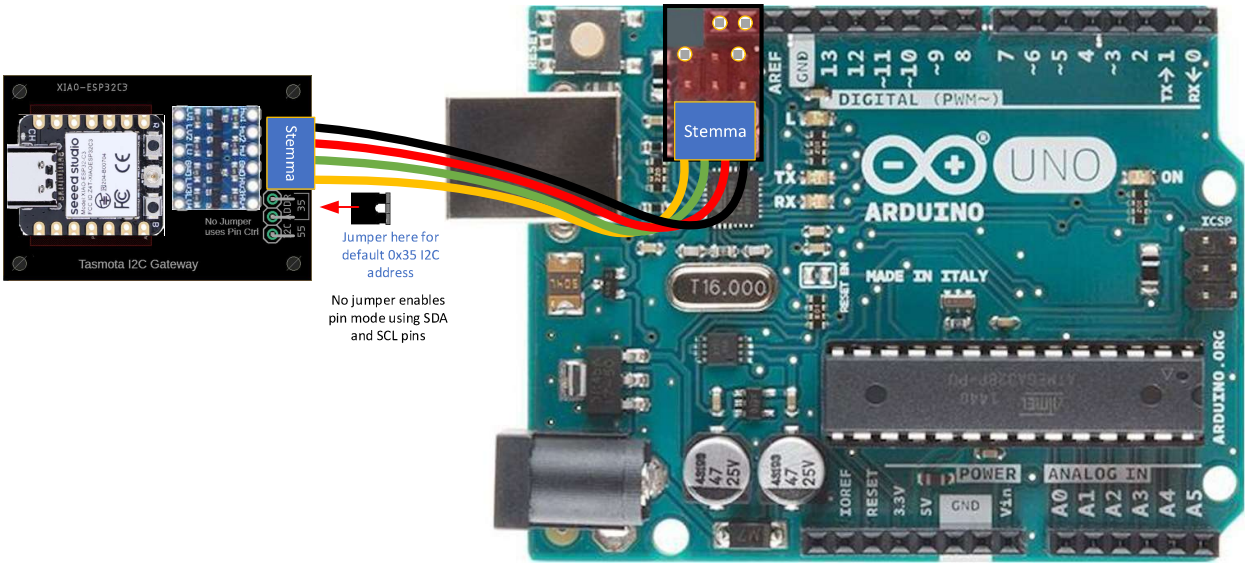
Arduino Tasmota Gateway

Problem definition:
Available Arduino solutions for controlling mains powered devices use solid state or electromechanical relays that could expose users to mains voltage. This solution uses wireless connectivity to control CE approved sealed Tasmota smartplugs.

The gateway connection is through a four pin Stemma (JST PH) connector to a small daughter board that plugs into an Arduino Uno. Connection can also be achieved using a Stemma to male DuPont pins plugged directly into the +5V, Ground and SDA/SCL pins.

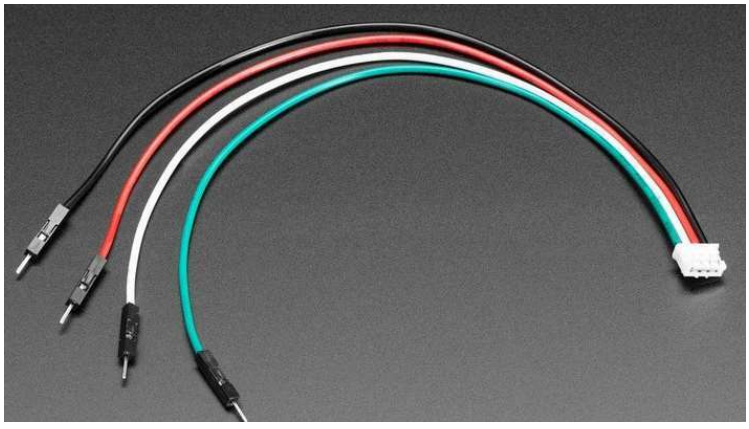


There are two control modes: Pin control and I2C.

Pin Control
With no jumper on the gateway PCB, the gateway will turn a smartplug on when the associated Arduino pin is High, off when the pin is Low. When using the daughter Stemma connector board, these pins are: SCL (PIN A4 or digital pin 18) and SDA (A5 or digital pin 19) but any digital pin can be used that the control wires are connected to.

Note: if I2C or analog pins A4 or A5 are used by the sketch, a Stemma to Dupont cable should be used to connect to unused pins.

THE PI HUT
SKU: 104250



I2C control
With the I2C jumper attached, the board will respond to requests through the Tasmota library.

This library allows control of as many smartplugs as are configured on the gateway, without tying up any pins other than the SDA and SCL pins required for I2C. It also enables reading of power and voltage statistics from connected smartplugs. It can also provide wireless signal quality between the gateway and connected smartplugs.

The default I2C address is 0x35 but 0x55 can be selected by changing the jumper if there is another I2C device at 0x35.

Here is a simple sketch that turns a smartplug on and off:

```
#include <SPI.h>
#include "tasmotaI2c.h"

TasmotaI2c tasmota(PRIMARY_I2C_ADDR);

void setup() {
  tasmota.begin();
  tasmota.powerOn();  // Turn on the power
  delay(1000);        // Wait for 1 second
  tasmota.powerOff(); // Turn off the power
}

void loop() {
  // your code here to call tasmota methods as required
}
```

Example calling convention with a single smartplug:

Calling powerOn() turns on the smartplug
Calling powerOff() turns off the smartplug
Calling getEnergyValues () returns a structure with energy values
Calling getRSSI() returns wifi signal strength percent((0-100)

If more than one smartplug is configured, the above calls operate on the first configured smartplug. To operate on other configured smartplugs, an argument must be given to the above commands indicating the index into the list of configured smartplugs.

For example, assuming the following config.json fiile:

```
{
  "esp_pin_map": [6,7],
  "plug_ip": [13,12],
  "plugs_per_ip":[1,1]
}
```

powerOn(0) turns on plug at IP address 192.168.4.13 (same as calling powerOn())
powerOn(1) turns on the plug at IP address 192.168.4.12

notes:

- all plugs have routing prefix of 192.168.4
- esp pin map values are ignored in I2C control mode

Below is a test sketch that uses the full library functionality:

```
#include "tasmotaI2c.h"

TasmotaI2c tasmota(PRIMARY_I2C_ADDR);

void setup() {
  Serial.begin(9600);
  Serial.print("\nStarting Tasmota I2C interface at 0X");
Serial.println(PRIMARY_I2C_ADDR, HEX);
  tasmota.begin();
  // test();
  printInstructions();   // Print instructions for using the commands
}

void test(){
  tasmota.powerOn();  // Turn on the power
  delay(1000);           // Wait for 1 second
  tasmota.powerOff(); // Turn off the power
  uint8_t rssi = tasmota.getRSSI();
  Serial.print("RSSI: ");
  Serial.println(rssi);

  EnergyValues energy = tasmota.getEnergyValues();
  Serial.print("Energy - Voltage: ");
  Serial.print(energy.Voltage);
  Serial.print(", Current: ");
  Serial.print(energy.Current);
  Serial.print(", Power: ");
  Serial.println(energy.Power);

}


void loop() {
  if (Serial.available() > 0) {
    String command = Serial.readStringUntil('\n');  // Read command from serial monitor
    command.trim();  // Remove any extraneous whitespace or newlines

    if (command == "on") {
      int ret = tasmota.powerOn();
      Serial.println("Power turned on");
    } else if (command == "off") {
      tasmota.powerOff();
      Serial.println("Power turned off");
    } else if (command.startsWith("on ")) {
      int ipIndex = command.substring(3).toInt();
      tasmota.powerOn(ipIndex);
      Serial.println("Power turned on for device at index: " + String(ipIndex));
    } else if (command.startsWith("off ")) {
      int ipIndex = command.substring(4).toInt();
      tasmota.powerOff(ipIndex);
      Serial.println("Power turned off for device at index: " + String(ipIndex));
    } else if (command == "rssi") {
      uint8_t rssi = tasmota.getRSSI();
```

```
      Serial.println("RSSI Value: " + String(rssi));
    } else if (command == "energy") {
      EnergyValues values = tasmota.getEnergyValues();
      printEnergyValues(values);
    } else if (command == "help") {
      printInstructions();
    } else {
      Serial.println("Invalid command. Type 'help' for a list of commands.");
    }
  }
}

void printInstructions() {
  Serial.println("Welcome to the tasmota Control Interface!");
  Serial.println("Available commands:");
  Serial.println("  'on'       - Turn on the power for the first device");
  Serial.println("  'off'      - Turn off the power for the first device");
  Serial.println("  'on [n]'   - Turn on the power for the nth device");
  Serial.println("  'off [n]'  - Turn off the power for the nth device");
  Serial.println("  'rssi'     - Get the RSSI value for the first device");
  Serial.println("  'energy'   - Get energy values for the first device");
  Serial.println("Type 'help' to display this message again.");
}

void printEnergyValues(const EnergyValues& values) {
  Serial.println("Energy Values:");
  Serial.print("Voltage: "); Serial.print(values.Voltage); Serial.println(" V");
  Serial.print("Current: "); Serial.print(values.Current); Serial.println(" A");
  Serial.print("Power: "); Serial.print(values.Power); Serial.println(" W");
  Serial.print("Energy Today: "); Serial.print(values.Today); Serial.println(" kWh");
  Serial.print("Yesterday: "); Serial.print(values.Yesterday); Serial.println(" kWh");
  Serial.print("Total Energy: "); Serial.print(values.Total); Serial.println(" kWh");
}
```