have a significant impact on the overall power demand in a region, which can lead to increased power loss in the grid due to the resistance and inefficiencies in the transmission and distribution of electricity. Power loss in the grid can occur as a result of heat dissipation, line losses, and transformers. It is estimated that power loss can be as high as 10% in some areas, which can result in increased costs for utilities and higher electricity prices for consumers. Power lines aren't built most efficiently and are usually made based on the demand for electricity. If we knew that a household would consume less power at a particular time of the day, we could find the best path for the electricity to reach that household and shut off lines that are not necessary.



Objective:

1. Design an IoT sensor suite for a household to predict its energy consumption based on the homeowner's preferences and compare it to the current condition. For example, what the preferred room temperature is vs the current ambient temperature can be used to predict energy consumption by the air conditioning. Implement a database to which each household uploads data.

2. Design a master node that is able to access the database and predict the shortest path to each house based on the length of the transmission line and the maximum load capacity of each line.

3. Build a dashboard for an FMC (facility management company) to observe the power consumption in a neighborhood.

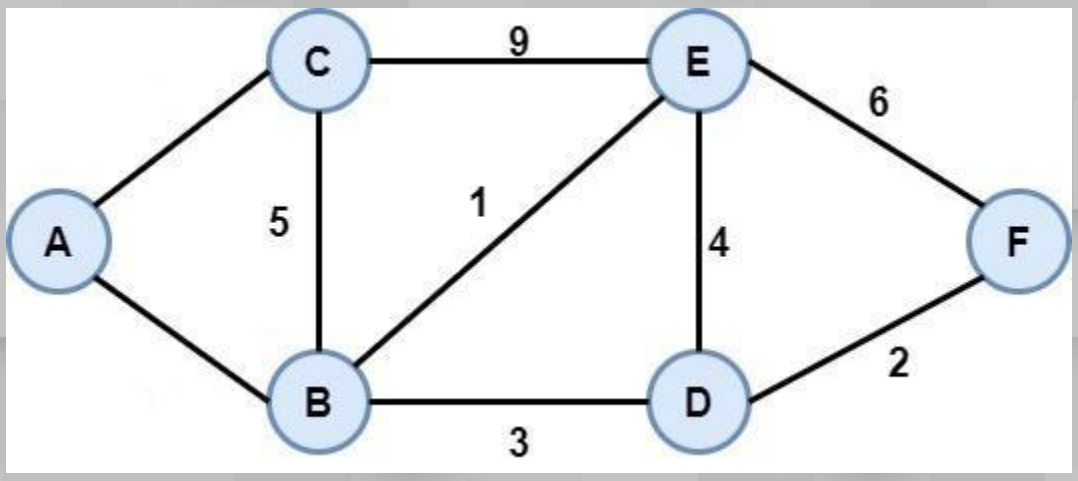
Software Used:

https://wokwi.com/ - esp 32 and network simulations
Tinkercad - Arduino simulations
Python - Dashboard

Implementation Rules:

- If a particular sensor is not available in the simulator, you can provide documentation and the code regarding the sensor in your implementation.
- Below is the structure of the neighborhood household grid.
    - A is the power station, and all other nodes are households
    - All the numbered transmission lines have a max power rating of 90KW (based on average US household power consumption of 29KWh per day).
    - The goal is to predict consumption throughout the day and find the best path to connect the household.

-



- Make sure the energy predictions are realistic (use 29KWh per day as reference) algorithms can also be made in python or in any other language that you are comfortable with.

Submission Rules:

- Final submission should be a PDF doc that includes:
    - Explanation for choice of the microcontroller.
    - Explanation for the Sensor suite.
    - Explanation about the choice of the database.
    - Code for the household IoT node.
    - Explanation and code for the master node.

- Final submission should contain explanations of **all** design decisions made during the design process.
- All code should be commented on to improve understanding.

Resources:

- Graph theory
- Arduino Basics
- ESP32 basics
- IoT Basics

Electronics
(Digital Design)

Problem Statements

DE01

Topic: I2C to UART

As the world rapidly advances in technology, we can see various communication protocols emerging and leaving. If one possesses various devices, he/she would have, at some point, faced compatibility issues between certain devices due to differences in protocols.
Instead of buying a new device that supports the protocol, one can use converters to act as a bridge between two protocols, reducing the amount of electronic waste generated over the lifetime of a system and promoting sustainability.

The problem statement demands you to design a circuit that does the following:

- Receive an RSA encrypted message transmitted by an I2C master
- Decrypt the message
- Transmit the decrypted message to another UART device
- Brownie: Display the decrypted message

Specifications:

Slave address: 07h

RSA keys: (n,e) = (143,11)

UART baud rate: 9600

UART parity: Odd

Rules and regulations:

- The circuit must be created using basic elements of digital electronics only
- Other basic circuit requirements like transistors, resistors, clocks, displays, switches, power supply etc., can be used if required.
- Any elements that abstract the decoding/ decrypting logic used are not allowed to be used. In case of any doubt regarding the component you're planning to use, clarify the same through the discord server.
- The use of microcontrollers is forbidden except for the generation of I2C and reception of UART.

- The circuit should strictly adhere to the communication protocols of I2C and UART.

Test case:

The following test cases are required to be simulated in your circuit, and a video displaying the same should be attached to the submission:

· 88 91 66 79 78 89

· 89 66 93 78 88 95

Apart from the given test cases, a few hidden test cases will be used to test your circuit after submission.

Submission guidelines:

Submission should include (Create a drive folder with the following files):

- Circuit link or file
- A full circuit downloaded image at high resolution
- A pdf explaining your logic (with necessary screenshots).
- A video (screen recording) containing a simulation of your circuit by testing it with provided test cases. (Maximum duration: 5 minutes)
- Approach and other essential information about your circuit that might be important for evaluation.

Resources:

SimulIDE https://www.simulide.com/p/home.html

TinkerCAD simulator https://www.tinkercad.com/

Proteus simulator software https://www.labcenter.com/simulation/

These are just examples of simulators which could be used to build the circuit. However, you are free to use the simulator of your choice.

DE02

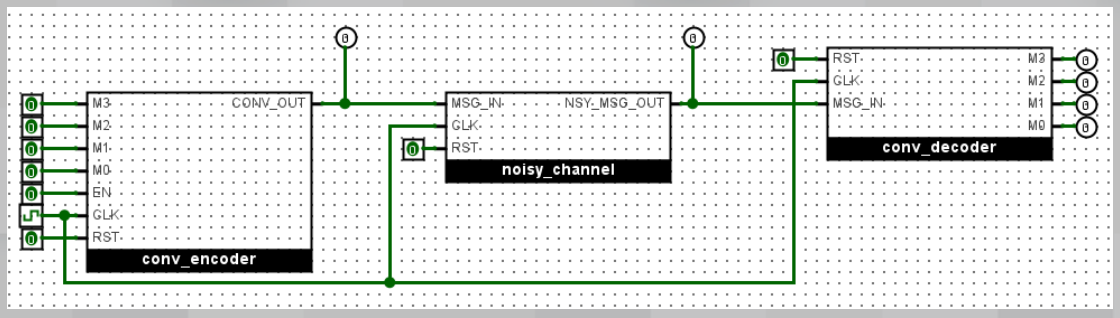
TOPIC: Convolutional Code Communication System

There's almost no place in today's world without digital communication. But where there's communication, there is also ambient noise which corrupts the transmitted signal. It is not always efficient to re-transmit the message, as it costs time and energy. This is especially true when high throughput is required. A more sustainable alternative is to code the message bits with error-correcting codes so that the receiver can decipher the error-free message while maintaining high throughput.

There are several classes of error-correcting codes, such as Hamming codes, Reed-Solomon codes, convolutional codes, etc. Convolutional codes are more sustainable since they have better error correction performance and because the same encoder circuit can be used for any length of the message.

The problem statement is to design a digital communication system based on convolutional codes. The problem statement is 3-fold:

1. You have to design a convolutional code encoder circuit.
2. You have to design a circuit that simulates channel noise.
3. You have to design a convolutional code decoder circuit.



1. **Convolutional Code Encoder:** The circuit must parallelly input a 4-bit message sequence. The parallel message sequence is fed serially (LSB first) to the input of the Convolutional Code Encoder. The encoder has a constraint length of 3, and parity equations are as follows:

   $p_0[n] = x[n] + x[n-1] + x[n-2]$
   $p_1[n] = x[n] + x[n-2]$

The parity bits generated above are transmitted serially, such that the p_0 bit is transmitted first. For example $p_0(0)$ is transmitted, then $p_1(0)$, then $p_0(1)$, then $p_1(1)$, and so on.

2. **Channel Noise Simulator:** The circuit must accept the output of the Convolutional Code Encoder, and for each code word received, it should introduce one bit of error. It must comprise a pseudo-noise sequence counter that counts in the following order and then repeats:
   8, 11, 5, 1, 2, 7, 0, 4, 3, 9, 10, 6
   Each time a new message sequence is given, the counter must increment its count once. The current count of the counter represents the bit position of the code word at which the error is introduced. The error-induced code word is then transmitted.
   For example, 110101001011 is transmitted LSB first. The noisy version of it is 110001001011, with an error in the 8th bit. If the same code word is transmitted the next time, its noisy version is 010101001011, with an error in the 11th bit.
   Note: Any counter(s) used in the Channel Noise Simulator must strictly be synchronous counters, i.e. all flip flops used in the counter must share the same clock.
3. **Convolutional Code Decoder:** The output of the Channel Noise Simulator is the input of the Convolutional Code Decoder. The input is stored in a shift register, and the Viterbi algorithm is used to retrieve the message bits from the input. This circuit is left to the creativity of the contestant.
   No hard coding is allowed except for the contents of the state diagram. Consider an appropriate number of Trellis stages. In case of conflicts between two states, choose the state whose binary representation is the lesser of the two. No paths in the Trellis must be ignored.

Software Used:
LogiSim (Find download link in Resources)

Implementation Rules:

- Only basic logic gates like AND, OR, NOT, NAND, NOR, XOR, XNOR, and basic sequential elements like D Flip-Flop, T Flip-Flop, J-K Flip-Flop, S-R Flip-Flop can be used.
- You cannot use any other element, such as multiplexers, shift registers, etc. If you wish to use such elements of abstraction, use subcircuits which you yourself built.
- The “data bits” property of all circuit elements must be 1. For e.g. you cannot use a single AND gate to perform $(0010)_2\&(1010)_2$, you must use 4. However, you can use multiple-input elements like 3-input XOR gates, etc.

- All sequential elements must trigger on Rising Edge.
- All elements must be active high, including asynchronous sets and resets.
- Importing any kind of HDL library is not allowed.
- No hard coding is allowed except for the contents of the state diagram. Actual convolutional code encoding and decoding must take place.
- Good design practices are expected, such as modularising your circuit with subcircuits.

Test Sentence:
Consider the message to be the position of each character in the English alphabet (a = 0, b = 1, c = 2, etc.). The following sentence is to be transmitted and received (you can ignore spaces):
book madam check deep map fan gap cable mimic jail
You are not allowed to "reset" the Channel Noise Simulator before completing the sentence.

Submission Rules:
1. You must submit the ".circ" circuit file.
2. Provide the steps to simulate your circuit in a .txt file.
3. Provide a .txt file containing the outputs of "Test Sentence" in the form of English alphabets.
4. Provide a .pdf file explaining the logic behind your circuit. Attach any paperwork like state tables, etc., if needed. Also, provide a description of all subcircuits used in your circuit.
5. Provide a screen recording of you simulating the "Test Sentence".

PARTIAL SUBMISSIONS ARE ACCEPTED!

Resources:
- LogiSim download link: https://github.com/logisim-evolution/logisim-evolution/releases/download/v3.8.0/logisim-evolution-3.8.0-x86.msi
- Convolutional Code: http://web.mit.edu/6.02/www/f2010/handouts/lectures/L8.pdf
- https://www.youtube.com/watch?v=kRIfpmiMCpU
- Viterbi Algorithm: https://www.youtube.com/watch?v=dKIf6mQUfnY