

CavyIoT Dev-Board:

CavyIoT.nodemcuV0.03.bin

Firmware is released as free open source, for the developers by **Cavy Agrotronics**. This firmware is useful to engineering students, teachers, hobbyist, enthusiast, programmers, for availing CavyIoT services. This firmware is free for all who want to add the power of IoT in their product.

It is language independent, means no matters what programming language the user is using in his project. User can interface **CavyIoT Development Board** with simple **Send-Response protocol**.

Flashing firmware converts IC ESP8266 into **CavyIoT Development Board (CavyIoT-DB)**. Developer of any community can develop and deploy a powerful IoT system/application to the end user. Simple interfacing API architecture makes easier for developers in development process .

Features of our platform and development board-

Hardware:

- Platform independant
- Simple API
- Settings for data usage
- Local data backup

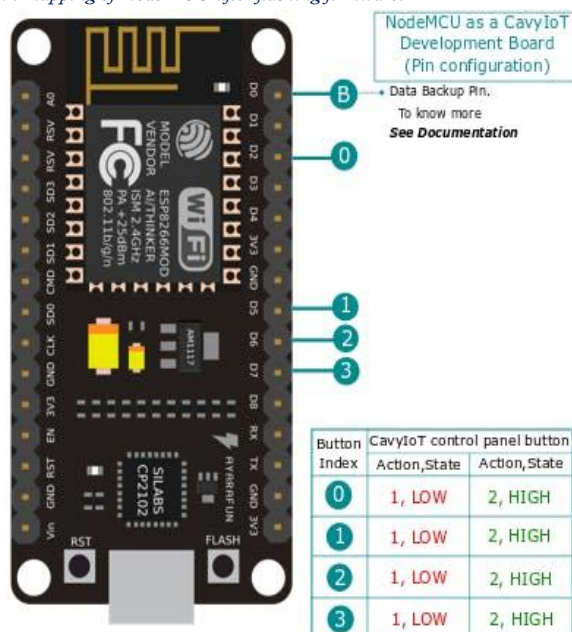
Software:

The CavyIoT service provides the subscriber access to **CavyIoT-DB** all over the internet anytime/anywhere via **Remote Control Panel**. Remote Control Panel is standard GUI software accessible over internet using standard internet browser.

Features of Remote Control Panel.

- Data visualization
- Report generation
- Full control over system
- Bi-directional & Synchronized with Dev-board
- Buttons to set the output of board's output pins
- Automation
- Log file of all operations and device status
- Triggers
- Multimaster Control Panel
- Automatic and manual mode changing for operation
- Live charts and Gauges
- Minimum Data consumption and maximum utility

Pin mapping of NodeMCU after flashing firmware.



Cavylot-DB's baud rate is 9600 and is fixed for serial communication (via rx , tx pins). Cavylot-DB recognize a line of Instruction when is send within enclosed chevrons followed by a carriage return. Device sends request to the Cavylot server via internet, in defined interval to get the status of remote control panel, for an example if the user clicks the button on control panel, the device will receive control signal issued in the next ping (request). The pinging keeps the device in synchronization with the remote control panel. Default ping interval is 5 sec, setting ping interval to higher values slows the response time of device and vice versa . And sensor data sent to Cavylot-DB is get displayed on remote control panel for monitoring and processing.

The output pins of **Cavylot-DB** shown above are responsive to the corresponding Buttons on the **Remote control panel**. The frequency of sending sensor data to server and the frequency of receiving control signal from remote control panel of device, depends upon **ping interval** and it is configurable. **ping interval** user can be set via programming device or from the remote control panel.

* **Local data backup:** **Cavylot-DB** has a many unique **Features** , local data backup is the most important as it is useful or data analysis, And for tracing the track of operation done. It keeps the track of all operations like it records the data , by which operation the state of output pins changed. As discussed above output pins of **Cavylot-DB** shown above are responsive to the corresponding Buttons on the **Remote control panel**. If the button on remote control panel is clicked, state of output pin changes or if user switches operation mode ie from "AUTO" to "MANUAL" .This change in operation mode or the change in state of output pin generates an event to write record to the end of **log file**. Even if any such event does not occurs the device record the state of device in predefined interval. **Recording interval** is configurable and can be set via programming device, or can be set easily from the remote control panel. Log file is log.csv as it is well known format. It keeps up to 500 record after reaching 500th record this log.csv file transfers all data to log_bac.csv . Thus user get upto 501 to 999 records for tracking operations and analysis of data.This data file is available for download any time.

How To download data from device : Keep the data-backup pin '**B**' **LOW** and restart the device.This will create a Wi-Fi hotspot to serve a page to download data log file. Browse to url ***http://100.100.100.100*** where the page with link for download will be available by ***local Wi-Fi web server*** of device. The default Wi-Fi ssid is "**Cavylot**" and password "**admin@123**". User can set Hotspot parameters (**SSID**, **password**) for his privilege.

Interfacing

(**Cavylot-DB** is abbreviated as 'device' for our convenience)

The basic interface is to send device a line of Instruction within enclosed chevrons as **<ShowDemo (WifiSSID,WifiPassword,cavylot-username,cavylot-password,device)>** then wait for the proper **response message** with open chevrons (>) followed by **ok!** or **error!**("**>ok!**" Or "**>Error!**") . This signals device has completed the parsing and executed the command. At times, Cavylot-DB may not respond immediately. This happens when device is busy doing something else mostly happened with **Show Demo ()** .

The **response messages** are strings terminated by a return. These messages are "pushed" from device to the user in response to a query or instruction to let the user what happened.

The primary way to talk to device is performed by sending it a command string of characters (Instruction within enclosed chevrons"<>"), followed by a carriage return. Device will then process the string and then reply back with a **response message**, also terminated by a return, to tell you how it went.

Writing an Interface for Cavylot-DB:

Streaming Protocol: Simple Send-Response [Recommended]

The send-response streaming protocol is the most fool-proof and simplest method to stream a command to device. The host PC/ MCU interface simply sends a line of command to Cavylot-DB and waits for an **Ok!** or **Error!** Response message before sending the next line of command.

So, no matter if device needs to wait for room in the look-ahead planner buffer to finish parsing and executing the last line of command or if the the host computer is busy doing something, this guarantees both to the host PC /MCU and device, the programmed command has been sent and received properly.

(Note : All commands for Cavylot-DB are **not Case Sensitive!**)

See the [Commands](#) document to see what they are and how they work.

1 . **DefineButtonLabels**(Label (0), Action1, Action2,Label (1), Action1, Action2,Label (2), Action1, Action2,Label (3), Action1, Action2)

Description: Sets the Label for Button(index) and Labels for the its actions

Button when is in Action1 OUTPUT Pin(index) is **LOW**

Button when is in Action2 OUTPUT Pin(index) is **HIGH**

Default:	Action1	Action2
Button(0) is LED1,	ON,	OFF
Button(1) is LED2,	ON,	OFF
Button(2) is LED3,	ON,	OFF
Button(3) is LED4,	ON,	OFF

Response message: <Buttons labelled >Ok!

2 **StartDevice**(WiFiSSID,WiFiPassword,CavlyIoT-username,password,device)

Ex: **StartDevice** (BlueSky,jki125nmio,robert555,oltrtrtv45pA2@,Demo)

Description: : Connects to the Wi-Fi and starts pinging with regular interval.

Response message:<Device Started ! Ready to send data >Ok!

And Ping response : : < *JSON String of device status* >Ok!

3. **UpdateSensorData** (Sensor1, S1_Value, S1_Unit, Sensor2, S2_Value, S2_Unit, Sensor3, S3_Value, S3_Unit, Sensor4, S4_Value, S4_Unit)

Description: Sends data to the server.

Response message : < *JSON String of device status* >Ok!

Ex:**UpdateSensorData**(Temperature,35.12,C,humidity,19.00,Rh,Speed,16.00,km/hr,pressure,4.12,Psi)

Will get response: < {"Device": "farm", "Mode": "MANUAL", "Buttons": ["OFF", "ON", "OFF", "OFF"]} >Ok!

Where,

Device is the device name.

Mode is the working mode set on remote control panel.

Buttons is the array of position corresponding to the button index on remote control panel.

4. **SetPingInterval** (Seconds)

Description: Sets the **Ping Interval**.

Response message : < *Ping Interval is set* >Ok!

Ex: **SetPingInterval** (10)

Will get response: < *Ping Interval is set* >Ok!

5. **SetRecordingInterval** (Minutes)

Description: Sets the **Recording Interval**.

Response message : < *Recording Interval is set* >Ok!

Ex: **SetRecordingInterval** (5)

Will get response: < *Recording Interval is set* >Ok!

6.**SetBackupHotspot** (Hotspot_name, Hotspot_password)

Description: Sets the Wi-Fi **Hotspot** for data backup.

Response message : < *Hotspot For BACKUP is defined* >Ok!

Ex: **SetBackupHotspot** (BlueSky, jki125nmio)

Will get response: < *Hotspot For BACKUP is defined* >Ok!

7. **ShowDemo**(WiFiSSID WiFiPassword, CavlyIoT-username,password,device)

Example:**ShowDemo**(BlueSky,jki125nmio,robert555,oltrtrtv45pA2@,Demo)

Description: Connects to the Wi-Fi and starts pinging. And returns Device Status after each ping .

Returns: : < A *JSON string of device stautus* >Ok!

Like this <{"Device": "farm", "Mode": "MANUAL", "Buttons": ["OFF", "ON", "OFF", "OFF"]} >Ok!

Library for Arduino

CavyloTdevelopmentBoard.h

Library for interfacing with **CavyloT Development Board** is available at- <https://github.com/CavyAgrotronics/CavyloTdevelopmentBoard>

. Now take a brief look on functions of **CavyloTdevelopmentBoard.h** library.

CavyloT Class variable members

String Status :-It is a **JSON String of device status**.

CavyloT Class has following **member functions**-

1. SetPort()

Description: Defines the arduino pins and Baud rate for communication with SmartKit using software serial library.

Syntax: SetPort(rx, tx, rst) where **rx** is the pin on which to receive serial data, **tx** is the pin on which to transmit serial data, **rst** is reset pin.

Example:SetPort(10,11,13);

Returns:Nothing

2. DefineButtonLables()

Description: Sets the Label for button(index) and Labels for the its actions

Button when is in Action1 OUTPUT Pin(index) is LOW

Button when is in Action2 OUTPUT Pin(index) is HIGH

Syntax:DefineButtonLables(**String** Lable (0), **String** Action1, **String** Action2, **String** Lable (1), **String** Action1, **String** Action2, **String** Lable (2), **String** Action1, **String** Action2, **String** Lable (3), **String** Action1, **String** Action2
);

Example: DefineButtonLables("LED1", "ON", "OFF",
"LED2", "ON", "OFF",
"LED3", "ON", "OFF",
"LED4", "ON", "OFF"
);

Returns: Boolean

and assigns current status of device to **Status variable**

Status={"Device":"farm", "Mode":"MANUAL", "Buttons":["OFF", "ON", "OFF", "OFF"]}

3. StartDevice(WiFiSSID,WiFiPassword,CavyloT-username,password,device)

Description: Connects to the WiFi and starts ping.

Returns: Boolean

Example: StartDevice("WiFiSSID", "WiFiPassword", "CavyloT-username", "password", "device");

4. UpdateSensorData(String Sensor1, String S1_Value, String S1_Unit, String Sensor2, String S2_Value, String S2_Unit, String Sensor3, String S3_Value, String S3_Unit, String Sensor4, String S4_Value, String S4_Unit)

Description: Updates the value of sensor on the server and update the Status variable;

Example: UpdateSensorData("Temperature", "35.12", "C",
"Humidity", "20.00", "RH",
"Speed", "40.00", "Km/hr",
"Pressure", "06.00", "Psi");

Returns: Nothing

Assigns current status of device to **Status variable**

Status={"Device":"farm", "Mode":"MANUAL", "Buttons":["OFF", "ON", "OFF", "OFF"]}

For Demo & Testing:

5. ShowDemo(*String* WiFiSSID, *String* WiFiPassword, *String* CavyIoT-username, *String* password,device)

Example:ShowDemo("BlueSky","jki125nmio","robert555","oltrtrtv45pA2@","Demo");

Description: Connects to the Wi-Fi and starts pinging.

Returns: Nothing

Ayush V khade
CEO,Cavy Agtronics
Akola –MS 444005
India