Phpoc Library

This Library is for using PHPoC Shields.

Member Classes

Phpoc Class

This class is a basic class of Phpoc Library.

Requirements

- Phpoc Library

Member Functions

- begin()
- localIP()
- subnetMask()
- gatewayIP()
- dnsServerIP()
- beginIP6()
- localIP6()
- globalIP6()
- gatewayIP6()
- dnsServerIP6()
- globalprefix6()

begin()

Description

Initializes the network parameters of PHPoC Shield for Arduino.

Syntax

Phpoc.begin()

Phpoc.begin(debug_flag)

Parameters

debug_flag - flags for debugging

Debug Flags	Descriptions
PF_LOG_SPI	debugging flag for SPI communication
PF_LOG_NET	debugging flag for network communication
PF_LOG_APP	debugging flag for applications such as sending an E-mail

Returns

1 - on success

0 - on failure

Example

```
#include <SPI.h>
#include <Phpoc.h>

void setup()
{
    Serial.begin(9600);

    if(Phpoc.begin() != 0)
        Serial.println("Success");
    else
        Serial.println("Fail");
}

void loop()
{
}
```

localIP()

Description

Returns the IP address of the device.

Syntax

Phpoc.localIP()

Parameters

none

Returns

the IP address of the device (IPAdress).

Example

```
#include <SPI.h>
#include <Phpoc.h>

void setup()
{
  Serial.begin(9600);

  if(Phpoc.begin() == 0)
  {
    Serial.println("Failed to initialize Network");
    for(;;)
      ;
  }
  Serial.println(Phpoc.localIP());
}

void loop()
{
}
```

subnetMask()

Description

Returns the subnet mask of the device.

Syntax

Phpoc.subnetMask()

Parameters

none

Returns

the subnet mask of the device (IPAddress).

Example

```
#include <SPI.h>
#include <Phpoc.h>

void setup()
{
  Serial.begin(9600);

  if(Phpoc.begin() == 0)
  {
    Serial.println("Failed to initialize Network");
    for(;;)
      ;
  }
  Serial.println(Phpoc.subnetMask());
}

void loop()
{
}
```

gatewayIP()

Description

Returns the gateway IP address for the device.

Syntax

Phpoc.gatewayIP()

Parameters

none

Returns

the gateway IP address for the device (IPAddress).

Example

```
#include <SPI.h>
#include <Phpoc.h>

void setup()
{
  Serial.begin(9600);

  if(Phpoc.begin() == 0)
  {
    Serial.println("Failed to initialize Network");
    for(;;)
      ;
  }
  Serial.println(Phpoc.gatewayIP());
}

void loop()
{
}
```

dnsServerIP()

Description

Returns the DNS server IP address for the device.

Syntax

Phpoc.dnsServerIP()

Parameters

none

Returns

the DNS server IP address for the device (IPAddress).

Example

```
#include <SPI.h>
#include <Phpoc.h>

void setup()
{
  Serial.begin(9600);

  if(Phpoc.begin() == 0)
  {
    Serial.println("Failed to initialize Network");
    for(;;)
      ;
  }
  Serial.println(Phpoc.dnsServerIP());
}

void loop()
{
}
```

beginIP6()

Description

Enables IPv6 feature.

Syntax

Phpoc.beginIP6()

Parameters

none

Returns

1 - on success
0 - on failure

example

```
#include <SPI.h>
#include <Phpoc.h>

void setup(){
  Serial.begin(9600);
  Phpoc.begin();
  if(Phpoc.beginIP6() != 0)
    Serial.println("Success");
  else
    Serial.println("Fail");
}

void loop(){
}
```

localIP6()

Description

Returns the IPv6 link local address of the device.

Syntax

Phpoc.localIP6()

Parameters

none

Returns

the IPv6 link local address of the device (IP6Adress).

Example

```
#include <SPI.h>
#include <Phpoc.h>

void setup(){
  Serial.begin(9600);
  if(Phpoc.begin() == 0){
    Serial.println("Failed to initialize Network");
    for(;;)
      ;
  }
  Phpoc.beginIP6();
  Serial.println(Phpoc.localIP6());
}

void loop(){
}
```

globalIP6()

Description

Returns the IPv6 global address of the device.

Syntax

Phpoc.globalIP6()

Parameters

none

Returns

the IPv6 global address of the device (IP6Adress).

Example

```
#include <SPI.h>
#include <Phpoc.h>

void setup(){
  Serial.begin(9600);
  if(Phpoc.begin() == 0){
    Serial.println("Failed to initialize Network");
    for(;;)
      ;
  }
  Phpoc.beginIP6();
  Serial.println(Phpoc.globalIP6());
}

void loop(){
}
```

gatewayIP6()

Description

Returns the IPv6 gateway address of the device.

Syntax

Phpoc.gatewayIP6()

Parameters

none

Returns

the IPv6 gateway address of the device (IP6Adress).

Example

```
#include <SPI.h>
#include <Phpoc.h>

void setup(){
  Serial.begin(9600);
  if(Phpoc.begin() == 0){
    Serial.println("Failed to initialize Network");
    for(;;)
      ;
  }
  Phpoc.beginIP6();
  Serial.println(Phpoc.gatewayIP6());
}

void loop(){
}
```

dnsServerIP6()

Description

Returns the IPv6 DNS server address of the device.

Syntax

Phpoc.dnsServerIP6()

Parameters

none

Returns

the IPv6 DNS server address of the device (IP6Adress).

Example

```
#include <SPI.h>
#include <Phpoc.h>

void setup(){
  Serial.begin(9600);
  if(Phpoc.begin() == 0){
    Serial.println("Failed to initialize Network");
    for(;;)
      ;
  }
  Phpoc.beginIP6();
  Serial.println(Phpoc.dnsServerIP6());
}

void loop(){
}
```

globalPrefix6()

Description

Returns the IPv6 prefix of the device.

Syntax

Phpoc.globalPrefix6()

Parameters

none

Returns

the IPv6 prefix of the device.

Example

```
#include <SPI.h>
#include <Phpoc.h>

void setup(){
  Serial.begin(9600);
  if(Phpoc.begin() == 0){
    Serial.println("Failed to initialize Network");
    for(;;)
      ;
  }
  Phpoc.beginIP6();
  Serial.println(Phpoc.globalPrefix6());
}

void loop(){
}
```

Server Class

This class is for using a PHPoC shield as a server.

Requirements

- Phpoc Library

Member Functions

- PhpocServer()
- begin()
- beginTelnet()
- beginWebSocket()
- beginWebSocketText()
- beginWebSocketBinary()
- available()
- write()

PhpocServer()

Description

Creates a server that listens for incoming connections on the specified port.

Syntax

PhpocServer(port)

Parameters

port - the port to listen on

Returns

none

Example

```
#include <Phpoc.h>

PhpocServer server(23);
boolean alreadyConnected = false; // whether or not the client was connected previously

void setup() {
  Serial.begin(9600);
  while(!Serial)
    ;

  // initialize PHPoC [WiFi] Shield:
  Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
  //Phpoc.begin();

  // start listening for TCP clients:
  server.begin();

  // print IP address of PHPoC [WiFi] Shield to serial monitor:
  Serial.print("Chat server address : ");
  Serial.println(Phpoc.localIP());
}

void loop() {
  // wait for a new client:
  PhpocClient client = server.available();

  // when the client sends the first byte, say hello:
  if (client) {
    if (!alreadyConnected) {
```

```
        // clear out the transmission buffer:
        client.flush();
        Serial.println("We have a new client");
        client.println("Hello, client!");
        alreadyConnected = true;
      }

      if (client.available() > 0) {
        // read the bytes incoming from the client:
        char thisChar = client.read();
        // echo the bytes back to all connected clients:
        server.write(thisChar);
        // echo the bytes to the server as well:
        Serial.write(thisChar);
      }
    }
  }
```

begin()

Description

Tells the server to begin listening for incoming connections.

Syntax

server.begin()

Parameters

none

Returns

none

Example

```
#include <Phpoc.h>

PhpocServer server(23);
boolean alreadyConnected = false; // whether or not the client was connected previously

void setup() {
  Serial.begin(9600);
  while(!Serial)
    ;

  // initialize PHPoC [WiFi] Shield:
  Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
  //Phpoc.begin();

  // start listening for TCP clients:
  server.begin();

  // print IP address of PHPoC [WiFi] Shield to serial monitor:
  Serial.print("Chat server address : ");
  Serial.println(Phpoc.localIP());
}

void loop() {
  // wait for a new client:
  PhpocClient client = server.available();

  // when the client sends the first byte, say hello:
  if (client) {
    if (!alreadyConnected) {
```

```
      // clear out the transmission buffer:
      client.flush();
      Serial.println("We have a new client");
      client.println("Hello, client!");
      alreadyConnected = true;
    }

    if (client.available() > 0) {
      // read the bytes incoming from the client:
      char thisChar = client.read();
      // echo the bytes back to all connected clients:
      server.write(thisChar);
      // echo the bytes to the server as well:
      Serial.write(thisChar);
    }
  }
}
```

beginTelnet()

Description

Tells the server to begin listening for incoming TELNET connections.

Syntax

server.beginTelnet()

Parameters

none

Returns

none

Example

```
#include <Phpoc.h>

PhpocServer server(23);
boolean alreadyConnected = false; // whether or not the client was connected previously

void setup() {
  Serial.begin(9600);
  while(!Serial)
    ;

  // initialize PHPoC [WiFi] Shield:
  Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
  //Phpoc.begin();

  // beginTelnet() enables telnet option negotiation & "character at a time".
  // In "character at a time" mode, text typed is immediately sent to server.
  server.beginTelnet();

  // print IP address of PHPoC [WiFi] Shield to serial monitor:
  Serial.print("Telnet server address : ");
  Serial.println(Phpoc.localIP());
}

void loop() {
  // wait for a new client:
  PhpocClient client = server.available();

  // when the client sends the first byte, say hello:
  if (client) {
```

```
    if (!alreadyConnected) {
      // clear out the transmission buffer:
      client.flush();
      Serial.println("We have a new client");
      client.println("Hello, client!");
      alreadyConnected = true;
    }

    if (client.available() > 0) {
      // read the bytes incoming from the client:
      char thisChar = client.read();
      // echo the bytes back to all connected clients:
      server.write(thisChar);
      // echo the bytes to the server as well:
      Serial.write(thisChar);
    }
  }
}
```

beginWebSocket()

Description

Tells the server to begin listening for an incoming Web Socket connection.

Syntax

server.beginWebSocket(path)

Parameters

path - URI of the web socket

Returns

none

Example

```
#include <Phpoc.h>

PhpocServer server(80);

void setup() {
  Serial.begin(9600);
  while(!Serial)
    ;

  // initialize PHPoC [WiFi] Shield:
  Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
  //Phpoc.begin();

  // start WebSocket server
  server.beginWebSocket("remote_push");

  // print IP address of PHPoC [WiFi] Shield to serial monitor:
  Serial.print("WebSocket server address : ");
  Serial.println(Phpoc.localIP());
}

void loop() {
  // wait for a new client:
  PhpocClient client = server.available();

  if (client) {
    if (client.available() > 0) {
      // read a byte incoming from the client:
      char thisChar = client.read();
```

```
        // when an user presses a button on the web apps, the web app sends an
        // uppercase character corresponding with the name of button to Arduino.
        // When an user releases a button on the web apps, the web app sends a
        // lowercase character corresponding with the name of button to Arduino.
        if(thisChar == 'A')
          Serial.println("button A press");
        if(thisChar == 'a')
          Serial.println("button A release");
        if(thisChar == 'B')
          Serial.println("button B press");
        if(thisChar == 'b')
          Serial.println("button B release");
        if(thisChar == 'C')
          Serial.println("button C press");
        if(thisChar == 'c')
          Serial.println("button C release");
      }
    }
}
```

beginWebSocketText()

Description

Tells the server to begin listening for an incoming Web Socket connection in text transmission mode.

Syntax

server.beginWebSocketText(path)
server.beginWebSocketText(path, proto)

Parameters

path - URI of the web socket
proto - protocol of the web socket

Returns

none

Example

```
#include <Phpoc.h>

PhpocServer server(80);

void setup() {
  Serial.begin(9600);
  while(!Serial)
    ;

  // initialize PHPoC [WiFi] Shield:
  Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
  //Phpoc.begin();

  // start WebSocket server
  server.beginWebSocketText("remote_push");

  // print IP address of PHPoC [WiFi] Shield to serial monitor:
  Serial.print("WebSocket server address : ");
  Serial.println(Phpoc.localIP());
}

void loop() {
  // wait for a new client:
  PhpocClient client = server.available();

  if (client) {
```

```
    if (client.available() > 0) {
      // read a byte incoming from the client:
      char thisChar = client.read();

      // when an user presses a button on the web apps, the web app sends an
      // uppercase character corresponding with the name of button to Arduino.
      // When an user releases a button on the web apps, the web app sends a
      // lowercase character corresponding with the name of button to Arduino.
      if(thisChar == 'A')
        Serial.println("button A press");
      if(thisChar == 'a')
        Serial.println("button A release");
      if(thisChar == 'B')
        Serial.println("button B press");
      if(thisChar == 'b')
        Serial.println("button B release");
      if(thisChar == 'C')
        Serial.println("button C press");
      if(thisChar == 'c')
        Serial.println("button C release");
    }
  }
}
```

beginWebSocketBinary()

Description

Tells the server to begin listening for an incoming Web Socket connection in binary transmission mode.

Syntax

server.beginWebSocketBinary(path)
server.beginWebSocketBinary(path, proto)

Parameters

path - URI of the web socket
proto - protocol of the web socket

Returns

none

Example

```
#include <Phpoc.h>

PhpocServer server(80);

void setup() {
  Serial.begin(9600);
  while(!Serial)
    ;

  // initialize PHPoC [WiFi] Shield:
  Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
  //Phpoc.begin();

  // start WebSocket server
  server.beginWebSocketBinary("remote_push");

  // print IP address of PHPoC [WiFi] Shield to serial monitor:
  Serial.print("WebSocket server address : ");
  Serial.println(Phpoc.localIP());
}

void loop() {
  // wait for a new client:
  PhpocClient client = server.available();

  if (client) {
```

```
    if (client.available() > 0) {
      // read a byte incoming from the client:
      char thisChar = client.read();

      // when an user presses a button on the web apps, the web app sends an
      // uppercase character corresponding with the name of button to Arduino.
      // When an user releases a button on the web apps, the web app sends a
      // lowercase character corresponding with the name of button to Arduino.
      if(thisChar == 'A')
        Serial.println("button A press");
      if(thisChar == 'a')
        Serial.println("button A release");
      if(thisChar == 'B')
        Serial.println("button B press");
      if(thisChar == 'b')
        Serial.println("button B release");
      if(thisChar == 'C')
        Serial.println("button C press");
      if(thisChar == 'c')
        Serial.println("button C release");
    }
  }
}
```

available()

Description

Gets a client that is connected to the server and has data available for reading.

Syntax

server.available()

Parameters

none

Returns

a client object - on success
false - on failure

```
#include <Phpoc.h>

PhpocServer server(23);
boolean alreadyConnected = false; // whether or not the client was connected previously

void setup() {
  Serial.begin(9600);
  while(!Serial)
    ;

  // initialize PHPoC [WiFi] Shield:
  Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
  //Phpoc.begin();

  // start listening for TCP clients:
  server.begin();

  // print IP address of PHPoC [WiFi] Shield to serial monitor:
  Serial.print("Chat server address : ");
  Serial.println(Phpoc.localIP());
}

void loop() {
  // wait for a new client:
  PhpocClient client = server.available();

  // when the client sends the first byte, say hello:
  if (client) {
    if (!alreadyConnected) {
      // clear out the transmission buffer:
      client.flush();
```

```
      Serial.println("We have a new client");
      client.println("Hello, client!");
      alreadyConnected = true;
    }

    if (client.available() > 0) {
      // read the bytes incoming from the client:
      char thisChar = client.read();
      // echo the bytes back to all connected clients:
      server.write(thisChar);
      // echo the bytes to the server as well:
      Serial.write(thisChar);
    }
  }
}
```

write()

Description

Writes data to all the clients connected to a server.
This data is sent as a byte or series of bytes.

Syntax

server.write(val)
server.write(buf, len)

Parameters

val - a value to send as a single byte (byte of char)
buf - an array to send as a series of bytes (byte or char)
len - the length of the buffer

Returns

Returns an int represents the number of bytes written.

Example

```
#include <Phpoc.h>

PhpocServer server(23);
boolean alreadyConnected = false; // whether or not the client was connected previously

void setup() {
  Serial.begin(9600);
  while(!Serial)
    ;

  // initialize PHPoC [WiFi] Shield:
  Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
  //Phpoc.begin();

  // start listening for TCP clients:
  server.begin();

  // print IP address of PHPoC [WiFi] Shield to serial monitor:
  Serial.print("Chat server address : ");
  Serial.println(Phpoc.localIP());
}

void loop() {
  // wait for a new client:
  PhpocClient client = server.available();
```

```
  // when the client sends the first byte, say hello:
  if (client) {
    if (!alreadyConnected) {
      // clear out the transmission buffer:
      client.flush();
      Serial.println("We have a new client");
      client.println("Hello, client!");
      alreadyConnected = true;
    }

    if (client.available() > 0) {
      // read the bytes incoming from the client:
      char thisChar = client.read();
      // echo the bytes back to all connected clients:
      server.write(thisChar);
      // echo the bytes to the server as well:
      Serial.write(thisChar);
    }
  }
}
```

Client Class

This class is for using a PHPoC shield as a client.

Requirements

- Phpoc Library

Member Functions

- PhpocClient()
- connected()
- connect()
- connectSSL()
- write()
- available()
- read()
- readLine()
- flush()
- stop()

PhpocClient()

Description

Creates a client which can connect to a server with specified internet IP address and port.

Syntax

PhpocClient()

Parameters

none

Returns

none

Example

```
#include <Phpoc.h>

// hostname of web server:
char server_name[] = "example.phpoc.com";
PhpocClient client;

void setup() {
  Serial.begin(9600);
  while(!Serial)
    ;

  Serial.println("Sending GET request to web server");

  // initialize PHPoC [WiFi] Shield:
  Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
  //Phpoc.begin();

  // connect to web server on port 80:
  if(client.connect(server_name, 80))
  {
    // if connected:
    Serial.println("Connected to server");
    // make a HTTP request:
    //client.println("GET / HTTP/1.0");
    client.println("GET /asciilogo.txt HTTP/1.0");
    //client.println("GET /remote_addr/ HTTP/1.0");
    client.println("Host: example.phpoc.com");
    client.println();
  }
```

```
  else // if not connected:
    Serial.println("connection failed");
}

void loop() {
  if(client.available())
  {
    // if there is an incoming byte from the server, read them and print them to
    // serial monitor:
    char c = client.read();
    Serial.print(c);
  }

  if(!client.connected())
  {
    // if the server's disconnected, stop the client:
    Serial.println("disconnected");
    client.stop();
    // do nothing forevermore:
    while(true)
      ;
  }
}
```

connected()

Description

Checks if the client is connected or not.
Note that a client might be considered connected if there is still unread data although the connection has been closed.

Syntax

client.connected()

Parameters

none

Returns

true - when client is connected
false - when client is not connected

Example

```
#include <Phpoc.h>

// hostname of web server:
char server_name[] = "example.phpoc.com";
PhpocClient client;

void setup() {
  Serial.begin(9600);
  while(!Serial)
    ;

  Serial.println("Sending GET request to web server");

  // initialize PHPoC [WiFi] Shield:
  Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
  //Phpoc.begin();

  // connect to web server on port 80:
  if(client.connect(server_name, 80))
  {
    // if connected:
    Serial.println("Connected to server");
    // make a HTTP request:
    //client.println("GET / HTTP/1.0");
    client.println("GET /asciilogo.txt HTTP/1.0");
    //client.println("GET /remote_addr/ HTTP/1.0");
```

```
    client.println("Host: example.phpoc.com");
    client.println();
  }
  else // if not connected:
    Serial.println("connection failed");
}

void loop() {
  if(client.available())
  {
    // if there is an incoming byte from the server, read them and print them to
    // serial monitor:
    char c = client.read();
    Serial.print(c);
  }

  if(!client.connected())
  {
    // if the server's disconnected, stop the client:
    Serial.println("disconnected");
    client.stop();
    // do nothing forevermore:
    while(true)
      ;
  }
}
```

connect()

Description

Connects to a server with specified IP address(or hostname) and port.

Syntax

client.connect(ip_addr, port)
client.connect(hostname, port)

Parameters

ip_addr - the IP address which the client will connect to
port - the port number that the client will connect to
hostname - the hostname that the client will connect to

Returns

1 - on success
0 - on failure

Example

```
#include <Phpoc.h>

// hostname of web server:
char server_name[] = "example.phpoc.com";
PhpocClient client;

void setup() {
  Serial.begin(9600);
  while(!Serial)
    ;

  Serial.println("Sending GET request to web server");

  // initialize PHPoC [WiFi] Shield:
  Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
  //Phpoc.begin();

  // connect to web server on port 80:
  if(client.connect(server_name, 80))
  {
    // if connected:
    Serial.println("Connected to server");
    // make a HTTP request:
    //client.println("GET / HTTP/1.0");
    client.println("GET /asciilogo.txt HTTP/1.0");
```

```
    //client.println("GET /remote_addr/ HTTP/1.0");
    client.println("Host: example.phpoc.com");
    client.println();
  }
  else // if not connected:
    Serial.println("connection failed");
}

void loop() {
  if(client.available())
  {
    // if there is an incoming byte from the server, read them and print them to
    // serial monitor:
    char c = client.read();
    Serial.print(c);
  }

  if(!client.connected())
  {
    // if the server's disconnected, stop the client:
    Serial.println("disconnected");
    client.stop();
    // do nothing forevermore:
    while(true)
      ;
  }
}
```

connectSSL()

Description

Connects to an SSL server with specified IP address(or hostname) and port.

Syntax

client.connectSSL(ip_addr, port)
client.connectSSL(hostname, port)

Parameters

ip_addr - the IP address which the SSL client will connect to
port - the port number that the SSL client will connect to
hostname - the hostname that the SSL client will connect to

Returns

1 - on success
0 - on failure

Example

```
#include <Phpoc.h>

// hostname of web server:
char server_name[] = "example.phpoc.com";

PhpocClient client;

void setup() {
  Serial.begin(9600);
  while(!Serial)
  ;

  Serial.println("Sending GET request to SSL web server");

  // initialize PHPoC [WiFi] Shield:
  Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
  //Phpoc.begin();

  // connect to web server on port 443:
  if(client.connectSSL(server_name, 443)) {
    // if connected:
    Serial.println("Connected to server");
    // make a HTTP request:
    //client.println("GET / HTTP/1.0");
    client.println("GET /asciilogo.txt HTTP/1.0");
```

```
    //client.println("GET /remote_addr/ HTTP/1.0");
    client.println("Host: example.phpoc.com");
    client.println();
  }
}

void loop() {
  // if there are incoming bytes available from the server, read them and print
  // them:
  while (client.available()) {
    char c = client.read();
    Serial.write(c);
  }

  // if the server's disconnected, stop the client:
  if (!client.connected()) {
    // if the server's disconnected, stop the client:
    Serial.println();
    Serial.println("disconnecting from server.");
    client.stop();

    // do nothing forevermore:
    while (true)
      ;
  }
}
```

write()

Description

Writes data to the server the client is connected to.
This data is sent as a byte or series of bytes.

Syntax

client.write(val)
client.write(buf, len)

Parameters

val - a value to send as a single byte (byte of char)
buf - an array to send as a series of bytes (byte or char)
len - the length of the buffer

Returns

Returns an int represents the number of bytes written.

```
#include <Phpoc.h>

// hostname of web server:
char server_name[] = "example.phpoc.com";
PhpocClient client;

void setup() {
  Serial.begin(9600);
  while(!Serial)
    ;

  Serial.println("Sending GET request to web server");

  // initialize PHPoC [WiFi] Shield:
  Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
  //Phpoc.begin();

  // connect to web server on port 80:
  if(client.connect(server_name, 80))
  {
    // if connected:
    Serial.println("Connected to server");
    // make a HTTP request:
    //client.println("GET / HTTP/1.0");
    client.write("GET /asciilogo.txt HTTP/1.0₩r₩n");
    //client.write("GET /remote_addr/ HTTP/1.0₩r₩n");
    client.write("Host: example.phpoc.com₩r₩n");
    client.write("₩r₩n");
```

```
  }
  else // if not connected:
    Serial.println("connection failed");
}

void loop() {
  if(client.available())
  {
    // if there is an incoming byte from the server, read them and print them to
    // serial monitor:
    char c = client.read();
    Serial.print(c);
  }

  if(!client.connected())
  {
    // if the server's disconnected, stop the client:
    Serial.println("disconnected");
    client.stop();
    // do nothing forevermore:
    while(true)
      ;
  }
}
```

available()

Description

Returns the number of bytes available to read from the server which is connected to.

Syntax

client.available()

Parameters

none

Returns

Returns an int represents the number of bytes available.

Example

```
#include <Phpoc.h>

// hostname of web server:
char server_name[] = "example.phpoc.com";
PhpocClient client;

void setup() {
  Serial.begin(9600);
  while(!Serial)
    ;

  Serial.println("Sending GET request to web server");

  // initialize PHPoC [WiFi] Shield:
  Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
  //Phpoc.begin();

  // connect to web server on port 80:
  if(client.connect(server_name, 80))
  {
    // if connected:
    Serial.println("Connected to server");
    // make a HTTP request:
    //client.println("GET / HTTP/1.0");
    client.println("GET /asciilogo.txt HTTP/1.0");
    //client.println("GET /remote_addr/ HTTP/1.0");
    client.println("Host: example.phpoc.com");
    client.println();
  }
```

```
  else // if not connected:
    Serial.println("connection failed");
}

void loop() {
  if(client.available())
  {
    // if there is an incoming byte from the server, read them and print them to
    // serial monitor:
    char c = client.read();
    Serial.print(c);
  }

  if(!client.connected())
  {
    // if the server's disconnected, stop the client:
    Serial.println("disconnected");
    client.stop();
    // do nothing forevermore:
    while(true)
      ;
  }
}
```

read()

Description

Reads the next byte received from the server the client is connected to.

Syntax

client.read()

Parameters

none

Returns

the next byte - on success
-1 - on failure

Example

```
#include <Phpoc.h>

// hostname of web server:
char server_name[] = "example.phpoc.com";
PhpocClient client;

void setup() {
  Serial.begin(9600);
  while(!Serial)
    ;

  Serial.println("Sending GET request to web server");

  // initialize PHPoC [WiFi] Shield:
  Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
  //Phpoc.begin();

  // connect to web server on port 80:
  if(client.connect(server_name, 80))
  {
    // if connected:
    Serial.println("Connected to server");
    // make a HTTP request:
    //client.println("GET / HTTP/1.0");
    client.println("GET /asciilogo.txt HTTP/1.0");
    //client.println("GET /remote_addr/ HTTP/1.0");
    client.println("Host: example.phpoc.com");
    client.println();
```

```
  }
  else // if not connected:
    Serial.println("connection failed");
}

void loop() {
  if(client.available())
  {
    // if there is an incoming byte from the server, read them and print them to
    // serial monitor:
    char c = client.read();
    Serial.print(c);
  }

  if(!client.connected())
  {
    // if the server's disconnected, stop the client:
    Serial.println("disconnected");
    client.stop();
    // do nothing forevermore:
    while(true)
      ;
  }
}
```

readLine()

Description

Reads line based data from the server the client is connected to.
The line based data means data are finished to CR(0x0d) and LF(0x0a).

Syntax

client.readLine()
client.readLine(buf, size)

Parameters

buf - buffer to store reading data
size - length(bytes) of buffer

Returns

the length of line - on success
0 - on failure

Example

```
#include <Phpoc.h>

PhpocServer server(80);

char slideName;
int slideValue;

void setup() {
  Serial.begin(9600);
  while(!Serial)
    ;

  // initialize PHPoC [WiFi] Shield:
  Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
  //Phpoc.begin();

  // start WebSocket server
  server.beginWebSocket("remote_slide");

  // print IP address of PHPoC [WiFi] Shield to serial monitor:
  Serial.print("WebSocket server address : ");
  Serial.println(Phpoc.localIP());
}

void loop() {
```

```
  // wait for a new client:
  PhpocClient client = server.available();

  if (client) {
    // read a string that is terminated by a carriage return and a newline
    // characters:
    String slideStr = client.readLine();

    if(slideStr)
    {
      // when an user moves a slider on web app, web app sends a string to
      // Arduino. The first character of string is slide name, followed by slide
      // value and terminated by a carriage return and a newline characters.
      slideName = slideStr.charAt(0);
      slideValue = slideStr.substring(1).toInt();

      Serial.print(slideName);
      Serial.print('/');
      Serial.println(slideValue);
    }
  }
}
```

flush()

Description

Waits until all outgoing data in buffer have been sent.

Syntax

client.flush()

Parameters

none

Returns

none

Example

```
#include <Phpoc.h>

PhpocServer server(23);
boolean alreadyConnected = false; // whether or not the client was connected previously

void setup() {
  Serial.begin(9600);
  while(!Serial)
    ;

  // initialize PHPoC [WiFi] Shield:
  Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
  //Phpoc.begin();

  // start listening for TCP clients:
  server.begin();

  // print IP address of PHPoC [WiFi] Shield to serial monitor:
  Serial.print("Chat server address : ");
  Serial.println(Phpoc.localIP());
}

void loop() {
  // wait for a new client:
  PhpocClient client = server.available();

  // when the client sends the first byte, say hello:
  if (client) {
    if (!alreadyConnected) {
```

```
        // clear out the transmission buffer:
        client.flush();
        Serial.println("We have a new client");
        client.println("Hello, client!");
        alreadyConnected = true;
      }

      if (client.available() > 0) {
        // read the bytes incoming from the client:
        char thisChar = client.read();
        // echo the bytes back to all connected clients:
        server.write(thisChar);
        // echo the bytes to the server as well:
        Serial.write(thisChar);
      }
    }
  }
```

stop()

Description

Disconnects from the server.

Syntax

client.stop()

Parameters

none

Returns

none

Example

```
#include <Phpoc.h>

// hostname of web server:
char server_name[] = "example.phpoc.com";
PhpocClient client;

void setup() {
  Serial.begin(9600);
  while(!Serial)
    ;

  Serial.println("Sending GET request to web server");

  // initialize PHPoC [WiFi] Shield:
  Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
  //Phpoc.begin();

  // connect to web server on port 80:
  if(client.connect(server_name, 80))
  {
    // if connected:
    Serial.println("Connected to server");
    // make a HTTP request:
    //client.println("GET / HTTP/1.0");
    client.println("GET /asciilogo.txt HTTP/1.0");
    //client.println("GET /remote_addr/ HTTP/1.0");
    client.println("Host: example.phpoc.com");
    client.println();
  }
```

```
  else // if not connected:
    Serial.println("connection failed");
}

void loop() {
  if(client.available())
  {
    // if there is an incoming byte from the server, read them and print them to
    // serial monitor:
    char c = client.read();
    Serial.print(c);
  }

  if(!client.connected())
  {
    // if the server's disconnected, stop the client:
    Serial.println("disconnected");
    client.stop();
    // do nothing forevermore:
    while(true)
      ;
  }
}
```

Email Class

This class is for sending an E-mail by a PHPoC shield.

Requirements

- Phpoc Library

Member Functions

- setOutgoingServer()
- setOutgoingLogin()
- setFrom()
- setTo()
- setSubject()
- beginMessage()
- endMessage()
- write()
- send()

setOutgoingServer()

Description

Sets the outgoing mail server.

Syntax

email.setOutgoingServer(hostname, port)

Parameters

hostname - hostname of the outgoing mail server
port - port number of the outgoing mail server

Returns

none

Example

```
#include <Phpoc.h>

PhpocEmail email;

void setup() {
  Serial.begin(9600);
  while(!Serial)
    ;

  // initialize PHPoC [WiFi] Shield:
  Phpoc.begin(PF_LOG_SPI | PF_LOG_NET | PF_LOG_APP);
  //Phpoc.begin();

  Serial.println("Sending email to gmail relay server");

  // [login using your private password]
  // Google may block sign-in attempts from some apps or devices that do not use
  // modern security standards.
  // Change your settings to allow less secure apps to access your account.
  // https://www.google.com/settings/security/lesssecureapps

  // [login using app password]
  // 1. turn on 2-step verification
  // 2. create app password
  // 3. apply app password as your login password

  // setup outgoing relay server - gmail.com:
  email.setOutgoingServer("smtp.gmail.com", 587);
```

```
  email.setOutgoingLogin("your_login_id", "your_login_password or app_password");

  // setup From/To/Subject:
  email.setFrom("from_email_address", "from_user_name");
  email.setTo("to_email_address", "to_user_name");
  email.setSubject("Mail from PHPoC Shield for Arduino");

  // write email message:
  email.beginMessage();
  email.println("Hello, world!");
  email.println("I am PHPoC Shield for Arduino");
  email.println("Good bye");
  email.endMessage();

  // send email:
  if(email.send() > 0)
    Serial.println("Email send ok");
  else
    Serial.println("Email send failed");
}

void loop() {
}
```

setOutgoingLogin()

Description

Sets log in information of the outgoing mail server.

Syntax

email.setOutgoingLogin(id, password)

Parameters

id - username or id of the account
password - password of the account

Returns

none

Example

```
#include <Phpoc.h>

PhpocEmail email;

void setup() {
  Serial.begin(9600);
  while(!Serial)
    ;

  // initialize PHPoC [WiFi] Shield:
  Phpoc.begin(PF_LOG_SPI | PF_LOG_NET | PF_LOG_APP);
  //Phpoc.begin();

  Serial.println("Sending email to gmail relay server");

  // [login using your private password]
  // Google may block sign-in attempts from some apps or devices that do not use
  // modern security standards.
  // Change your settings to allow less secure apps to access your account.
  // https://www.google.com/settings/security/lesssecureapps

  // [login using app password]
  // 1. turn on 2-step verification
  // 2. create app password
  // 3. apply app password as your login password

  // setup outgoing relay server - gmail.com:
  email.setOutgoingServer("smtp.gmail.com", 587);
```

```
  email.setOutgoingLogin("your_login_id", "your_login_password or app_password");

  // setup From/To/Subject:
  email.setFrom("from_email_address", "from_user_name");
  email.setTo("to_email_address", "to_user_name");
  email.setSubject("Mail from PHPoC Shield for Arduino");

  // write email message:
  email.beginMessage();
  email.println("Hello, world!");
  email.println("I am PHPoC Shield for Arduino");
  email.println("Good bye");
  email.endMessage();

  // send email:
  if(email.send() > 0)
    Serial.println("Email send ok");
  else
    Serial.println("Email send failed");
}

void loop() {
}
```

setFrom()

Description

Sets a sender's e-mail address and name

Syntax

email.setFrom(email_addr, name)

Parameters

email_addr - sender's e-mail address
name - sender's name

Returns

none

Example

```
#include <Phpoc.h>

PhpocEmail email;

void setup() {
  Serial.begin(9600);
  while(!Serial)
    ;

  // initialize PHPoC [WiFi] Shield:
  Phpoc.begin(PF_LOG_SPI | PF_LOG_NET | PF_LOG_APP);
  //Phpoc.begin();

  Serial.println("Sending email to server directly");

  // setup From/To/Subject:
  email.setFrom("from_email_address", "from_user_name");
  email.setTo("to_email_address", "to_user_name");
  email.setSubject("Mail from PHPoC Shield for Arduino");

  // write email message:
  email.beginMessage();
  email.println("Hello, world!");
  email.println("I am PHPoC Shield for Arduino");
  email.println("Good bye");
  email.endMessage();

  // send email:
```

```
  if(email.send() > 0)
    Serial.println("Email send ok");
  else
    Serial.println("Email send failed");
}

void loop() {
}
```

setTo()

Description

Sets a receiver's e-mail address and name

Syntax

email.setTo(email_addr, name)

Parameters

email_addr - receiver's e-mail address
name - receiver's name

Returns

none

Example

```
#include <Phpoc.h>

PhpocEmail email;

void setup() {
  Serial.begin(9600);
  while(!Serial)
    ;

  // initialize PHPoC [WiFi] Shield:
  Phpoc.begin(PF_LOG_SPI | PF_LOG_NET | PF_LOG_APP);
  //Phpoc.begin();

  Serial.println("Sending email to server directly");

  // setup From/To/Subject:
  email.setFrom("from_email_address", "from_user_name");
  email.setTo("to_email_address", "to_user_name");
  email.setSubject("Mail from PHPoC Shield for Arduino");

  // write email message:
  email.beginMessage();
  email.println("Hello, world!");
  email.println("I am PHPoC Shield for Arduino");
  email.println("Good bye");
  email.endMessage();

  // send email:
```

```
  if(email.send() > 0)
    Serial.println("Email send ok");
  else
    Serial.println("Email send failed");
}

void loop() {
}
```

setSubject()

Description

Sets a subject of the e-mail.

Syntax

email.setSubject(subject)

Parameters

subject - subject of the e-mail

Returns

none

Example

```
#include <Phpoc.h>

PhpocEmail email;

void setup() {
  Serial.begin(9600);
  while(!Serial)
    ;

  // initialize PHPoC [WiFi] Shield:
  Phpoc.begin(PF_LOG_SPI | PF_LOG_NET | PF_LOG_APP);
  //Phpoc.begin();

  Serial.println("Sending email to server directly");

  // setup From/To/Subject:
  email.setFrom("from_email_address", "from_user_name");
  email.setTo("to_email_address", "to_user_name");
  email.setSubject("Mail from PHPoC Shield for Arduino");

  // write email message:
  email.beginMessage();
  email.println("Hello, world!");
  email.println("I am PHPoC Shield for Arduino");
  email.println("Good bye");
  email.endMessage();

  // send email:
  if(email.send() > 0)
```

```
    Serial.println("Email send ok");
  else
    Serial.println("Email send failed");
}

void loop() {
}
```

beginMessage()

Description

Gets ready to put contents into the e-mail body.
The write(), print() or println() can be used for writing the e-mail after calling this function.
To end writing e-mail, endMassage() is used.

Syntax

email.beginMessage()

Parameters

none

Returns

none

Example

```
#include <Phpoc.h>

PhpocEmail email;

void setup() {
  Serial.begin(9600);
  while(!Serial)
    ;

  // initialize PHPoC [WiFi] Shield:
  Phpoc.begin(PF_LOG_SPI | PF_LOG_NET | PF_LOG_APP);
  //Phpoc.begin();

  Serial.println("Sending email to server directly");

  // setup From/To/Subject:
  email.setFrom("from_email_address", "from_user_name");
  email.setTo("to_email_address", "to_user_name");
  email.setSubject("Mail from PHPoC Shield for Arduino");

  // write email message:
  email.beginMessage();
  email.println("Hello, world!");
  email.println("I am PHPoC Shield for Arduino");
  email.println("Good bye");
  email.endMessage();
```

```
  // send email:
  if(email.send() > 0)
    Serial.println("Email send ok");
  else
    Serial.println("Email send failed");
}

void loop() {
}
```

endMessage()

Description

Gets finished putting contents into the e-mail body.

> Calling this function after writing messages is highly recommended. If you don't use this function, you may loose the last line of the messages.

Syntax

email.endMessage()

Parameters

none

Returns

none

Example

```
#include <Phpoc.h>

PhpocEmail email;

void setup() {
  Serial.begin(9600);
  while(!Serial)
    ;

  // initialize PHPoC [WiFi] Shield:
  Phpoc.begin(PF_LOG_SPI | PF_LOG_NET | PF_LOG_APP);
  //Phpoc.begin();

  Serial.println("Sending email to server directly");

  // setup From/To/Subject:
  email.setFrom("from_email_address", "from_user_name");
  email.setTo("to_email_address", "to_user_name");
  email.setSubject("Mail from PHPoC Shield for Arduino");

  // write email message:
  email.beginMessage();
  email.println("Hello, world!");
  email.println("I am PHPoC Shield for Arduino");
```

```
  email.println("Good bye");
  email.endMessage();

  // send email:
  if(email.send() > 0)
    Serial.println("Email send ok");
  else
    Serial.println("Email send failed");
}

void loop() {
}
```

write()

Description

Writes data to e-mail body.
This data can be written as a byte or series of bytes.

Syntax

email.write(val)
email.write(buf, len)

Parameters

val - a value to write as a single byte
buf - an array to write as series of byes
len - the length of the buffer

Returns

Returns an int represents the number of bytes written.

Example

```
#include <Phpoc.h>

PhpocEmail email;

void setup() {
  Serial.begin(9600);
  while(!Serial)
    ;

  // initialize PHPoC [WiFi] Shield:
  Phpoc.begin(PF_LOG_SPI | PF_LOG_NET | PF_LOG_APP);
  //Phpoc.begin();

  Serial.println("Sending email to server directly");

  // setup From/To/Subject:
  email.setFrom("from_email_address", "from_user_name");
  email.setTo("to_email_address", "to_user_name");
  email.setSubject("Mail from PHPoC Shield for Arduino");

  // write email message:
  email.beginMessage();
  email.write("Hello, world!₩r₩n");
  email.write("I am PHPoC Shield for Arduino₩r₩n");
  email.write("Good bye₩r₩n");
```

```
  email.endMessage();

  // send email:
  if(email.send() > 0)
    Serial.println("Email send ok");
  else
    Serial.println("Email send failed");
}

void loop() {
}
```

send()

Description

Sends an e-mail.

> Before calling this function, you need to set required parameters such as e-mail addresses of receiver and sender.

Syntax

email.send()

Parameters

none

Returns

1 - on success
0 - on failure

Example

```
#include <Phpoc.h>

PhpocEmail email;

void setup() {
  Serial.begin(9600);
  while(!Serial)
    ;

  // initialize PHPoC [WiFi] Shield:
  Phpoc.begin(PF_LOG_SPI | PF_LOG_NET | PF_LOG_APP);
  //Phpoc.begin();

  Serial.println("Sending email to server directly");

  // setup From/To/Subject:
  email.setFrom("from_email_address", "from_user_name");
  email.setTo("to_email_address", "to_user_name");
  email.setSubject("Mail from PHPoC Shield for Arduino");

  // write email message:
  email.beginMessage();
  email.println("Hello, world!");
```

```
  email.println("I am PHPoC Shield for Arduino");
  email.println("Good bye");
  email.endMessage();

  // send email:
  if(email.send() > 0)
    Serial.println("Email send ok");
  else
    Serial.println("Email send failed");
}

void loop() {
}
```

DateTime Class

This class is for getting date and time on a PHPoC shield.

Requirements

- Phpoc Library

Member Functions

- date()
- year()
- month()
- day()
- dayofWeek()
- hour()
- minute()
- second()

date()

Description

Gets the current date and time from RTC of PHPoC Shield for Arduino.

Syntax

datetime.date()
datetime.date(format)

Parameters

format - time format

Format	Description
Y	A full numeric representation of a year, 4 digits (example: 2016)
y	A two digit representation of a year (example: 16)
M	A short texture representation of a month, three letters (example: Mar)
m	Numeric representation of a month with leading zeros (example: 03)
n	Numeric representation of a month without leading zeros (example: 3)
d	Day of the month, 2 digits with leading zeros (01 to 31)
j	Day of the month without leading zeros (1 to 31)
D	A textual representation of a day, three letters (example: Mon)
g	12-hour format of an hour without leading zeros (1 to 12)
G	24-hour format of an hour without leading zeros (0 to 23)
h	12-hour format of an hour with leading zeros (01 to 12)
H	24-hour format of an hour with leading zeros (00 to 23)
i	Minutes with leading zeros (00 to 59)
s	Seconds with leading zeros (00 to 59)
a	Lowercase Ante meridiem and Post meridiem (am or pm)
A	Uppercase Ante meridiem and Post meridiem (AM or PM)

Returns

With a given format, it returns a formatted string represents the current date and time.
Without a given format, it returns a string with the last given format.
The default format is "D M j H:i:s".

Example

```
#include <Phpoc.h>

PhpocDateTime datetime;

void setup() {
  Serial.begin(9600);
  while(!Serial)
    ;
```

```
  // initialize PHPoC [WiFi] Shield:
  Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);

  Serial.println("Get year/month/day/dayofWeek/hour/minute/second from RTC in PHPoC Shield");

  Serial.print(datetime.year());
  Serial.print('-');
  Serial.print(datetime.month());
  Serial.print('-');
  Serial.print(datetime.day());
  Serial.print(' ');
  Serial.print(datetime.dayofWeek());
  Serial.print(' ');
  Serial.print(datetime.hour());
  Serial.print(':');
  Serial.print(datetime.minute());
  Serial.print(':');
  Serial.print(datetime.second());
  Serial.println();

  // set date and time format:
  datetime.date(F("Y-m-d H:i:s"));
}

void loop() {
  Serial.println(datetime.date());
  delay(10);
}
```

year()

Description

Gets the current year from RTC of PHPoC Shield for Arduino.

Syntax

datetime.year()

Parameters

none

Returns

the current year - on success (2000 ~ 2099)
0 - on failure

Example

```
#include <Phpoc.h>

PhpocDateTime datetime;

void setup() {
  Serial.begin(9600);
  while(!Serial)
    ;

  // initialize PHPoC [WiFi] Shield:
  Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);

  Serial.println("Get year/month/day/dayofWeek/hour/minute/second from RTC in PHPoC Shield");

  Serial.print(datetime.year());
  Serial.print('-');
  Serial.print(datetime.month());
  Serial.print('-');
  Serial.print(datetime.day());
  Serial.print(' ');
  Serial.print(datetime.dayofWeek());
  Serial.print(' ');
  Serial.print(datetime.hour());
  Serial.print(':');
  Serial.print(datetime.minute());
  Serial.print(':');
  Serial.print(datetime.second());
  Serial.println();
```

```
  // set date and time format:
  datetime.date(F("Y-m-d H:i:s"));
}

void loop() {
  Serial.println(datetime.date());
  delay(10);
}
```

month()

Description

Gets the current month from RTC of PHPoC Shield for Arduino.

Syntax

datetime.month()

Parameters

none

Returns

the current month - on success (1~12)
0 - on failure

Example

```
#include <Phpoc.h>

PhpocDateTime datetime;

void setup() {
  Serial.begin(9600);
  while(!Serial)
    ;

  // initialize PHPoC [WiFi] Shield:
  Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);

  Serial.println("Get year/month/day/dayofWeek/hour/minute/second from RTC in PHPoC Shield");

  Serial.print(datetime.year());
  Serial.print('-');
  Serial.print(datetime.month());
  Serial.print('-');
  Serial.print(datetime.day());
  Serial.print(' ');
  Serial.print(datetime.dayofWeek());
  Serial.print(' ');
  Serial.print(datetime.hour());
  Serial.print(':');
  Serial.print(datetime.minute());
  Serial.print(':');
  Serial.print(datetime.second());
  Serial.println();
```

```
  // set date and time format:
  datetime.date(F("Y-m-d H:i:s"));
}

void loop() {
  Serial.println(datetime.date());
  delay(10);
}
```

day()

Description

Gets the current day from RTC of PHPoC Shield for Arduino.

Syntax

datetime.day()

Parameters

none

Returns

the current day - on success (1~31)
0 - on failure

Example

```
#include <Phpoc.h>

PhpocDateTime datetime;

void setup() {
  Serial.begin(9600);
  while(!Serial)
    ;

  // initialize PHPoC [WiFi] Shield:
  Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);

  Serial.println("Get year/month/day/dayofWeek/hour/minute/second from RTC in PHPoC Shield");

  Serial.print(datetime.year());
  Serial.print('-');
  Serial.print(datetime.month());
  Serial.print('-');
  Serial.print(datetime.day());
  Serial.print(' ');
  Serial.print(datetime.dayofWeek());
  Serial.print(' ');
  Serial.print(datetime.hour());
  Serial.print(':');
  Serial.print(datetime.minute());
  Serial.print(':');
  Serial.print(datetime.second());
  Serial.println();
```

```
  // set date and time format:
  datetime.date(F("Y-m-d H:i:s"));
}

void loop() {
  Serial.println(datetime.date());
  delay(10);
}
```

dayofWeek()

Description

Gets the current day of week from RTC of PHPoC Shield for Arduino.

Syntax

datetime.dayofWeek()

Parameters

none

Returns

the current day of week - on success (1 ~ 7)

Return	Day of Week
1	Monday
2	Tuesday
3	Wednesday
4	Thursday
5	Friday
6	Saturday
7	Sunday

0 - on failure

Example

```
#include <Phpoc.h>

PhpocDateTime datetime;

void setup() {
  Serial.begin(9600);
  while(!Serial)
    ;

  // initialize PHPoC [WiFi] Shield:
  Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);

  Serial.println("Get year/month/day/dayofWeek/hour/minute/second from RTC in PHPoC Shield");

  Serial.print(datetime.year());
  Serial.print('-');
  Serial.print(datetime.month());
  Serial.print('-');
  Serial.print(datetime.day());
```

```
  Serial.print(' ');
  Serial.print(datetime.dayofWeek());
  Serial.print(' ');
  Serial.print(datetime.hour());
  Serial.print(':');
  Serial.print(datetime.minute());
  Serial.print(':');
  Serial.print(datetime.second());
  Serial.println();

  // set date and time format:
  datetime.date(F("Y-m-d H:i:s"));
}

void loop() {
  Serial.println(datetime.date());
  delay(10);
}
```

hour()

Description

Gets the current hour from RTC of PHPoC Shield for Arduino.

Syntax

datetime.hour()

Parameters

none

Returns

the current hour - on success (0 ~ 23)
0 - on failure

Example

```
#include <Phpoc.h>

PhpocDateTime datetime;

void setup() {
  Serial.begin(9600);
  while(!Serial)
    ;

  // initialize PHPoC [WiFi] Shield:
  Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);

  Serial.println("Get year/month/day/dayofWeek/hour/minute/second from RTC in PHPoC Shield");

  Serial.print(datetime.year());
  Serial.print('-');
  Serial.print(datetime.month());
  Serial.print('-');
  Serial.print(datetime.day());
  Serial.print(' ');
  Serial.print(datetime.dayofWeek());
  Serial.print(' ');
  Serial.print(datetime.hour());
  Serial.print(':');
  Serial.print(datetime.minute());
  Serial.print(':');
  Serial.print(datetime.second());
  Serial.println();
```

```
  // set date and time format:
  datetime.date(F("Y-m-d H:i:s"));
}

void loop() {
  Serial.println(datetime.date());
  delay(10);
}
```

minute()

Description

Gets the current minute from RTC of PHPoC Shield for Arduino.

Syntax

datetime.minute()

Parameters

none

Returns

the current minute - on success (0 ~ 59)
0 - on failure

Example

```
#include <Phpoc.h>

PhpocDateTime datetime;

void setup() {
  Serial.begin(9600);
  while(!Serial)
    ;

  // initialize PHPoC [WiFi] Shield:
  Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);

  Serial.println("Get year/month/day/dayofWeek/hour/minute/second from RTC in PHPoC Shield");

  Serial.print(datetime.year());
  Serial.print('-');
  Serial.print(datetime.month());
  Serial.print('-');
  Serial.print(datetime.day());
  Serial.print(' ');
  Serial.print(datetime.dayofWeek());
  Serial.print(' ');
  Serial.print(datetime.hour());
  Serial.print(':');
  Serial.print(datetime.minute());
  Serial.print(':');
  Serial.print(datetime.second());
  Serial.println();
```

```
  // set date and time format:
  datetime.date(F("Y-m-d H:i:s"));
}

void loop() {
  Serial.println(datetime.date());
  delay(10);
}
```

second()

Description

Gets the current second from RTC of PHPoC Shield for Arduino.

Syntax

datetime.second()

Parameters

none

Returns

the current second - on success (0 ~ 59)
0 - on failure

Example

```
#include <Phpoc.h>

PhpocDateTime datetime;

void setup() {
  Serial.begin(9600);
  while(!Serial)
    ;

  // initialize PHPoC [WiFi] Shield:
  Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);

  Serial.println("Get year/month/day/dayofWeek/hour/minute/second from RTC in PHPoC Shield");

  Serial.print(datetime.year());
  Serial.print('-');
  Serial.print(datetime.month());
  Serial.print('-');
  Serial.print(datetime.day());
  Serial.print(' ');
  Serial.print(datetime.dayofWeek());
  Serial.print(' ');
  Serial.print(datetime.hour());
  Serial.print(':');
  Serial.print(datetime.minute());
  Serial.print(':');
  Serial.print(datetime.second());
  Serial.println();
```

```
  // set date and time format:
  datetime.date(F("Y-m-d H:i:s"));
}

void loop() {
  Serial.println(datetime.date());
  delay(10);
}
```

PhpocExpansion Library

This Library is for using smart expansion boards of PHPoC Shield 2.

Member Classes

ExpansionRelayOutput Class

This class is for using a 4-port relay output board for PHPoC shield 2.

Requirements

- PhpocExpansion Library

Member Functions

- ExpansionRelayOutput()
- on()
- off()
- isOn()
- isoff()
- setdelay()

ExpansionRelayOutput()

Description

This function creates a new instance of the ExpansionRelayOutput class that represents a particular output port.

Syntax

ExpansionRelayOutput(sid, port)

Parameters

sid - slave ID (1 ~ 14)
port - the number of a output port (0 ~ 3)

Returns

none

Example

```
#include <Phpoc.h>
#include <PhpocExpansion.h>

byte spcId = 1;

ExpansionRelayOutput relay0(spcId, 0);
ExpansionRelayOutput relay1(spcId, 1);
ExpansionRelayOutput relay2(spcId, 2);
ExpansionRelayOutput relay3(spcId, 3);

void relayPrintState(void) {
    if(relay0.isOn())
        Serial.println("Relay 0 is ON");
    else
        Serial.println("Relay 0 is OFF");

    if(relay1.isOn())
        Serial.println("Relay 1 is ON");
    else
        Serial.println("Relay 1 is OFF");

    if(relay2.isOff())
        Serial.println("Relay 2 is OFF");
    else
        Serial.println("Relay 2 is ON");

    if(relay3.isOff())
```

```
        Serial.println("Relay 3 is OFF");
    else
        Serial.println("Relay 3 is ON");

    Serial.println();
}

void setup() {
    Serial.begin(9600);
    while(!Serial)
        ;

    // begin PHPoC Shield or PHPoC WiFi Shield
    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    //Phpoc.begin();

    // begin expansion board
    Expansion.begin();

    // get name and print it to serial
    Serial.println(relay0.getName());
}

void loop() {
    // turn all relays on
    relay0.on();
    relay1.on();
    relay2.on();
    relay3.on();
    delay(1000);

    // get current state and pint to serial
    relayPrintState();

    // turn all relays off
    relay0.off();
    relay1.off();
    relay2.off();
    relay3.off();
    delay(1000);

    // get current state and pint to serial
    relayPrintState();
}
```

On()

Description

This function turns the output ON.

Syntax

output.on()

Parameters

none

Returns

none

Example

```
#include <Phpoc.h>
#include <PhpocExpansion.h>

byte spcId = 1;

ExpansionRelayOutput relay0(spcId, 0);
ExpansionRelayOutput relay1(spcId, 1);
ExpansionRelayOutput relay2(spcId, 2);
ExpansionRelayOutput relay3(spcId, 3);

void relayPrintState(void) {
    if(relay0.isOn())
        Serial.println("Relay 0 is ON");
    else
        Serial.println("Relay 0 is OFF");

    if(relay1.isOn())
        Serial.println("Relay 1 is ON");
    else
        Serial.println("Relay 1 is OFF");

    if(relay2.isOff())
        Serial.println("Relay 2 is OFF");
    else
        Serial.println("Relay 2 is ON");

    if(relay3.isOff())
        Serial.println("Relay 3 is OFF");
    else
```

```
        Serial.println("Relay 3 is ON");

    Serial.println();
}

void setup() {
    Serial.begin(9600);
    while(!Serial)
        ;

    // begin PHPoC Shield or PHPoC WiFi Shield
    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    //Phpoc.begin();

    // begin expansion board
    Expansion.begin();

    // get name and print it to serial
    Serial.println(relay0.getName());
}

void loop() {
    // turn all relays on
    relay0.on();
    relay1.on();
    relay2.on();
    relay3.on();
    delay(1000);

    // get current state and pint to serial
    relayPrintState();

    // turn all relays off
    relay0.off();
    relay1.off();
    relay2.off();
    relay3.off();
    delay(1000);

    // get current state and pint to serial
    relayPrintState();
}
```

Off()

Description

This function turns the output OFF.

Syntax

output.off()

Parameters

none

Returns

none

Example

```
#include <Phpoc.h>
#include <PhpocExpansion.h>

byte spcId = 1;

ExpansionRelayOutput relay0(spcId, 0);
ExpansionRelayOutput relay1(spcId, 1);
ExpansionRelayOutput relay2(spcId, 2);
ExpansionRelayOutput relay3(spcId, 3);

void relayPrintState(void) {
    if(relay0.isOn())
        Serial.println("Relay 0 is ON");
    else
        Serial.println("Relay 0 is OFF");

    if(relay1.isOn())
        Serial.println("Relay 1 is ON");
    else
        Serial.println("Relay 1 is OFF");

    if(relay2.isOff())
        Serial.println("Relay 2 is OFF");
    else
        Serial.println("Relay 2 is ON");

    if(relay3.isOff())
        Serial.println("Relay 3 is OFF");
    else
```

```
        Serial.println("Relay 3 is ON");

    Serial.println();
}

void setup() {
    Serial.begin(9600);
    while(!Serial)
        ;

    // begin PHPoC Shield or PHPoC WiFi Shield
    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    //Phpoc.begin();

    // begin expansion board
    Expansion.begin();

    // get name and print it to serial
    Serial.println(relay0.getName());
}

void loop() {
    // turn all relays on
    relay0.on();
    relay1.on();
    relay2.on();
    relay3.on();
    delay(1000);

    // get current state and pint to serial
    relayPrintState();

    // turn all relays off
    relay0.off();
    relay1.off();
    relay2.off();
    relay3.off();
    delay(1000);

    // get current state and pint to serial
    relayPrintState();
}
```

isOn()

Description

This function checks if the output is ON.

Syntax

output.isOn()

Parameters

none

Returns

ture - when the output is ON
false - when the outputs is OFF

Example

```
#include <Phpoc.h>
#include <PhpocExpansion.h>

byte spcId = 1;

ExpansionRelayOutput relay0(spcId, 0);
ExpansionRelayOutput relay1(spcId, 1);
ExpansionRelayOutput relay2(spcId, 2);
ExpansionRelayOutput relay3(spcId, 3);

void relayPrintState(void) {
    if(relay0.isOn())
        Serial.println("Relay 0 is ON");
    else
        Serial.println("Relay 0 is OFF");

    if(relay1.isOn())
        Serial.println("Relay 1 is ON");
    else
        Serial.println("Relay 1 is OFF");

    if(relay2.isOff())
        Serial.println("Relay 2 is OFF");
    else
        Serial.println("Relay 2 is ON");

    if(relay3.isOff())
        Serial.println("Relay 3 is OFF");
```

```
    else
        Serial.println("Relay 3 is ON");

    Serial.println();
}

void setup() {
    Serial.begin(9600);
    while(!Serial)
        ;

    // begin PHPoC Shield or PHPoC WiFi Shield
    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    //Phpoc.begin();

    // begin expansion board
    Expansion.begin();

    // get name and print it to serial
    Serial.println(relay0.getName());
}

void loop() {
    // turn all relays on
    relay0.on();
    relay1.on();
    relay2.on();
    relay3.on();
    delay(1000);

    // get current state and pint to serial
    relayPrintState();

    // turn all relays off
    relay0.off();
    relay1.off();
    relay2.off();
    relay3.off();
    delay(1000);

    // get current state and pint to serial
    relayPrintState();
}
```

isOff()

Description

This function checks if the output is OFF.

Syntax

output.isOff()

Parameters

none

Returns

ture - when the output is OFF
false - when the outputs is ON

Example

```
#include <Phpoc.h>
#include <PhpocExpansion.h>

byte spcId = 1;

ExpansionRelayOutput relay0(spcId, 0);
ExpansionRelayOutput relay1(spcId, 1);
ExpansionRelayOutput relay2(spcId, 2);
ExpansionRelayOutput relay3(spcId, 3);

void relayPrintState(void) {
    if(relay0.isOn())
        Serial.println("Relay 0 is ON");
    else
        Serial.println("Relay 0 is OFF");

    if(relay1.isOn())
        Serial.println("Relay 1 is ON");
    else
        Serial.println("Relay 1 is OFF");

    if(relay2.isOff())
        Serial.println("Relay 2 is OFF");
    else
        Serial.println("Relay 2 is ON");

    if(relay3.isOff())
        Serial.println("Relay 3 is OFF");
```

```
    else
        Serial.println("Relay 3 is ON");

    Serial.println();
}

void setup() {
    Serial.begin(9600);
    while(!Serial)
        ;

    // begin PHPoC Shield or PHPoC WiFi Shield
    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    //Phpoc.begin();

    // begin expansion board
    Expansion.begin();

    // get name and print it to serial
    Serial.println(relay0.getName());
}

void loop() {
    // turn all relays on
    relay0.on();
    relay1.on();
    relay2.on();
    relay3.on();
    delay(1000);

    // get current state and pint to serial
    relayPrintState();

    // turn all relays off
    relay0.off();
    relay1.off();
    relay2.off();
    relay3.off();
    delay(1000);

    // get current state and pint to serial
    relayPrintState();
}
```

setDelay()

Description

This function gives delay time before output.

Syntax

output.setDelay(ms)

Parameters

ms - delay time in millisecond unit (1 ~ 30000)

Returns

none

Example

```
#include <Phpoc.h>
#include <PhpocExpansion.h>

byte spcId = 1;

ExpansionRelayOutput relay0(spcId, 0);
ExpansionRelayOutput relay1(spcId, 1);
ExpansionRelayOutput relay2(spcId, 2);
ExpansionRelayOutput relay3(spcId, 3);

void relayPrintState(void) {
    if(relay0.isOn())
        Serial.println("Relay 0 is ON");
    else
        Serial.println("Relay 0 is OFF");

    if(relay1.isOn())
        Serial.println("Relay 1 is ON");
    else
        Serial.println("Relay 1 is OFF");

    if(relay2.isOff())
        Serial.println("Relay 2 is OFF");
    else
        Serial.println("Relay 2 is ON");

    if(relay3.isOff())
        Serial.println("Relay 3 is OFF");
    else
```

```
        Serial.println("Relay 3 is ON");

    Serial.println();
}

void setup() {
    Serial.begin(9600);
    while(!Serial)
        ;

    // begin PHPoC Shield or PHPoC WiFi Shield
    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    //Phpoc.begin();

    // begin expansion board
    Expansion.begin();

    // get name and print it to serial
    Serial.println(relay0.getName());

    // set delay
    relay0.setDelay(900);
    relay1.setDelay(1000);
    relay2.setDelay(1100);
    relay3.setDelay(1200);
}

void loop() {
    // turn all relays on
    relay0.on();
    relay1.on();
    relay2.on();
    relay3.on();
    delay(1000);

    // get current state and pint to serial
    relayPrintState();

    // turn all relays off
    relay0.off();
    relay1.off();
    relay2.off();
    relay3.off();
    delay(1000);

    // get current state and pint to serial
    relayPrintState();
}
```

ExpansionPhotoInput Class

This class is for using a 4-port degital input board for PHPoC shield 2.

Requirements

- PhpocExpansion Library

Member Functions

- ExpansionPhotoInput()
- isOn()
- isOff()
- setdelay()

ExpansionPhotoInput()

Description

This function creates a new instance of the ExpansionPhotoInput class that represents a particular input port.

Syntax

ExpansionPhotoInput(sid, port)

Parameters

sid - slave ID (1 ~ 14)
port - the number of a input port (0 ~ 3)

Returns

none

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>

byte spcId = 1;

ExpansionPhotoInput input0(spcId, 0);
ExpansionPhotoInput input1(spcId, 1);
ExpansionPhotoInput input2(spcId, 2);
ExpansionPhotoInput input3(spcId, 3);

void setup() {
    Serial.begin(9600);
    while(!Serial)
        ;

    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    Expansion.begin();

    Serial.println(input0.getPID());
    Serial.println(input0.getName());
}

void loop() {
    if(input0.isOn())
        Serial.println("Port 0: 1");
    else
        Serial.println("Port 0: 0");
```

```
    if(input1.isOn())
        Serial.println("Port 1: 1");
    else
        Serial.println("Port 1: 0");

    if(input2.isOff())
        Serial.println("Port 2: 0");
    else
        Serial.println("Port 2: 1");

    if(input3.isOff())
        Serial.println("Port 3: 0");
    else
        Serial.println("Port 3: 1");

    Serial.println();

    delay(1000);
}
```

isOn()

Description

This function checks if the input is ON.

Syntax

input.isOn()

Parameters

none

Returns

ture - when the input is ON
false - when the input is OFF

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>

byte spcId = 1;

ExpansionPhotoInput input0(spcId, 0);
ExpansionPhotoInput input1(spcId, 1);
ExpansionPhotoInput input2(spcId, 2);
ExpansionPhotoInput input3(spcId, 3);

void setup() {
    Serial.begin(9600);
    while(!Serial)
        ;

    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    Expansion.begin();

    Serial.println(input0.getPID());
    Serial.println(input0.getName());
}

void loop() {
    if(input0.isOn())
        Serial.println("Port 0: 1");
    else
        Serial.println("Port 0: 0");
```

```
    if(input1.isOn())
        Serial.println("Port 1: 1");
    else
        Serial.println("Port 1: 0");

    if(input2.isOff())
        Serial.println("Port 2: 0");
    else
        Serial.println("Port 2: 1");

    if(input3.isOff())
        Serial.println("Port 3: 0");
    else
        Serial.println("Port 3: 1");

    Serial.println();

    delay(1000);
}
```

isOff()

Description

This function checks if the input is OFF.

Syntax

input.isOff()

Parameters

none

Returns

ture - when the input is OFF
false - when the input is ON

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>

byte spcId = 1;

ExpansionPhotoInput input0(spcId, 0);
ExpansionPhotoInput input1(spcId, 1);
ExpansionPhotoInput input2(spcId, 2);
ExpansionPhotoInput input3(spcId, 3);

void setup() {
    Serial.begin(9600);
    while(!Serial)
        ;

    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    Expansion.begin();

    Serial.println(input0.getPID());
    Serial.println(input0.getName());
}

void loop() {
    if(input0.isOn())
        Serial.println("Port 0: 1");
    else
        Serial.println("Port 0: 0");
```

```
    if(input1.isOn())
        Serial.println("Port 1: 1");
    else
        Serial.println("Port 1: 0");

    if(input2.isOff())
        Serial.println("Port 2: 0");
    else
        Serial.println("Port 2: 1");

    if(input3.isOff())
        Serial.println("Port 3: 0");
    else
        Serial.println("Port 3: 1");

    Serial.println();

    delay(1000);
}
```

setDelay()

Description

This function sets the valid time for the judgment of the input signal.

Syntax

input.setDlay(ms)

Parameters

ms - valid time in millisecond unit (1 ~ 30000)

Returns

none

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>

byte spcId = 1;

ExpansionPhotoInput input0(spcId, 0);
ExpansionPhotoInput input1(spcId, 1);
ExpansionPhotoInput input2(spcId, 2);
ExpansionPhotoInput input3(spcId, 3);

void setup() {
    Serial.begin(9600);
    while(!Serial)
        ;

    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    Expansion.begin();

    input0.setDelay(1000);
    input1.setDelay(1000);
    input2.setDelay(1000);
    input3.setDelay(1000);

    Serial.println(input0.getPID());
    Serial.println(input0.getName());
}

void loop() {
    if(input0.isOn())
```

```
        Serial.println("Port 0: 1");
    else
        Serial.println("Port 0: 0");

    if(input1.isOn())
        Serial.println("Port 1: 1");
    else
        Serial.println("Port 1: 0");

    if(input2.isOff())
        Serial.println("Port 2: 0");
    else
        Serial.println("Port 2: 1");

    if(input3.isOff())
        Serial.println("Port 3: 0");
    else
        Serial.println("Port 3: 1");

    Serial.println();

    delay(1000);
}
```

ExpansionDCMotor Class

This class is for using a DC Motor Controller for PHPoC shield 2.

Requirements

- PhpocExpansion Library

Member Functions

- ExpansionDCMotor()
- setPolarity()
- setDeriection()
- setPeriod()
- setWidth()
- setDecay()
- setEncoderPolarity()
- setEncoderPosition()
- setEncoderPSR()
- getEncoderPosition()
- getEncoderPeriod()
- setFilterFrequency()
- setFilterPNC()
- getFilterPNC()

ExpansionDCMotor()

Description

This function creates a new instance of the ExpansionDCMotor class that represents a particular DC motor port.

Syntax

ExpansionDCMotor(sid, port)

Parameters

sid - slave ID(1 ~ 14)
port - the number of a motor port(1 or 2)

Returns

none

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>

byte spcId = 1;

ExpansionDCMotor dcmotor(spcId, 1);

int width = 3000;

void setup() {
    Serial.begin(9600);
    while(!Serial)
    ;

    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    Expansion.begin();

    Serial.println(dcmotor.getPID());
    Serial.println(dcmotor.getName());

    dcmotor.setPeriod(10000);
    dcmotor.setWidth(width);
}

void loop() {
    if(width > 0) {
        width -= 100;
```

```
            dcmotor.setWidth(width);
            delay(100);
        }
    }
```

setPolarity()

Description

This function sets the PWM polarity.

Syntax

dcmotor.setPolarity(pol)

Parameters

pol - a integer value which represents the polarity

pol	polarity
0 or greater than 0	positive (default)
otherwise	negative

Returns

none

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>

byte spcId = 1;

ExpansionDCMotor dcmotor(spcId, 1);

int width = 3000;

void setup() {
    Serial.begin(9600);
    while(!Serial)
    ;

    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    Expansion.begin();

    Serial.println(dcmotor.getPID());
    Serial.println(dcmotor.getName());

    // set PWM polarity
    dcmotor.setPolarity(1);
    dcmotor.setPeriod(10000);
    dcmotor.setWidth(width);
}
```

```
void loop() {
    if(width > 0) {
        width -= 100;
        dcmotor.setWidth(width);
        delay(100);
    }
}
```

setDirection()

Description

This function sets the direction of rotation.

Syntax

dcmotor.setDirection(dir)

Parameters

dir - a integer value which represents the direction of rotation

dir	direction
0 or greater than 0	forward (default)
otherwise	reverse

Returns

none

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>

byte spcId = 1;

ExpansionDCMotor dcmotor(spcId, 1);

int width = 3000;

void setup() {
    Serial.begin(9600);
    while(!Serial)
    ;

    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    Expansion.begin();

    Serial.println(dcmotor.getPID());
    Serial.println(dcmotor.getName());

    dcmotor.setPolarity(1);

    // set the direction of rotation
    dcmotor.setDirection(1);
    dcmotor.setPeriod(10000);
    dcmotor.setWidth(width);
```

```
}

void loop() {
    if(width > 0) {
        width -= 100;
        dcmotor.setWidth(width);
        delay(100);
    }
}
```

setPeriod()

Description

This function sets the PWM period.

Syntax

dcmotor.setPeriod(period)

Parameters

period - a integer value which represents the PWM period in microsecond unit

Returns

none

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>

byte spcId = 1;

ExpansionDCMotor dcmotor(spcId, 1);

int width = 3000;

void setup() {
    Serial.begin(9600);
    while(!Serial)
    ;

    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    Expansion.begin();

    Serial.println(dcmotor.getPID());
    Serial.println(dcmotor.getName());

    // set the PWM period
    dcmotor.setPeriod(10000);
    dcmotor.setWidth(width);
}

void loop() {
    if(width > 0) {
        width -= 100;
        dcmotor.setWidth(width);
```

```
        delay(100);
    }
}
```

setWidth()

Description

This function sets the HIGH duration of PWM.

Syntax

dcmotor.setWidth(width)

Parameters

width - a integer value which represents the HIGH duration of PWM in microsecond unit

Returns

none

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>

byte spcId = 1;

ExpansionDCMotor dcmotor(spcId, 1);

int width = 3000;

void setup() {
    Serial.begin(9600);
    while(!Serial)
    ;

    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    Expansion.begin();

    Serial.println(dcmotor.getPID());
    Serial.println(dcmotor.getName());

    dcmotor.setPeriod(10000);

    // set the HIGH duration of PWM
    dcmotor.setWidth(width);
}

void loop() {
    if(width > 0) {
        width -= 100;
```

```
        dcmotor.setWidth(width);
        delay(100);
    }
}
```

setDecay()

Description

This function sets the decay mode.

Syntax

dcmotor.setDecay(decay)

Parameters

decay - a integer value which represents the decay mode

decay	mode
0	slow decay mode
otherwise	fast decay mode (default)

Returns

none

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>

byte spcId = 1;

ExpansionDCMotor dcmotor(spcId, 1);

void setup() {
    Serial.begin(9600);
    while(!Serial)
    ;

    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    Expansion.begin();

    Serial.println(dcmotor.getPID());
    Serial.println(dcmotor.getName());

    dcmotor.setPeriod(10000);

}

void loop() {

    // fast decay mode
    dcmotor.setDecay(1);
```

```
    dcmotor.setWidth(10000);
    dcmotor.setWidth(0);
    delay(5000);

    // slow decay mode
    dcmotor.setDecay(0);
    dcmotor.setWidth(10000);
    dcmotor.setWidth(0);
    delay(5000);
}
```

setEncoderPolarity()

Description

This function sets the count direction of an encoder.

Syntax

dcmotor.setEncoderPolarity(pol)

Parameters

pol - a integer value which represents count direction of an encoder

pol	count direction
0 or grater than 0	normal (default)
otherwise	invert

Returns

none

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>

byte spcId = 1;

ExpansionDCMotor dcmotor(spcId, 1);

void setup() {
    Serial.begin(9600);
    while(!Serial)
        ;

    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    Expansion.begin();

    Serial.println(dcmotor.getPID());
    Serial.println(dcmotor.getName());

    dcmotor.setPolarity(1);
    dcmotor.setDirection(1);
    dcmotor.setPeriod(10000);
    dcmotor.setWidth(3000);

    //normal count direction
    dcmotor.setEncoderPolarity(0);
}
```

```
void loop() {

    Serial.println(dcmotor.getEncoderPosition());
    delay(100);
}
```

setEncoderPosition()

Description

This function sets the counter value of an encoder.

Syntax

dcmotor.setEncoderPosition(pos)

Parameters

pos - a integer value which represents counter value of an encoder (-1 billion to +1 billion)

Returns

none

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>

byte spcId = 1;

ExpansionDCMotor dcmotor(spcId, 1);

void setup() {
    Serial.begin(9600);
    while(!Serial)
        ;

    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    Expansion.begin();

    Serial.println(dcmotor.getPID());
    Serial.println(dcmotor.getName());

    dcmotor.setPeriod(10000);
    dcmotor.setWidth(3000);

    // set counter value
    dcmotor.setEncoderPosition(10000);
}

void loop() {

    Serial.println(dcmotor.getEncoderPosition());
    delay(100);
```

```
}
```

setEncoderPSR()

Description

This function sets the sampling count of an encoder.

Syntax

dcmotor.setEncoderPSR(psr)

Parameters

psr - a integer value which represents sampling count of an encoder (1 ~ 64)

Returns

none

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>

byte spcId = 1;

ExpansionDCMotor dcmotor(spcId, 1);

void setup() {
    Serial.begin(9600);
    while(!Serial)
        ;

    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    Expansion.begin();

    Serial.println(dcmotor.getPID());
    Serial.println(dcmotor.getName());

    dcmotor.setPeriod(10000);
    dcmotor.setWidth(3000);

    // set sampling count
    dcmotor.setEncoderPSR(16);
}

void loop() {

    Serial.println(dcmotor.getEncoderPosition());
    delay(100);
```

```
}
```

getEncoderPosition()

Description

This function gets the counter value of an encoder.

Syntax

dcmotor.getEncoderPosition()

Parameters

none

Returns

the current counter value of an encoder

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>

byte spcId = 1;

ExpansionDCMotor dcmotor(spcId, 1);

void setup() {
    Serial.begin(9600);
    while(!Serial)
        ;

    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    Expansion.begin();

    Serial.println(dcmotor.getPID());
    Serial.println(dcmotor.getName());

    dcmotor.setPeriod(10000);
    dcmotor.setWidth(3000);
}

void loop() {
    // get the counter value of an encoder
    Serial.println(dcmotor.getEncoderPosition());
    delay(100);
}
```

getEncoderPeriod()

Description

This function gets the period of an encoder's output.

Syntax

dcmotor.getEncoderPeriod()

Parameters

none

Returns

the period of an encoder's output in microsecond unit

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>

byte spcId = 1;

ExpansionDCMotor dcmotor(spcId, 1);

int width = 3000;

void setup() {
    Serial.begin(9600);
    while(!Serial)
    ;

    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    Expansion.begin();

    Serial.println(dcmotor.getPID());
    Serial.println(dcmotor.getName());

    dcmotor.setEncoderPSR(64);
    dcmotor.setPeriod(10000);
    dcmotor.setWidth(width);
}

void loop() {
    if(width > 1000) {
        width -= 100;
        dcmotor.setWidth(width);
```

```
        delay(500);

        Serial.print(width);
        Serial.print("=>");

        // get the period of an encoder's output
        Serial.println(dcmotor.getEncoderPeriod());
    }
    else
        dcmotor.setWidth(0);
}
```

setFilterFrequency()

Description

This function sets the cut-off frequency of the low-pass filter.

Syntax

dcmotor.setFilterFrequency(freq)

Parameters

freq - a integer value which represents cut-off frequency of the low-pass filter

Returns

none

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>

byte spcId = 1;

int freq = 1000;
int count_prev = 0;
int count;
int diff;

ExpansionDCMotor dcmotor(spcId, 1);

void setup() {
    Serial.begin(9600);
    while(!Serial)
        ;

    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    Expansion.begin();

    Serial.println(dcmotor.getPID());
    Serial.println(dcmotor.getName());

    dcmotor.setPeriod(10000);
    dcmotor.setWidth(3000);
}

void loop() {
    if(freq > 7000) {
```

```
        dcmotor.setWidth(0);
        return;
    }

    // set the cut-off frequency
    dcmotor.setFilterFrequency(freq);
    count = dcmotor.getFilterPNC();
    diff = count - count_prev;
    Serial.print("noise count at freq");
    Serial.print(freq);
    Serial.print(" : ");
    Serial.println(diff);

    freq += 200;
    count_prev = count;
    delay(200);
}
```

setFilterPNC()

Description

This function sets the noise counter value of the low-pass filter.

Syntax

dcmotor.setFilterPNC(pnc)

Parameters

pnc - a integer value which represents the noise counter value of the low-pass filter

Returns

none

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>

byte spcId = 1;

int freq = 1000;
int count_prev = 0;
int count;
int diff;

ExpansionDCMotor dcmotor(spcId, 1);

void setup() {
    Serial.begin(9600);
    while(!Serial)
        ;

    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    Expansion.begin();

    Serial.println(dcmotor.getPID());
    Serial.println(dcmotor.getName());

    dcmotor.setPeriod(10000);
    dcmotor.setWidth(3000);

    // set the noise counter value
    dcmotor.setFilterPNC(0);
}
```

```
void loop() {
    if(freq > 7000) {
        dcmotor.setWidth(0);
        return;
    }
    dcmotor.setFilterFrequency(freq);
    count = dcmotor.getFilterPNC();
    diff = count - count_prev;
    Serial.print("noise count at freq");
    Serial.print(freq);
    Serial.print(" : ");
    Serial.println(diff);

    freq += 200;
    count_prev = count;
    delay(200);
}
```

getFilterPNC()

Description

This function gets the noise counter value of the low-pass filter.

Syntax

dcmotor.getFilterPNC()

Parameters

none

Returns

the current noise counter value of the low-pass filter

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>

byte spcId = 1;

int freq = 1000;
int count_prev = 0;
int count;
int diff;

ExpansionDCMotor dcmotor(spcId, 1);

void setup() {
    Serial.begin(9600);
    while(!Serial)
        ;

    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    Expansion.begin();

    Serial.println(dcmotor.getPID());
    Serial.println(dcmotor.getName());

    dcmotor.setPeriod(10000);
    dcmotor.setWidth(3000);
}

void loop() {
    if(freq > 7000) {
```

```
        dcmotor.setWidth(0);
        return;
    }
    dcmotor.setFilterFrequency(freq);

    // set the noise counter value
    count = dcmotor.getFilterPNC();
    diff = count - count_prev;
    Serial.print("noise count at freq");
    Serial.print(freq);
    Serial.print(" : ");
    Serial.println(diff);

    freq += 200;
    count_prev = count;
    delay(200);
}
```

ExpansionStepper Class

This class is for using a stepper motor controller for PHPoC shield 2.

Requirements

- PhpocExpansion Library

Member Functions

- ExpansionStepper()
- reset()
- unlock()
- setMode()
- setVrefStop()
- setVrefDrive()
- setVrefLock()
- setResonance()
- setSpeed()
- setAccel()
- setPosition()
- getState()
- getPosition()
- setMove()
- setGoto()
- setGotoSW()
- stop()
- setEioMode()
- getEio()

ExpansionStepper()

Description

This function creates a new instance of the ExpansionStepper class that represents a stepper motor.

Syntax

ExpansionStepper(sid)

Parameters

sid - slave ID (1 ~ 14)

Returns

none

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>

byte spcId = 1;

// creates a new instance
ExpansionStepper step(spcId);

void setup() {
    Serial.begin(9600);
    while(!Serial)
        ;

    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    Expansion.begin();

    Serial.println(step.getPID());
    Serial.println(step.getName());

    step.setMode(4);
    step.setVrefStop(2);
    step.setVrefDrive(8);
    step.setResonance(120, 250);
    step.setSpeed(400);
    step.setAccel(0, 0);
    step.setPosition(-400);

    step.stepGoto(400);
    delay(1000);
```

```
    step.stepGoto(-400);
    delay(1000);
    step.stepGoto(400);

    while(step.getState()) {
        delay(1);
    }
}

void loop() {

}
```

reset()

Description

This function does:

- stops driving a motor immediately.
- resets the position of motoer to zero.
- resets the value configured by the setSpeed() function to default.
- resets the value configured by the setAccel() function to default.
- resets the value configured by the setDecel() function to default.
- resets the value configured by the setEioMode() function to default.

Syntax

step.reset()

Parameters

none

Returns

none

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>

byte spcId = 1;

ExpansionStepper step(spcId);

void setup() {
    Serial.begin(9600);
    while(!Serial)
        ;

    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    Expansion.begin();

    Serial.println(step.getPID());
    Serial.println(step.getName());

    step.setMode(1);
    step.setVrefStop(2);
    step.setVrefDrive(8);
    step.setResonance(120, 250);
```

```
    step.setSpeed(400);
    step.setAccel(0, 0);
    step.setPosition(0);

    step.stepGoto(400);
    delay(2000);
    step.stepGoto(0);
    delay(500);

    // resets the stepper motor
    step.reset();

    while(step.getState()) {
        delay(1);
    }
}

void loop() {

}
```

unlock()

Description

This function does:

- changes the motor state to 0 from 1.
- resets the value configured by the setEioMode() function to default.

Syntax

step.unlock()

Parameters

none

Returns

none

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>

byte spcId = 1;

ExpansionStepper step(spcId);

int state;

void setup() {
    Serial.begin(9600);
    while(!Serial)
        ;

    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    Expansion.begin();

    Serial.println(step.getPID());
    Serial.println(step.getName());

    step.setMode(4);
    step.setVrefStop(2);
    step.setVrefDrive(8);
    step.setVrefLock(8);
    step.setSpeed(400);
    step.setAccel(4000);
```

```
    step.setEioMode(0, 1);
    step.setEioMode(1, 1);
    step.setEioMode(2, 1);
    step.setEioMode(3, 1);

    step.stepGoto(4000);

    while(step.getState() > 1) {
        delay(1);
    }

    // state: 0 - stop, 1 - locked
    Serial.print("step_state ");
    Serial.println(step.getState());

    step.unlock();

    // state: 0 - stop, 1 - locked
    Serial.print("step_state ");
    Serial.println(step.getState());
}

void loop() {

}
```

setMode()

Description

This function sets the division rate of microstepping.

Syntax

step.setMode(mode)

Parameters

mode - one of following values which represents the division rate

mode	description
1	full-step
2	half-step
4	1/4-step
8	1/8-step
16	1/16-step
32	1/32-step

Returns

none

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>

byte spcId = 1;

ExpansionStepper step(spcId);

void setup() {
    Serial.begin(9600);
    while(!Serial)
        ;

    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    Expansion.begin();

    Serial.println(step.getPID());
    Serial.println(step.getName());

    // sets the division rate
    step.setMode(4);
    step.setVrefStop(2);
    step.setVrefDrive(8);
```

```
    step.setResonance(120, 250);
    step.setSpeed(400);
    step.setAccel(0, 0);
    step.setPosition(-400);

    step.stepGoto(400);
    delay(1000);
    step.stepGoto(-400);
    delay(1000);
    step.stepGoto(400);

    while(step.getState()) {
        delay(1);
    }
}

void loop() {

}
```

setVrefStop()

Description

This function sets the current limiting for keeping stop state of a stepper motor.

Syntax

step.setVrefStop(vref)

Parameters

vref - a integer value which represents the current limiting (0 ~ 15)

Returns

none

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>

byte spcId = 1;

ExpansionStepper step(spcId);

void setup() {
    Serial.begin(9600);
    while(!Serial)
        ;

    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    Expansion.begin();

    Serial.println(step.getPID());
    Serial.println(step.getName());

    step.setMode(4);

    // sets the current limiting for stop state
    step.setVrefStop(2);
    step.setVrefDrive(8);
    step.setResonance(120, 250);
    step.setSpeed(400);
    step.setAccel(0, 0);
    step.setPosition(-400);

    step.stepGoto(400);
```

```
    delay(1000);
    step.stepGoto(-400);
    delay(1000);
    step.stepGoto(400);

    while(step.getState()) {
        delay(1);
    }
}

void loop() {

}
```

setVrefDrive()

Description

This function sets the current limiting for keeping run state of a stepper motor.

Syntax

step.setVrefDrive(vref)

Parameters

vref - a integer value which represents the current limiting (0 ~ 15)

Returns

none

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>

byte spcId = 1;

ExpansionStepper step(spcId);

void setup() {
    Serial.begin(9600);
    while(!Serial)
        ;

    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    Expansion.begin();

    Serial.println(step.getPID());
    Serial.println(step.getName());

    step.setMode(4);
    step.setVrefStop(2);

    // sets the current limiting for run state
    step.setVrefDrive(8);
    step.setResonance(120, 250);
    step.setSpeed(400);
    step.setAccel(0, 0);
    step.setPosition(-400);

    step.stepGoto(400);
```

```
    delay(1000);
    step.stepGoto(-400);
    delay(1000);
    step.stepGoto(400);

    while(step.getState()) {
        delay(1);
    }
}

void loop() {

}
```

setVrefLock()

Description

This function sets the current limiting for keeping lock state of a stepper motor.

Syntax

step.setVrefLock(vref)

Parameters

vref - a integer value which represents the current limiting (0 ~ 15)

Returns

none

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>

byte spcId = 1;

ExpansionStepper step(spcId);

int state;

void setup() {
    Serial.begin(9600);
    while(!Serial)
        ;

    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    Expansion.begin();

    Serial.println(step.getPID());
    Serial.println(step.getName());

    step.setMode(4);
    step.setVrefStop(2);
    step.setVrefDrive(8);

    // sets the current limiting for run state
    step.setVrefLock(8);
    step.setSpeed(400);
    step.setAccel(4000);
```

```
    step.setEioMode(0, 1);
    step.setEioMode(1, 1);
    step.setEioMode(2, 1);
    step.setEioMode(3, 1);

    step.stepGoto(4000);

    while(step.getState() > 1) {
        delay(1);
    }

    // state: 0 - stop, 1 - locked
    Serial.print("step_state ");
    Serial.println(step.getState());

    step.unlock();

    // state: 0 - stop, 1 - locked
    Serial.print("step_state ");
    Serial.println(step.getState());
}

void loop() {

}
```

setResonance()

Description

This function sets a resonance range.

Syntax

step.setResonance(low, high)

Parameters

low - the minimum integer value of the resonance range in pps unit
high - the maximum integer value of the resonance range in pps unit

※ pps: pulse per second

Returns

none

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>

byte spcId = 1;

ExpansionStepper step(spcId);

void setup() {
    Serial.begin(9600);
    while(!Serial)
        ;

    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    Expansion.begin();

    Serial.println(step.getPID());
    Serial.println(step.getName());

    step.setMode(4);
    step.setVrefStop(2);
    step.setVrefDrive(8);

    // sets a resonance range
    step.setResonance(120, 250);
    step.setSpeed(400);
    step.setAccel(0, 0);
```

```
    step.setPosition(-400);

    step.stepGoto(400);
    delay(1000);
    step.stepGoto(-400);
    delay(1000);
    step.stepGoto(400);

    while(step.getState()) {
        delay(1);
    }
}

void loop() {

}
```

setSpeed()

Description

This function sets a rotating speed of a stepper motor.

Syntax

step.setSpeed(speed)

Parameters

speed - a integer value which represents the rotating speed in pps unit (~ 240000)

※ pps: pulse per second

Returns

none

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>

byte spcId = 1;

ExpansionStepper step(spcId);

void setup() {
    Serial.begin(9600);
    while(!Serial)
        ;

    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    Expansion.begin();

    Serial.println(step.getPID());
    Serial.println(step.getName());

    step.setMode(4);
    step.setVrefStop(2);
    step.setVrefDrive(8);
    step.setResonance(120, 250);

    // sets a speed
    step.setSpeed(400);
    step.setAccel(0, 0);
    step.setPosition(-400);
```

```
    step.stepGoto(400);
    delay(1000);
    step.stepGoto(-400);
    delay(1000);
    step.stepGoto(400);

    while(step.getState()) {
        delay(1);
    }
}

void loop() {

}
```

setAccel()

Description

This function sets a acceleration speed and a deceleration speed of a stepper motor.

Syntax

step.setAccel(accel)
step.setAccel(accel, decel)

Parameters

accel - a integer value which represents the acceleration speed in pps/s unit (~ 2400000)
decel - a integer value which represents the deceleration speed in pps/s unit (~ 2400000)

※ pps: pulse per second

Returns

none

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>

byte spcId = 1;

ExpansionStepper step(spcId);

void setup() {
    Serial.begin(9600);
    while(!Serial)
        ;

    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    Expansion.begin();

    Serial.println(step.getPID());
    Serial.println(step.getName());

    step.setMode(4);
    step.setVrefStop(2);
    step.setVrefDrive(8);
    step.setResonance(120, 250);
    step.setSpeed(400);

    // sets a acceleration speed and a deceleration spped
```

```
    step.setAccel(800, 800);
    step.setPosition(-400);

    step.stepGoto(400);
    delay(1000);
    step.stepGoto(-400);
    delay(1000);
    step.stepGoto(400);

    while(step.getState()) {
        delay(1);
    }
}

void loop() {

}
```

setPosition()

Description

This function sets or resets a counter position of a stepper motor.

Syntax

step.setPosition(pos)

Parameters

pos - a integer value which represents a counter position (-1 billion ~ +1 billion)

Returns

none

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>

byte spcId = 1;

ExpansionStepper step(spcId);

void setup() {
    Serial.begin(9600);
    while(!Serial)
        ;

    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    Expansion.begin();

    Serial.println(step.getPID());
    Serial.println(step.getName());

    step.setMode(4);
    step.setVrefStop(2);
    step.setVrefDrive(8);
    step.setResonance(120, 250);
    step.setSpeed(400);
    step.setAccel(0, 0);

    // sets a counter position
    step.setPosition(-400);

    step.stepGoto(400);
```

```
    delay(1000);
    step.stepGoto(-400);
    delay(1000);
    step.stepGoto(400);

    while(step.getState()) {
        delay(1);
    }
}

void loop() {

}
```

getState()

Description

This function gets a status code of a stepper motor.

Syntax

state = step.getState()

Parameters

none

Returns

a integer value represents the status of a stepper motor

status code	description
0	stop state
1	lock state
otherwise	run state

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>

byte spcId = 1;

ExpansionStepper step(spcId);

void setup() {
    Serial.begin(9600);
    while(!Serial)
        ;

    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    Expansion.begin();

    Serial.println(step.getPID());
    Serial.println(step.getName());

    step.setMode(4);
    step.setVrefStop(2);
    step.setVrefDrive(8);
    step.setResonance(120, 250);
    step.setSpeed(400);
    step.setAccel(0, 0);
    step.setPosition(-400);
```

```
    step.stepGoto(400);
    delay(1000);
    step.stepGoto(-400);
    delay(1000);
    step.stepGoto(400);

    // gets a status code
    while(step.getState()) {
        delay(1);
    }
}

void loop() {

}
```

getPosition()

Description

This function gets the current counter position of a stepper motor.

Syntax

position = step.getPosition()

Parameters

none

Returns

a integer value represents the current counter position of a stepper motor

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>

byte spcId = 1;

ExpansionStepper step(spcId);

void setup() {
    Serial.begin(9600);
    while(!Serial)
        ;

    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    Expansion.begin();

    Serial.println(step.getPID());
    Serial.println(step.getName());

    step.setMode(4);
    step.setVrefStop(2);
    step.setVrefDrive(8);
    step.setResonance(120, 250);
    step.setSpeed(400);
    step.setAccel(0, 0);
    step.setPosition(-400);

    step.stepGoto(1600);

    while(step.getState()) {
```

```
        // gets the current counter position
        Serial.println(step.getPosition());
        delay(1000);
    }
}

void loop() {

}
```

stepMove()

Description

This function controls a stepper motor based on the current counter position, not the initial position. This function is only available when the motor is stopped.

Syntax

step.stepMove(step)
step.stepMove(step, speed)
step.stepMove(step, speed, accel)

Parameters

step - a integer value which represents a target position (-1 billion ~ +1 billion)
speed - a integer value which represents the rotating speed in pps unit (~ 240000)
accel - a integer value which represents the deceleration speed in pps/s unit (~ 2400000)

※ pps: pulse per second

Returns

none

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>

byte spcId = 1;

ExpansionStepper step(spcId);

void setup() {
    Serial.begin(9600);
    while(!Serial)
        ;

    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    Expansion.begin();

    Serial.println(step.getPID());
    Serial.println(step.getName());

    step.setMode(4);
    step.setVrefStop(2);
    step.setVrefDrive(8);
    step.setResonance(120, 250);
```

```
    step.setSpeed(400);
    step.setAccel(0, 0);
    step.setPosition(0);

    step.stepMove(400);
    delay(2000);
    step.stepMove(-400);
    delay(2000);
    step.stepMove(400);
    delay(2000);

    while(step.getState()) {
        delay(1);
    }
}

void loop() {

}
```

stepGoto()

Description

This function controls a stepper motor based on the initial position, not the current counter position. This function can be used even when the motor is running.

Syntax

step.stepGoto(pos)
step.stepGoto(pos, speed)
step.stepGoto(pos, speed, accel)

Parameters

pos - a integer value which represents a target position (-1 billion ~ +1 billion)
speed - a integer value which represents the rotating speed in pps unit (~ 240000)
accel - a integer value which represents the deceleration speed in pps/s unit (~ 2400000)

※ pps: pulse per second

Returns

none

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>

byte spcId = 1;

ExpansionStepper step(spcId);

void setup() {
    Serial.begin(9600);
    while(!Serial)
        ;

    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    Expansion.begin();

    Serial.println(step.getPID());
    Serial.println(step.getName());

    step.setMode(4);
    step.setVrefStop(2);
    step.setVrefDrive(8);
    step.setResonance(120, 250);
```

```
    step.setSpeed(400);
    step.setAccel(0, 0);
    step.setPosition(0);

    step.stepGoto(400);
    delay(2000);
    step.stepGoto(-400);
    delay(2000);
    step.stepGoto(400);
    delay(2000);

    while(step.getState()) {
        delay(1);
    }
}

void loop() {

}
```

stepGotoSW()

Description

This function controls a stepper motor until a limit switch is closed.

Syntax

step.stepGotoSW(id, dir)
step.stepGotoSW(id, dir, speed)
step.stepGotoSW(id, dir, speed, accel)

Parameters

id - a integer value which represents the port number of a limit switch (0 ~ 3)
dir - a integer value which represents the direction of rotation

dir	description
0 or grater than 0	forward
otherwise	reverse

speed - a integer value which represents the rotating speed in pps unit (~ 240000)
accel - a integer value which represents the deceleration speed in pps/s unit (~ 2400000)

※ pps: pulse per second

Returns

none

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>

byte spcId = 1;

ExpansionStepper step(spcId);

void setup() {
    Serial.begin(9600);
    while(!Serial)
        ;

    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    Expansion.begin();

    Serial.println(step.getPID());
    Serial.println(step.getName());
```

```
    step.setMode(4);
    step.setVrefDrive(8);
    step.setSpeed(400);
    step.setAccel(0);

    // rotate until digital input 0 is LOW
    Serial.print("find positive limit ...");
    step.stepGotoSW(0, 1);
    while(step.getState()) {
        delay(1);
    }
    Serial.println("done");

    delay(1000);

    // rotate until digital input 1 is LOW
    Serial.print("find negative limit ...");
    step.stepGotoSW(1, -1);
    while(step.getState()) {
        delay(1);
    }
    Serial.println("done");
}

void loop() {

}
```

stop()

Description

This function stops driving a motor immediately.

Syntax

step.stop()
step.stop(decel)

Parameters

decel - a positive integer value which represents the deceleration speed in pps/s unit(~ 2400000)

※ pps: pulse per second

Returns

none

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>

byte spcId = 1;

ExpansionStepper step(spcId);

int state;

void setup() {
    Serial.begin(9600);
    while(!Serial)
        ;

    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    Expansion.begin();

    Serial.println(step.getPID());
    Serial.println(step.getName());

    step.setMode(4);
    step.setVrefStop(2);
    step.setVrefDrive(8);
    step.setSpeed(400);
    step.setAccel(1000);
```

```
    step.stepGoto(40000);
}

void loop() {
    state = step.getEio(0);

    if(state == 0)
        step.stop();

    delay(1);
}
```

setEioMode()

Description

This function sets a type of a digital input port.

Syntax

step.setEioMode(id, mode)

Parameters

id - a integer value which represents the port number of a limit switch (0 ~ 3)
mode - a integer value which represents the type of a digital input port

mode	description
0	normal input
otherwise	control lock

Returns

none

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>

byte spcId = 1;

ExpansionStepper step(spcId);

int state;

void setup() {
    Serial.begin(9600);
    while(!Serial)
        ;

    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    Expansion.begin();

    Serial.println(step.getPID());
    Serial.println(step.getName());

    step.setMode(4);
    step.setVrefStop(2);
    step.setVrefDrive(8);
    step.setVrefLock(8);
```

```
    step.setSpeed(400);
    step.setAccel(4000);

    step.setEioMode(0, 1);
    step.setEioMode(1, 1);
    step.setEioMode(2, 1);
    step.setEioMode(3, 1);

    step.stepGoto(4000);

    while(step.getState() > 1) {
        delay(1);
    }

    // state: 0 - stop, 1 - locked
    Serial.print("step_state ");
    Serial.println(step.getState());

    step.unlock();

    // state: 0 - stop, 1 - locked
    Serial.print("step_state ");
    Serial.println(step.getState());
}

void loop() {

}
```

getEio()

Description

This function gets a status code of a digital input port.

Syntax

state = step.getEio(id)

Parameters

id - a integer value which represents the port number of a limit switch (0 ~ 3)

※ pps: pulse per second

Returns

a integer value represents the status of a digital input port

status code	description
0	low
1	high (default)

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>

byte spcId = 1;

ExpansionStepper step(spcId);

int state;

void setup() {
    Serial.begin(9600);
    while(!Serial)
        ;

    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    Expansion.begin();

    Serial.println(step.getPID());
    Serial.println(step.getName());

    step.setMode(4);
    step.setVrefDrive(8);
    step.setSpeed(400);
```

```
    step.setAccel(1000);

    step.stepGoto(40000);
}

void loop() {
    state = step.getEio(0);

    if(state == 0)
        step.stop();

    delay(1);
}
```

ExpansionSerial Class

This class is for using both a RS-232 board and a RS-422/485 board for PHPoC shield 2.

Requirements

- PhpocExpansion Library

Member Functions

- ExpansionSerial()
- begin()
- available()
- peek()
- read()
- availableForWrite()
- flush()
- write()

begin()

Description

This function sets the parameters for serial data communication.

Syntax

port.begin(baud)
port.begin(sets)

Parameters

baud - a integer value which represents the data rate in bit per second
sets - a string value which represents the data rate, parity, data bit, stop bit and flow control

- string format of the sets - (): required, []: optional

```
"(baudrate)[parity[data bit[stop bit[flow control]]]]"
```

※ Please refer to the product's specification for the available setting values.

Returns

none

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>
#define BUFFER_SIZE 100  // read and write buffer size, reduce it if memory of Arduino is not
enough

byte spcId = 1;

ExpansionSerial port(spcId);

byte rwbuf[BUFFER_SIZE];  // read and write buffer

void setup() {
    Serial.begin(9600);
    while(!Serial)
        ;

    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    Expansion.begin();
```

```
    // sets the parameters for serial data communication
    port.begin("115200N81N");
}

void loop() {
    int txfree = port.availableForWrite();
    int rxlen = port.available();

    if(rxlen > 0) {
        if(rxlen <= txfree) {
            int rwlen; // read and write length

            if(rxlen <= BUFFER_SIZE)
                rwlen = rxlen;
            else
                rwlen = BUFFER_SIZE;

            // receive data
            rwlen = port.readBytes(rwbuf, rwlen);

            // send data
            port.write(rwbuf, rwlen);

            // print data to serial monitor of Arduino IDE
            Serial.write(rwbuf, rwlen);
        }
    }

    delay(1);
}
```

available()

Description

Gets the number of bytes (characters) available for reading from the serial port. This is data that's already arrived and stored in the serial receive buffer. The available() inherits from the Stream utility class.

Syntax

port.available()

Parameters

none

Returns

a integer value which represents the size of received data

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>
#define BUFFER_SIZE 100  // read and write buffer size, reduce it if memory of Arduino is not
enough

byte spcId = 1;

ExpansionSerial port(spcId);

byte rwbuf[BUFFER_SIZE];  // read and write buffer

void setup() {
    Serial.begin(9600);
    while(!Serial)
        ;

    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    Expansion.begin();
    port.begin("115200N81N");
}

void loop() {
    int txfree = port.availableForWrite();

    // Get the number of bytes available for reading
    int rxlen = port.available();
```

```
    if(rxlen > 0) {
        if(rxlen <= txfree) {
            int rwlen; // read and write length

            if(rxlen <= BUFFER_SIZE)
                rwlen = rxlen;
            else
                rwlen = BUFFER_SIZE;

            // receive data
            rwlen = port.readBytes(rwbuf, rwlen);

            // send data
            port.write(rwbuf, rwlen);

            // print data to serial monitor of Arduino IDE
            Serial.write(rwbuf, rwlen);
        }
    }

    delay(1);
}
```

peek()

Description

Returns the next byte (character) of incoming serial data without removing it from the internal serial buffer. That is, successive calls to peek() will return the same character, as will the next call to read(). The peek() inherits from the Stream utility class.

Syntax

port.peek()

Parameters

none

Returns

the first byte of incoming serial data available, or -1 if no data is available (int)

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>
#define BUFFER_SIZE 100  // read and write buffer size, reduce it if memory of Arduino is not
enough

byte spcId = 1;

ExpansionSerial port(spcId);

byte rwbuf[BUFFER_SIZE];  // read and write buffer

void setup() {
  Serial.begin(9600);
  while(!Serial)
    ;

  Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
  Expansion.begin();
  port.begin("115200N81N");
}

void loop() {
  int txfree = port.availableForWrite();

  // gets the size of received data
  int rxlen = port.available();
```

```
  if(rxlen > 0) {
    // gets the next byte of incoming serial data without removing
    int len = port.available();
    Serial.print("available length before peek(): ");
    Serial.println(len);

    char value = port.peek();
    Serial.print("peek() value: ");
    Serial.println(value);

    len = port.available();
    Serial.print("available length after peek(): ");
    Serial.println(len);

    Serial.println("----------");

    // remove one byte from the buffer
    char data = port.read();
  }
  delay(1);
}
```

read()

Description

Reads incoming serial data. The read() inherits from the Stream utility class.

Syntax

port.read()

Parameters

none

Returns

the first byte of incoming serial data available, or -1 if no data is available (int)

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>
#define BUFFER_SIZE 100  // read and write buffer size, reduce it if memory of Arduino is not
enough

byte spcId = 1;

ExpansionSerial port(spcId);

byte rwbuf[BUFFER_SIZE];  // read and write buffer

void setup() {
    Serial.begin(9600);
    while(!Serial)
        ;

    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    Expansion.begin();
    port.begin("115200N81N");
}

void loop() {
    int txfree = port.availableForWrite();

    // gets the size of received data
    int rxlen = port.available();

    if(rxlen > 0) {
```

```
        // reads the next byte of incoming serial data
        int value = port.read();
        Serial.print("read : ");
        Serial.println(value);

    }
    delay(1);
}
```

availableForWrite()

Description

Gets the number of bytes (characters) available for writing in the serial buffer without blocking the write operation.

Syntax

port.availableForWrite()

Parameters

none

Returns

the number of bytes available to write

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>
#define BUFFER_SIZE 100  // read and write buffer size, reduce it if memory of Arduino is not enough

byte spcId = 1;

ExpansionSerial port(spcId);

byte rwbuf[BUFFER_SIZE];  // read and write buffer

void setup() {
    Serial.begin(9600);
    while(!Serial)
        ;

    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    Expansion.begin();
    port.begin("115200N81N");
}

void loop() {

    // Get the number of bytes available for writing
    int txfree = port.availableForWrite();

    int rxlen = port.available();
```

```
    if(rxlen > 0) {
        if(rxlen <= txfree) {
            int rwlen; // read and write length

            if(rxlen <= BUFFER_SIZE)
                rwlen = rxlen;
            else
                rwlen = BUFFER_SIZE;

            // receive data
            rwlen = port.readBytes(rwbuf, rwlen);

            // send data
            port.write(rwbuf, rwlen);

            // print data to serial monitor of Arduino IDE
            Serial.write(rwbuf, rwlen);
        }
    }

    delay(1);
}
```

flush()

Description

Waits for the transmission of outgoing serial data to complete.

Syntax

port.flush()

Parameters

none

Returns

none

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>

byte spcId = 1;

ExpansionSerial port(spcId);

// read and write buffer
byte wbuf[] = "abcdefghijklmnopqrstuvwxyz0123456789abcdefghijklmnopqrstuvwxyz0123456789";

void setup() {
  Serial.begin(9600);
  while(!Serial)
    ;

  Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
  Expansion.begin();

  port.begin("9600N81N");
}

void loop() {
  int txfree = port.availableForWrite();
  Serial.println(txfree);

  if(txfree >= 72) {

    port.write(wbuf, 72);
  }
```

```
    else {

      Serial.println("waits until sending data ...");

      // waits for the transmission of outgoing serial data to complete
      port.flush();
    }

    delay(1);
  }
```

write()

Description

Writes binary data to the serial port. This data is sent as a byte or series of bytes.

Syntax

port.write(byte)
port.write(wbuf, wlen)

Parameters

byte - a value to send as a single byte
wbuf - a string to send as a series of bytes
wlen - the number of bytes to send

Returns

This function returns the number of bytes written.

Example

```
#include <PhpocExpansion.h>
#include <Phpoc.h>
#define BUFFER_SIZE 100  // read and write buffer size, reduce it if memory of Arduino is not
enough

byte spcId = 1;

ExpansionSerial port(spcId);

byte rwbuf[BUFFER_SIZE];  // read and write buffer

void setup() {
    Serial.begin(9600);
    while(!Serial)
        ;

    Phpoc.begin(PF_LOG_SPI | PF_LOG_NET);
    Expansion.begin();
    port.begin("115200N81N");
}

void loop() {

    int txfree = port.availableForWrite();
    int rxlen = port.available();
```

```
    if(rxlen > 0) {
        if(rxlen <= txfree) {
            int rwlen; // read and write length

            if(rxlen <= BUFFER_SIZE)
                rwlen = rxlen;
            else
                rwlen = BUFFER_SIZE;

            // receive data
            rwlen = port.readBytes(rwbuf, rwlen);

            // send data
            port.write(rwbuf, rwlen);

            // print data to serial monitor of Arduino IDE
            Serial.write(rwbuf, rwlen);
        }
    }

    delay(1);
}
```