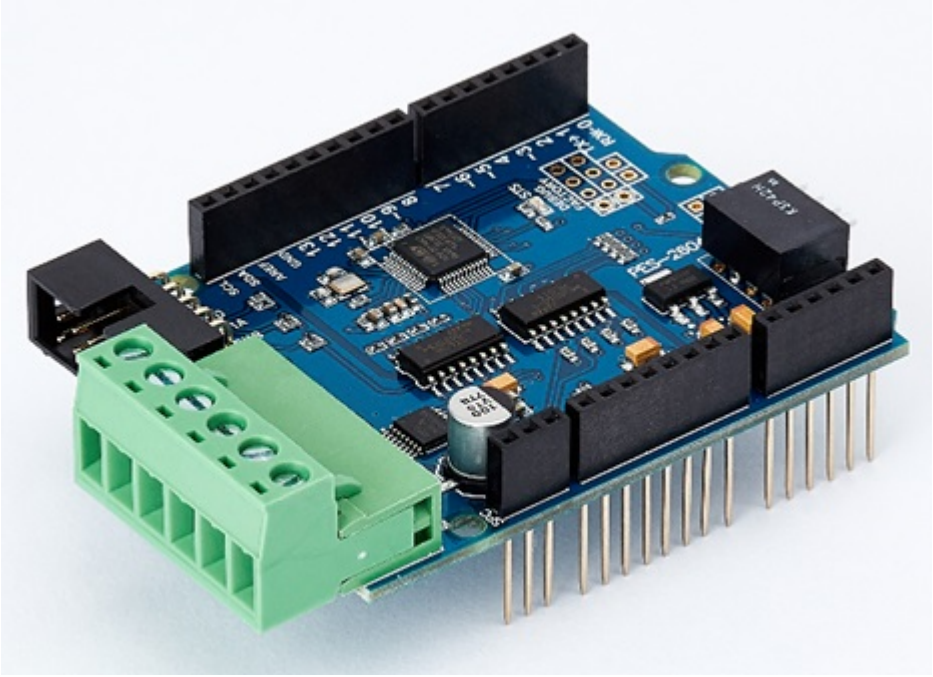
Introduction



PES-2604

PES-2604, DC motor controller, is a smart expansion board for PHPoC shield for Arduino. You can easily control two brushed DC motors with this board via the Arduino sketch.

Highlights of PES-2604

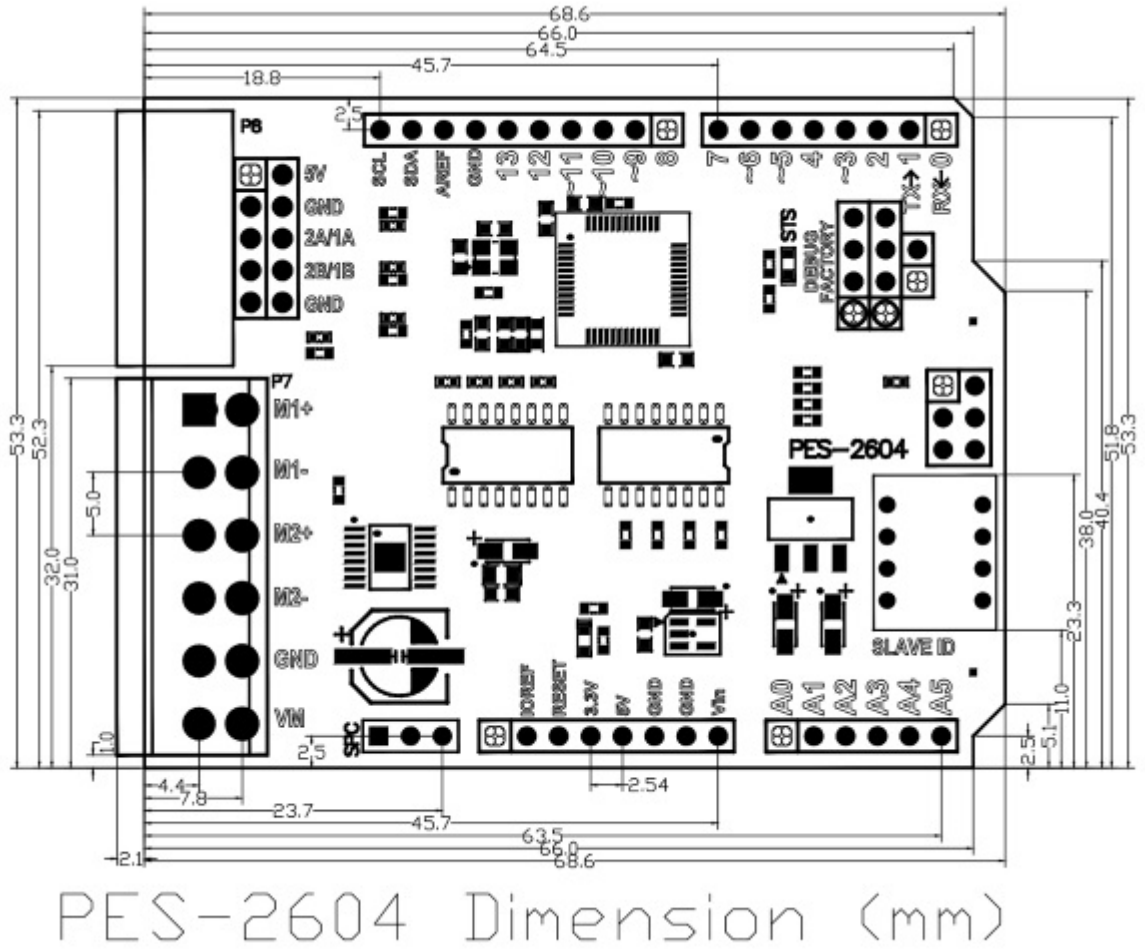
- brushed DC motor controller
- dual DC motor ports with encoder ports
- motor voltage: 4 ~ 18V [DC]
- motor current: Maximum 1A on each port

※ Caution : Both a PHPoC shield with R2 or later version and an Arduino board are required to use this board!

> What is the Smart Expansion Board for PHPoC shield?
>
> A smart expansion board for PHPoC shield has own devices and firmware. This board communicate with a PHPoC shield in a master-slave protocol through the designated port. Two or more smart expansion boards can be connected to one PHPoC shield and each of them required to be setting a slave id.

Dimension

Body



PES-2604 Dimension (mm)

※ Dimensions(unit : mm) may vary according to a method of measurement.

Terminal Block

This board uses two types of 6-pole terminal block. Refer to each datasheet for dimension.

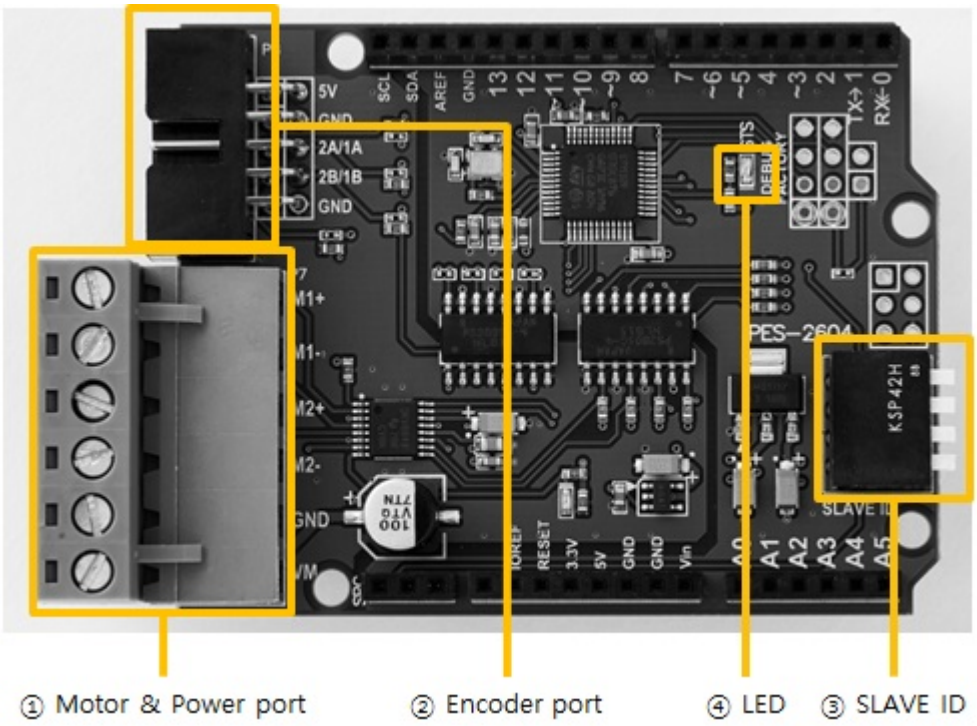
- Datasheet of T-type Terminal Block
- Datasheet of S-type Terminal Block

Schematic

This is the schematic of PES-2604.

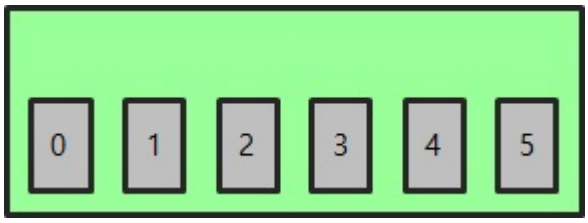
- PES-2604-V11-PO.pdf

Layout



1. Motor & Power port

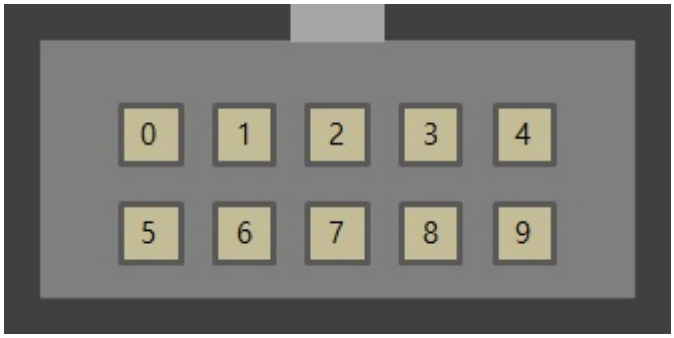
This port is for connecting PES-2604 to motor and the motor's power supply. It has six terminals and a 5mm pitch.



Number	Name	I/O	Description
0	M1+	Out	Motor port 1, positive(+)
1	M1-	Out	Motor port 1, negative(-)
2	M2+	Out	Motor port 2, positive(+)
3	M2-	Out	Motor port 2, negative(-)
4	GND	-	Power supply for motors (Ground)
5	VM	In	Power supply for motors (DC 4 ~ 18V)

2. Encoder port

This port is for connecting encoders.



Number	Name	I/O	Description
0	5V	Out	Power supply for encoders (DC 5V)
1	GND	-	Power supply for encoders (Ground)
2	1A	In	Encoder A phase for motor port 1
3	1B	In	Encoder B phase for motor port 1
4	GND	-	Power supply for encoders (Ground)
5	5V	Out	Power supply for encoders (DC 5V)
6	GND	-	Power supply for encoders (Ground)
7	2A	In	Encoder A phase for motor port 2
8	2B	In	Encoder A phase for motor port 2
9	GND	-	Power supply for encoders (Ground)

※ 1A, 1B, 2A and 2B are internally pulled up.

3. SLAVE ID Switch

A slave ID is used when PHPoC shield identifies each smart expansion board. So, each smart expansion board, which is connected to a PHPoC shield, should have a unique slave ID. The slave ID can be set one of the numbers from 1 to 14 by 4 DIP switches as follows:

4. LED

This board has STS LED which indicates the board's status.

State	Operation
Normal	Repeat On/Off in every second
Invalid slave ID	Blinks very quickly
Fail to communicate with PHPoC shield	Off

How to Use

This board can be used by steps as follows.

1. Connect to a PHPoC Shield and an Arduino

It is not possible to use this board alone. Please be sure that connection to a PHPoC Shield and an Arduino.

2. Install Libraries for Arduino

Install PHPoC and PhpocExpansion library via library manager on Arduino IDE. Both libraries are required to use PHPoC shield and this board. Refer to the manual pages below for detail about the libraries.

- PHPoC shield library reference

3. Use Sample Codes

Use sample codes in libraries and examples in this manual.

Class and Functions

Class

To use this extension board, use the ExpansionDCMotor class of the PHPoC Expansion library.

Member Functions

Available member functions of the ExpansionDCMotor class are as follows:

Member Function	Description
int getPID(void)	get the product's ID
char *getName(void)	get the product's name
ExpansionDCMotor(int sid, int port)	create an instance of a motor port
setPolarity(int pol)	set a PWM polarity
setDirection(int dir)	set a direction of rotation
setPeriod(long period)	set a PWM period
setWidth(long width)	set a HIGH duration (a duty cycle)
setDecay(int decay)	set decay mode
setEncoderPolarity(int pol)	set a count direction of encoder
setEncoderPosition(long pos)	set a counter value of encoder
setEncoderPSR(int psr)	set a sampling count of encoder
getEncoderPosition(void)	get a counter value of encoder
getEncoderPeriod(void)	get a period of encoder output
setFilterFrequency(long freq)	set a cut-off frequency of LPF
setFilterPNC(int pnc)	set a noise counter of LPF
getFilterPNC(void)	get a noise counter of LPF

Controlling Motors and Setting PWM

Major Functions for a Motor

Creating a Instance of a Motor Port

Connect a motor to a motor port of this board and create a instance of the port by using ExpansionDCMotor() function.

```
ExpansionDCMotor dcmotor(sid, port);
```

- sid: slave ID(1 ~ 14)
- port: port number(1 or 2)

Setting a PWM period

Set a PWM period by using setPeriod() function.

```
dcmotor.setPeriod(period);
```

- period: PWM period in microsecond unit

Controlling a Motor

Set a HIGH duration for controlling a motor by using setWidth() function.

```
dcmotor.setWidth(width);
```

The HIGH duration is the time during which the HIGH signal is output within one cycle of the PWM signal. Setting the HIGH duration determines the duty cycle of the PWM signal.

```
Duty Cycle(%) = HIGH duration / period * 100
```

PWM output starts simultaneously with this setting, so this function drives the motor.

Example

- source code for Arduino

```
#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);
    }
}
```

Other Functions

Setting a PWM Polarity

Set a PWM polarity by using setPolarity() function.

```
dcmotor.setPolarity(pol);
```

- pol : PWM polarity

pol	polarity
0 or greater than 0	normal polarity(default)
less than 0	reverse polarity

Setting a Direction of Rotation

Set a direction of rotation by using setDirection() function.

```
dcmotor.setDirection(dir);
```

- dir : direction of rotation

pol	polarity
0 or greater than 0	forward(default)
less than 0	reverse

※ Note : The direction of rotation is affected by both direction of rotation and PWM polarity.

PWM polarity value	direction of rotation value	actual direction of rotation
normal	forward	clockwise
normal	reverse	counterclockwise
reverse	forward	counterclockwise
reverse	reverse	clockwise

Setting Decay Mode

Set decay mode by using setDecay() function.

```
dcmotor.setDecay(decay)
```

- decay : decay mode

decay	polarity
0	slow decay
otherwise	fast decay

Setting and Monitoring Encoders

Major Functions for an encoder

Monitoring counter values of encoders

Connect a motor's encoder to an encoder port of this board and monitor counter values by using getEncoderPosition() function.

```
dcmotor.getEncoderPosition();
```

- 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);
}
```

Monitoring periods of encoder outputs.

Monitor periods of encoder outputs by using getEncoderPeriod() function.

```
period = dcmotor.getEncoderPeriod();
```

The unit of the return values is is microseconds (us). To reduce the error when monitoring periods of encoder outputs, set the encoder sampling count to a large value.

- 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);
}
```

Other Functions

Setting count direction

Set count direction of encoder by using setEncoderPolarity() function.

```
dcmotor.setEncoderPolarity(pol);
```

- pol - count direction

pol	polarity
0 or greater than 0	normal(default)
less than 0	reverse

※ Note : The count direction of encoder is affected by motor's direction of rotation.

count direction of encoder	motor's direction of rotation	count value
normal	clockwise	increase
normal	counterclockwise	decrease
reverse	counterclockwise	increase
reverse	clockwise	decrease

Setting a counter value

Set or initialize counter values of encoder by using setEncoderPosition() function.

```
dcmotor.setEncoderPosition(pos);
```

- pos - a counter value of encoder

  You can set the counter value from -1000000000 (-1 billion) to +1000000000 (1 billion).

Setting a sampling count

Set a sampling count of encoder by using setEncoderPSR() function.

```
dcmotor.setEncoderPSR(psr);
```

- psr - sampling count

  The encoder sampling count is the number of pulses used to measure the period of the encoder output pulse. The more sampling counts, the smaller the error. The encoder sampling count can be set from 1 to 64.

Setting and Monitoring a Low-pass Filter

Major Functions for a Low-pass Filter

Setting a Cut-off Frequency

Set a cut-off frequency of a low-pass filter by using setFilterFrequency() function.

```
dcmotor.setFilterFrequency(freq);
```

- freq - cut-off frequency

When a cut-off frequency of the low-pass filter is set, signals corresponding to frequencies higher than the cut-off frequency are not counted during encoder monitoring.

Monitoring Noise Counter Values

Monitoring noise counter values by using getFilterPNC() function.

```
pnc = dcmotor.getFilterPNC();
```

Example

- source code for Arduino

```
#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);
}
```

Other Functions

Setting a Noise Counter Value

Set or initialize a noise counter value by using setFilterPNC() function.

```
dcmotor.setFilterPNC(pnc);
```

- pnc - a noise counter value