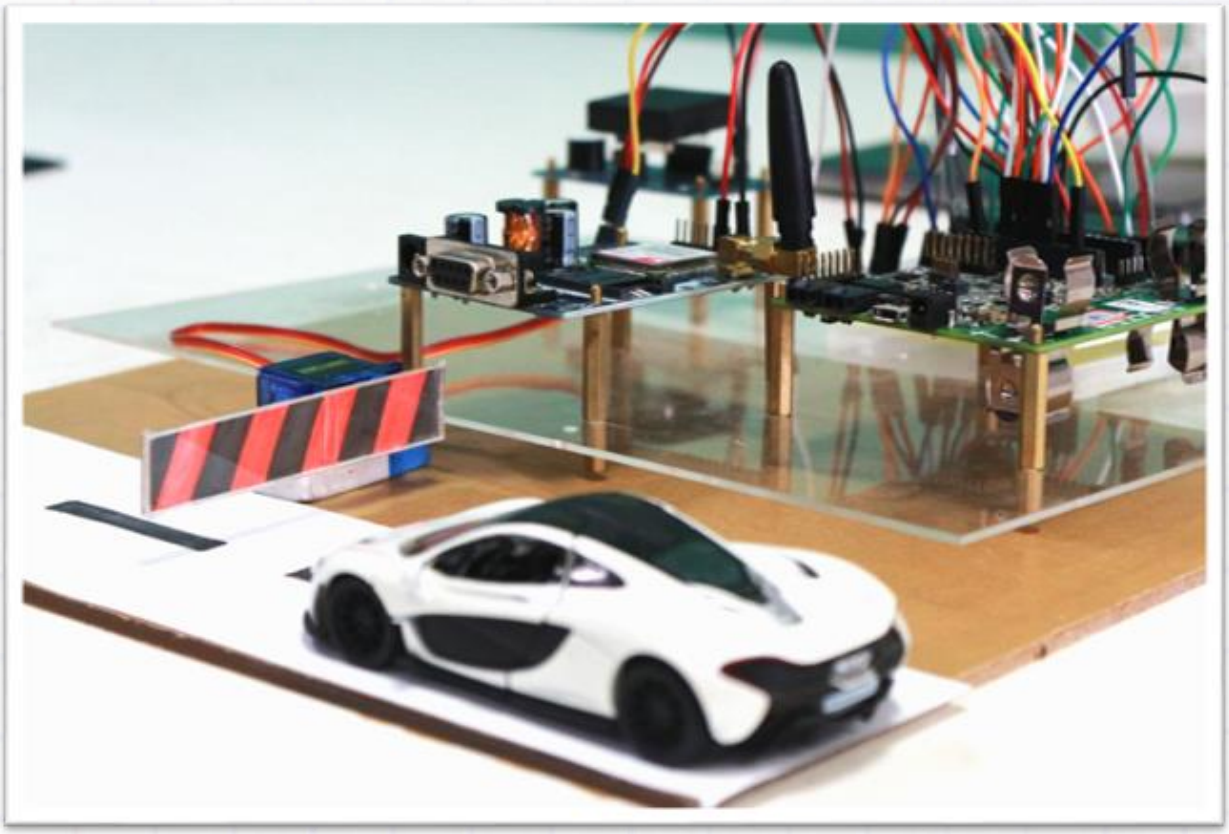
Build a FAST TAG Reader

5



Index

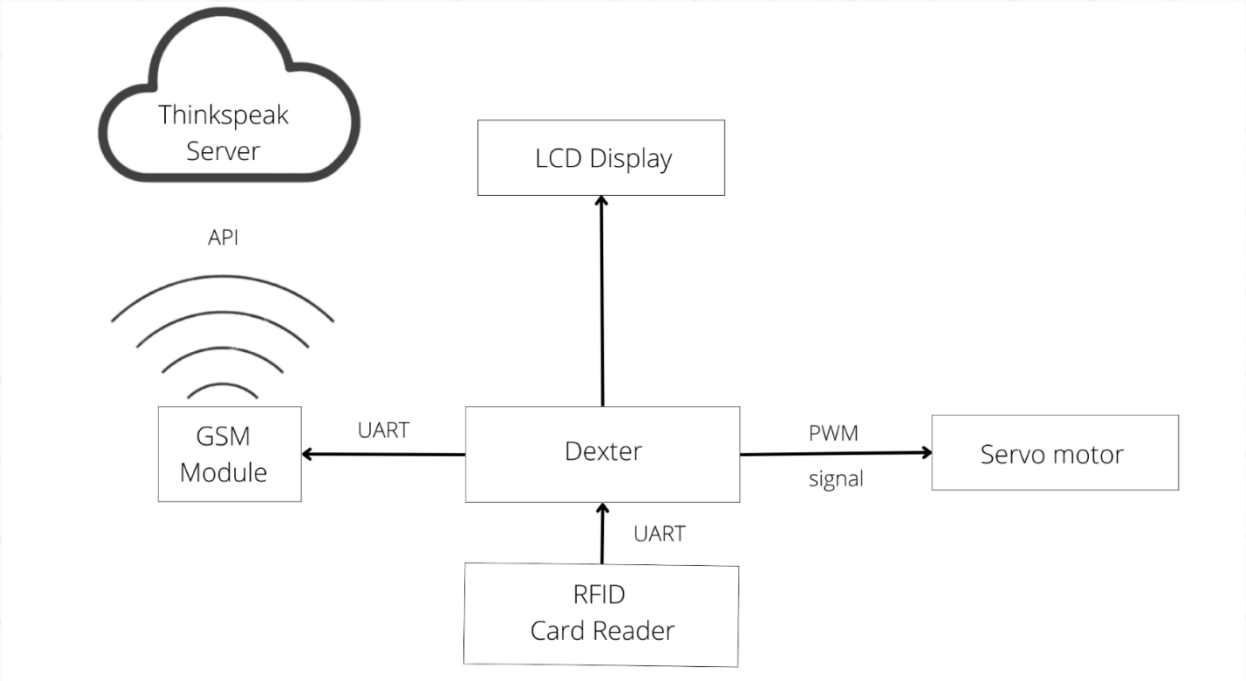
A. AIM

Build a RFID Fast Tag Reader that registers RFID card data, processes it, create actions, publishes the data to an online dashboard.

B. CONCEPT

In this project, we mimic the working of a fast tag reader which reads RFID card data and operates the boom barrier in toll gates. Here we read RFID cards , verifies it with the registry in the memory, open/closes the gate and finally sent the data to online dashboard.

The RFID card data is read by 'RFID card reader', the data is stored in the 'FRAM memory' of dexter board since it needs to be accessed later. 'LCD display' is used to display status of each steps in the process. The data regarding the number of cards, entries permitted/ not permitted is sent to an online dashboard with the help of a 'GSM SIM module'. 'thingspeak' is the web dashboard we use here for the purpose.



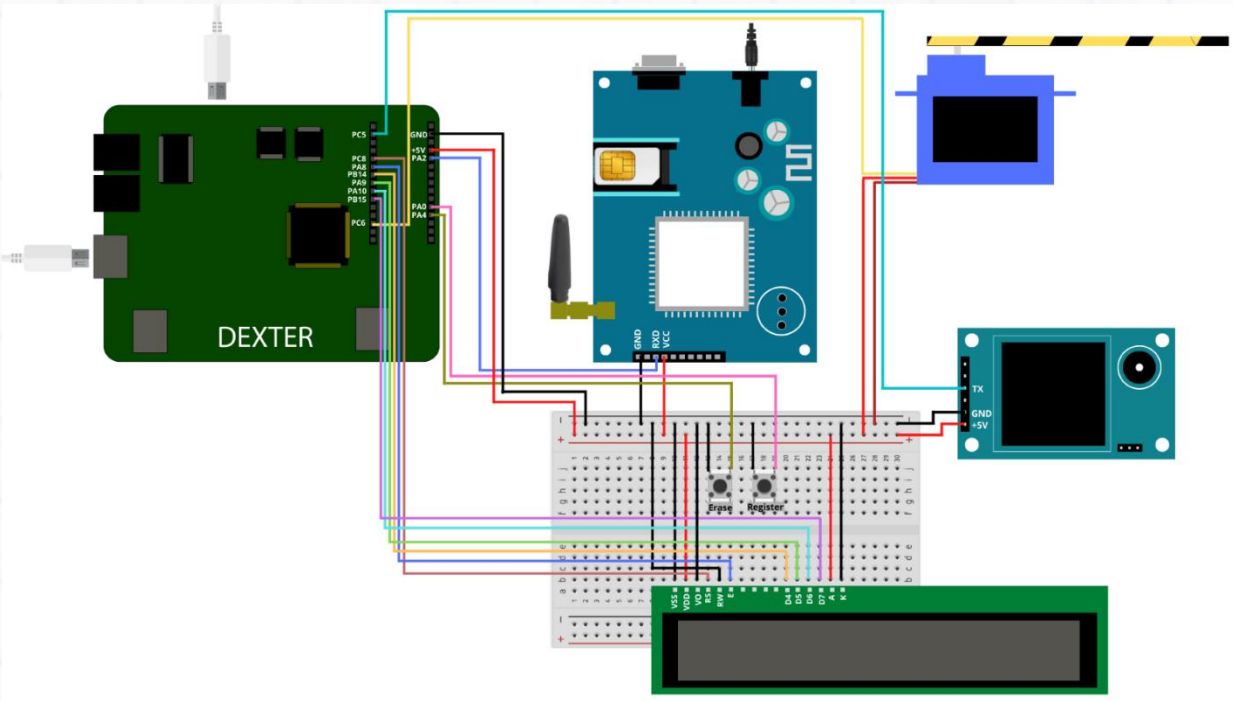
C. Components

1. Dexter board
2. RFID Card Reader
3. GSM Module
4. Servo Motor
5. 16*2 LCD Display
6. Jumper wires
7. USB cables
8. USB to TTL converter (Component to execute extra activities in the project. Procure externally if not available in the kit, CP2102(6-pin) USB 2.0 to TTL UART serial converter)

D. Connections

Circuit Diagram

- **NOTE**: Before starting the connections, verify using a multimeter that all the jumper wires are working. Also ensure that the connections are strong, else the setup may not work.

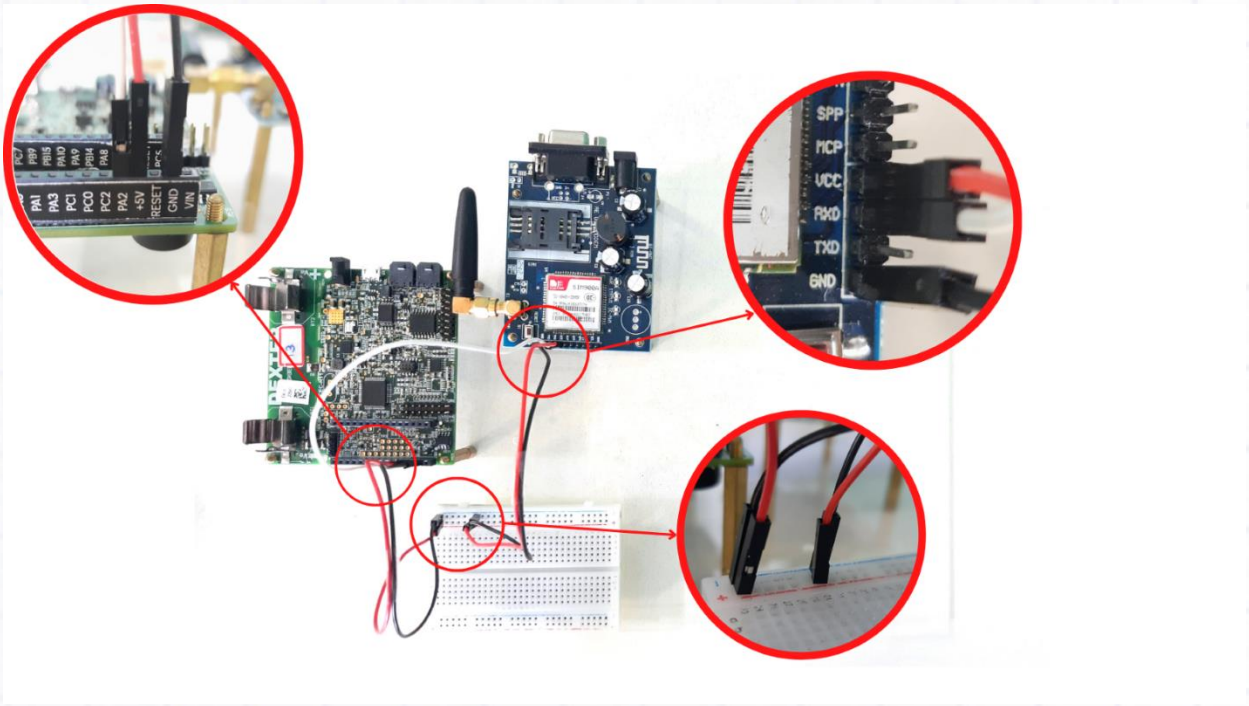


Detailed Connection Steps

Step 1

Take 2 male-to-male and 3 female to male jumper wires connect dexter board, GSM module and breadboard as shown below -

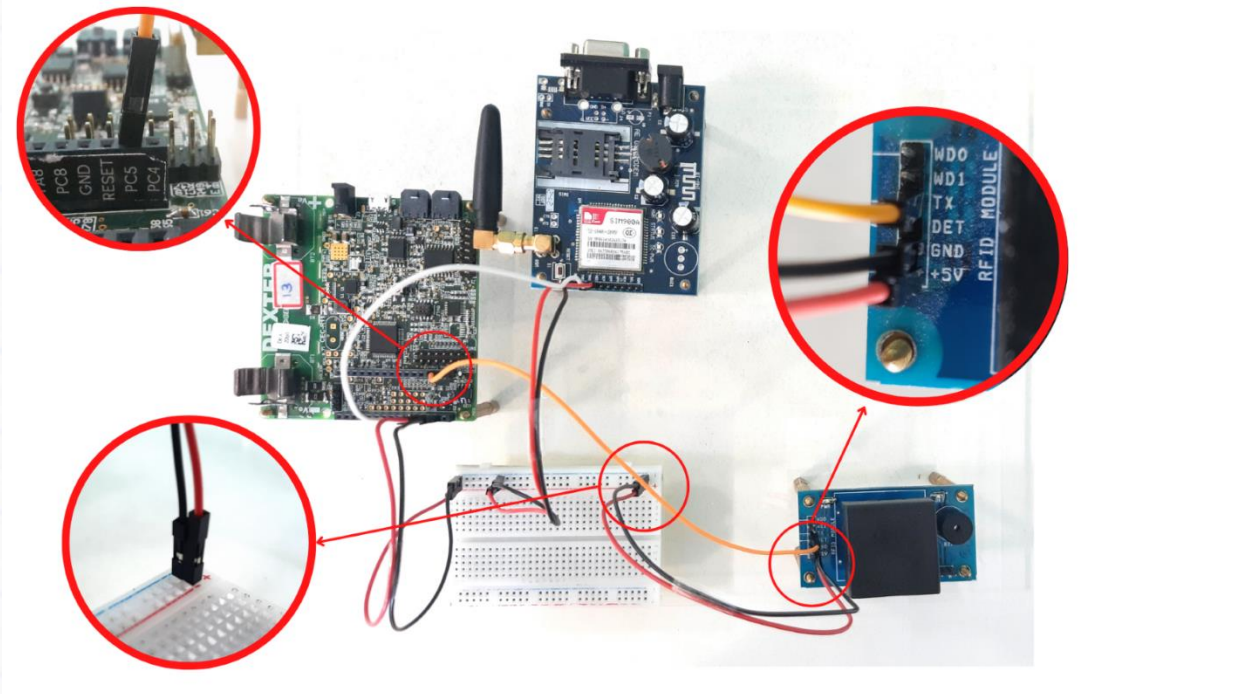
- **GND** of the dexter to **blue line** of the breadboard
- **+5V** of the dexter to **red line** of the breadboard
- **RXD** of the GSM module to **PA2** of the breadboard
- **GND** of the GSM module to the **blue line** of the breadboard
- **VCC** of the GSM module to the **red line** of the breadboard



Step 2

Take 3 female-to-male jumper wires and connect the pins of the RFID card reader as shown:

- **+5V** to **Red line** of breadboard
- **GND** to **Blue line** of breadboard
- **TX** to **PC5** of the Dexter

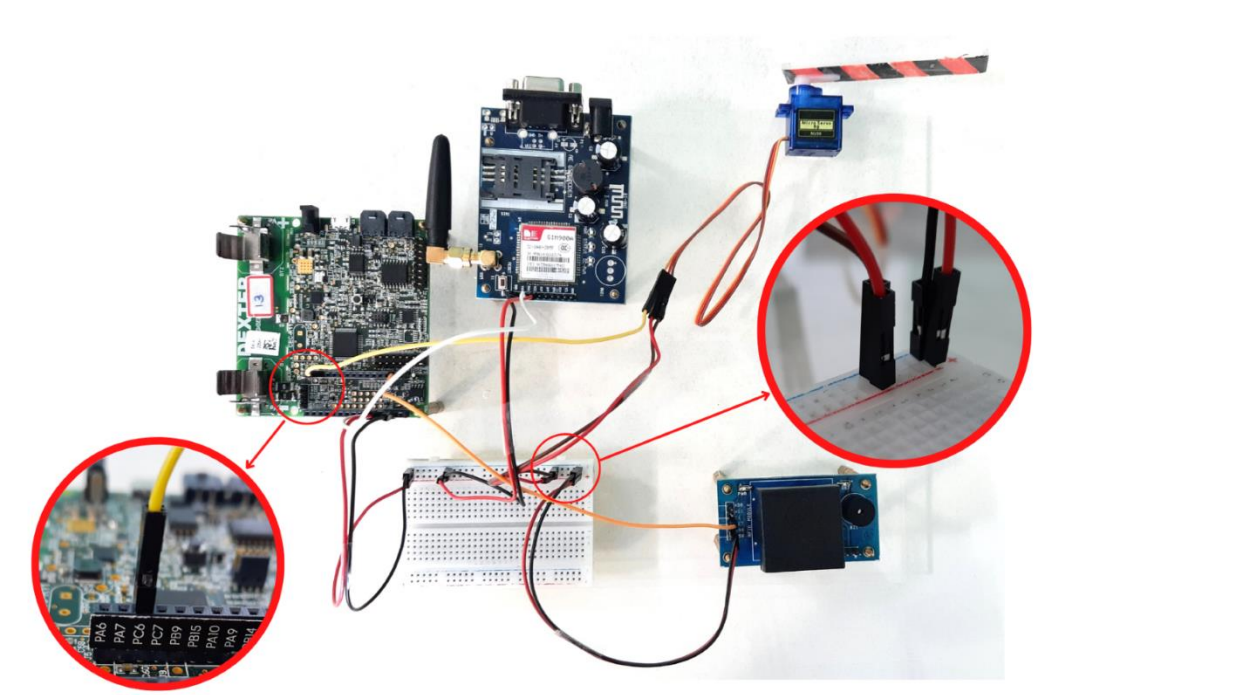


Step 3

Take 3 male-to-male jumper wires and connect the pins of DC servo motor :

- **Yellow cable** to **PC6** of the dexter board

- **Red cable** to **Red line** of the breadboard
- **Brown cable** to **Blue line** of the breadboard



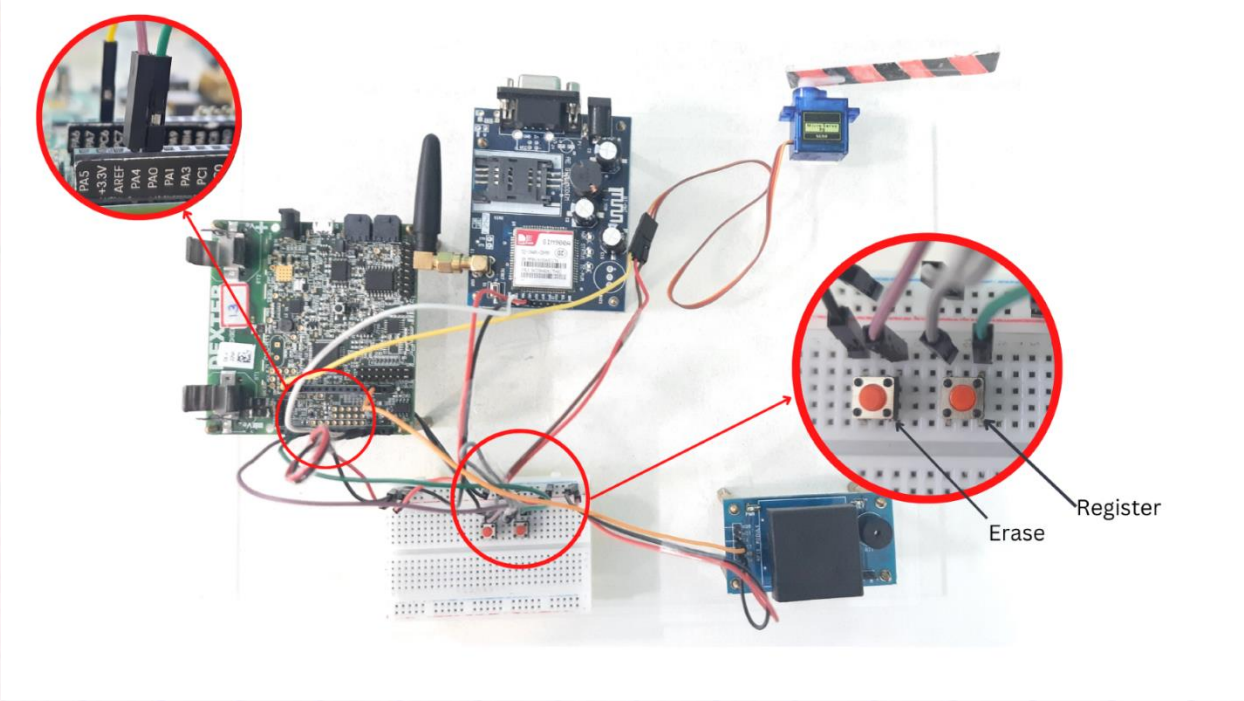
Step 4

Take 2 male to male jumper wires and connect the register button as shown below :

- One terminal of register button to **blue line** of the breadboard.
- Second terminal to the **PA0** of the dexter board.

Take 2 more male to male jumper wires and connect the erase button as shown below :

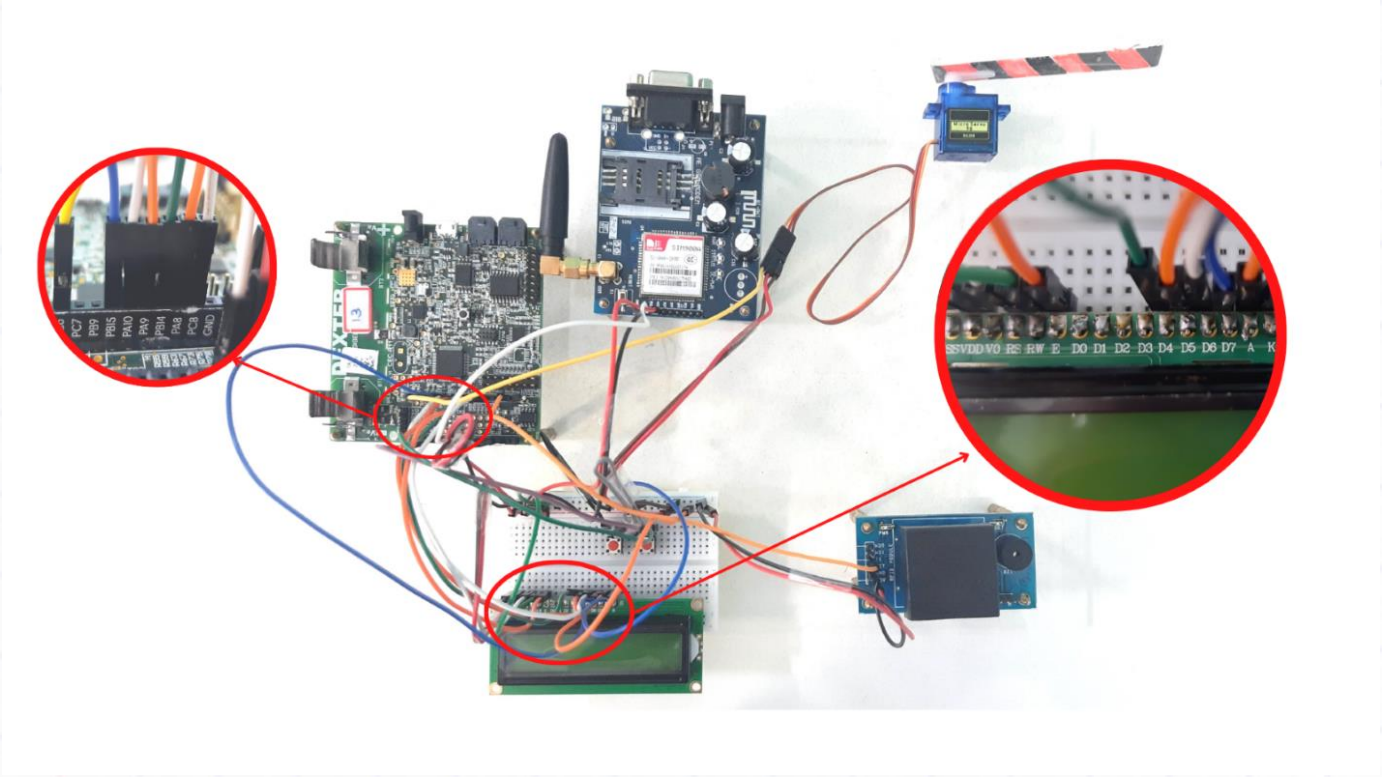
- One terminal of the erase button to the **blue line** of the breadboard.
- Second terminal to the **PA4** of the dexter board



Step 5

Take 11 male to male jumper wires and connect the LCD display and dexter as shown below :

- **VSS** to the **blue line** of the breadboard
- **VDD** to the **red line** of the breadboard
- **RS** to the **PC8** of the dexter
- **RW** to the **blue line** of the breadboard
- **E** to the **PA8** of the dexter
- **D4** to the **PB14** of the dexter
- **D5** to the **PA9** of the dexter
- **D6** to the **PA10** of the dexter
- **D7** to the **PB15** of the dexter
- **A** to the **red line** of the breadboard
- **K** to the **blue line** of the breadboard



Step 6

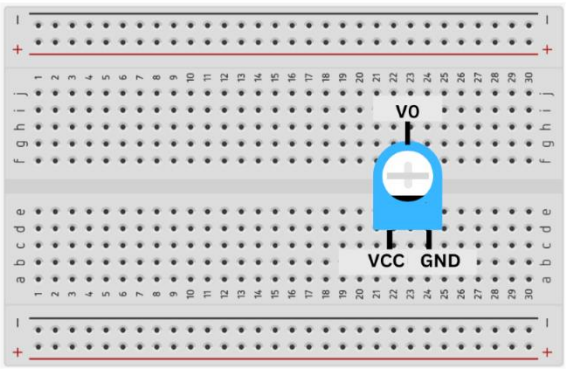
Take 1 male to male jumper wires and connect the **V0** of LCD display and GND of dexter boards as below :

- **V0 (contrast control)** to the **blue line** of the breadboard

Note: For improving the display contrast we can add a 10k potentiometer (potentiometer is not available in the kit and needs to be procured externally) as per connections shown below or solder it permanently to the display between the pins **VSS (GND), VDD (VCC) & V0.** Adjusting the potentiometer will improve the contrast.

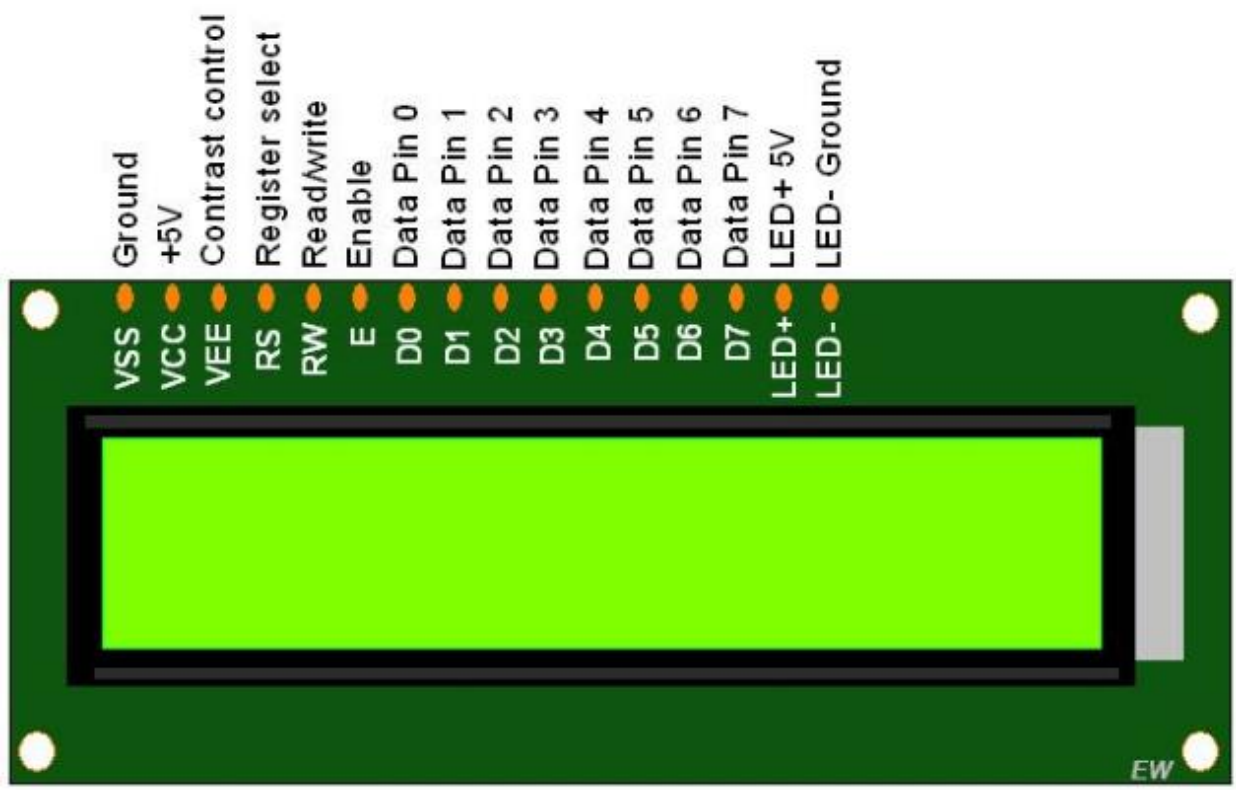
Link to purchase 10k Pot:

> https://robu.in/product/rm065-10k-ohm-trimpot-trimmer-potentiometer-pack-of-10/?gclid=CjwKCAjwpqCZBhAbEiwAa7pXeTvc2ayxgG4xkVVJmMQ3tDy-



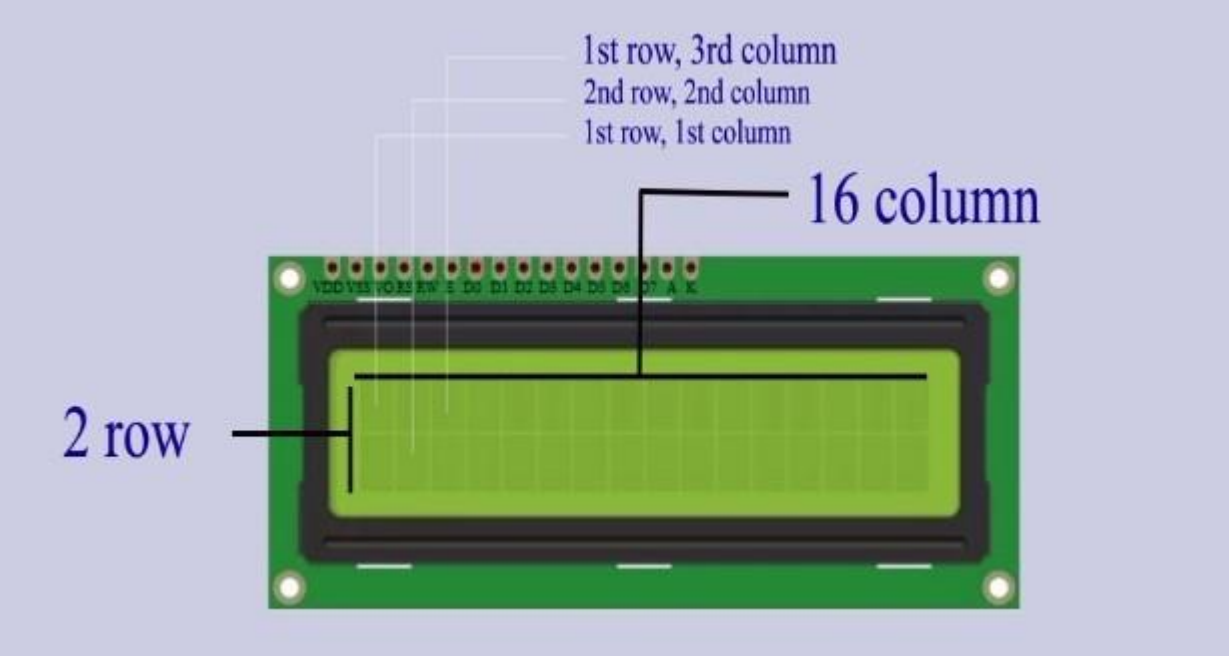
E. Part 1: Experimenting with the components

1. 16*2 LCD Display



LCD (Liquid Crystal Display) is a type of flat panel display which uses liquid crystals for displaying information. It consists an array of small segments called pixels, which can be manipulated for displaying status or parameters.

A 16*2 LCD display can display 16 characters or numbers in column wise and 2 in row wise.



The LCD display have a total of 16 pins

The use of pins listed below:

1. **GND (VSS)**

Connect the ground pin of the power supply to this pin.

2. **VCC (VDD)**

   Connect this pin to 5v

3. **Contrast (VEE)(V0)**

   This pin is used to adjust the contrast of Display. Connect a potentiometer (POT) to this pin. Rotate the knob of the POT to adjust the contrast.

4. **RS**

   RS pin means Register select pin. Selects command register when the pin is LOW. Selects data register when this pin is HIGH. Command register and data registers are used to send LCD command and data to LCD

5. **RW**

   It represents the Read Write pin. When this pin is LOW, the MCU write to register. And when the pin is HIGH, MCU

read from the register. Here we want to write. Connect it permanently to GND.

6. **EN (E)**
   EN pin means the Enable pin. Send data to data pins when a HIGH to LOW pulse is given.
7. **D0-D7 (DB0-DB7)**
   These are 8 data pins. Here I interface this LCD with dexter board in 4 bit mode. So we need only D4 to D7.
8. **Backlight(+)**
   This is the anode pin of the backlight of the display
9. **Backlight(-)**
   This is the cathode pin of the backlight of the display

Note:

There are two different modes of operation of LCD which can be controlled by our Control Pins.

4-bit and 8-bit Mode of LCD:

The LCD can work in two different modes, namely the 4-bit mode and the 8-bit mode. In **4 bit mode** we send the data nibble by nibble, first upper nibble and then lower nibble. (**nibble** is a group of four bits), so the lower four bits (D0-D3) of a byte form the lower nibble while the upper four bits (D4-D7) of a byte form the higher nibble. This will help us to send 8 bit data.

Whereas **in 8 bit mode** we can send the 8-bit data directly in one stroke since we use all the 8 data lines.

8-bit mode is faster than 4-bit mode of data transfer. But we will run out of I/O pins on our MCU, so 4-bit mode is widely used

For detailed theory refer this link

> https://circuitdigest.com/article/16x2-lcd-display-module-pinout-datasheet

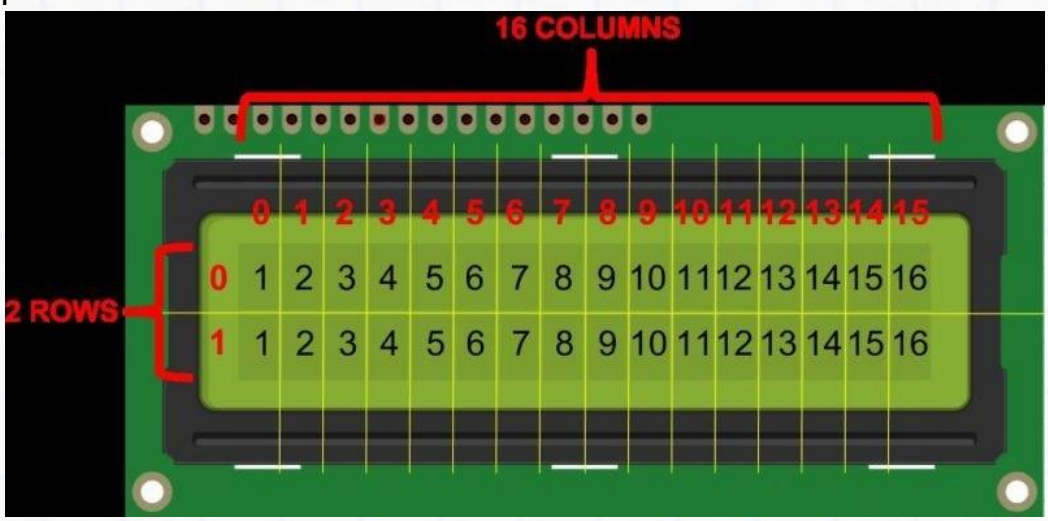
Variables & Functions

1. **lcd_initialise()**
   This function initializes the LCD display by sending some preset commands which configures the LCD and also sets it in 4 bit data mode
2. **lcd_clear()**
   Clears the display, which enables to write new character or numbers
3. **lcd_put_cur(**x**,** x**)**
   After clearing the display this function can be used to change the position of cursor to write a character in a particular position

For example:

lcd_put_cur(1, 15);

This function puts the character at 2nd row and 16 column of the lcd display, after which we can write the character in that position



4. **lcd_send_string ("x")**
   Writes a character or a string on the current position of cursor. Remember each block can only hold one character.

For example:

lcd_send_string ("hello")

5. **HAL_Delay(50)**
   Creates a delay in milliseconds

Activities

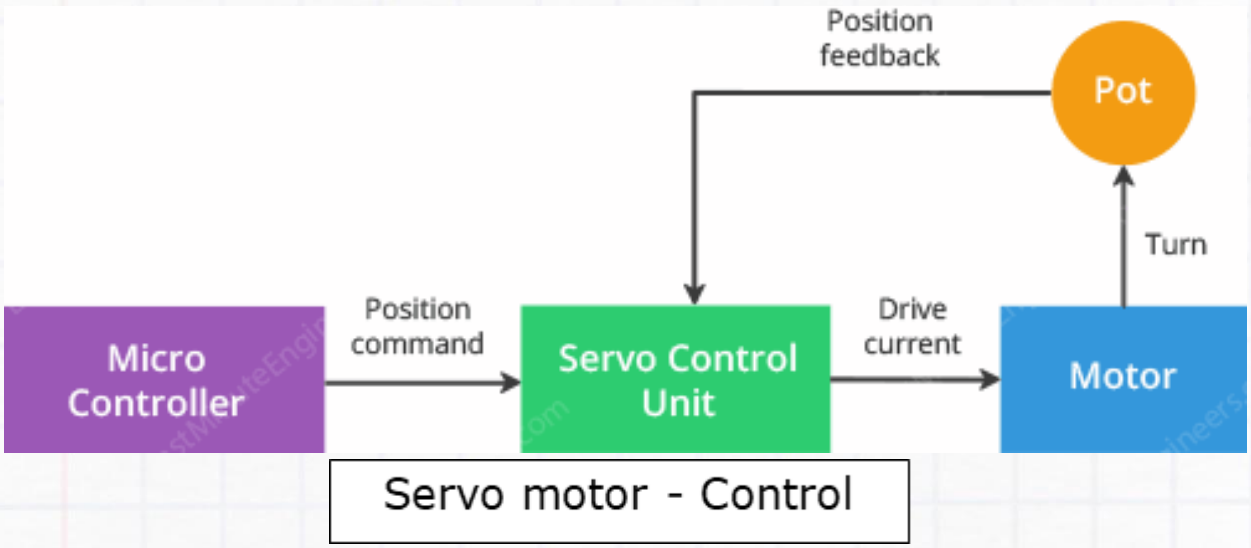
1.Write 'IIT MRP' on 1st row and 'Build Club' on 2nd row

2. Write a text to display your name in first row, college name in second row. Clear the display after a delay. Again write 'IIT MRP' and 'Build Club'. Write these texts in a loop so that it displays continuously.

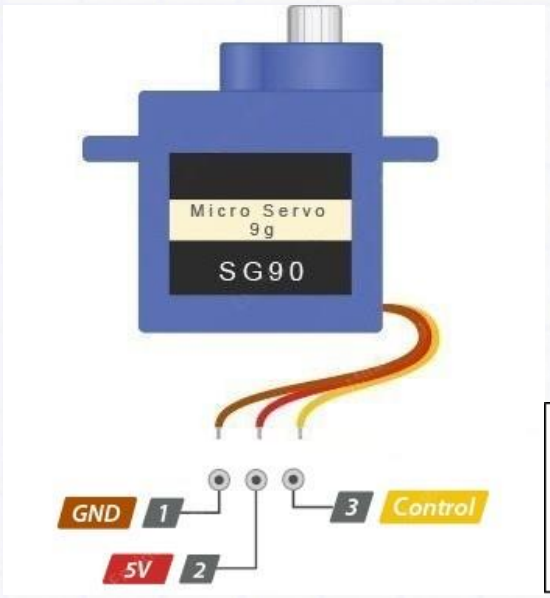
2. DC Servo motor

A servo system primarily consists of three basic components a controlled device, a output sensor, a feedback system. This is an automatic closed loop control system. Here instead of controlling a device by applying the variable input signal, the device is controlled by a feedback signal generated by comparing output signal and reference input signal. Any electrical motor can be utilized as servo motor if it is controlled by servomechanism.

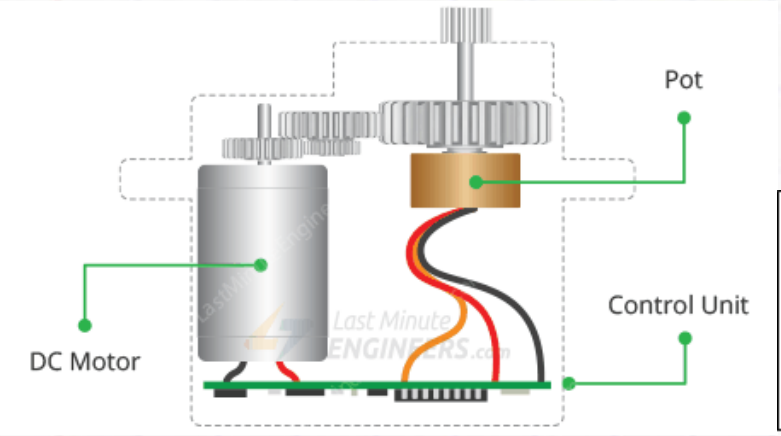
In DC servo motor, a small DC motor connected to the output shaft through the gears. The output shaft drives a servo arm and is also connected to a potentiometer (pot). The potentiometer provides position feedback to the servo control unit where the current position of the motor is compared to the target position. According to the error, the control unit corrects the actual position of the motor so that it matches the target position



Servo motor - Control



DC Servo motor - Wiring connections



DC Servo motor - Internal Illustration

Note : For a detailed understanding refer [link]

> https://lastminuteengineers.com/servo-motor-arduino-tutorial/

Functions

1. **set_servo_angle()**

This model of DC servo SG90 can rotate from 0 –180 degrees. The command rotates the servo the particular angle.

For example:

set_servo_angle(x**)**

;for x degrees of rotation

Activities

1.Write a code to rotate the servo to 0 , have a delay ,move to 45 ,have a delay, move to 90

2. Create a repeated half circle to & fro motion **ie,** a code to move from 0 to 180 degrees and from 180 to 0 . Try using a **for loop** to change the x values , inside the **while loop**

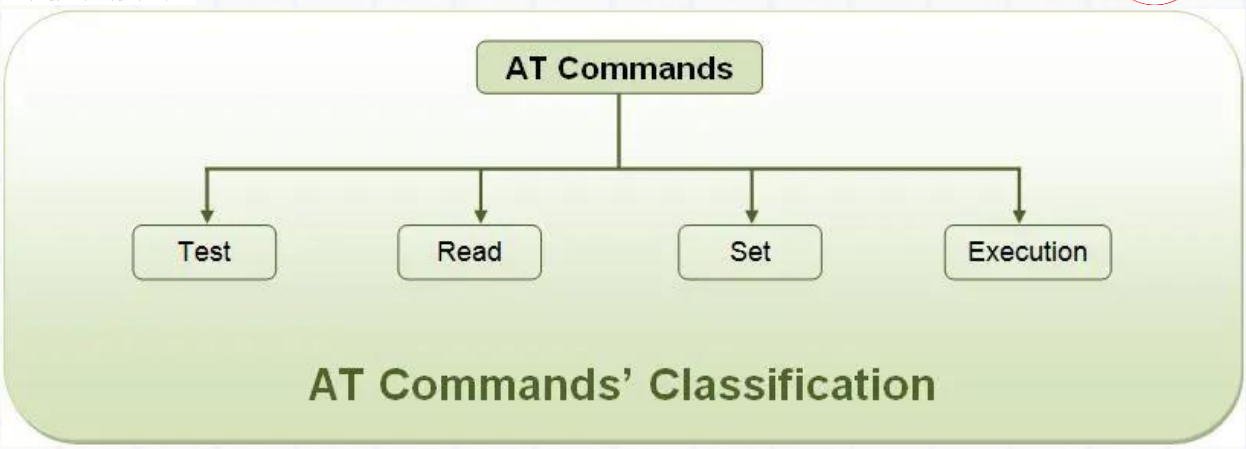
Note: If there are any loose connections/electrical disturbances/improper voltage the servo motor can behave abnormally.

3) GSM Module

A GSM modem or GSM module is a device that uses GSM mobile telephone technology to provide a wireless data link to a network. GSM modems are used in mobile telephones and other equipment that communicates with mobile telephone networks. They use SIMs to identify their device to the network

SIM900A GSM Module, which we use in this project is a small and cheap module for GPRS/GSM communication. This can work with any GSM network operator SIM card just like a mobile phone with its own unique phone number. We use this device to communicate with our web dashboard.

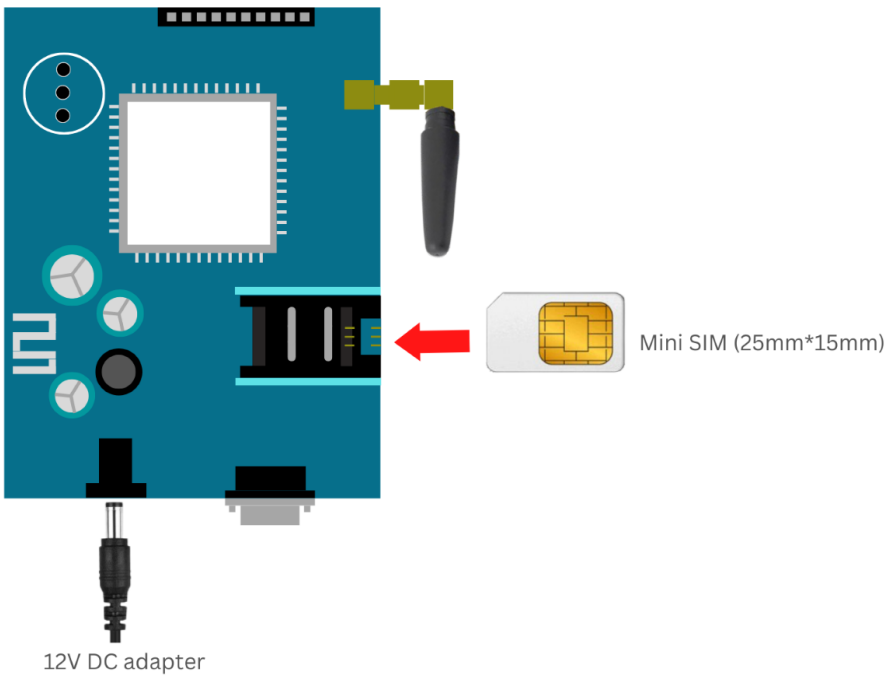
Note: AT or Attention commands are used for controlling the GSM modems. There are four types of AT commands: Test commands, Read commands, Set commands & Execution commands



For more detailed understanding refer

> https://www.engineersgarage.com/at-commands-gsm-at-command-set/

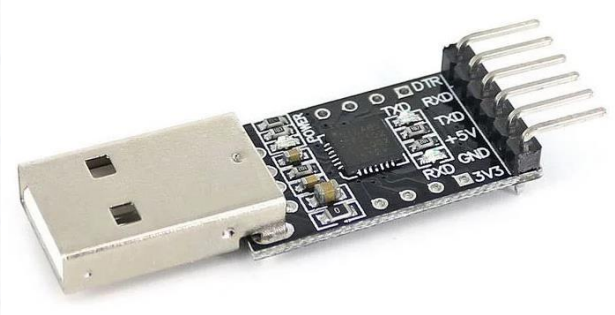
GSM Module Connections



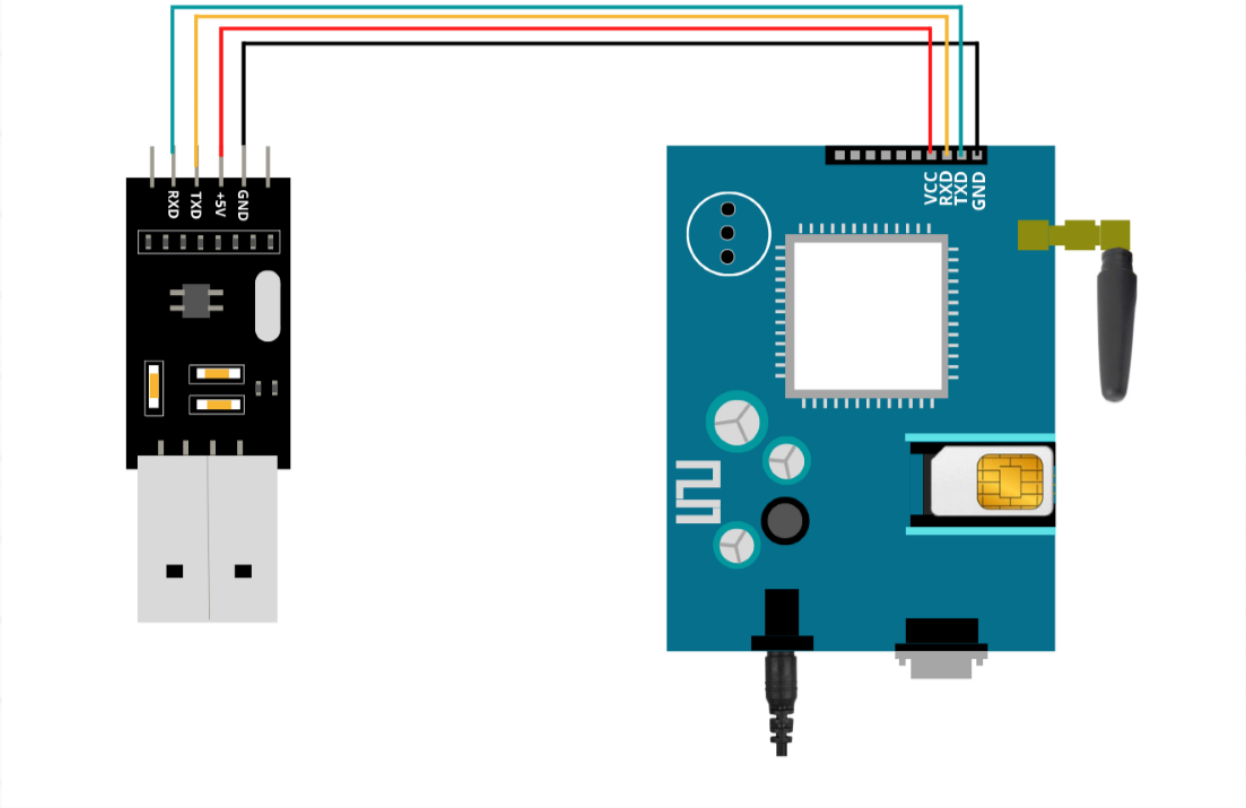
Note : Extra acitivites are not part of the main schedule. You may/maynot need to disconnect the components from existing connections for performing these activites. Remember to connect it back for executing the main activities

Extra Activity – Testing with AT commands

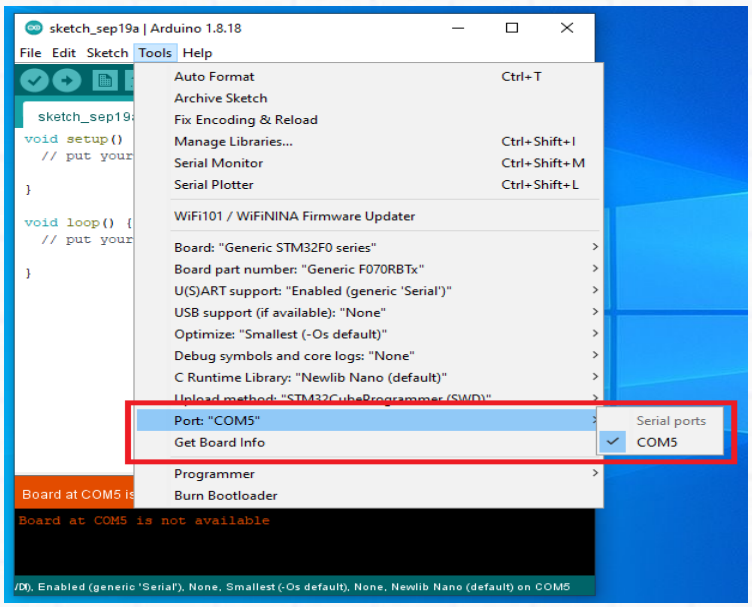
1. Here we use a USB-TTL converter to communicate with the GSM Module.



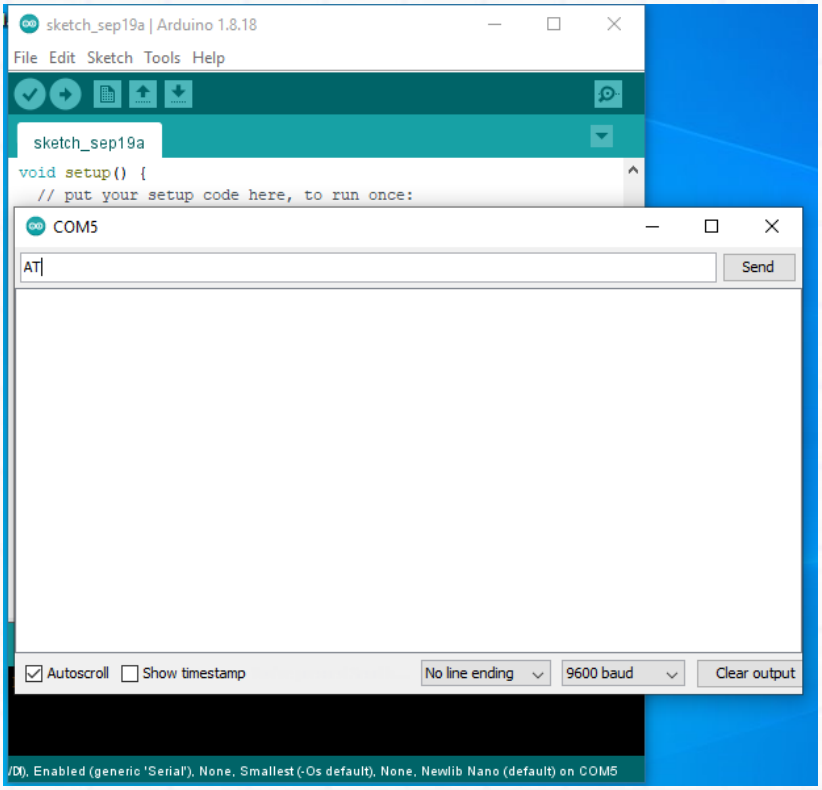
2. Connect a USB to TTL converter to the send commands to GSM module and also receive data from GSM network. Connections as below:



3. Check whether SIM card is inserted properly and USB –TTL converter is connected to PC.
4. Since you have worked on arduino IDE, open serial monitor in arduino. Click the COMx Port on which the TTL Converter is connected.



5. Type “AT” in the serial monitor. Press ‘send’



6. Check whether “OK” is getting as output

Note: **AT** command is used to check communication between the module and the computer. If its displaying “OK”, then it indicates there is proper communication.

7. Type ATD and ‘phone number’ and press ‘send’ to call to a phone number

For example)

Type **ATD9976390289**

This ‘9976390289’ will be the phone number

Note: ATD command is used to dial a number.

8. Ring to the GSM sim that you have placed in GSM module and check the output in serial monitor. It should show “Ringing” as output.

Note : Use SIM that supports GSM network with messaging balance

List of network providers:

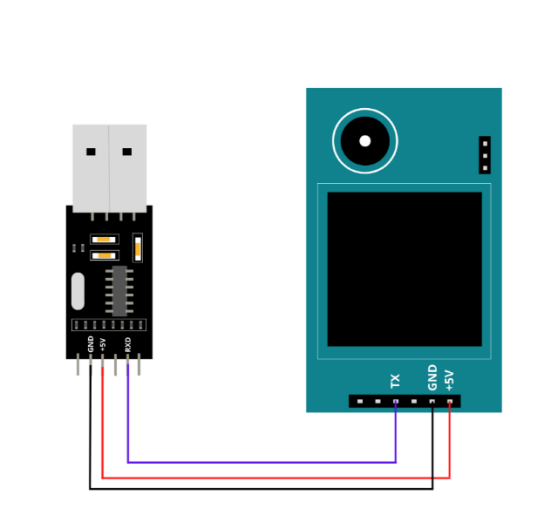
1) Airtel
2) Vodafone
3) BSNL

4) RFID Card reader Module

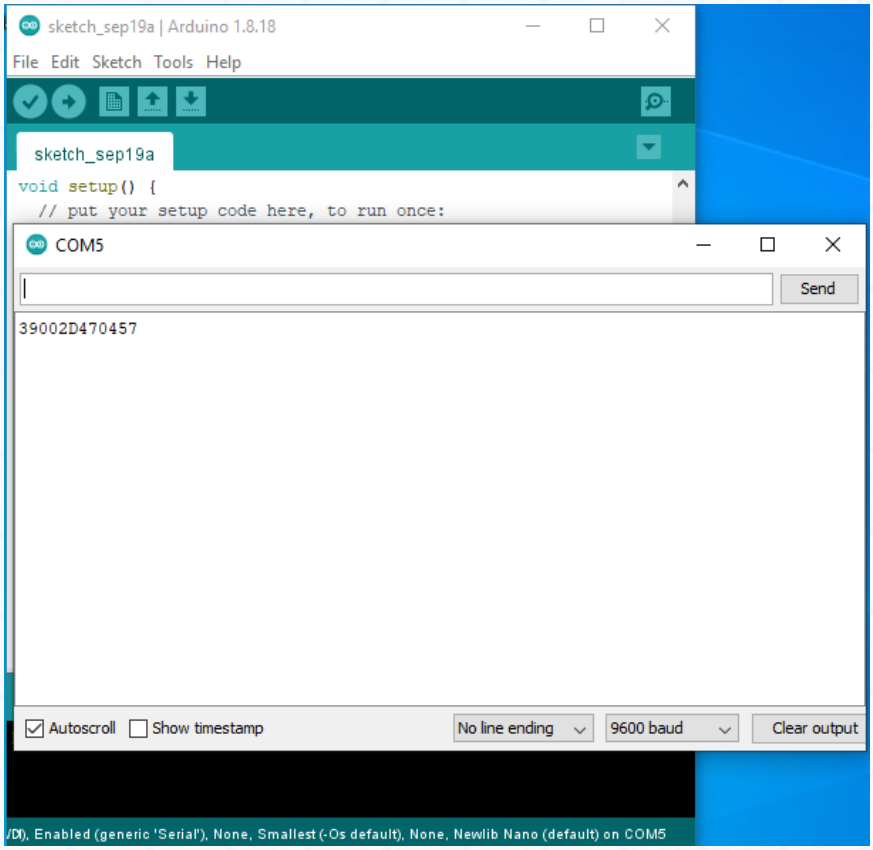
A radio frequency identification reader (RFID reader) module is a device used to gather information from an RFID tag, which is used to track individual tags/cards. Radio waves are used to transfer data from the tag to a reader. RFID is a technology similar to bar codes.

Extra Activity– Reading RFID values

1. Connect a USB to TTL converter to RFID card reader as below:



2. Show a card to the reader and check the 12 digit RFID code appearing on Serial Monitor



Note: Each RFID cards will have unique values

F. Part 2 : Implementing the Fast tag reader project

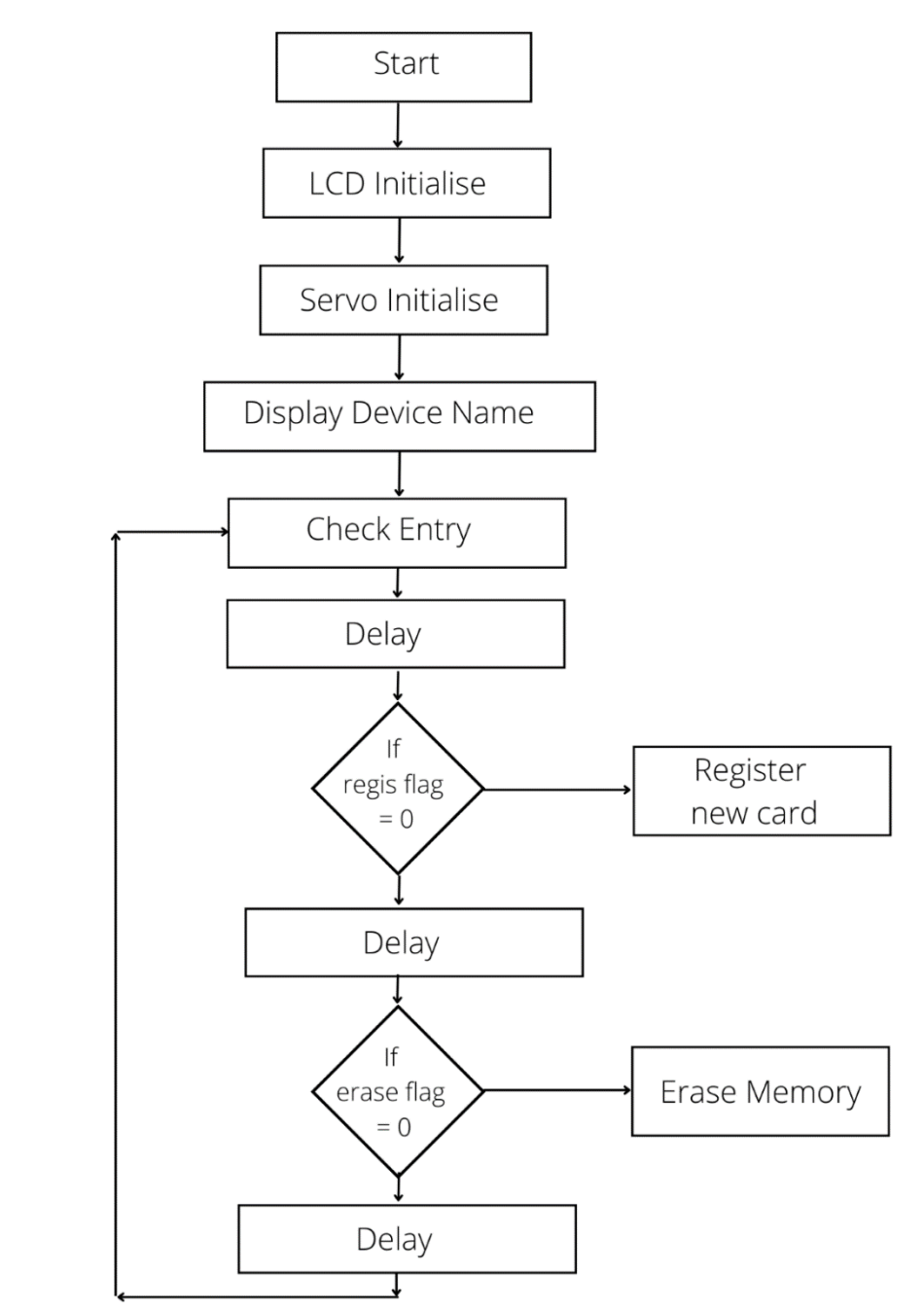
Downloads & Installation

1. Download the Project Workspace file **RFID_FastTag_Reader.zip**' given in the project page on the Build Club website.
2. In the **Workspace** folder in your C: drive, create a new folder named '**RFID_FastTag_Reader**'.
3. Then i) Launch the STM IDE, ii) Select the **RFID_FastTag_Reader** folder as workspace, iii) Import the ZIP file **RFID_FastTag_Reader.zip**, iv) Navigate to **app.c**

NOTE: Before getting into the software for the project please go through the **Concept** section at the start of the manual, where we explain how this project works.

Flowchart of the Code

The flowchart diagram represents the flow of the RFID Reader project. Along with the flowchart the program part can be easily implemented.



Variables & Functions

Variables & Functions

Note : The functions below are derived from functions mentioned in part 1

1. **lcd_initialise()**

This function calls the lcd_init() & lcd_clear() functions defined in part 1 of this manual which initializes the lcd parameters and clears the existing values in display

2. **servo_initialise()**

Includes set_servo_angle() function to set the angle to zero, so that the gate is closed initially

3. **lcd_display_device_name()**

This function displays “FAST TAG READER” on the display using lcd_put_cur(), lcd_send_string(), lcd_put_cur()

4. **check_entry()**

This Function is used to check for card entries continuously, which verifies the card data with the data stored in the FRAM memory. The gate (servo motor) is closed/opened based on the verification. This function also sends the updates to LCD display & to the thingspeak server

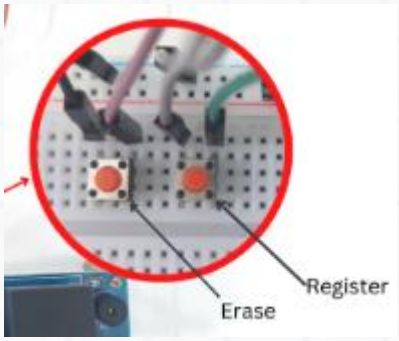
5. **regis_flag**

This is a flag register that checks and tells whether a new card needs to be registered to the memory. It is connected to a push button to receive user input to add a new RFID card

6. **erase_flag**

This is also a flag register that tells the dexter board to erase the card data stored in the FRAM memory. It is connected to a push button to receive user input to add a new RFID card

Note : regis_flag & erase_flag is updated from external interrupts in dexter board, ie from the button clicks(**erase** & **register**)



7. **register_newcard()**

This function sets the device to 'new card registration mode'. It is executed once the **regis_flag** becomes '1'.
The new card data is stored to FRAM memory and the card registry is updated

8**. erase_cardmem()**
Erases the complete card data stored in the FRAM memory. The status is shown in the LCD Display

Implementing the Code

Refer the flowchart and functions above to implement the code in provided space in the **main.c** function. Follow the steps below:

1. Initialize LCD and servo motor using **lcd_initialise()** and **servo_initialise()** functions
2. Display the device name with

lcd_ display_device_name()

3. The rest of the code should enter an infinite while loop

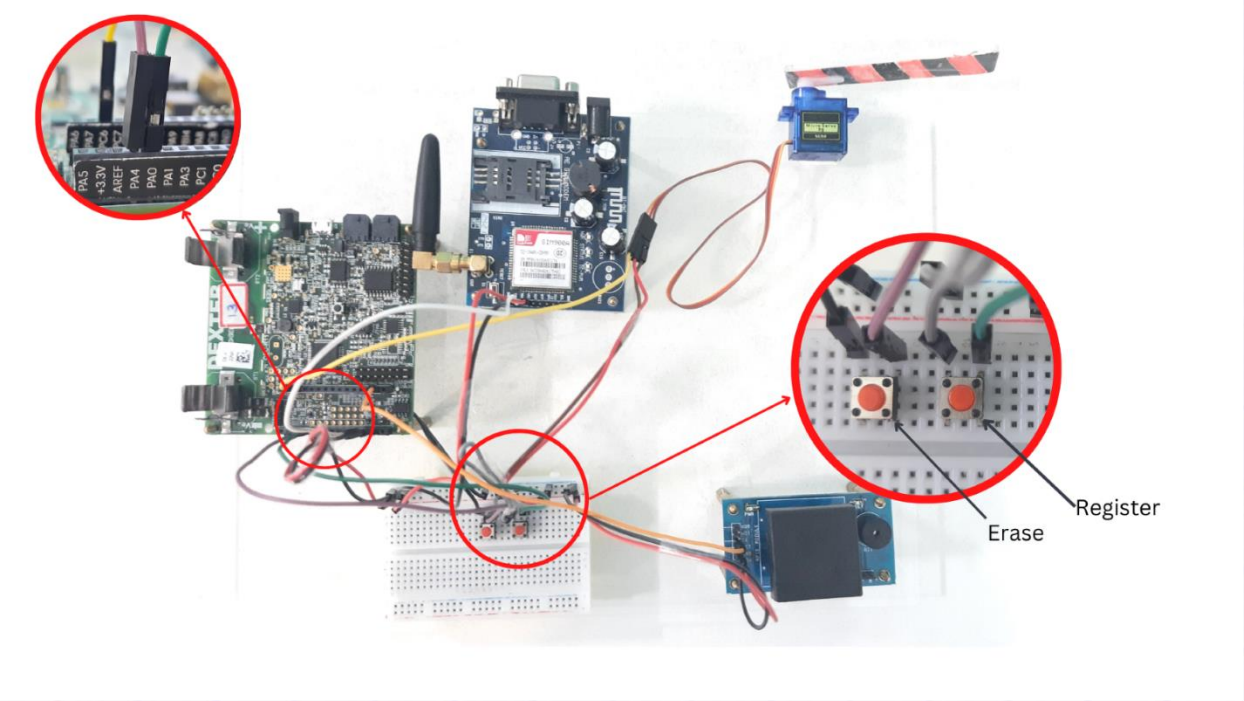
```
while(1)
{
    // Rest of code
}
```

4. Continuously check for card entries using **check_entry()** function. Add a 50ms delay so data is read properly
5. Is **regis_flag** equal to 1. Then execute the function for ‘new registration card’. Add delay after the check
6. Is **erase_flag** equal to 1. Then execute the function for ‘memory erase’. Add delay.

Implement these code commands in the **main.c** function in the order specified

After writing these codes in STM32 IDE. The code can now be uploaded to the Dexter board by hitting ‘Run’.

7. Try adding new cards by pressing the push button (register button), also try erasing the data (erase button). Check whether access is granted for registered cards and gate is opened.

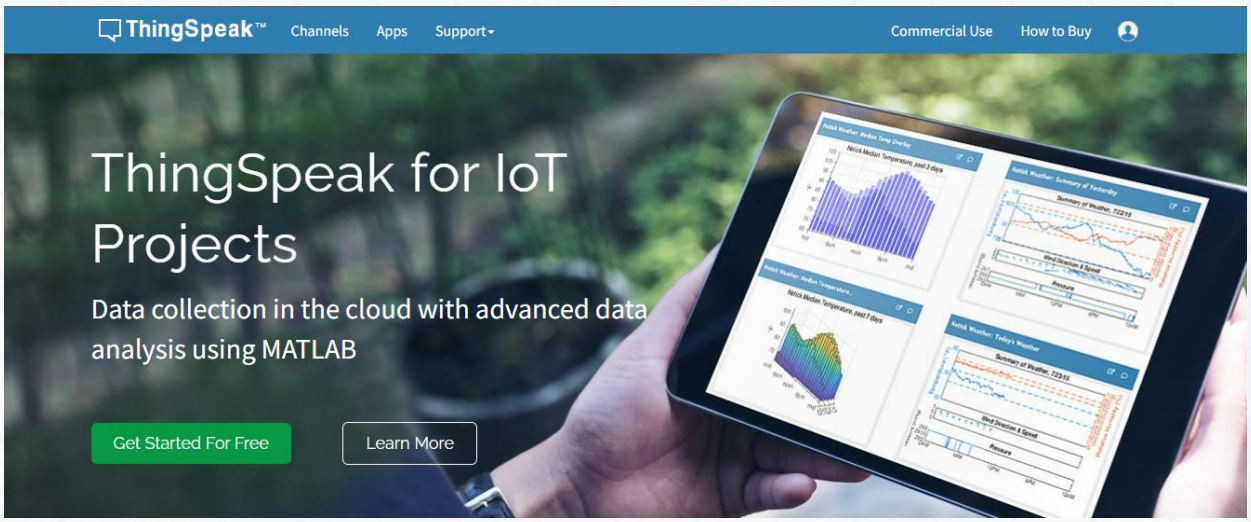


Now we have completed adding, verifying card data and also operating gates.

Next step is to check whether the entry data is send to the **online dashboard** - **thingspeak**

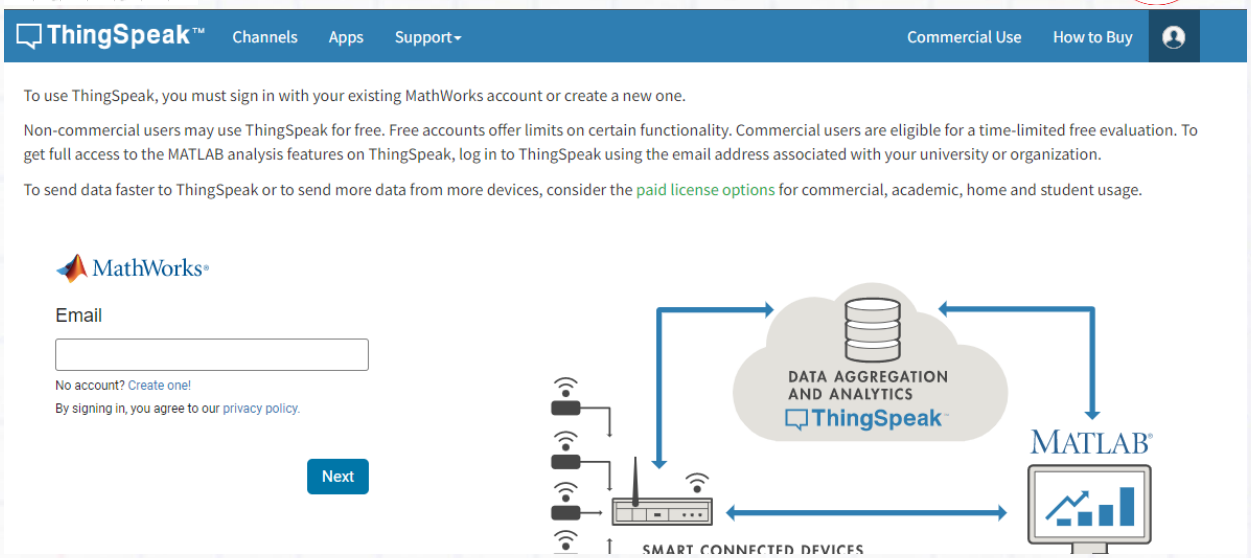
Setting up web dashboard – thingspeak

In previous projects we had used ubidots web platform, this time we will try thingspeak. ThingSpeak is an IoT analytics platform service that allows you to aggregate, visualize, and analyze live data streams in the cloud.

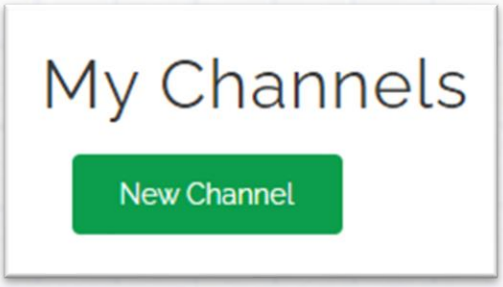


Configuration steps

1. Visit https://thingspeak.com/ website.
2. Create an account



3. Create a new channel in ‘My channels’

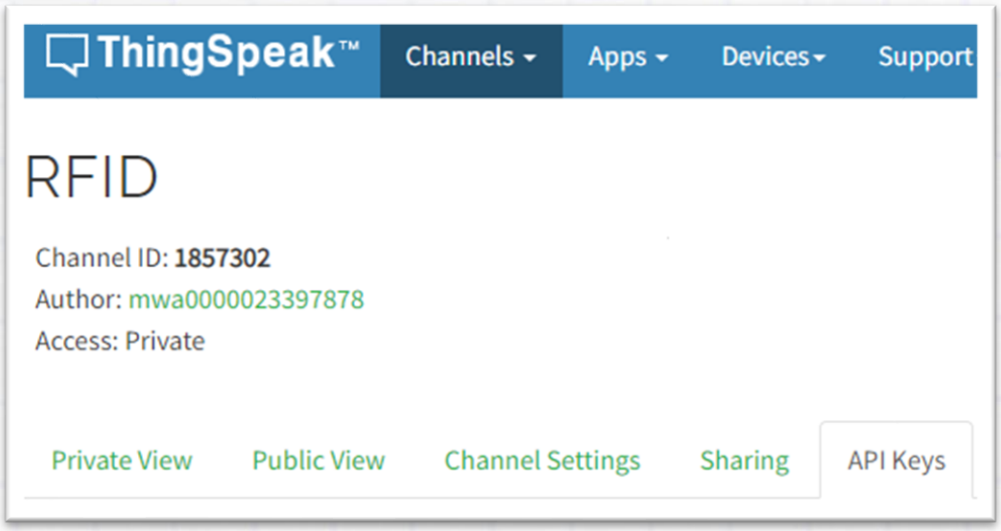


4. Name the channel as you may prefer like RFID. Tick the fields Field1,Field2,Field3,Field4 since we intent to fill the fields with following data

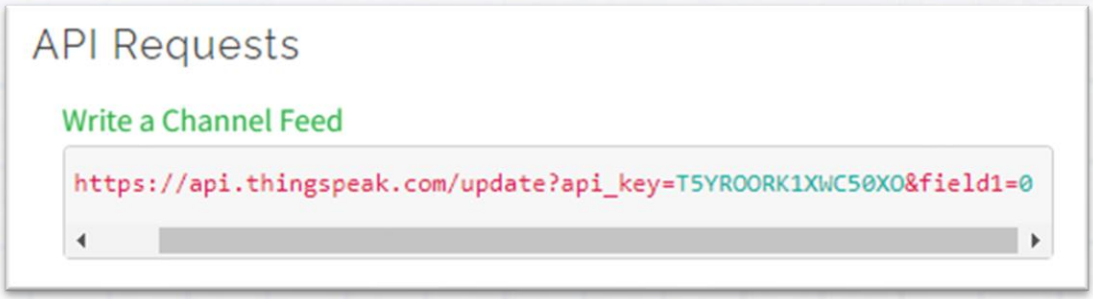
Field1=No of ID’s registered
Field2= Total no. of times card entries
Field3 = Total no. of access granted
Field4 = Total no. of access denied

5. Now we have to link this thingspeak dashboard with our dexter code.

6. Go to ‘API keys’ section in RFID channel



7. Copy the whole text in ‘Write a channel feed’ section



Note : Remember to copy the entire text except ‘0’ in the end

For example

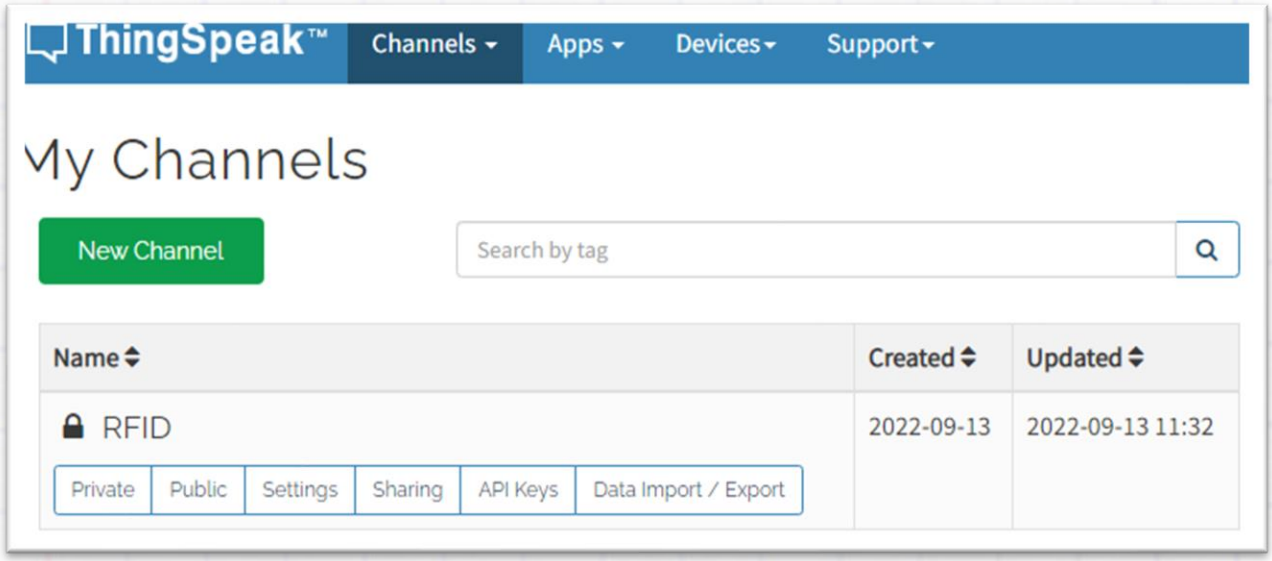
GET https://api.thingspeak.com/update?api_key=T5YROORK1XWC50XO&field1=

8. Replace **char stri[]** in **App.c** with this code.

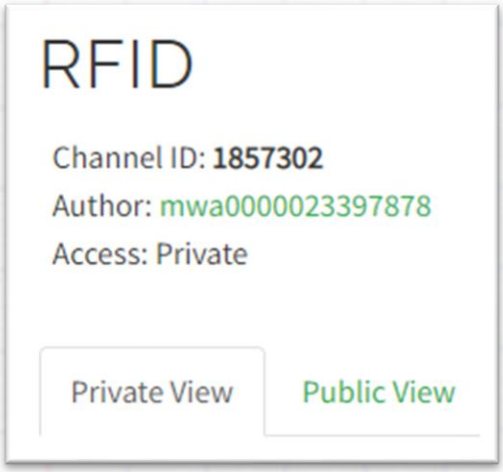
9. Run the code in dexter board

Note : The code run on dexter board will automatically fill the 4 fields with data

10. Now go to your ‘RFID’ channel dashboard



11. Click RFID channel and navigate to ‘private view’



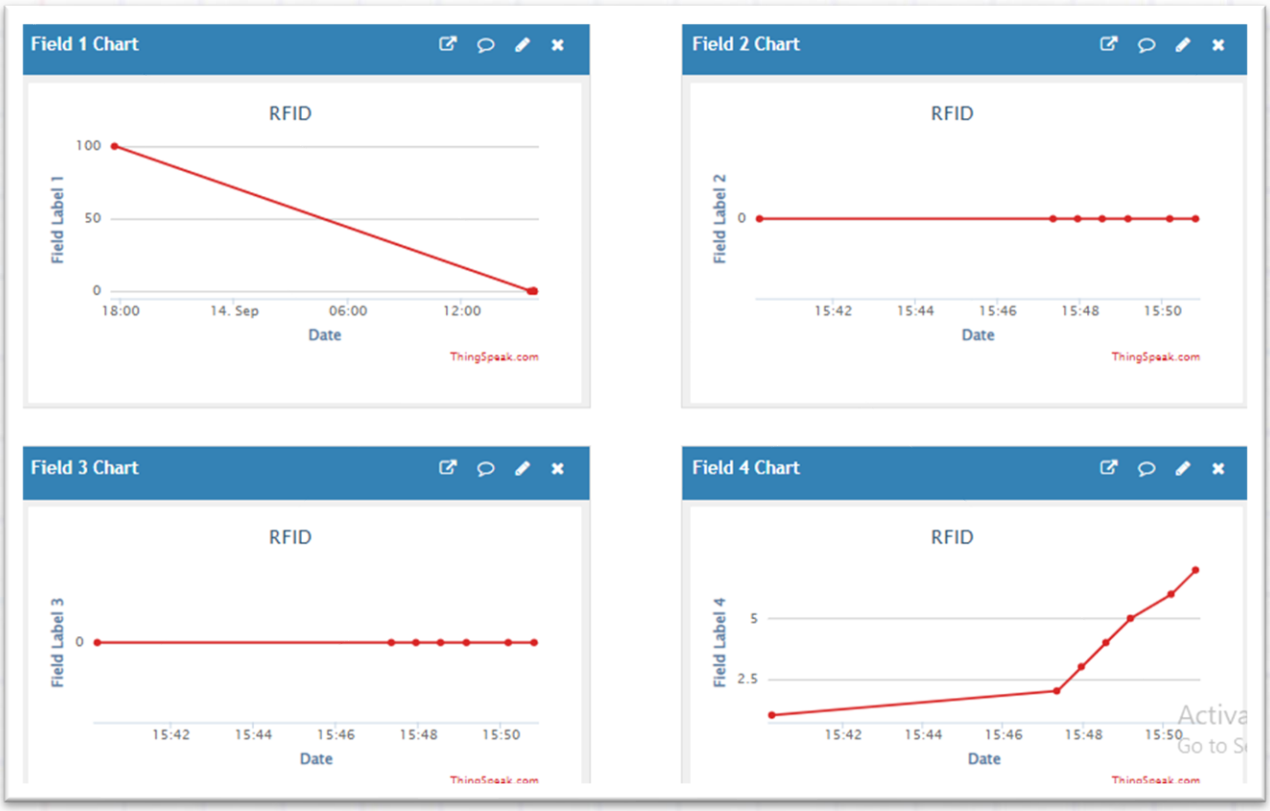
12. Now it’s time to receive data to the web dashboard.

13. Power up the GSM module (with 12v adapter). Remember to insert GSM sim in the slot

14. Check whether all other components like RFID reader, LCD display servo motor, dexter board are powered up properly.

15. Try using different RFID cards, register a card. Check whether servomotor is working and entry is permitted. Try with non-registered cards, check whether entry is denied.

16. Now visit the thingspeak dashboard. You will see the fields are getting populated



17. Project Finished 😊

Task

1. After erasing the current memory. Take 3 new RFID cards , add 2 cards. Keep one card unregistered. Check how the device is behaving with the 3 cards. Check whether the data is received on Web dashboard.
2. Make a video presenting, the build of RFID reader, how you completed the activities and challenges. Share it with the Build Club Community on the Discord Server!