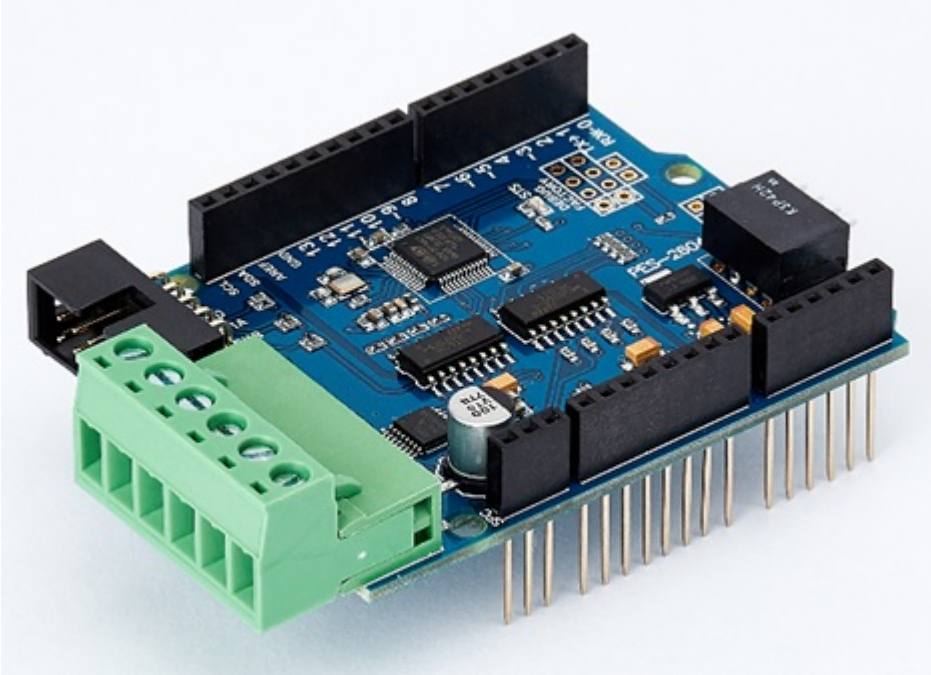
제품 소개



PES-2604

DC모터 제어기 PES-2604는 아두이노용 PHPoC 쉴드 제품 전용 스마트 확장보드입니다. 이 보드를 이용하면 아두이노 스케치를 통해 원격으로 2개의 brushed DC모터를 손쉽게 컨트롤 할 수 있습니다.

PES-2604의 주요 특징

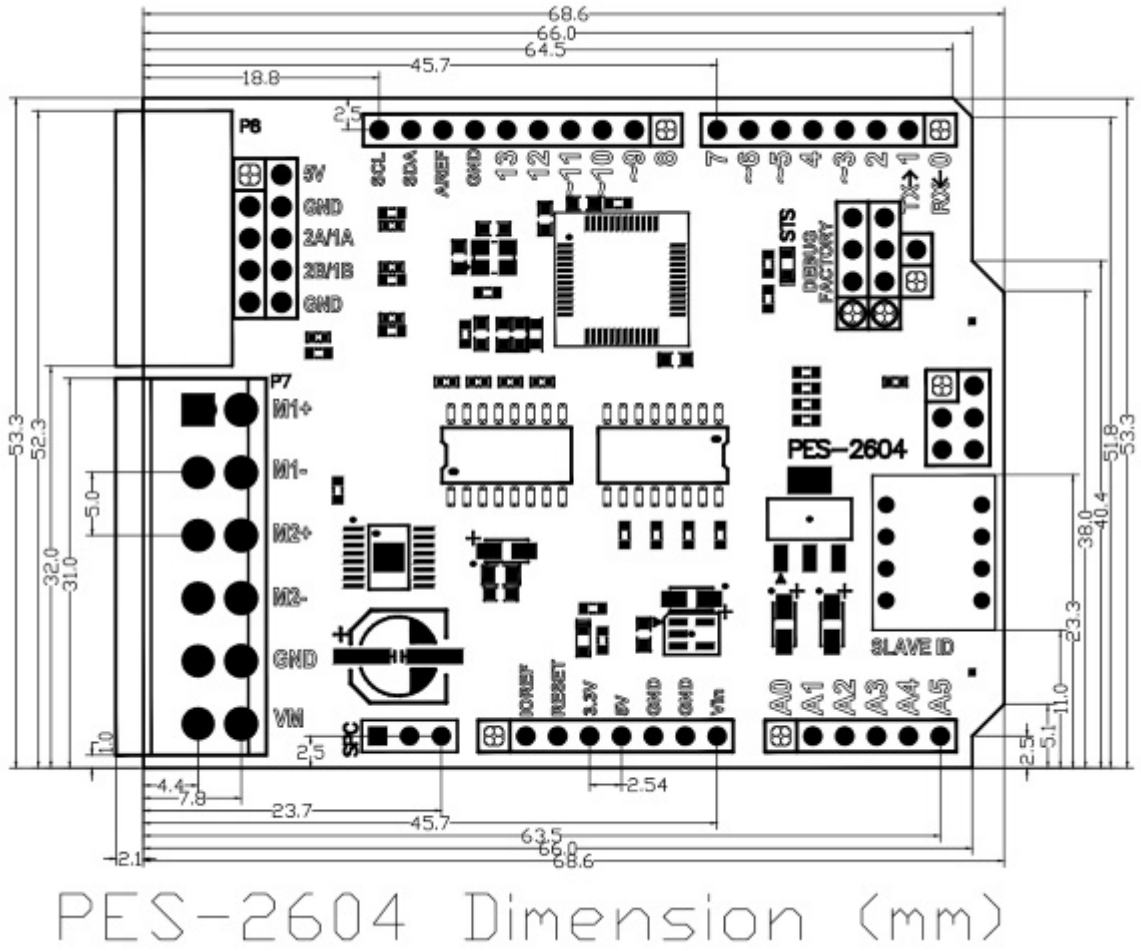
- Brushed DC모터 제어기
- 2개의 DC모터 및 엔코더 연결포트
- 모터 전압: 4 ~ 18V [DC]
- 모터 전류: 포트당 최대 1A

※ 주의 : 이 확장보드를 사용하기 위해서는 반드시 아두이노와 R2 이상 버전의 PHPoC 쉴드가 필요합니다!

> PHPoC 쉴드용 스마트 확장보드란?
>
> PHPoC 쉴드용 스마트 확장보드는 자체 디바이스와 전용 펌웨어를 내장하고 있습니다. 이 보드는 PHPoC 쉴드와 전용 통신 포트를 이용해 마스터-슬레이브 방식으로 통신합니다. 하나의 PHPoC 쉴드에 여러 개의 스마트 확장보드를 연결할 수 있으며 각각의 스마트 확장보드에는 반드시 슬레이브 아이디를 설정해야 합니다.

치수

제품 본체



PES-2604 Dimension (mm)

※ 치수(단위 : mm)는 제품 상태 및 재는 각도 등에 따라 약간의 오차가 있을 수 있습니다.

터미널블록

이 보드는 6폴 터미널블록을 사용합니다. 치수는 각 터미널블록의 데이터시트를 참조하시기 바랍니다.

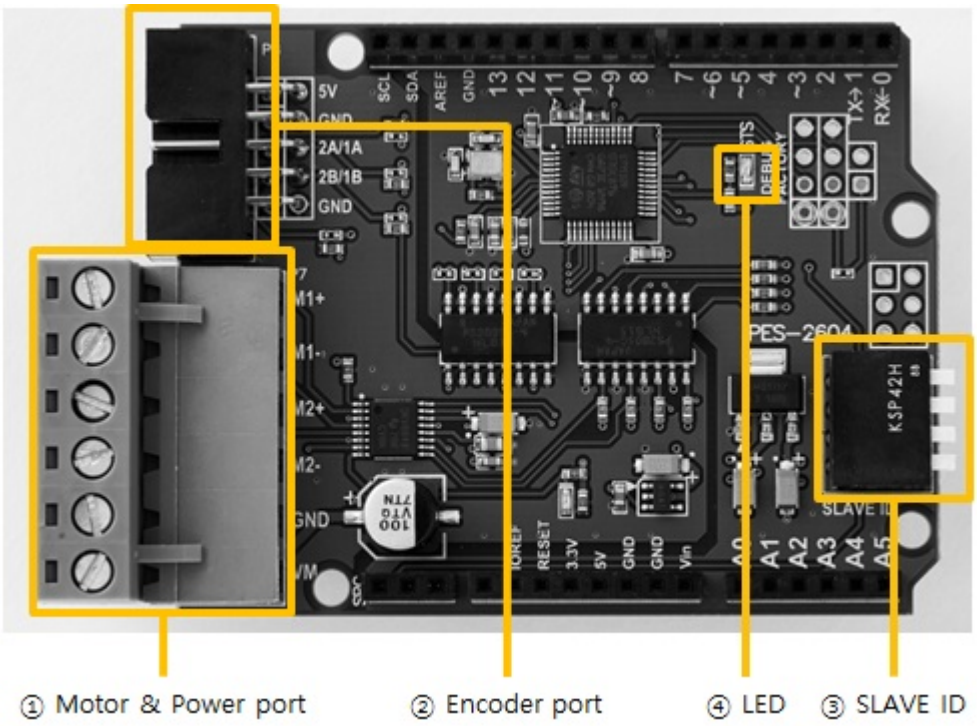
- T형 터미널블록 데이터시트
- S형 터미널블록 데이터시트

회로도

PES-2604의 회로도 입니다.

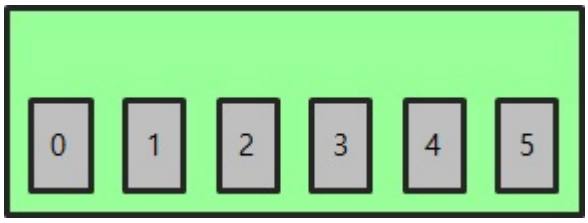
- PES-2604-V11-PO.pdf

레이아웃



1. 모터 & 전원포트

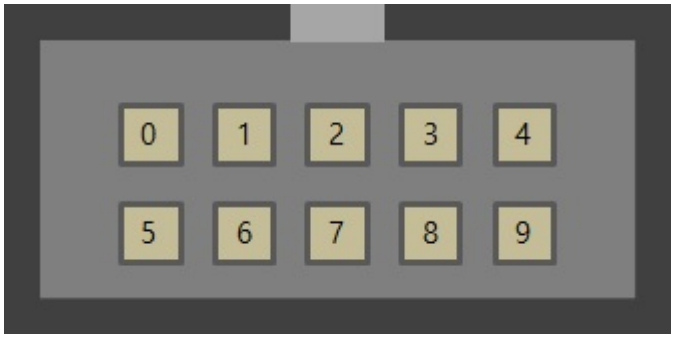
모터 & 전원포트는 모터와 모터 구동을 위한 전원을 연결하는 포트입니다. 이 포트는 5mm간격의 1 by 6 터미널 블록으로 되어 있습니다.



번호	이름	방향	설명
0	M1+	출력	모터 연결포트 1, 양극(+)
1	M1-	출력	모터 연결포트 1, 음극(-)
2	M2+	출력	모터 연결포트 2, 양극(+)
3	M2-	출력	모터 연결포트 2, 음극(-)
4	GND	-	모터 구동용 전원 (Ground)
5	VM	입력	모터 구동용 전원 (DC 4 ~ 18V)

2. 엔코더 연결포트

이 포트는 엔코더를 연결하는 포트입니다.

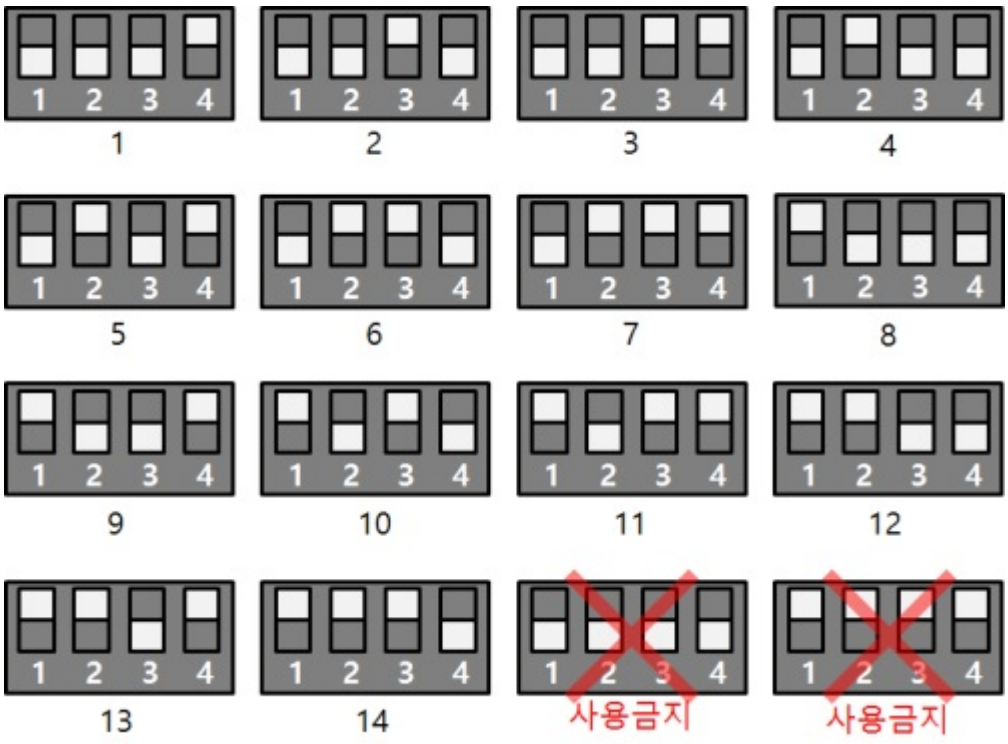


번호	이름	방향	설명
0	5V	출력	엔코더 전원 (DC 5V)
1	GND	-	엔코더 전원 (Ground)
2	1A	입력	포트 1번의 엔코더 A상
3	1B	입력	포트 1번의 엔코더 B상
4	GND	-	엔코더 전원 (Ground)
5	5V	출력	엔코더 전원 (DC 5V)
6	GND	-	엔코더 전원 (Ground)
7	2A	입력	포트 2번의 엔코더 A상
8	2B	입력	포트 2번의 엔코더 B상
9	GND	-	엔코더 전원 (Ground)

※ 1A, 1B, 2A 및 2B는 내부적으로 풀업(pull-up)되어 있습니다.

3. 슬레이브 아이디 스위치

슬레이브 아이디는 PHPoC 쉴드가 스마트 확장보드 각각을 구분하는데 사용됩니다. 따라서 PHPoC 쉴드에 연결되는 각 스마트 확장보드는 고유한 슬레이브 아이디를 사용해야 합니다. 슬레이브 아이디는 1부터 14까지 14개 중 하나로 설정할 수 있으며 다음과 같이 4개의 딥스위치를 조정하여 설정합니다.



4. LED

이 보드에는 상태를 나타내는 STS LED가 있습니다.

상태	LED 동작
정상	1초마다 켜짐/꺼짐 반복
슬레이브 아이디 설정 이상	매우 빠르게 깜박임
PHPoC보드와 통신 불가능	꺼짐

사용하기

이 보드를 사용하는 방법은 다음과 같습니다.

1. PHPoC 쉴드와 아두이노에 연결

이 보드는 단독으로 사용할 수 없습니다. 반드시 아두이노와 아두이노용 PHPoC 쉴드에 연결하여 사용하시기 바랍니다.

2. 아두이노용 라이브러리 설치

아두이노 IDE의 라이브러리 매니저를 통해 Phpoc 라이브러리와 PhpocExpansion 라이브러리를 설치합니다. 아두이노용 PHPoC 쉴드와 스마트 확장보드를 사용하려면 반드시 두 라이브러리를 모두 설치해야 합니다. 라이브러리에 대한 자세한 내용은 다음 문서를 참조하시기 바랍니다.

- PHPoC 쉴드 라이브러리 레퍼런스

3. 예제코드 활용

본 매뉴얼과 라이브러리에 포함된 예제코드를 활용하여 프로그래밍 하시기 바랍니다.

클래스 및 함수

클래스

이 확장보드를 사용하기 위해서는 아두이노 PHPoC 라이브러리의 ExpansionDCMotor 클래스를 사용합니다.

멤버 함수

ExpansionDCMotor 클래스의 사용 가능한 멤버함수는 다음과 같습니다.

멤버 함수	설명
int getPID(void)	제품 아이디 읽기
char *getName(void)	제품명 읽기
ExpansionDCMotor(int sid, int port)	모터 포트의 인스턴스 생성
setPolarity(int pol)	PWM 극성 설정
setDirection(int dir)	모터의 회전 방향 설정
setPeriod(long period)	PWM 주기 설정
setWidth(long width)	유효 시간(듀티 사이클) 설정
setDecay(int decay)	감쇠모드 설정
setEncoderPolarity(int pol)	엔코더 카운트 방향 설정
setEncoderPosition(long pos)	엔코더 카운터값 설정
setEncoderPSR(int psr)	엔코더 샘플링 카운트 설정
getEncoderPosition(void)	엔코더 카운터값 모니터링
getEncoderPeriod(void)	엔코더 출력 펄스 주기 모니터링
setFilterFrequency(long freq)	저역통과필터 컷오프 주파수 설정
setFilterPNC(int pnc)	저역통과필터 노이즈 카운터값 설정
getFilterPNC(void)	저역통과필터 노이즈 카운터값 모니터링

모터 제어 및 PWM 설정

모터 제어 관련 주요 함수

모터 포트 인스턴스 생성

모터를 이 보드에 연결하고 ExpansionDCMotor() 함수를 이용해 특정 해당 포트에 대한 인스턴스를 생성합니다.

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

- sid: 슬래이브 아이디(1 ~ 14)
- port: 모터 포트 번호(1 또는 2)

PWM 주기 설정

setPeriod() 함수를 이용해 PWM의 주기를 설정합니다.

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

- period: PWM 주기(마이크로 초 단위)

모터 제어

setWidth() 함수를 이용해 모터제어를 위한 유효시간을 설정합니다.

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

유효시간은 PWM의 HIGH신호가 출력되는 시간입니다. 이 시간과 PWM주기에 따라 듀티사이클이 결정됩니다.

```
듀티사이클(%) = 유효시간 / 주기 * 100
```

이 함수가 실행되는 순간 듀티사이클에 따라 모터가 구동됩니다.

예제

- 아두이노 소스코드

```
#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);
    }
}
```

기타 함수

PWM 극성 설정

setPolarity() 함수를 이용해 PWM신호의 극성을 설정할 수 있습니다.

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

- pol : PWM 극성

pol	polarity
0보다 크거나 같음	정상 극성(기본 값)
0보다 작음	반대 극성

회전 방향 설정

setDirection() 함수를 이용해 모터의 회전 방향을 설정할 수 있습니다.

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

- dir : 회전 방향

pol	polarity
0보다 크거나 같음	정방향(기본 값)
0보다 작음	역방향

※참고: 실제 모터의 회전 방향은 PWM 극성 설정과 회전 방향 설정에 영향을 받습니다.

PWM 극성 설정	회전 방향 설정	실제 회전 방향
정상극성	정방향	시계방향
정상극성	역방향	반시계방향
반대극성	정방향	반시계방향
반대극성	역방향	시계방향

감쇠모드 설정

setDecay() 함수를 이용해 감쇠모드를 설정할 수 있습니다.

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

- decay : 감쇠 모드

decay	polarity
0	느린 감쇠
그 외	빠른 감쇠

엔코더 설정 및 모니터링

엔코더 관련 주요 함수

엔코더 카운터값 모니터링

모터의 엔코더포트를 이 보드에 연결하고 getEncoderPosition() 함수를 이용해 엔코더 카운터값을 모니터링 하십시오.

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

- 예제

```
#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);
}
```

엔코더 출력 펄스 주기 모니터링

getEncoderPeriod() 함수를 이용해 엔코더 출력 펄스 주기 모니터링 하십시오.

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

엔코더 출력 펄스 주기의 단위는 마이크로 초(us) 입니다. 엔코더 출력 펄스 주기를 모니터링할 때 오차를 줄이기 위해서는 엔코더 샘플링 카운트를 높은 값으로 설정하시기 바랍니다.

- 예제

```
#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);
}
```

기타 함수

엔코더 카운트 방향 설정

setEncoderPolarity() 함수를 이용해 엔코더 카운트 방향을 설정할 수 있습니다.

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

- pol - 엔코더 카운트 방향

pol	polarity
0보다 크거나 같음	정방향(기본 값)
0보다 작음	역방향

※참고: 실제 엔코더 카운터값은 모터의 회전 방향에 영향을 받습니다.

엔코더 카운트 방향 설정	모터의 회전 방향	카운터 값
정방향	시계방향	증가
정방향	반시계방향	감소
역방향	반시계방향	증가
역방향	시계방향	감소

엔코더 카운터값 설정

setEncoderPosition() 함수를 이용해 엔코더 카운터값을 초기화 또는 변경할 수 있습니다.

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

- pos - 엔코더 카운터값

  카운터값은 -1000000000(-10억) 부터 1000000000(10억)까지 설정할 수 있습니다.

엔코더 샘플링 카운트 설정

setEncoderPSR() 함수를 이용해 엔코더 샘플링 카운트를 설정할 수 있습니다.

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

- psr - 엔코더 샘플링 카운트

  엔코더 샘플링 카운트는 엔코더 출력 펄스의 주기를 측정할 때 사용할 펄스의 개수를 의미합니다. 샘플링 카운트가 많을수록 오차는 줄어듭니다. 엔코더 샘플링 카운트는 1부터 64까지 설정할 수 있습니다.

저역통과필터 설정 및 모니터링

저역통과필터 관련 주요 함수

저역통과필터 컷오프 주파수 설정

setFilterFrequency() 함수를 이용해 저역통과필터 컷오프 주파수를 설정합니다.

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

- freq - 컷오프 주파수

저역통과필터 컷오프 주파수를 설정하면 엔코더 모니터링 시 해당 주파수보다 높은 주파수에 해당하는 신호는 카운트하지 않습니다.

저역통과필터 노이즈 카운터값 모니터링

getFilterPNC() 함수를 이용해 저역통과필터 노이즈 카운터값을 모니터링 하십시오.

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

저역통과필터 노이즈 카운터값 모니터링 예

```
#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);
}
```

기타 함수

저역통과필터 노이즈 카운터값 설정

setFilterPNC() 함수를 이용해 저역통과필터 노이즈 카운터값을 초기화 또는 변경할 수 있습니다.

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

- pnc - 노이즈 카운터값