APPLICATION OF IOT AUTOMATION IN DIFFERENT ENGINEERING FIELDS

ABSTRACT

Everyone has their own definition of the term Internet of Things (IoT); I assume it to be the technology that provides us the ability to bring life into that plain old circuitry so that they can have a smart and simple way to communicate among themselves and with users. Just imagine the fact that we can have real time chat with our home appliances or that good old dc motor over social network from anywhere in the world. The devices can also communicate about their present status with any Artificial Intelligence (AI) of our choice. Let us create a world where not only humans but devices also have their own social network platform where we can command them to perform certain action just the similar way we would ask a person to do a favour and monitor them with more efficiency. In this paper I am proposing a prototype which can be controlled over the internet through Artificial Intelligence of social network like Facebook (Chatbot) and Google (Google Assistant), if required we can also switch our choice of AI over the variety available. The paper also brings an approach towards bridging the gap between social networks (SN) and Internet of Things which enables the connection of people towards ubiquitous automation universe based on Social Internet of Things (SIoT).

TABLE OF CONTENTS

LIST OF FIGURES

INTRODUCTION

Internet of Things has been the hot topic for the last decade. The ability to control physical devices over the internet and monitor sensor values with live feed from anywhere in the world gives us that simplicity, transparency, efficiency and security that is required in both home and industrial automation that our present system lacks. Nowadays, as sensing, actuation, communication and control become even more sophisticated and ubiquitous; there is a significant overlap in the field of IoT, sometimes from slightly different perspectives. The gadgets would weave themselves into the fabric of our everyday life to support us in carrying out daily life activities, tasks and rituals in an easy, natural way using information and intelligence, hidden in the network connecting the gadgets.

With the advent of smart phones, more people are connected with social network, so why not shift the control of devices to a simple chat on Facebook or just speak "OK Google, run motor clockwise", to perform the task.

In this paper, for the prototype demonstrated, Adafruit MQTT broker [1] is used for communication among devices, IFTTT (If This Then That) [2] service is used to configure our AI and Chatfuel [3] is used to create a simple Chatbot. The system developed is also the cheapest solution of its kind as it is build on all open source platforms. For microcontroller NodeMCU [4] is used which have an inbuilt Wi-Fi support and Arduino IDE is used for programming the NodeMCU.

The prototype demonstrated has three modules:

Module 1: Switching home appliances.

Module 2: Control speed and rotation of a DC motor.

Module 3: Live feed of sensor data.

Section II of this paper contains literature survey where a brief about the papers I have studied to write this article is presented. The next section is methodology where I have described in detail the process used in developing the prototype with configuration pictures and circuit diagram. It also contains separate discussion on the AI used, communication protocol, and hardware and software implementation. Section IV contains the result that we can achieve after successful functioning of the prototype. The next section is about the future scope where I have discussed about Social Network and BlockChain implementation into IoT, and the potential they hold for future connected devices.

LITERATURE SURVEY

R.Piyare [7] and team has introduced design and implementation of a low cost, flexible and wireless solution to the home automation using Bluetooth. But, Bluetooth has range limitation.

Deepali Javale [8] and team proposed Home Automation using GSM and Arduino which is costlier than the system proposed in this paper and is also based on platform dependent technology.

Chandrappa D N [9] and team implemented a similar concept using MQTT protocol but the paper lacks any direct discussion on using Artificial Intelligence in IoT Automation.

Prof. Niranjan M [10] and team illustrated an IoT based Industrial Automation scenario describing the use of sensors and actuators that can automate the process with minimum human intervention and the process can be monitored live over the internet.

Ms. C. Hemalatha [11] and team proposed the control of Brushless DC (BLDC) motor using Arduino over the internet, but the system lacks speed and rotation control facility.

Antonio M. Ortiz [12] and team beautifully described the concept of SIoT starting from a general architecture description and going to a deep review of the challenges and open research issues that must be solved to make SIoT a reality.

What is Node MCU?

NodeMCU is an open source IoT platform It includes firmware which runs on the ESP8266 Wi-Fi SoC from Espressif Systems, and hardware which is based on the ESP-12 module. The term "NodeMCU" by default refers to the firmware rather than the development kits. The firmware uses the Lua scripting language. It is based on the eLua project, and built on the Espressif Non-OS SDK for ESP8266. It uses many open source projects, such as lua-cjson, and spiffs.

As Arduino.cc began developing new MCU boards based on non-AVR processors like the ARM/SAM MCU and used in the Arduino Due, they needed to modify the Arduino IDE so that it would be relatively easy to change the IDE to support alternate tool chains to allow Arduino C/C++ to be compiled down to these new processors. They did this with the introduction of the Board Manager and the SAM Core. A "core" is the collection of software components required by the Board Manager and the Arduino IDE to compile an Arduino C/C++ source file down to the target MCU's machine language. Some creative ESP8266 enthusiasts have developed an Arduino core for the ESP8266 WiFi SoC that is available at the GitHub ESP8266 Core webpage. This is what is popularly called the "ESP8266 Core for the Arduino IDE" and it has become one of the leading software development platforms for the various ESP8266 based modules and development boards, including NodeMCUs.

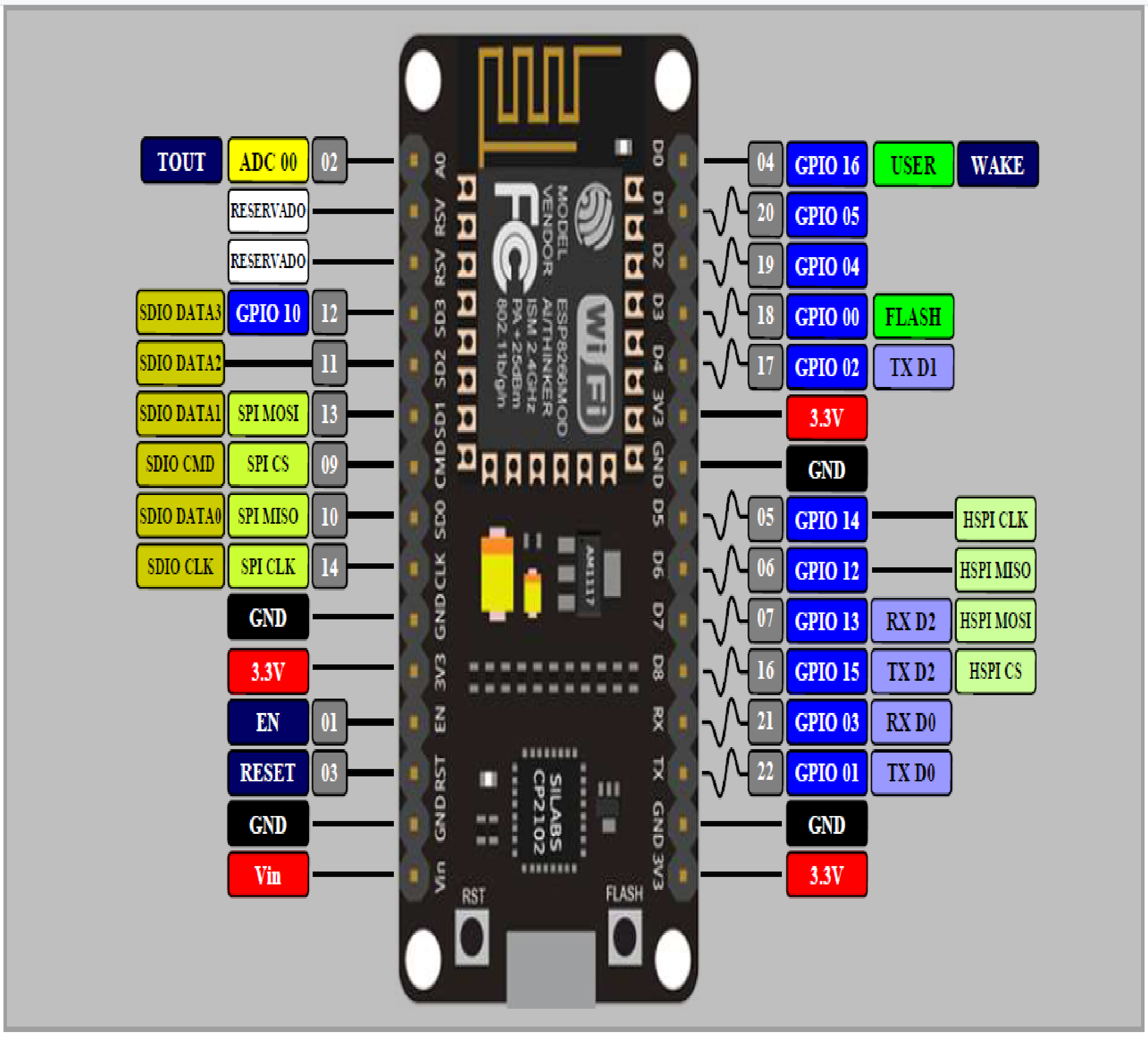
IO index	ESP8266 pin	IO index	ESP8266 pin
0 [*]	GPIO16	7	GPIO13
1	GPIO5	8	GPIO15

2	GPIO4	9	GPIO3
3	GPIO0	10	GPIO1
4	GPIO2	11	GPIO9
5	GPIO14	12	GPIO10
6	GPIO12		

Connection to MQTT broker

```
-- init mqtt client with keepalive timer 120sec
m = mqtt.Client("clientid", 120, "user", "password")

-- setup Last Will and Testament (optional)
-- Broker will publish a message with qos = 0, retain = 0,
data = "offline"
-- to topic "/lwt" if client don't send keepalive packet
m:lwt("/lwt", "offline", 0, 0)

m:on("connect", function(con) print ("connected") end)
m:on("offline", function(con) print ("offline") end)

-- on publish message receive event
m:on("message", function(conn, topic, data)
  print(topic .. ":" )
  if data ~= nil then
    print(data)
  end
end)

-- for secure: m:connect("192.168.11.118", 1880,
1)m:connect("192.168.11.118", 1880, 0, function(conn)
print("connected") end)
```

```
-- subscribe topic with qos = 0
m:subscribe("/topic",0, function(conn) print("subscribe
success") end)
-- or subscribe multiple topic (topic/0, qos = 0; topic/1, qos
= 1; topic2 , qos = 2)
-- m:subscribe({["topic/0"]=0,["topic/1"]=1,topic2=2},
function(conn) print("subscribe success") end)
-- publish a message with data = hello, QoS = 0, retain = 0
m:publish("/topic","hello",0,0, function(conn) print("sent")
end)

m:close();
-- you can call m:connect again
```



Node MCU pin out

Why Node MCU?

Low-power, highly-integrated Wi-Fi solution
A minimum of 7 external components
Wide temperature range: -40°C to +125°C
ESP8285 — 8 Mbit flash

The ESP8266EX microcontroller integrates a Tensilica L106 32-bit RISC processor, which achieves extra-low power consumption and reaches a maximum clock speed of 160 MHz. The Real-Time Operating System (RTOS) and Wi-Fi stack allow about 80% of the processing power to be available for user application programming and development.

Engineered for mobile devices, wearable electronics and IoT applications, ESP8266EX achieves low power consumption with a combination of several proprietary technologies. The power-saving architecture features three modes of operation: active mode, sleep mode and deep sleep mode. This allows battery-powered designs to run longer.

ESP8266EX is integrated with a 32-bit Tensilica processor, standard digital peripheral interfaces, antenna switches, RF balun, power amplifier, low noise receive amplifier, filters and power management modules. All of them are included in one small package, our ESP8266EX.

ESP8266EX is capable of functioning consistently in industrial environments, due to its wide operating temperature range. With highly-integrated on-chip features and minimal external discrete component count, the chip offers reliability, compactness and robustness.

COMPONENTS DESCRIPTION

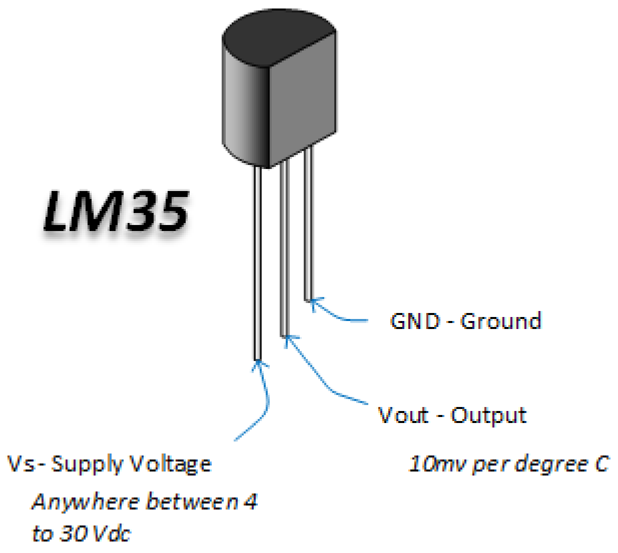
A. LM35 Temperature Sensor

The LM35 series are precision integrated-circuit temperature devices with an output voltage linearly-proportional to the Centigrade temperature. The LM35 device has an advantage over linear temperature sensors calibrated in Kelvin, as the user is not required to subtract a large constant voltage from the output to obtain convenient Centigrade scaling. The LM35 device does not require any external calibration or trimming to provide typical accuracies of ±¼°C at room temperature and ±¾°C over a full −55°C to 150°C temperature range. Lower cost is assured by trimming and calibration at the wafer level. The low-output impedance, linear output, and precise inherent calibration of the LM35 device makes interfacing to readout or control circuitry especially easy. The device is used with single power supplies, or with plus and minus supplies. As the LM35 device draws only 60 µA from the supply, it has very low self-heating of less than 0.1°C in still air. The LM35 device is rated to operate over a −55°C to 150°C temperature range, while the LM35C device is rated for a −40°C to 110°C range (−10° with improved accuracy). The LM35-series devices are available packaged in hermetic TO transistor packages, while the LM35C, LM35CA, and LM35D devices are available in the plastic TO-92 transistor package. The LM35D device is available in an 8-lead surface-mount small-outline package and a plastic TO-220 package.

Features

- Calibrated Directly in Celsius (Centigrade)
- Linear + 10-mV/°C Scale Factor
- 0.5°C Ensured Accuracy (at 25°C)
- Rated for Full −55°C to 150°C Range
- Suitable for Remote Applications
- Low-Cost Due to Wafer-Level Trimming

- Operates From 4 V to 30 V
- Less Than 60-µA Current Drain
- Low Self-Heating, 0.08°C in Still Air
- Non-Linearity Only ±¼°C Typical
- Low-Impedance Output, 0.1 Ω for 1-mA Load



LM35 Pinout

B. L293D Motor Driver

The L293 and L293D devices are quadruple high-current half-H drivers. The L293 is designed to provide bidirectional drive currents of up to 1 A at voltages from 4.5 V to 36 V. The L293D is designed to provide bidirectional drive currents of up to 600-mA at voltages from 4.5 V to 36 V. Both devices are designed to drive inductive loads such as relays, solenoids, DC and bipolar stepping motors, as well as other high-current/high-voltage loads in positive-supply applications.

Each output is a complete totem-pole drive circuit, with a Darlington transistor sink and a pseudo- Darlington source. Drivers are enabled in pairs, with drivers 1 and 2 enabled by 1,2EN and drivers 3 and 4 enabled by 3,4EN.

The L293 and L293D are characterized for operation from 0°C to 70°C.

Concept

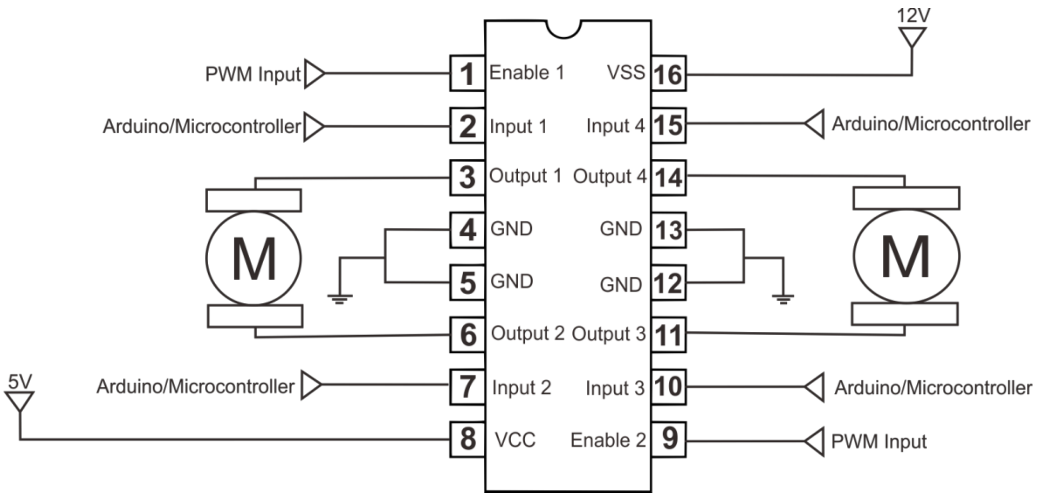
It works on the concept of H-bridge. H-bridge is a circuit which allows the voltage to be flown in either direction. As you know voltage need to change its direction for being able to rotate the motor in clockwise or anticlockwise direction, Hence H-bridge IC are ideal for driving a DC motor.

In a single L293D chip there are two h-Bridge circuit inside the IC which can rotate two dc motor independently. Due its size it is very much used in robotic application for controlling DC motors. Given below is the pin diagram of a L293D motor controller.

There are two Enable pins on l293d. Pin 1 and pin 9, for being able to drive the motor, the pin 1 and 9 need to be high. For driving the motor with left H-bridge you need to enable pin 1 to high. And for right H-Bridge you need to make the pin 9 to high. If anyone of the either pin1 or pin9 goes low then the motor in the corresponding section will suspend working. It's like a switch.

Let's consider a Motor connected on left side output pins (pin 3,6). For rotating the motor in clockwise direction the input pins has to be provided with Logic 1 and Logic 0.

• **Pin 2** = **Logic 1** and **Pin 7** = **Logic 0** | Clockwise Direction
• **Pin 2** = **Logic 0** and **Pin 7** = **Logic 1** | Anticlockwise Direction
• **Pin 2** = **Logic 0** and **Pin 7** = **Logic 0** | Idle [No rotation] [Hi-Impedance state]
• **Pin 2** = **Logic 1** and **Pin 7** = **Logic 1** | Idle [No rotation



L293d pinout diagram

C. Two channel Relay Module

A **relay** is an electrically operated switch. Many relays use an electromagnet to mechanically operate a switch, but other operating principles are also used, such as solid-state relays. Relays are used where it is necessary to control a circuit by a separate low-power signal, or where several circuits must be controlled by one signal.

In relation to mains relay have three connection:

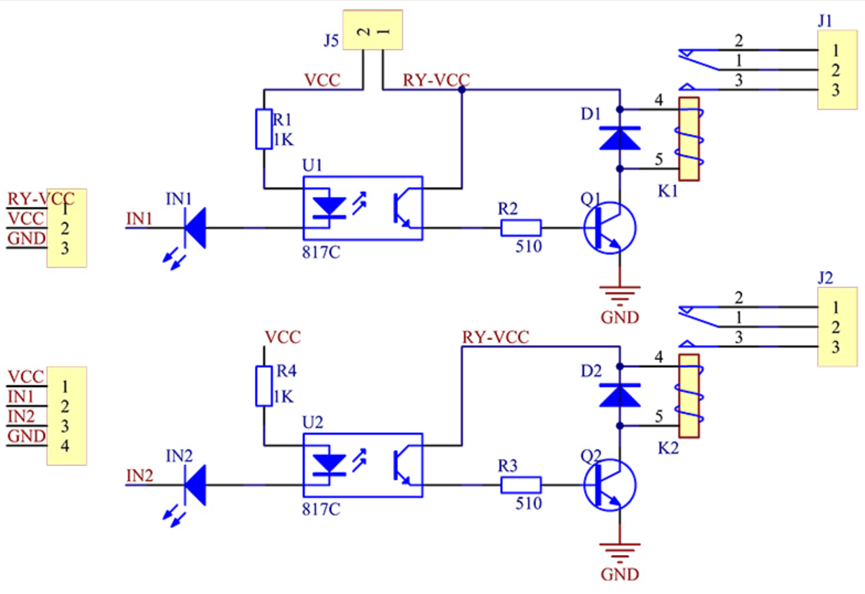
- **COM**: common pin
- **NO (Normally Open):** there is no contact between the common pin and the normally open pin. So, when you trigger the relay, it connects to the COM pin and supply is provided to a load
- **NC (Normally Closed):** there is contact between the common pin and the normally closed pin. There is always connection between the COM and NC pins, even when the relay is turned off. When you trigger the relay, the circuit is opened and there is no supply provided to a load.

The connections between the relay module and the MCU are:

- **GND**: goes to ground
- **IN1**: controls the first relay (it will be connected to an Arduino digital pin)
- **IN2**: controls the second relay (it should be connected to an Arduino digital pin if you are using this second relay. Otherwise, you don't need to connect it)
- **VCC**: goes to 5V

Controlling relay with MCU

```
int relay = 2;
 void setup() {
 // initialize the digital pin as an output.
 pinMode(relay, OUTPUT);
 }

 void loop() {
 digitalWrite(relay, HIGH);   // turn the relay on(HIGH is the
voltage level)
 delay(1000);                // wait for a second
 digitalWrite(relay, LOW);    // turn the relay on by making
the voltage LOW
 delay(1000);                // wait for a second
 }
```

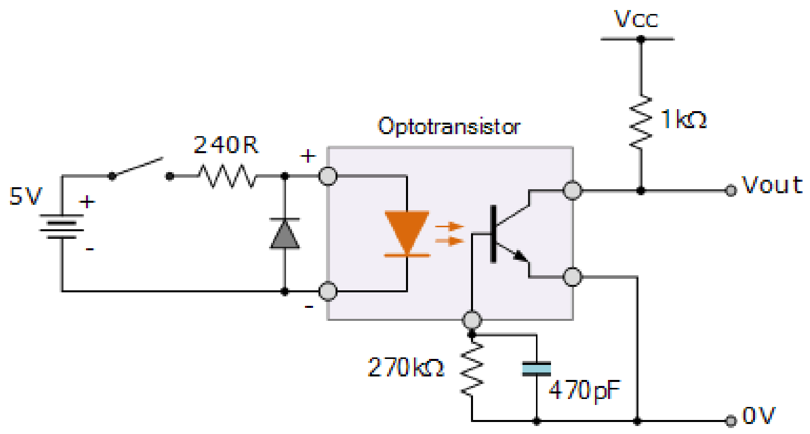


2 channel relay module

D. Optocoupler 817c

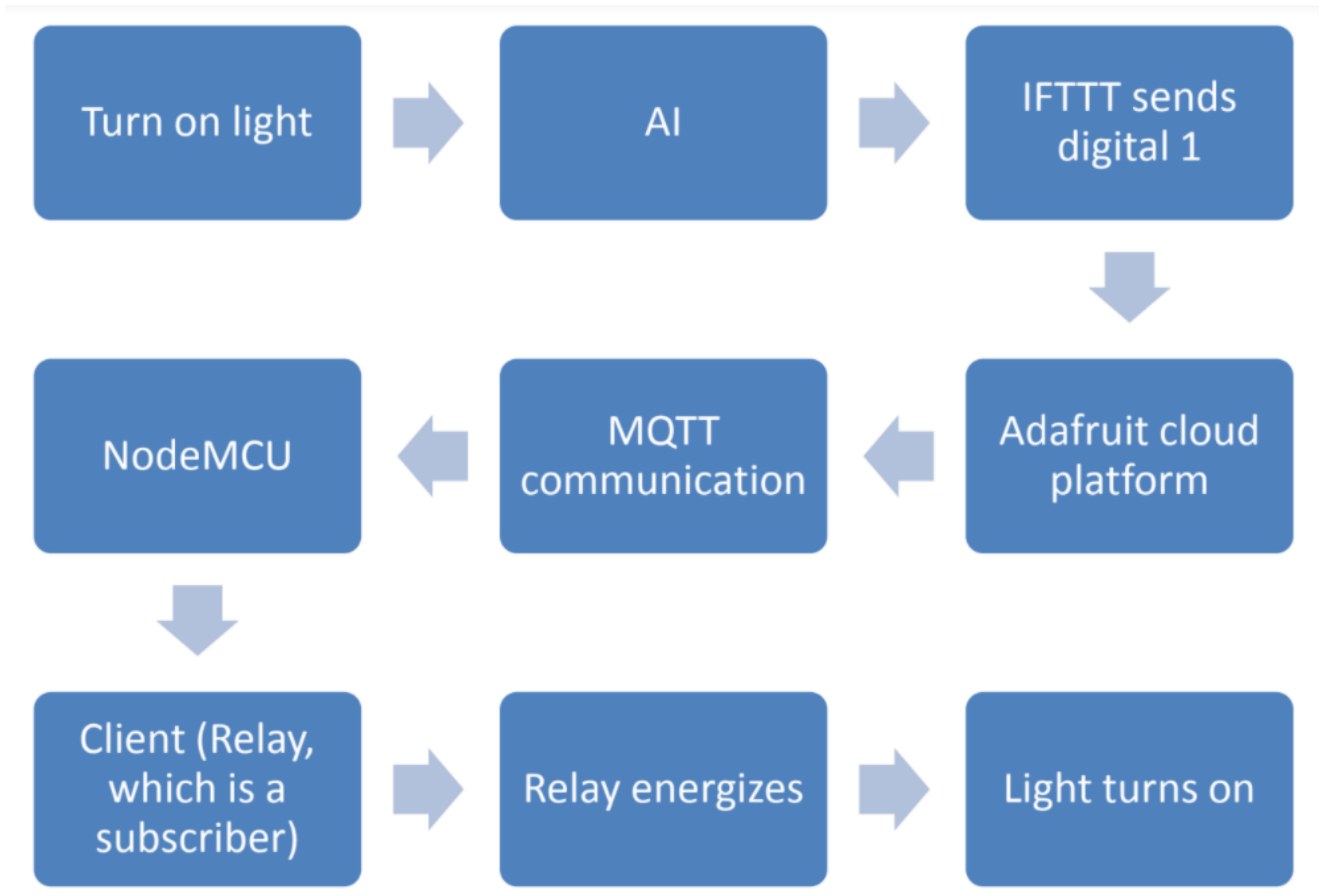
Transformers isolate the primary input voltage from the secondary output voltage using electromagnetic coupling and this is achieved using the magnetic flux circulating within their laminated iron core. But we can also provide electrical isolation between an input source and an output load using just light by using a very common and valuable electronic component called an **Optocoupler**.

The basic design of an optocoupler, also known as an **Opto-isolator**, consists of an LED that produces infra-red light and a semiconductor photo-sensitive device that is used to detect the emitted infra-red beam. Both the LED and photo-sensitive device are enclosed in a light-tight body or package with metal legs for the electrical connections as shown.



817c pinout diagram

FLOWCHART


Turn on light
AI
IFTTT sends digital 1
Adafruit cloud platform
MQTT communication
NodeMCU
Client (Relay, which is a subscriber)
Relay energizes
Light turns on

METHODOLOGY

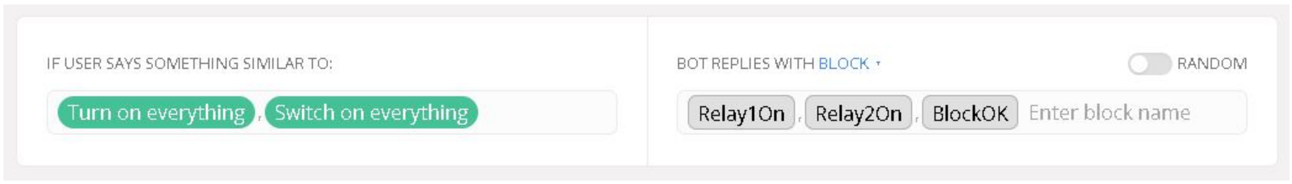
When a command is given using Google Assistant (Artificial Intelligence of Google) or by using Facebook Chatbot, the data goes straight to IFTTT (If This Then That). The service of IFTTT is used to program and process the input command to be sent to Adafruit cloud platform. The hardware is programmed to connect itself directly to the cloud using MQTT (Message Queue Telemetry Transport) protocol. Suppose the user sends a command “Turn on light” via Google or Facebook from anywhere in the world over the internet from his/her own account, then IFTTT decodes the message as “1” to Adafruit cloud platform. The cloud communicates with Node MCU by MQTT and the microcontroller sends digital “1” to the respective GPIO (General Purpose Input Output) pin. The input signal of relay module is connected to this pin and hence it receives a switching on command. Hence relay coil energises and the required light is turned on.

Similar action happens in motor control, where the binary “0” and “1” commands are replaced by PWM (Pulse Width Modulation) signal ranging from “0” to “1023”. In all these command receiving actions the microcontroller is programmed to listen to the server and in this mode of communication the microprocessor is said to be subscribed to the cloud.

When any live monitoring of sensor data is to happen, in this case data from LM35 sensor, the microprocessor is said to publish data to the cloud so it is programmed to be a publisher. Data from sensor is captured by Node MCU at specific delay as provided in program and this data is updated to cloud at given delay interval. Further processing of data can be done from this point onwards so as to trigger some action on respective sensor data.

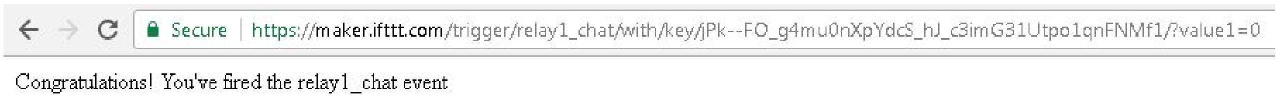
A. Artificial Intelligence setup

To begin with, a Facebook page is created for the appliances and the service of Chatfuel is used to program the page. With few AI rules it is possible to make an autonomous chatbot which can respond and send http request to IFTTT platform. The web hook service on IFTTT is used to accept the request and send corresponding data to Adafruit cloud. For switching purposes 1 and 0 are the values send and for PWM [5] (Pulse Width Modulation) control of dc motor, values ranging from 0 to 1023 are send from IFTTT to Adafruit.



An AI rule on Chatfuel to turn on both the relays.

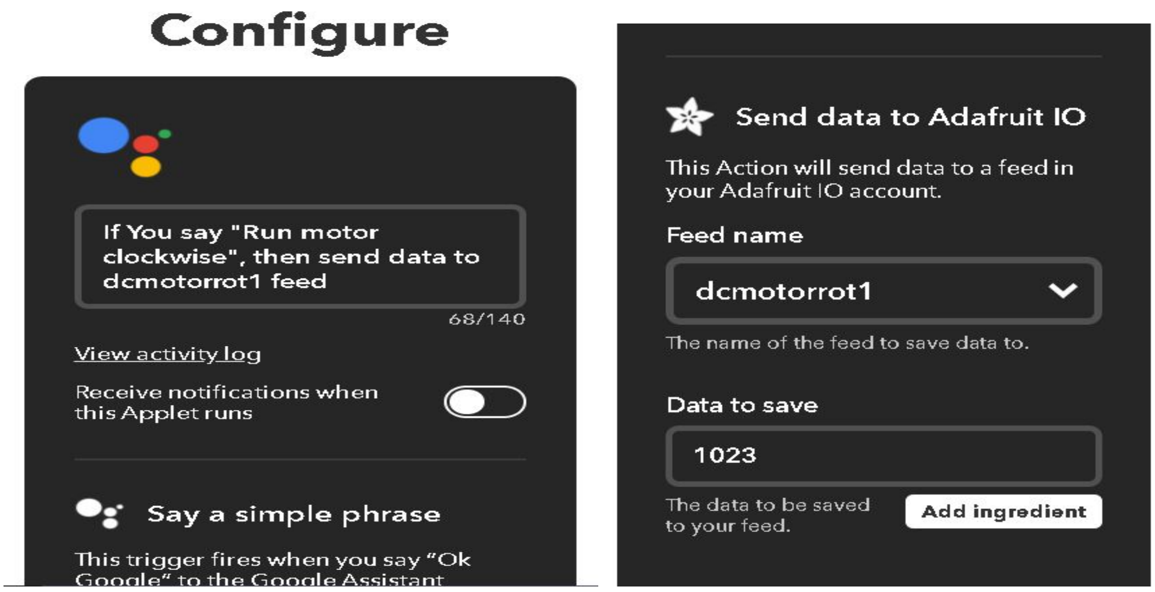
Chatfuel on receiving command from Facebook would send a post type http request to IFTTT using the JSON API [3]



```
https://maker.ifttt.com/trigger/relay1_chat/with/key/jPk--FO_g4mu0nXpYdcS_hJ_c3imG31Utpo1qnFNMf1/?value1=0
Congratulations! You've fired the relay1_chat event
```

Successful communication between Chatfuel and IFTTT.

Similarly, Google Assistant, Alexa and other Artificial Intelligence service are available on IFTTT platform which can be configured accordingly with Adafruit cloud.



Configure

If You say "Run motor clockwise", then send data to dcmotorrot1 feed

68/140

View activity log

Receive notifications when this Applet runs

Say a simple phrase

This trigger fires when you say "Ok Google" to the Google Assistant

Send data to Adafruit IO

This Action will send data to a feed in your Adafruit IO account.

Feed name

dcmotorrot1

The name of the feed to save data to.

Data to save

1023

The data to be saved to your feed.

Add ingredient

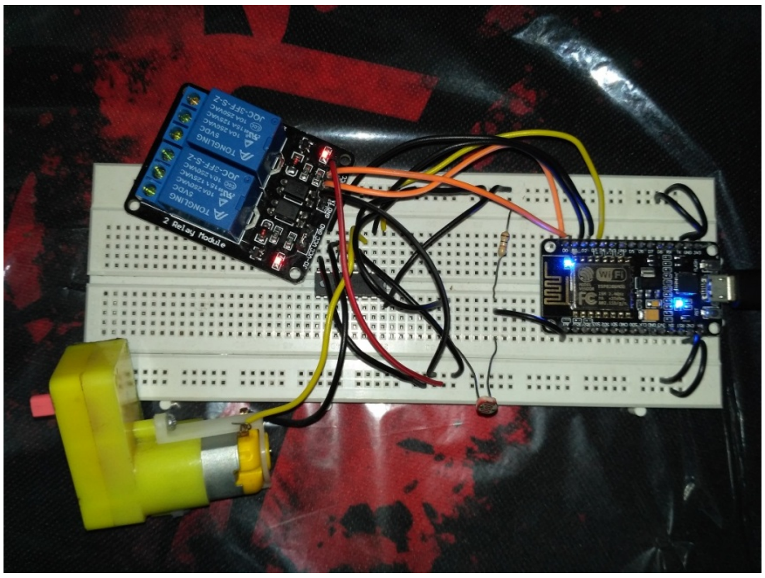
Configuration of Google Assistant and Adafruit on IFTTT

B. MQTT

MQTT or Message Queue Telemetry Transport is an extremely lightweight communication protocol developed by IBM. MQTT have a publish/subscribe design to communicate among machines (M2M). In this project MQTT is used as a message transfer binding protocol. MQTT consists of broker and client. A client is denoted as a subscriber or publisher to/of a certain topic. A subscriber listens to the server whereas a publisher sends value to the server. For the prototype Adafruit service is used as MQTT broker and NodeMCU as the client. Adafruit provides the cloud platform and a control dashboard as well. While programming the client, subscriber and publisher are defined. Relay and dc motor is the subscriber as they listen to the server for external command, whereas the sensor (photocell) is the publisher as it reads data and releases it to the server.

C. Hardware Design

NodeMCU is the heart of the project. It is the cheapest available microcontroller with inbuilt Wi-Fi support running on ESP8266, open source platform for developing IoT projects. It integrates GPIO, PWM, IIC and ADC all in one board. PWM (Pulse Width Modulation) feature is used for motor speed control [5], ADC (Analog to Digital Converter) is used for reading the sensor data [6]. GPIO (General Purpose Input Output) pins are used for connecting the relay and motor input pins. To run the motor, L293d motor driver IC is used. The IC runs on 3V DC and can support DC motor up to 24V. 5V relays can easily be used for switching home appliances. And NodeMCU requires 5V DC supply to function.



Actual circuit representation.

D. Software Implementation

1. Arduino IDE is installed.

2. From preferences ESP8266 board package is installed and NodeMCU 1.0 (ESP 12E) module is selected as the preferred board. Adafruit MQTT Library is also included.

3. Baud rate is set to 115200 and com port is selected accordingly.

4. Pins are defined, one for each relay. Since a single motor is used for demonstration purpose, two input pin and an enable pin for the motor are defined.

5. The SSID and Password of Wi-Fi for the internet connection are mentioned in the program.

6. Server ports are defined for Adafruit setup. For security purpose Adafruit also provides a unique key which is defined in the program.

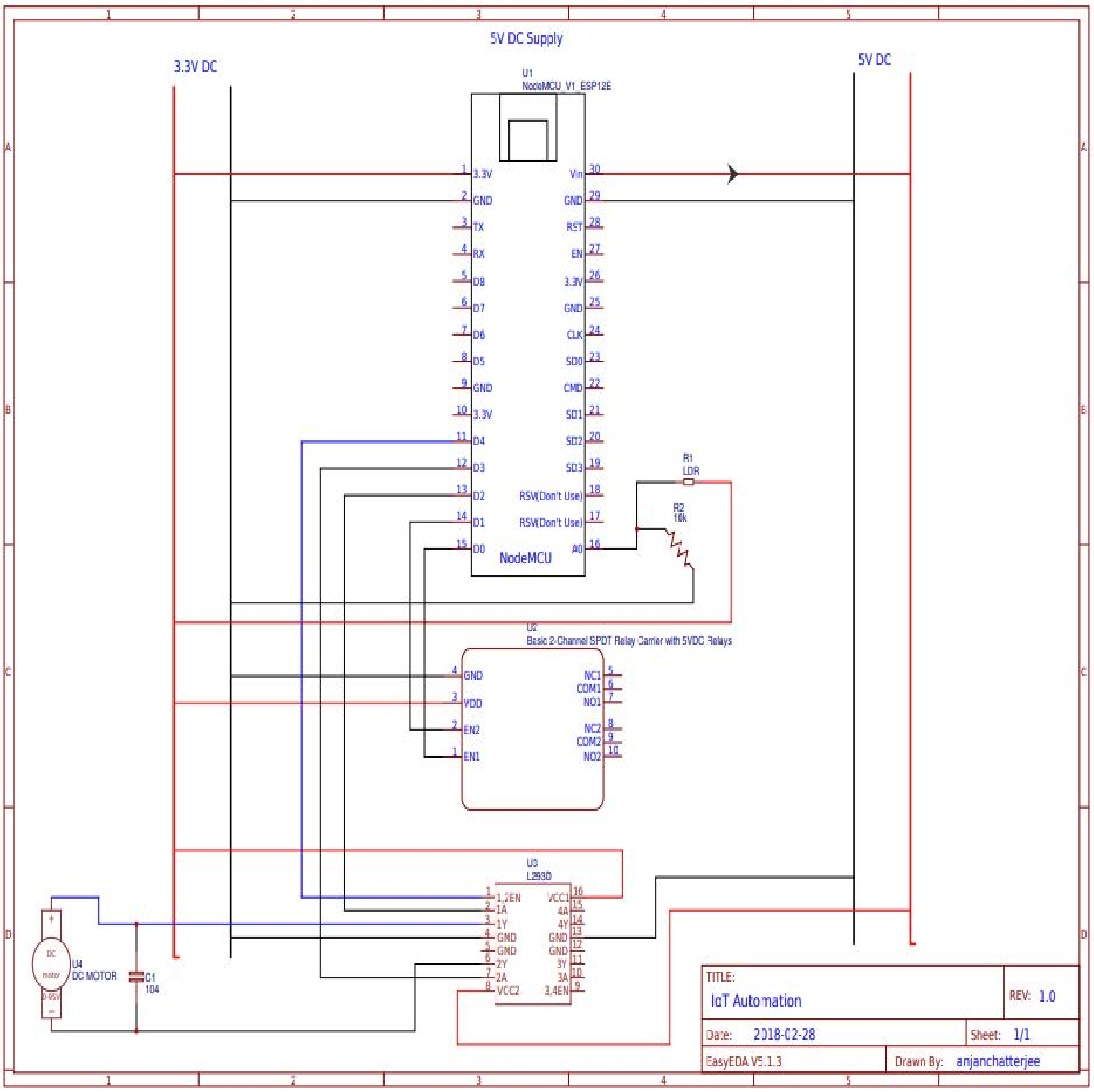
7. An ESP8266 Wi-Fi client class is created by passing in the login and server details for MQTT server secure client access.

8. Feeds are defined for MQTT paths in the form <username>/feeds/<feedname>. A feed may be defined for publishing values or for subscribing to a certain topic. Likewise, sensor feed is defined as publisher and relay and motor feeds are defined as subscribers.

9. Feeds are updated accordingly on Adafruit cloud platform and state of device can be monitored and controlled directly from there.

10. For Google Assistant, just speak out to Google or type out the action that you want to perform. For Facebook Chatbot, open the respective page of the project and type the command. Each service has their layer of security and can be accessed only via user's account. But the user if required can add multiple people on the platform. This adds security of devices that is mostly feared in the field of Internet of Things.

CIRCUIT DIAGRAM



Circuit diagram

PROGRAM

```
/**************************************************
  Adafruit MQTT Library ESP8266

  Must use ESP8266 Arduino from:
    https://github.com/esp8266/Arduino

  Works great with Adafruit's Huzzah ESP board & Feather
  ----> https://www.adafruit.com/product/2471
  ----> https://www.adafruit.com/products/2821

  Adafruit invests time and resources providing this open source code,
  please support Adafruit and open-source hardware by purchasing
  products from Adafruit!

  Written by Tony DiCola for Adafruit Industries.
  MIT license, all text above must be included in any redistribution

  Code is edited by Anjan Chatterjee for a project called
  IoT based Automation

 **************************************************/

#include <ESP8266WiFi.h>
#include "Adafruit_MQTT.h"
#include "Adafruit_MQTT_Client.h"

/************************* Pin Definition *********************************/

//Relays for switching appliances
#define relay_1 D0
#define relay_2 D1

//Motor pins
#define in_1 D2
#define in_2 D3
#define en_1 D4

//Analog pin to read sensor value
#define sensor A0

/************************* WiFi Access Point *********************************/

#define WLAN_SSID       "TrojanWorm"
#define WLAN_PASS       "@WifiAC!"
```

```
/************************* Adafruit.io Setup *********************************/

#define AIO_SERVER      "io.adafruit.com"
#define AIO_SERVERPORT  1883                   // use 8883 for SSL
#define AIO_USERNAME    "anjanchatterjee"
#define AIO_KEY         "8a69fe611fbf4c8d9e65a13afe8984f5"

/************ Global State (no need to change this!) ******************/

// Create an ESP8266 WiFiClient class to connect to the MQTT server.
WiFiClient client;
// or... use WiFiFlientSecure for SSL
//WiFiClientSecure client;

// Setup the MQTT client class by passing in the WiFi client and MQTT server and login
details.
Adafruit_MQTT_Client mqtt(&client, AIO_SERVER, AIO_SERVERPORT,
AIO_USERNAME, AIO_KEY);

/****************************** Feeds
***************************************/

// Setup feed for publishing.
// Notice MQTT paths for AIO follow the form: <username>/feeds/<feedname>
Adafruit_MQTT_Publish photocell = Adafruit_MQTT_Publish(&mqtt, AIO_USERNAME
"/feeds/photocell");
Adafruit_MQTT_Publish tempf = Adafruit_MQTT_Publish(&mqtt, AIO_USERNAME
"/feeds/tempf");
Adafruit_MQTT_Publish tempc = Adafruit_MQTT_Publish(&mqtt, AIO_USERNAME
"/feeds/tempc");

// Setup feed for subscribing to changes.
Adafruit_MQTT_Subscribe onoffbutton1 = Adafruit_MQTT_Subscribe(&mqtt,
AIO_USERNAME "/feeds/relay1");
Adafruit_MQTT_Subscribe onoffbutton2 = Adafruit_MQTT_Subscribe(&mqtt,
AIO_USERNAME "/feeds/relay2");
Adafruit_MQTT_Subscribe onoffbutton3 = Adafruit_MQTT_Subscribe(&mqtt,
AIO_USERNAME "/feeds/dcmotorrot1");
Adafruit_MQTT_Subscribe onoffbutton4 = Adafruit_MQTT_Subscribe(&mqtt,
AIO_USERNAME "/feeds/dcmotorrot2");

/************************* Sketch Code
************************************/

// Bug workaround for Arduino 1.6.6, it seems to need a function declaration
// for some reason (only affects ESP8266, likely an arduino-builder bug).
```

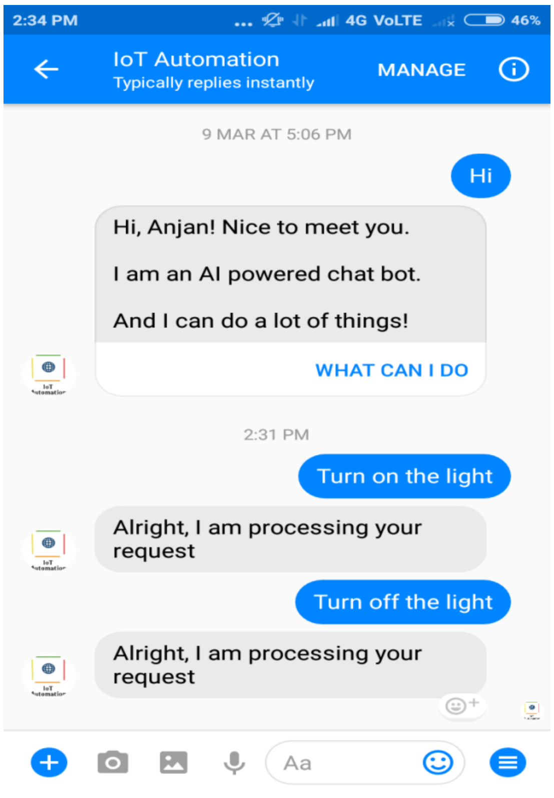
```
void MQTT_connect();

void setup() {
  Serial.begin(115200);
  delay(10);
  pinMode(relay_1,OUTPUT);
  pinMode(relay_2,OUTPUT);
  pinMode(in_1,OUTPUT);
  pinMode(in_2,OUTPUT);
  pinMode(en_1,OUTPUT);
  pinMode(sensor,INPUT);
Serial.println(F("Adafruit MQTT project"));

  // Connect to WiFi access point.
  Serial.println(); Serial.println();
  Serial.print("Connecting to ");
  Serial.println(WLAN_SSID);

  WiFi.begin(WLAN_SSID, WLAN_PASS);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println();

  Serial.println("WiFi connected");
  Serial.println("IP address: "); Serial.println(WiFi.localIP());

  // Setup MQTT subscription for onoff feed.
  mqtt.subscribe(&onoffbutton1);
  mqtt.subscribe(&onoffbutton2);
  mqtt.subscribe(&onoffbutton3);
  mqtt.subscribe(&onoffbutton4);
}

uint32_t x=0;

void loop() {
  // Ensure the connection to the MQTT server is alive (this will make the first
  // connection and automatically reconnect when disconnected).  See the MQTT_connect
  // function definition further below.
  MQTT_connect();

  // this is our 'wait for incoming subscription packets' busy subloop
  // try to spend your time here
```

```
Adafruit_MQTT_Subscribe *subscription;
  while ((subscription = mqtt.readSubscription(5000))) {
    if (subscription == &onoffbutton1) {
      Serial.print(F("Got Relay 1: "));
      Serial.println((char *)onoffbutton1.lastread);
      uint16_t state = atoi((char *)onoffbutton1.lastread);  // convert to a number
      digitalWrite(relay_1,state);
    }
      if (subscription == &onoffbutton2) {
      Serial.print(F("Got Relay 2: "));
      Serial.println((char *)onoffbutton2.lastread);
      uint16_t state = atoi((char *)onoffbutton2.lastread);  // convert to a number
      digitalWrite(relay_2,state);
    }
      if (subscription == &onoffbutton3) {
      Serial.print(F("Got DC Motor Rotation 1: "));
 Serial.println((char *)onoffbutton3.lastread);
      uint16_t state = atoi((char *)onoffbutton3.lastread);  // convert to a number
      analogWrite(en_1, state);   //sets the motors speed
      digitalWrite(in_1, HIGH);
      digitalWrite(in_2, LOW);
    }
      if (subscription == &onoffbutton4) {
      Serial.print(F("Got DC Motor Rotation 2: "));
      Serial.println((char *)onoffbutton4.lastread);
      uint16_t state = atoi((char *)onoffbutton4.lastread);  // convert to a number
      analogWrite(en_1, state);   //sets the motors speed
      digitalWrite(in_1, LOW);
      digitalWrite(in_2, HIGH);
    }
  }

//   Now we can publish stuff!
  Serial.print(F("\nSending photocell val "));
  Serial.print("Photocell");
  Serial.println(analogRead(sensor));
  Serial.print("...");
  int sensor_val = analogRead(sensor);

//  For LDR value
  if (! photocell.publish(sensor_val)) {
    Serial.println(F("Failed"));
  } else {
    Serial.println(F("OK!"));
  }
```

```
//  For temperature in Celsius
  float millivolts = (sensor_val/1024.0) * 3300; //3300 is the voltage provided by NodeMCU
  float celsius = millivolts/10;
  Serial.print("in DegreeC=   ");
  Serial.println(celsius);

  if (! tempc.publish(celsius)) {
    Serial.println(F("Failed"));
  } else {
    Serial.println(F("OK!"));
  }

//  For temperature in Fahrenheit
  float fahrenheit = ((celsius * 9)/5 + 32);
  Serial.print(" in Farenheit=   ");
  Serial.println(fahrenheit);

  if (! tempf.publish(fahrenheit)) {
    Serial.println(F("Failed"));
  } else {
    Serial.println(F("OK!"));
  }
 // delay(1000);

  // ping the server to keep the mqtt connection alive
  // NOT required if you are publishing once every KEEPALIVE seconds
  /*
  if(! mqtt.ping()) {
    mqtt.disconnect();
  }
  */
}

// Function to connect and reconnect as necessary to the MQTT server.
// Should be called in the loop function and it will take care if connecting.
void MQTT_connect() {
  int8_t ret;

  // Stop if already connected.
  if (mqtt.connected()) {
    return;
  }

  Serial.print("Connecting to MQTT... ");

  uint8_t retries = 3;
 while ((ret = mqtt.connect()) != 0) { // connect will return 0 for connected
      Serial.println(mqtt.connectErrorString(ret));
      Serial.println("Retrying MQTT connection in 5 seconds...");
      mqtt.disconnect();
      delay(5000);  // wait 5 seconds
      retries--;
      if (retries == 0) {
        // basically die and wait for WDT to reset me
        while (1);
      }
  }
  Serial.println("MQTT Connected!");
}
```

RESULT

After preparing the setup, end user may communicate with the hardware from anywhere in the world using the configured AI. It should be made sure that the hardware receives adequate power supply. In this case, a simple Facebook chat or a Google assistant voice command from any Smartphone device can trigger the hardware as shown in Fig. 7.



9 MAR AT 5:06 PM

Hi

Hi, Anjan! Nice to meet you.

I am an AI powered chat bot.

And I can do a lot of things!

2:31 PM

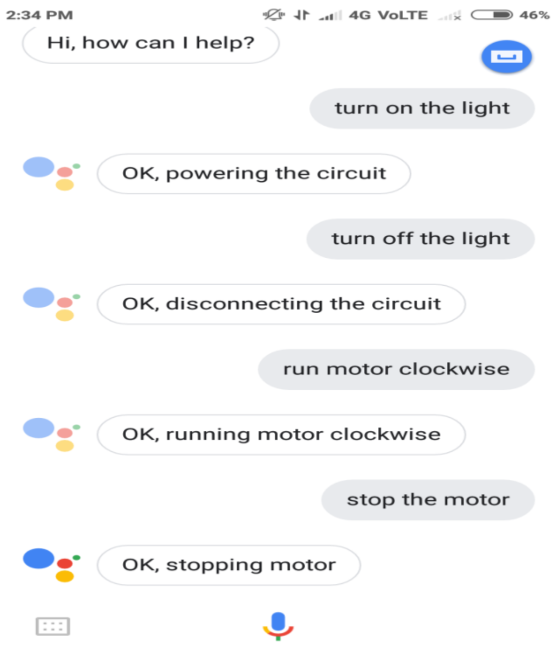
Turn on the light

Alright, I am processing your request

Turn off the light

Alright, I am processing your request

Facebook chat


2:34 PM 4G VoLTE 46%

Hi, how can I help?

turn on the light

OK, powering the circuit

turn off the light

OK, disconnecting the circuit

run motor clockwise

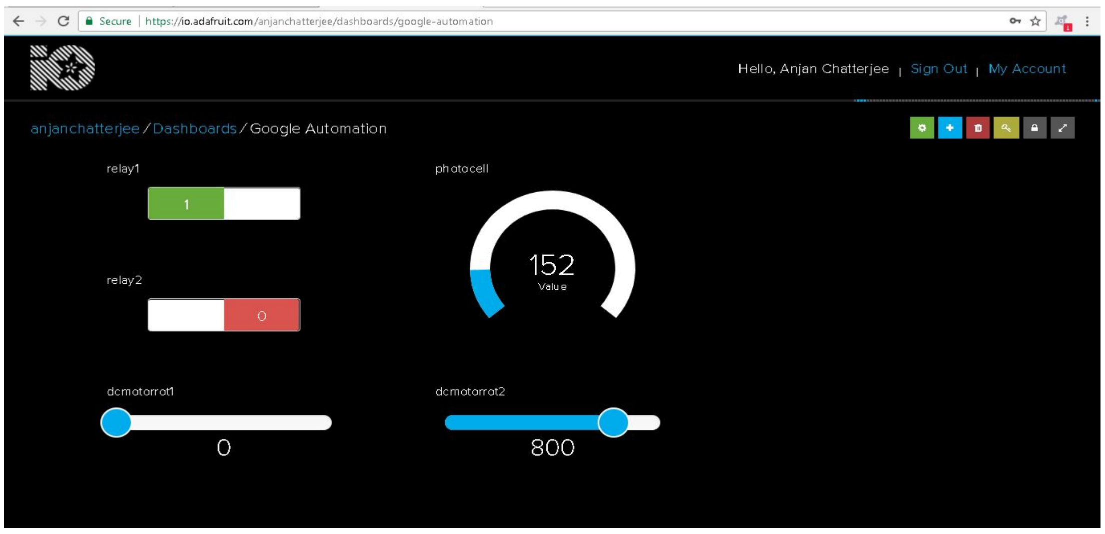
OK, running motor clockwise

stop the motor

OK, stopping motor

Google Assistant chat

The AI of choice would channel the request of end user via IFTTT (If This Then That) service to Adafruit cloud platform. Adafruit is integrated into the hardware using MQTT protocol as explained before. So we can monitor any connected devices to NodeMCU and their live status on Adafruit dashboard. We can also control our devices from there.


Secure | https://io.adafruit.com/anjanchatterjee/dashboards/google-automation

Hello, Anjan Chatterjee | Sign Out | My Account

anjanchatterjee / Dashboards / Google Automation

relay1: 1

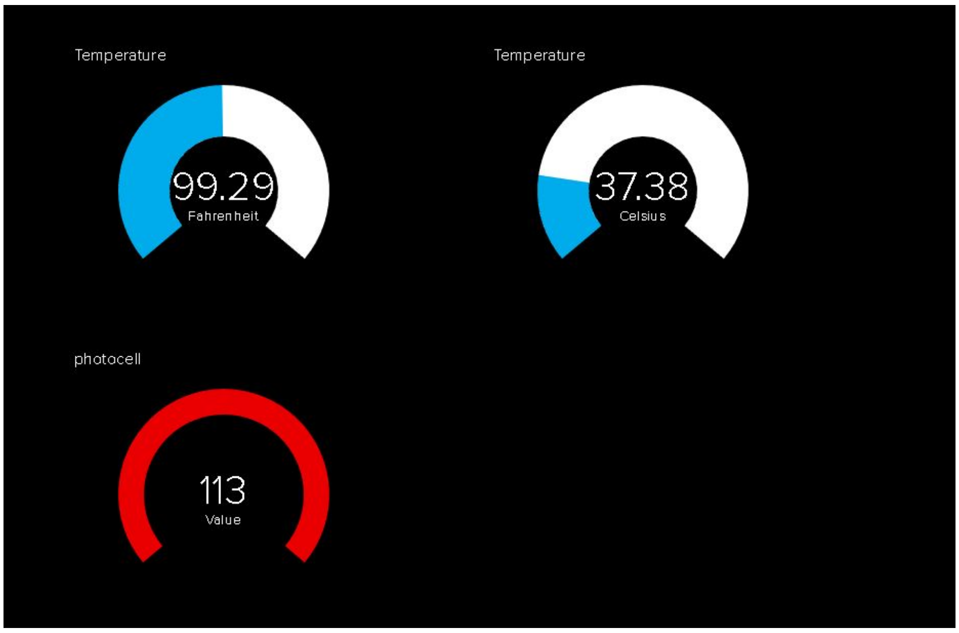
photocell: 152 Value

relay2: 0

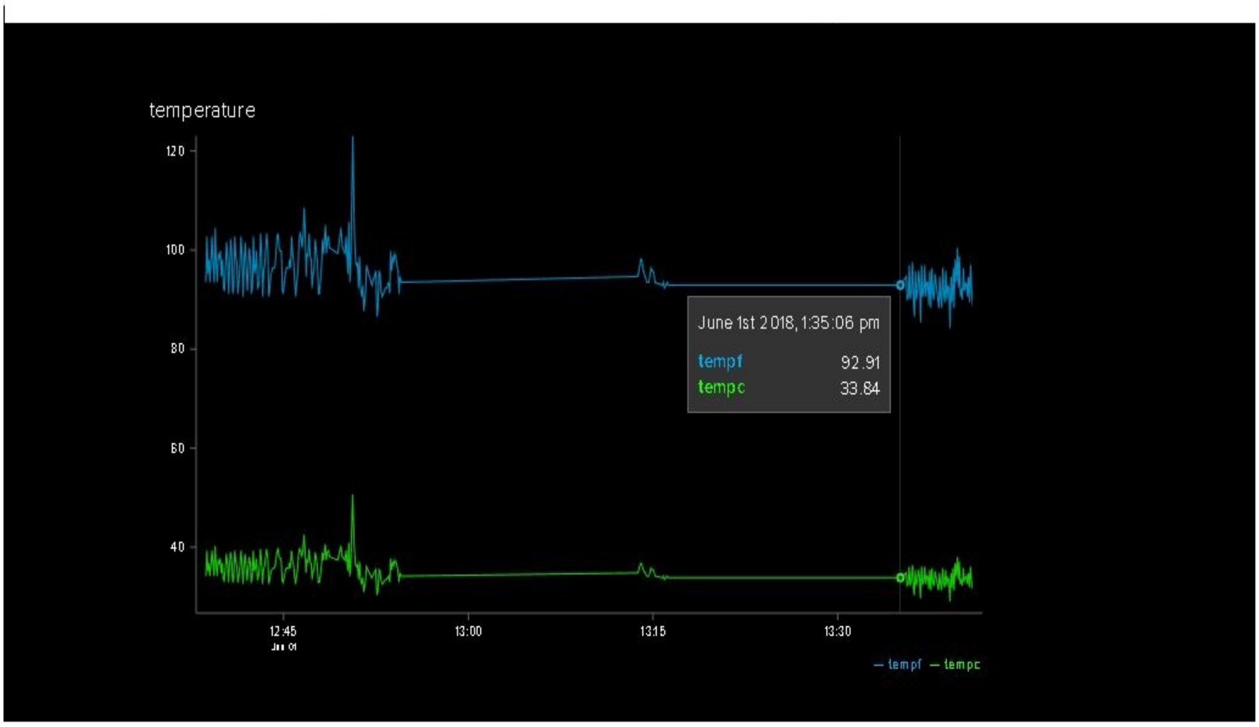
dcmotorrot1: 0

dcmotorrot2: 800

Adafruit control dashboard



Real time data monitoring



Sensor data in graph

FUTURE SCOPE

A. Social Internet of Things

With their last decade spent on connecting people with each other, social media giants and other organisations are now looking at the coming ten years to connect people to their devices and gain valuable insight from it. As enterprises around the world seek an efficient way to monitor, listen, and analyse data gathered from social media, IoT offers them a convenient method for social data aggregation without affecting their time and energy. By gathering valuable insights with the help of IoT-connected social media monitoring tools, businesses can make informative and crucial decisions across a variety of internal departments.

In order to practically integrate the ubiquitous computing in our future daily life with high Quality of Experience [12], we need to improve the connectivity of all the relationships between users and things, and to enhance the availability of computational power via sets of things surrounding us. Therefore, we take into consideration social networks (SNs) of all entities (i.e., humans and things) for ubiquitous computing as an evolution beyond the IoT. In other words, things should be socialized for allowing humans to establish relationships with them in an easy way. It does not only mean physical connections between humans and things, but also logical configurations of social communities involving humans as well as things. This logical configuration can be realized through exhibiting features from people's SN and adopting them for the suggested universal SN of all entities. The future SN of things can include the interactivity scheme, profiling system, recommendation, and mash up of services that include both human and machine on the same platform.

B. Blockchain into IoT

The blockchain-based approach, being trust-free is becoming a central feature of people’s relationships. The Economist describes blockchain (BC) as “the trust machine”, indicating that it takes care of trust issues between individuals (Economist 2015). In other words, the economic system, which is built on blockchain technology, runs without people, thus making a transaction “trust-free.” Historically, trust has underpinned business, often involving a reliable third party, which is expensive. Blockchain technology provides a viable alternative to eliminate intermediaries, thereby lowering operational costs and increasing the efficiency of a sharing service, such as a SIoT enabled device in the perspective of this paper. With blockchain technology, the world’s most fundamental commercial interactions can be re-imagined; the door to invent new styles of digital interactions in trust-free sharing services has been opened.

IoT devices would benefit from a private immutable ledger that acts as per the technology of peer to peer, but can be managed centrally, to optimize energy consumption [13]. BlockChain (BC) holds promise for privacy and security in IoT. However, applying BC in IoT is not straightforward due to various associated challenges including: high resource consumption, scalability, and processing time. High resource devices create an overlay network to implement a publicly accessible distributed BC that ensures end-to-end security and privacy. It employs a hierarchical architecture that uses a centralized private Immutable Ledger (IL) at the local IoT network level to reduce overhead, and a decentralized public BC at higher end devices for stronger trust. A distributed trust method is employed to decrease new block processing overhead. Security and privacy of the design could be evaluated that shows the robustness of the new architecture against several attacks. Security in IoT is challenging due to low resource capabilities of the vast majority of devices, immense scale, heterogeneity among the devices, and lack of standardization. Moreover, many of these IoT devices collect and share large amounts of data from our personal spaces, thus opening up significant privacy concerns. BlockChain can come into effect at this stage to protect user’s privacy, by anonymizing the data and its origin. This security requirement is necessary since we are standing on the brink of an era with real ubiquitous computing and communication where many gadgets, such as sensors, RF identification (RFID) tags, and smart electronic/electromechanical devices, surrounding us will be on the real-time network.

CONCLUSION

In this paper, the system proposed is a smart way of integrating electronic appliances with our daily life. The paper implies on using Artificial Intelligence along with the Internet of Things but it also shows an approach to the new world of Social Internet of Things (SIoT), where the potentialities of social networking concepts can be merged with IoT so that a physical appliance can be integrated directly into a social platform. The resulting paradigm has the potential to support novel applications and networking services for the IoT in more effective, efficient and secure ways. The technology can be beneficial for establishment and management of social relationships between objects in such a way that the resulting social network is navigable just like humans are connected on social media. The paper also describes a possible architecture for the IoT that includes the functionalities required to integrate things into a social network.

The technology of BlockChain can also be utilised to develop a cryptic ledger of millions of devices across the globe so that they can socialize among themselves and end users with secure and private design and help devices to learn about human preferences and function accordingly. A lot of work is required in this field so that in future we can have a social media dedicated to physical devices which are also connected on BlockChain so that they can communicate among themselves and take decisions and make human machine interaction to the next level of innovation.

REFERENCES

Adafruit Learning MQTT, Cloud Automation (https://learn.adafruit.com/mqtt-adafruit-io-and-you/getting-started-on-adafruit-io)

IFTTT Documentation (https://platform.ifttt.com/docs)

Chatfuel and JSON API (http://docs.chatfuel.com/plugins/plugin-documentation/json-api)

Wikipedia 2018, 29th March, NodeMCU (https://en.wikipedia.org/wiki/NodeMCU)

Cosmas Tatenda Katsambe, Vinukumar Luckose, Nurul Shahrizan Shahabuddin, "EFFECT OF PULSE WIDTH MODULATION ON DC MOTOR SPEED", International Journal of Students' Research In Technology & Management, Vol 5, No 2, June 2017, pp 42-45.

Mr. Nerella Ome1 , Mr. G. Someswara Rao, "INTERNET OF THINGS (IOT) BASED SENSORS TO CLOUD SYSTEM USING ESP8266 AND ARDUINO DUE", International Journal of Advanced Research in Computer and Communication Engineering, Vol. 5, Issue 10, October 2016, pp 337-343.

R.Piyare, M.Tazi "BLUETOOTH BASED HOME AUTOMATION SYSTEM USING CELL PHONE", 2011 IEEE 15th International Symposium on Consumer Electronics

Deepali Javale, Mohd. Mohsin, Shreerang Nandanwar, Mayur Shingate, "HOME AUTOMATION AND SECURITY SYSTEM USING ANDROID ADK", International Journal of Electronics Communication and Computer Technology (IJECCT), Volume 3 Issue 2 (March 2013), pp 382-385.

Chandra Sekar B, Nikhil K S, Raju K N , Sanjay M, Chandrappa D N, "INTERNET OF THINGS BASED AUTOMATION USING ARTIFICIAL INTELLIGENCE", International Journal of Emerging Research in Management &Technology, Volume-6, Issue-7 (July 2017), pp 142-145.

Prof. Niranjan M, Madhukar N, Ashwini A, Muddsar J, Saish M, "IOT BASED INDUSTRIAL AUTOMATION", IOSR Journal of Computer Engineering, National Conference On Advances In Computational Biology, Communication, And Data Analytics (ACBCDA 2017), pp 36-40.

Ms. C. Hemalatha, Mr. R. Nagarajan, P. Suresh, G. Ganesh Shankar, “BRUSHLESS DC MOTOR CONTROLLED BY USING INTERNET OF THINGS”, International Journal of Science Technology & Engineering, Volume 3, Issue 09, March 2017, pp 373-377.

Antonio M. Ortiz, Dina Hussein, Soochang Park, and Son N. Han, “THE CLUSTER BETWEEN INTERNET OF THINGS AND SOCIAL NETWORKS: REVIEW AND RESEARCH CHALLENGES”, IEEE Internet of Things Journal, Volume 1, No. 3, JUNE 2014, pp 206-215.

Ali Dorri, Salil S. Kanhere, Raja Jurdak, “TOWARDS AN OPTIMIZED BLOCKCHAIN FOR IOT”, IEEE International Conference on Internet-of-Things Design and Implementation (IoTDI 2017), Pittsburgh, PA USA, April 2017, pp 173-178.

Luigi Atzori, Antonio Iera, Giacomo Morabito, and Michele Nitti, “THE SOCIAL INTERNET OF THINGS (SIOT) – WHEN SOCIAL NETWORKS MEET THE INTERNET OF THINGS: CONCEPT, ARCHITECTURE AND NETWORK CHARACTERIZATION”, Computer Networks (Elsevier), Volume-56, Issue-16 (November 2012), pp 3594-3608.

https://roboindia.com/tutorials/direct.php?route=nodemcu-motor-driver-pwm

https://electronics-tutorials.ws/blog/pulse-width-modulation