보이지 않는 곳에서 세상을 편리하게 하는힘
오토닉스 센서 & 제어기기

Autonics
Sensors & Controllers

Motion Controller

# PMC-2HSP/2HSN

## 라이브러리 매뉴얼


PMC-2HSP/2HSN

저희 (주)오토닉스 제품을 구입해 주셔서 감사합니다.
사용 전에 안전을 위한 주의 사항을 반드시 읽고 정확하게 사용하여 주십시오.

# 제품 구입 감사 안내문

㈜오토닉스 제품을 구입해 주셔서 감사합니다.

먼저 안전을 위한 주의사항을 반드시 읽고 제품을 올바르게 사용해 주십시오.

본 라이브러리 매뉴얼은 제품에 대한 안내와 바른 사용 방법에 대한 내용을 담고 있으므로 사용자가 쉽게 찾아 볼 수 있는 장소에 보관하여 주십시오.

# 라이브러리 매뉴얼 안내

- 라이브러리 매뉴얼의 내용을 충분히 숙지한 후에 제품을 사용하여 주십시오.
- 라이브러리 매뉴얼은 사용자에게 제공되는 라이브러리 함수를 자세하게 설명한 것으로 라이브러리 매뉴얼 이외의 내용에 대해서는 보증하지 않습니다.
- 라이브러리 매뉴얼의 일부 또는 전부를 무단으로 편집 또는 복사하여 사용할 수 없습니다.
- 라이브러리 매뉴얼은 제품과 함께 제공하지 않습니다.
  당사 홈페이지(www.autonics.co.kr)에서 다운로드하여 사용하십시오.
- 라이브러리 매뉴얼의 내용은 해당 제품의 성능 및 소프트웨어 개선에 따라 사전 예고없이 변경될 수 있으며, 업그레이드 공지는 당사 홈페이지를 통해 제공해 드립니다.
- 당사에서는 라이브러리 매뉴얼의 내용을 조금 더 쉽게, 정확하게 작성하고자 많은 노력을 기울였습니다. 그럼에도 불구하고 수정해야 될 부분이나 질문사항이 있으시면 당사 홈페이지를 통하여 의견을 주시기 바랍니다.
- 각 함수별 사용예제와 함수를 활용한 MFC 기반의 예제프로그램을 제공합니다.

# 라이브러리 매뉴얼의 공통 기호

| 기호 | 설명 |
|---|---|
|  Note | 해당 기능에 대한 보충 설명 |
|  Warning | 지시 사항을 위반할 경우 심각한 상해나 사망 사고의 위험이 있는 내용 |
|  Caution | 지시 사항을 위반할 경우 경미한 상해나 제품 손상이 발생할 수 있는 내용 |
|  Ex. | 해당 기능에 대한 예시 |
| ※1 | 주석 설명 표시 |

# 안전을 위한 주의사항

- 안전을 위한 주의사항은 제품을 안전하고 올바르게 사용하여 사고나 위험을 미리 막기 위한 것이므로 반드시 지키십시오.
- 주의사항은 경고와 주의로 구분되어 있으며 각각의 의미는 다음과 같습니다.

| | | |
|---|---|---|
| Warning | 경고 | 지시 사항을 위반하였을 때, 심각한 상해나 사망 사고가 발생할 가능성이 있는 경우 |
| Caution | 주의 | 지시 사항을 위반하였을 때, 경미한 상해나 제품 손상이 발생할 가능성이 있는 경우 |

## Warning

- 인명이나 재산상에 영향이 큰 기기(예: 원자력 제어 장치, 의료기기, 선박, 차량, 철도, 항공기, 연소장치, 안전장치, 방범/방재장치 등)에 사용할 경우에는 반드시 2 중으로 안전장치를 부착한 후 사용하십시오.

  화재, 인사사고, 재산상의 막대한 손실이 발생할 수 있습니다.
- 제품을 사용하기 전에 라이브러리 매뉴얼을 충분히 숙지한 다음 설치 및 운전하여 주십시오. 조작 실수로 인한 기계적인 손실과 인명사고 및 오동작의 원인이 됩니다.
- 가연성 가스 또는 폭발성 가스 사용 장소, 직사광선이 비추는 장소에서는 사용 하지 마십시오. 감전, 화재, 인명사고, 재산상의 손실이 발생할 수 있습니다.
- 제품 설치 시 위험 예상 지점에는 반드시 비상 정지 스위치, 리미트 스위치를 설치하십시오. 인명사고, 재산상의 손실이 발생할 수 있습니다.
- 제품을 설치할 때에는 정전 대책을 세운 후 설치하십시오. 인명사고, 재산상의 손실이 발생할 수 있습니다.
- 제품의 통풍창에 금속성 등의 이물질이 들어가지 않도록 하십시오. 화재, 감전의 우려가 있습니다.
- 전원 입력은 반드시 절연 트랜스를 사용하여 정류된 전원을 사용하십시오. 화재나 감전, 부상의 우려가 있습니다.
- 전원 입력 사양을 반드시 확인하시고, 전원 연결 시 반드시 단자를 확인한 후 연결하십시오. 화재의 우려가 있습니다.
- 전원이 인가된 상태에서 결선 및 점검, 보수를 하지 마십시오. 감전, 제품의 손상, 오동작의 원인이 됩니다.
- 운전 중에 전원을 차단하거나 커넥터를 분리하지 마십시오. 인명사고, 재산상의 손실이 발생할 수 있습니다.
- 제품을 분해 및 개조하지 마십시오. 감전이나 화재의 우려가 있습니다.

**Caution**

- 제품의 전원선과 신호선은 단단히 고정시키십시오. 감전, 제품손상의 원인이 됩니다.
- 전원 커넥터와 RS485 용 커넥터의 나사는 0.4N·m 이하의 토크로 조이십시오. 나사가 파손되어 접촉불량의 원인이 됩니다.
- 전원 배선은 AWG28-16 규격의 배선을 사용하십시오. 화재의 원인이 됩니다.
- 입/출력 배선에 리본 케이블을 이용할 시에는 케이블을 올바르게 접속시켜 주시고 리본 케이블에 의한 접촉 불량이 일어나지 않도록 하십시오. 오동작의 원인이 됩니다.
- 반드시 커넥터의 사양 및 형식을 확인 후 배선하여 주십시오. 화재나 감전 및 제품 파손의 우려가 있습니다.
- 반드시 정격/성능 범위에서 사용하십시오. 제품의 수명이 짧아지는 원인이 되며 화재의 우려가 있습니다.
- 케이블은 노이즈의 영향을 받지 않도록 가능한 한 전원선, 동력선, 부하선 등으로부터 분리 배선하여 사용하십시오. 오동작 및 제품 손상의 원인이 됩니다.
- 청소 시 물, 유기 용제를 사용하지 마십시오. 감전, 화재, 제품 손상의 원인이 됩니다.
- 제품의 폐기 시에는 산업 폐기물로서 처리하십시오.
- 이 기기는 업무용(A 급)으로 전자파 적합등록을 한 기기이오니 판매자 또는 사용자는 이점을 주의하시기 바라며, 가정 이외의 지역에서 사용하는 것을 목적으로 합니다.

# Table of Contents



# 1 초기화

## 1.1 autpmc_Open

autpmc_Open 함수는 PMC-2HSP/2HSN 에 통신 연결을 합니다.

**(1) 함수명**

```
int autpmc_Open(
int PortNum,
int BaudRate
);
```

**(2) 파라미터**

- PortNum
  접속하려는 Serial Port 숫자를 입력합니다.
- BaudRate
  Serial Port 의 Baudrate 을 입력합니다.

| 구분 | 입력 | 내용 | 상수값 |
|---|---|---|---|
| PMC_BAUDRATE | FPMC_BAUD_9600 | 9,600bps | 9600 |
| | FPMC_BAUD_19200 | 19,200bps | 19200 |
| | FPMC_BAUD_38400 | 38,400bps | 38400 |
| | FPMC_BAUD_57600 | 57,600bps | 57600 |
| | FPMC_BAUD_115200 | 115,200bps | 115200 |

**(3) 리턴값**

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력 에러 | FPMC_INVALID_PORT | 1 | 입력한 Port 가 사용 중이거나 잘못된 Port 번호를 입력하였습니다. |
| | FPMC_INVALID_BAUDRATE | 2 | 잘못된 Baudrate 을 입력하였습니다. |

**(4) 사용 예**

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define COMPORT 15

void main()
{

 int stat=0;
 int i;
```

```
// PMC-2HSP(N)의 통신 접속 함수
// 리턴값 : 정상적으로 명령어 수행시에는 FPMC_OK 를 리턴한다.
// 인자값 : 접속하려는 Serial Port Number, Serial Port Baudrate
// stat : 현재 접속 가능한 Comport 의 상태 확인

for(i=0;i<COMPORT;i++)
{
        switch(i)
        {
        case  0: stat = autpmc_Open( 0, FPMC_BAUD_115200 );
                 break;
        case  1: stat = autpmc_Open( 1, FPMC_BAUD_115200 );
                 break;
        case  2: stat = autpmc_Open( 2, FPMC_BAUD_115200 );
                 break;
        case  3: stat = autpmc_Open( 3, FPMC_BAUD_115200 );
                 break;
        case  4: stat = autpmc_Open( 4, FPMC_BAUD_115200 );
                 break;
        case  5: stat = autpmc_Open( 5, FPMC_BAUD_115200 );
                 break;
        case  6: stat = autpmc_Open( 6, FPMC_BAUD_115200 );
                 break;
        case  7: stat = autpmc_Open( 7, FPMC_BAUD_115200 );
                 break;
        case  8: stat = autpmc_Open( 8, FPMC_BAUD_115200 );
                 break;
        case  9: stat = autpmc_Open( 9, FPMC_BAUD_115200 );
                 break;
        case 10: stat = autpmc_Open(10, FPMC_BAUD_115200);
                 break;
        case 11: stat = autpmc_Open(11, FPMC_BAUD_115200);
                 break;
        case 12: stat = autpmc_Open(12, FPMC_BAUD_115200);
                 break;
        case 13: stat = autpmc_Open(13, FPMC_BAUD_115200);
                 break;
        case 14: stat = autpmc_Open(14, FPMC_BAUD_115200);
                 break;
        case 15: stat = autpmc_Open(15, FPMC_BAUD_115200);
                 break;
        }

        if (stat == FPMC_OK)
        {
                 printf("MESSAGE : Found and open 'PMC-2HSP(N) (ID=%d)'
ComPort\n", i);
        }
 }

}
```

## 1.2 autpmc_Reset

autpmc_Reset 함수는 PMC-2HSP/2HSN 을 브로드캐스트 기능으로 모션 IC 를 초기화합니다.

### (1) 함수명

```
int autpmc_Reset(
int PortNum,
char nNodeId
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다. 또한, Broadcast (0x80)을 입력하면 브로드캐스트 기능으로 현재 PC 와 연결된 모든 PMC-2HSP/2HSN 에 데이터를 전송합니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM          3

void main()
{

 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);

 Flag = autpmc_Reset (PORTNUM, Node01);

 if(Flag!=FPMC_OK)
 {
          printf("error!\n");
          return;
 }


 autpmc_Close(PORTNUM);
}
```

## 1.3 autpmc_IsCon

autpmc_IsCon 함수는 PMC-2HSP/2HSN 과 통신 접속을 해서 데이터를 정상적으로 주고 받는지 확인합니다.

**(1) 함수명**

```
int autpmc_IsCon(
int PortNum,
char nNodeId,
BOOL *bOn
);
```

**(2) 파라미터**

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- bOn

  명령 성공 시 데이터에 1 이 저장되고, 실패 시 0 이 저장됩니다.

**(3) 리턴값**

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |

**(4) 사용 예**

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM            3

void main()
{
 int Flag=0; //함수의 동작 상태 확인

 BOOL bOn=0; //본체 접속 여부 (접속 ON(1), 접속 OFF(0))

 autpmc_Open(PORTNUM, FPMC_BAUD_115200); //통신 접속 함수


 //PMC-2HSP(N)의 연결 상태 확인 함수
 //리턴값 : 정상적으로 명령어 수행시에는 FPMC_OK 를 리턴한다.
 //인자 : 설정 할 노드 ID, 본체 접속 여부 : 접속 ON(1), 접속 OFF(0)
```

```
 Flag = autpmc_IsCon(PORTNUM, Node01, &bOn);

 printf("%d\n", bOn);

 if (bOn == 1)
 {
        printf("Connection!\n");
 }
 else
 {
        printf("Connection Failed\n");
 }

 if(Flag!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }

 autpmc_Close(PORTNUM);
}
```

## 1.4 autpmc_SetBaudrate

autpmc_SetBaudrate 함수는 PMC-2HSP/2HSN 을 브로드캐스트 기능으로 통신 속도(baudrate)를 변경합니다.

### (1) 함수명

```
int autpmc_SetBaudrate(
int PortNum,
char nNodeId,
int BaudRate
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다. 또한, Broadcast (0x80)을 입력하면 브로드캐스트 기능으로 현재 PC 와 연결된 모든 PMC-2HSP/2HSN 에 데이터를 전송합니다.

- BaudRate

  변경하고자 하는 통신 속도를 입력합니다.

| 구분 | 입력 | 내용 | 상수값 |
|---|---|---|---|
| PMC_BAUDRATE | FPMC_BAUD_9600 | 9,600bps | 9600 |
| | FPMC_BAUD_19200 | 19,200bps | 19200 |
| | FPMC_BAUD_38400 | 38,400bps | 38400 |
| | FPMC_BAUD_57600 | 57,600bps | 57600 |
| | FPMC_BAUD_115200 | 115,200bps | 115200 |

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_BAUDRATE | 2 | 잘못된 Baudrate 을 입력하였습니다. |
| | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM          3
```

```
void main()
{

 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_SetBaudrate(PORTNUM, Node01, FPMC_BAUD_9600);

 if(Flag!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }



 autpmc_Close(PORTNUM);
}
```

## 1.5 autpmc_ClrINCPos

autpmc_ClrINCPos 함수는 PMC-2HSP/2HSN 의 상대 위치를 초기화합니다.

### (1) 함수명

```
int autpmc_ClrINCPos(
int PortNum,
char nNodeId,
char axis
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{

 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);
```

```
 Flag = autpmc_ClrINCPos(PORTNUM, Node01, FPMC_X_Y_AXIS);

 if(Flag!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }


 autpmc_Close(PORTNUM);
}
```

## 1.6 autpmc_Timeout

autpmc_Timeout 함수는 PMC-2HSP/2HSN 의 송/수신 명령어의 대기시간을 설정합니다. Port 의 값을 읽어 올때 아무런 설정도 하지 않으면 지정된 수신 버퍼에 명령어가 모두 들어올 때까지 무한정 기다리게 됩니다. 일정 시간이 지난 후 수신 동작에 대한 대기시간을 설정 할 경우 이 함수를 설정합니다.

### (1) 함수명

```
int autpmc_Timeout(
int PortNum,
int RITimeout,
int RTTimeoutMultiplier,
int RTTimeoutConstant,
int WTTimeoutMultiplier,
int WTTimeoutConstant
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- RITimeout(ReadIntervalTimeout)

  읽기 간격 대기시간은 바이트와 바이트 사이에 도착 시간을 설정하는 것으로 설정 할 시간 안에 다음 바이트가 도착하지 않으면 대기시간이 발생합니다.

  도착하는 2 바이트 사이의 최대 시간 간격을 ms 단위로 설정 할 수 있습니다.

  읽기 동작 시, 첫 번째 바이트가 수신되면 대기시간 간격이 시작되고 만약 1 바이트가 수신된 후에 읽기 간격 대기시간이 지나도 다음 1 바이트가 수신되지 않으면 읽기 동작은 완료됩니다.

  읽기 간격 대기시간을 설정하고 싶지 않으면 0 을 입력합니다.

- RTTimeoutMultiplier(ReadTotalTimeoutMultiplier)

  읽기 전체 대기시간 곱하기 계수는 바이트당 읽기 대기시간값과 상수값을 ms 단위로 지정합니다.

- RTTimeoutConstant(ReadTotalTimeoutConstant)

  읽기 전체 대기시간 상수는 읽기 전체 대기시간 간격을 계산하기 위해 사용되는 상수로 단위는 ms 입니다.

  각각의 읽기 동작 시, 이 값은 읽기 요청된 바이트 수와 ReadTotalTimeoutMultiplier 의 곱에 더해집니다.

  만약 읽을 바이트 수가 n 이라면 읽기 대기시간 값은 아래와 같이 계산합니다.

  읽기 대기시간(ms) = n×ReadTotalTimeoutMultiplier+ReadTotalTimeoutConstant

  단, ReadTotalTimeoutMultiplier 와 ReadTotalTimeoutConstant 를 0 으로 설정하면 읽기 대기시간을 사용하지 않습니다.

- WTTimeoutMultiplier(WriteTotalTimeoutMultiplier)

쓰기 전체 대기시간 곱하기 계수는 쓰기 작업을 위한 전체 대기시간을 계산하기 위해 ms 단위의 곱하기 계수를 지정합니다.

- WTTimeoutConstant(WriteTotalTimeoutConstant)

  쓰기 전체 대기시간 상수는 쓰기 작업을 위한 전체 대기시간을 계산하기 위해 사용되는 상수로 단위는 ms 입니다.

  만약 보낼 바이트 수가 n 이라면 쓰기 대기시간값은 아래와 같이 계산합니다.

  쓰기 대기시간 (ms) = n×WriteTotalTimeoutMultiplier+WriteTotalTimeoutConstant

  단, WriteTotalTimeoutMultiplier 와 WriteTotalTimeoutConstant 를 0 으로 설정하면 쓰기 대기시간을 사용하지 않습니다.

  ※대기시간이 올바르게 설정되지 않았을 경우에는 데이터를 읽기 위해 많은 시간을 소비하거나, 대기시간을 너무 짧게 설정하여 데이터를 읽지 못하는 경우가 있을 수도 있습니다. 이럴 경우 적정한 대기시간을 설정하십시오.

**(3) 리턴값**

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_PORT | 1 | 유효한 Port 가 아닙니다. |

**(4) 사용 예**

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0; // 함수의 동작 상태 확인

 autpmc_Open(PORTNUM, FPMC_BAUD_115200); //통신 접속 함수

 Flag = autpmc_Timeout(PORTNUM, 0, 1, 1, 0, 0);   //송,수신 명령어의 대기시간 변경

 if(Flag!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }

 autpmc_Close(PORTNUM); //통신 연결 해제
}
```

# 2 정지, 종료

## 2.1 autpmc_Close

autpmc_Close 함수는 PMC-2HSP/2HSN 의 통신 연결을 해제합니다.

**(1) 함수명**

int autpmc_Close(PORTNUM);

**(2) 파라미터**

없음.

**(3) 리턴값**

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_PORT | 1 | 유효한 Port 가 아닙니다. |

**(4) 사용 예**

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{

 int Flag=0; // 함수의 동작 상태 확인

 Flag = autpmc_Open(PORTNUM, FPMC_BAUD_115200); //통신 접속 함수


 if(Flag!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }



 autpmc_Close(PORTNUM); //통신 연결 해제
}
```

## 2.2 autpmc_SlowStop

autpmc_SlowStop 함수는 PMC-2HSP/2HSN 을 감속 정지합니다.

### (1) 함수명

```
int autpmc_SlowStop(
int PortNum,
char nNodeId,
char axis
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{

 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);
```

```
 Flag = autpmc_SlowStop(PORTNUM, Node01, FPMC_X_Y_AXIS);

 if(Flag!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }


 autpmc_Close(PORTNUM);
}
```

## 2.3 autpmc_EmgStop

autpmc_EmgStop 함수는 PMC-2HSP/2HSN 을 브로드캐스트 기능으로 긴급 정지합니다.

### (1) 함수명

```
int autpmc_EmgStop(
int PortNum,
char nNodeId
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  Broadcast (0x80)을 입력하면 브로드캐스트 기능으로 현재 PC 와 연결된 모든 PMC-2HSP/2HSN 에 데이터를 전송합니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{

 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_EmgStop(PORTNUM, Broadcast);

 if(Flag!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }


 autpmc_Close(PORTNUM);
}
```

# 3 파라미터 설정

## 3.1 autpmc_GetParaAll

autpmc_GetParaAll 함수는 PMC-2HSP/2HSN 의 저장된 모든 설정값을 가져옵니다.

**(1) 함수명**

```
struct PMC_PARADATA *autpmc_GetParaAll(
int PortNum,
char nNodeId,
char axis,
PMC_PARADATA *pData
);
```

**(2) 파라미터**

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- pData

  모든 파라미터의 정보를 한 번에 가져와서 저장합니다.

| 구조체명 | 변수 형식 | 내용 | 데이터값 |
|---|---|---|---|
| PMC_ PARADATA | int iErrorState | 오류 상태 확인 | 0 : 함수가 정상적으로 명령을 수행하였습니다.<br>3 : 잘못된 노드 ID 를 입력하였습니다.<br>4 : 잘못된 축을 입력하였습니다.<br>5 : 잘못된 데이터를 입력하였습니다. |
| | BOOL bLmtStopMod[2] | 리미트 정지 모드 Enable/Disable<br>(X 축 : bLmtStopMod[0],<br>Y 축 : bLmtStopMod[1]) | 0 : Instant<br>1 : Slow |

| 구조체명 | 변수 형식 | 내용 | 데이터값 |
|---|---|---|---|
| PMC_ PARADATA | BOOL bLmtActLev[2] | 리미트 신호 논리 레벨 Low/High (X 축 : bLmtActLev [0], Y 축 : bLmtActLev [1]) | 0 : Low<br>1 : High |
| | BOOL bSCurve[2] | S 자 가감속의 사용 Enable/Disable (X 축 : bSCurve [0], Y 축 : bSCurve [1]) | 0 : Disable<br>1 : Enable |
| | BOOL bEndPEnable[2] | 드라이브 종료 펄스의 대칭/비대칭 (X 축 : bEndPEnable [0], Y 축 : bEndPEnable [1]) | 0 : Disable<br>1 : Enable |
| | BOOL bDecValue[2] | 사다리꼴 가감속 드라이브의 대칭/비대칭 (X 축 : bDecValue [0], Y 축 : bDecValue [1]) | 0 : Accel<br>1 : Decel |
| | BOOL bSofLmtEnable[2] | 소프트웨어 리미트의 Enable/Disable (X 축 : bSofLmtEnable [0], Y 축 : bSofLmtEnable [1]) | 0 : Enable<br>1 : Disable |
| | BOOL bPowHomStart[2] | 파워 온 원점 복귀 자동 스타트 Enable/Disable (X 축 : bPowHomStart [0], Y 축 : bPowHomStart [1]) | 0 : Disable<br>1 : Enable |
| | BOOL bPowPgmStart[2] | 파워 온 프로그램 자동 스타트 Enable/Disable (X 축 : bPowPgmStart [0], Y 축 : bPowPgmStart [1]) | 0 : Disable<br>1 : Enable |
| | BOOL bInput0Lev[2] | 범용 입력 0 번의 액티브 레벨 Low/High (X 축 : bInput0Lev [0], Y 축 : bInput0Lev [1]) | 0 : Low<br>1 : High |
| | BOOL bInput1Lev[2] | 범용 입력 1 번의 액티브 레벨 Low/High (X 축 : bInput1Lev [0], Y 축 : bInput1Lev [1]) | 0 : Low<br>1 : High |
| | int iPulseType | 펄스 입력 방식 : 1PULSE, 2PULSE | 1 : 1PULSE<br>2 : 2PULSE |
| | int iSpdMul[2] | 속도 배율 (X 축 : iSpdMul [0], Y 축 : iSpdMul [1]) | 1~500 |
| | int iJrkSpd[2] | 가가속도 (X 축 : iJrkSpd [0], Y 축 : iJrkSpd [1]) | 1~65535 |
| | int iAccSpdRate[2] | 가속률(X 축 : iAccSpdRate [0], Y 축 : iAccSpdRate [1]) | 1~8000 |
| | int iDecSpdRate[2] | 감속률 (X 축 : iDecSpdRate [0], Y 축 : iDecSpdRate [1]) | 1~8000 |
| | int iStrSpd[2] | 초기 속도 (X 축 : iStrSpd [0], Y 축 : iStrSpd [1]) | 1~8000 |
| | int iDrvSpd[2] | 구동 속도 (X 축 : iDrvSpd [0], Y 축 : iDrvSpd [1]) | 1~8000 |

| 구조체명 | 변수 형식 | 내용 | 데이터값 |
|---|---|---|---|
| PMC_ PARADATA | int iDrvSpd1Pgm[2] | 프로그램 모드에서 사용하는 구동 속도 1 (X 축 : iDrvSpd1Pgm [0], Y 축 : iDrvSpd1Pgm [1]) | 1~8000 |
| | int iDrvSpd2Pgm[2] | 프로그램 모드에서 사용하는 구동 속도 2 (X 축 : iDrvSpd2Pgm [0], Y 축 : iDrvSpd2Pgm [1]) | 1~8000 |
| | int iDrvSpd3Pgm[2] | 프로그램 모드에서 사용하는 구동 속도 3 (X 축 : iDrvSpd3Pgm [0], Y 축 : iDrvSpd3Pgm [1]) | 1~8000 |
| | int iDrvSpd4Pgm[2] | 프로그램 모드에서 사용하는 구동 속도 4 (X 축 : iDrvSpd4Pgm [0], Y 축 : iDrvSpd4Pgm [1]) | 1~8000 |
| | int iTim1Pgm[2] | 프로그램 모드에서 사용하는 포스트 타이머 1 (X 축 : iTim1Pgm [0], Y 축 : iTim1Pgm [1]) | 1~65535 |
| | int iTim2Pgm[2] | 프로그램 모드에서 사용하는 포스트 타이머 2 (X 축 : iTim2Pgm [0], Y 축 : iTim2Pgm [1]) | 1~65535 |
| | int iTim3Pgm[2] | 프로그램 모드에서 사용하는 포스트 타이머 3 (X 축 : iTim3Pgm [0], Y 축 : iTim3Pgm [1]) | 1~65535 |
| | long lSofLmtP[2] | +방향 소프트웨어 리미트 (X 축 : lSofLmtP [0], Y 축 : lSofLmtP [1]) | -8388608 ~8388607 |
| | long lSofLmtM[2] | -방향 소프트웨어 리미트 (X 축 : lSofLmtM [0], Y 축 : lSofLmtM [1]) | - 8388608 ~ 8388607 |
| | int iEndPWidth[2] | 드라이브 종료 펄스의 폭 (X 축 : iEndPWidth [0], Y 축 : iEndPWidth [1]) | 1~65535 |
| | int iPulSclNum[2] | 펄스 스케일 (X 축 : iPulSclNum [0], Y 축 : iPulSclNum [1]) | 1~65535 |
| | int iPulSclDen[2] | 펄스 스케일 (X 축 : iPulSclDen [0], Y 축 : iPulSclDen [1]) | 1~65535 |
| | BOOL bHomMod1[2] | 원점 복귀 모드의 스텝 1 Enable/Disable (X 축 : bHomMod1 [0], Y 축 : bHomMod1 [1]) | 0 : Disable 1 : Enable |
| | BOOL bHomMod1Dir[2] | 원점 복귀 모드의 스텝 1 탐색 방향 (X 축 : bHomMod1Dir [0], Y 축 : bHomMod1Dir [1]) | 0 : +방향 1 : -방향 |

| 구조체명 | 변수 형식 | 내용 | 데이터값 |
|---|---|---|---|
| PMC_ PARADATA | BOOL bHomMod2[2] | 원점 복귀 모드의 스텝 2 Enable/Disable<br>(X 축 : bHomMod2 [0],<br>Y 축 : bHomMod2 [1]) | 0 : Disable<br>1 : Enable |
| | BOOL bHomMod2Dir[2] | 원점 복귀 모드의 스텝 2 탐색 방향<br>(X 축 : bHomMod2Dir [0],<br>Y 축 : bHomMod2Dir [1]) | 0 : +방향<br>1 : -방향 |
| | BOOL bHomMod3[2] | 원점 복귀 모드의 스텝 3 Enable/Disable<br>(X 축 : bHomMod3 [0],<br>Y 축 : bHomMod3 [1]) | 0 : Disable<br>1 : Enable |
| | BOOL bHomMod3Dir[2] | 원점 복귀 모드의 스텝 3 탐색 방향<br>(X 축 : bHomMod3Dir [0],<br>Y 축 : bHomMod3Dir [1]) | 0 : +방향<br>1 : -방향 |
| | BOOL bHomMod4[2] | 원점 복귀 모드의 스텝 4 Enable/Disable<br>(X 축 : bHomMod4 [0],<br>Y 축 : bHomMod4 [1]) | 0 : Disable<br>1 : Enable |
| | BOOL bHomMod4Dir[2] | 원점 복귀 모드 4 의 방향<br>(X 축 : bHomMod4Dir [0],<br>Y 축 : bHomMod4Dir [1]) | 0 : +방향<br>1 : -방향 |
| | BOOL bHomEndPosClr[2] | 위치 카운터의 초기화 Enable/Disable<br>(X 축 : bHomEndPosClr [0],<br>Y 축 : bHomEndPosClr [1]) | 0 : Disable<br>1 : Enable |
| | BOOL bHomSig0Lev[2] | 원점 근접 신호(STOP 0) 논리 레벨 Low/High<br>(X 축 : bHomSig0Lev [0],<br>Y 축 : bHomSig0Lev [1]) | 0 : Low<br>1 : High |
| | BOOL bHomSig1Lev[2] | 원점 신호(STOP1) 논리 레벨 Low/High<br>(X 축 : bHomSig1Lev [0],<br>Y 축 : bHomSig1Lev [1]) | 0 : Low<br>1 : High |
| | BOOL bHomSig2Lev[2] | 엔코더 Z 상 신호(STOP2) 논리 레벨<br>Low/High<br>(X 축 : bHomSig2Lev [0],<br>Y 축 : bHomSig2Lev [1]) | 0 : Low<br>1 : High |
| | int iHomLowSpd[2] | 저속 원점 복귀 속도<br>(X 축 : iHomLowSpd [0],<br>Y 축 : iHomLowSpd [1]) | 1~8000 |
| | int iHomLowSpd[2] | 저속 원점 복귀 속도<br>(X 축 : iHomLowSpd [0],<br>Y 축 : iHomLowSpd [1]) | 1~8000 |

| 구조체명 | 변수 형식 | 내용 | 데이터값 |
|---|---|---|---|
| PMC_ PARADATA | int iHomHighSpd[2] | 고속 원점 복귀 속도 (X 축 : iHomHighSpd [0], Y 축 : iHomHighSpd [1]) | 1~8000 |
| | long lHomOffset[2] | 원점 복귀 스텝 4 의 고속 오프셋 이동의 이동량 (X 축 : lHomOffset [0], Y 축 : lHomOffset [1]) | 1~8000 |

**(3) 리턴값**

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |

**(4) 사용 예**

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);



 PMC_PARADATA Data;    // 구조체 변수 선언
 PMC_PARADATA *pData = &Data;

 autpmc_GetParaAll(PORTNUM, Node01, FPMC_X_Y_AXIS, pData);

printf("X 축 Limit Stop Mode : %d\nY 축 Limit Stop Mode : %d\n", pData->bLmtStopMod[0],
pData->bLmtStopMod[1]);

 if(pData->iErrorState!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }



 autpmc_Close(PORTNUM);
}
```

## 3.2 autpmc_GetParaOPMAll

autpmc_GetParaOPMAll 함수는 PMC-2HSP/2HSN 의 동작 모드 설정값을 가져옵니다.

### (1) 함수명

```
struct PMC_PARADATA *autpmc_GetParaOPMAll(
int PortNum,
char nNodeId,
char axis,
PMC_PARADATA *pData
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- pData

  동작 모드에 관련된 모든 파라미터의 정보를 한 번에 가져와서 저장합니다.

| 구조체명 | 변수 형식 | 내용 | 데이터값 |
|---|---|---|---|
| PMC_ PARADATA | int iErrorState | 오류 상태 확인 | 0 : 함수가 정상적으로 명령을 수행하였습니다.<br>3 : 잘못된 노드 ID 를 입력하였습니다.<br>4 : 잘못된 축을 입력하였습니다.<br>5 : 잘못된 데이터를 입력하였습니다. |
| | BOOL bLmtStopMod[2] | 리미트 정지 모드 Enable/Disable<br>(X 축 : bLmtStopMod[0],<br>Y 축 : bLmtStopMod[1]) | 0 : Instant<br>1 : Slow |
| | BOOL bLmtActLev[2] | 리미트 신호 논리 레벨 Low/High<br>(X 축 : bLmtActLev [0],<br>Y 축 : bLmtActLev [1]) | 0 : Low<br>1 : High |

| 구조체명 | 변수 형식 | 내용 | 데이터값 |
|---|---|---|---|
| PMC_ PARADATA | BOOL bSCurve[2] | S 자 가감속의 사용 Enable/Disable (X 축 : bSCurve [0], Y 축 : bSCurve [1]) | 0 : Disable 1 : Enable |
| | BOOL bEndPEnable[2] | 드라이브 종료 펄스의 대칭/비대칭 (X 축 : bEndPEnable [0], Y 축 : bEndPEnable [1]) | 0 : Disable 1 : Enable |
| | BOOL bDecValue[2] | 사다리꼴 가감속 드라이브의 대칭/비대칭 (X 축 : bDecValue [0], Y 축 : bDecValue [1]) | 0 : Accel 1 : Decel |
| | BOOL bSofLmtEnable[2] | 소프트웨어 리미트의 Enable/Disable (X 축 : bSofLmtEnable [0], Y 축 : bSofLmtEnable [1]) | 0 : Enable 1 : Disable |
| | BOOL bPowHomStart[2] | 파워 온 원점 복귀 자동 스타트 Enable/Disable (X 축 : bPowHomStart [0], Y 축 : bPowHomStart [1]) | 0 : Disable 1 : Enable |
| | BOOL bPowPgmStart[2] | 파워 온 프로그램 자동 스타트 Enable/Disable (X 축 : bPowPgmStart [0], Y 축 : bPowPgmStart [1]) | 0 : Disable 1 : Enable |
| | BOOL bInput0Lev[2] | 범용 입력 0 번의 액티브 레벨 Low/High (X 축 : bInput0Lev [0], Y 축 : bInput0Lev [1]) | 0 : Low 1 : High |
| | BOOL bInput1Lev[2] | 범용 입력 1 번의 액티브 레벨 Low/High (X 축 : bInput1Lev [0], Y 축 : bInput1Lev [1]) | 0 : Low 1 : High |

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);



 PMC_PARADATA Data; //구조체 변수 선언
 PMC_PARADATA *pData = &Data;


 autpmc_GetParaOPMAll (PORTNUM, Node01, FPMC_X_Y_AXIS, pData);

 printf(" X 축 Limit Stop Mode : %d\n Y 축 Limit Stop Mode : %d\n", pData-
>bLmtStopMod[0], pData->bLmtStopMod[1]);

 if(pData->iErrorState!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }



 autpmc_Close(PORTNUM);
}
```

## 3.3 autpmc_GetParaPMAll

autpmc_GetParaPMAll 함수는 PMC-2HSP/2HSN 의 파라미터 설정값을 가져옵니다.

### (1) 함수명

```
struct PMC_PARADATA *autpmc_GetParaPMAll(
int PortNum,
char nNodeId,
char axis,
PMC_PARADATA *pData
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- pData

  파라미터에 관련된 모든 정보를 한 번에 가져와서 저장합니다.

| 구조체명 | 변수 형식 | 내용 | 데이터값 |
|---|---|---|---|
| PMC_ PARADATA | int iErrorState | 오류 상태 확인 | 0 : 함수가 정상적으로 명령을 수행하였습니다.<br>3 : 잘못된 노드 ID 를 입력하였습니다.<br>4 : 잘못된 축을 입력하였습니다.<br>5 : 잘못된 데이터를 입력하였습니다. |
| | int iPulseType | 펄스 입력 방식 : 1PULSE, 2PULSE | 1 : 1PULSE<br>2 : 2PULSE |
| | int iSpdMul[2] | 속도 배율 (X 축 : iSpdMul [0], Y 축 : iSpdMul [1]) | 1~500 |

| 구조체명 | 변수 형식 | 내용 | 데이터값 |
|---|---|---|---|
| PMC_ PARADATA | int iJrkSpd[2] | 가가속도 (X 축 : iJrkSpd [0], Y 축 : iJrkSpd [1]) | 1~65535 |
| | int iAccSpdRate[2] | 가속률 (X 축 : iAccSpdRate [0], Y 축 : iAccSpdRate [1]) | 1~8000 |
| | int iDecSpdRate[2] | 감속률 (X 축 : iDecSpdRate [0], Y 축 : iDecSpdRate [1]) | 1~8000 |
| | int iStrSpd[2] | 초기 속도 (X 축 : iStrSpd [0], Y 축 : iStrSpd [1]) | 1~8000 |
| | int iDrvSpd[2] | 구동 속도 (X 축 : iDrvSpd [0], Y 축 : iDrvSpd [1]) | 1~8000 |
| | int iDrvSpd1Pgm[2] | 프로그램 모드에서 사용하는 구동 속도 1 (X 축 : iDrvSpd1Pgm [0], Y 축 : iDrvSpd1Pgm [1]) | 1~8000 |
| | int iDrvSpd2Pgm[2] | 프로그램 모드에서 사용하는 구동 속도 2 (X 축 : iDrvSpd2Pgm [0], Y 축 : iDrvSpd2Pgm [1]) | 1~8000 |
| | int iDrvSpd3Pgm[2] | 프로그램 모드에서 사용하는 구동 속도 3 (X 축 : iDrvSpd3Pgm [0], Y 축 : iDrvSpd3Pgm [1]) | 1~8000 |
| | int iDrvSpd4Pgm[2] | 프로그램 모드에서 사용하는 구동 속도 4 (X 축 : iDrvSpd4Pgm [0], Y 축 : iDrvSpd4Pgm [1]) | 1~8000 |
| | int iTim1Pgm[2] | 프로그램 모드에서 사용하는 포스트 타이머 1 (X 축 : iTim1Pgm [0], Y 축 : iTim1Pgm [1]) | 1~65535 |
| | int iTim2Pgm[2] | 프로그램 모드에서 사용하는 포스트 타이머 2 (X 축 : iTim2Pgm [0], Y 축 : iTim2Pgm [1]) | 1~65535 |
| | int iTim3Pgm[2] | 프로그램 모드에서 사용하는 포스트 타이머 3 (X 축 : iTim3Pgm [0], Y 축 : iTim3Pgm [1]) | 1~65535 |
| | long lSofLmtP[2] | +방향 소프트웨어 리미트 (X 축 : lSofLmtP [0], Y 축 : lSofLmtP [1]) | -8388608 ~8388607 |
| | long lSofLmtM[2] | -방향 소프트웨어 리미트 (X 축 : lSofLmtM [0], Y 축 : lSofLmtM [1]) | -8388608 ~8388607 |

| 구조체명 | 변수 형식 | 내용 | 데이터값 |
|---|---|---|---|
| PMC_ PARADATA | int iEndPWidth[2] | 드라이브 종료 펄스의 폭 (X 축 : iEndPWidth [0], Y 축 : iEndPWidth [1]) | 1~65535 |
| | int iPulSclNum[2] | 펄스 스케일 (X 축 : iPulSclNum [0], Y 축 : iPulSclNum [1]) | 1~65535 |
| | int iPulSclDen[2] | 펄스 스케일 (X 축 : iPulSclDen [0], Y 축 : iPulSclDen [1]) | 1~65535 |

**(3) 리턴값**

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |

**(4) 사용 예**

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);



 PMC_PARADATA Data; //구조체 변수 선언
 PMC_PARADATA *pData = &Data;

 autpmc_GetParaPMAll (PORTNUM, Node01, FPMC_X_AXIS, pData);

 printf("PulseType : %d\n", pData->iPulseType);

 if(pData->bErrorState[0]||pData->bErrorState[1]!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }



 autpmc_Close(PORTNUM);
}
```

## 3.4 autpmc_GetParaHSMAll

autpmc_GetParaHSMAll 함수는 PMC-2HSP/2HSN 에 설정된 원점 복귀 모드에 관련된 파라미터를 모두 가져옵니다.

**(1) 함수명**

```
struct PMC_PARADATA *autpmc_GetParaHSMAll(
int PortNum,
char nNodeId,
char axis,
PMC_PARADATA *pData
);
```

**(2) 파라미터**

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- pData

  원점 복귀에 관련된 모든 파라미터의 정보를 한 번에 가져와서 저장합니다.

| 구조체명 | 변수 형식 | 내용 | 데이터값 |
|---|---|---|---|
| PMC_ PARADATA | int iErrorState | 오류 상태 확인 | 0 : 함수가 정상적으로 명령을 수행하였습니다.<br>3 : 잘못된 노드 ID 를 입력하였습니다.<br>4 : 잘못된 축을 입력하였습니다.<br>5 : 잘못된 데이터를 입력하였습니다. |
| | BOOL bHomMod1[2] | 원점 복귀 모드의 스텝 1 Enable/Disable<br>(X 축 : bHomMod1 [0],<br>Y 축 : bHomMod1 [1]) | 0 : Disable<br>1 : Enable |

| 구조체명 | 변수 형식 | 내용 | 데이터값 |
|---|---|---|---|
| PMC_<br>PARADATA | BOOL bHomMod1Dir[2] | 원점 복귀 모드의 스텝 1 탐색 방향<br>(X 축 : bHomMod1Dir [0], Y 축 :<br>bHomMod1Dir [1]) | 0 : +방향<br>1 : -방향 |
| | BOOL bHomMod2[2] | 원점 복귀 모드의 스텝 2<br>Enable/Disable<br>(X 축 : bHomMod2 [0],<br>Y 축 : bHomMod2 [1]) | 0 : Disable<br>1 : Enable |
| | BOOL bHomMod2Dir[2] | 원점 복귀 모드의 스텝 2 탐색 방향<br>(X 축 : bHomMod2Dir [0],<br>Y 축 : bHomMod2Dir [1]) | 0 : +<br>1 : - |
| | BOOL bHomMod3[2] | 원점 복귀 모드의 스텝 3<br>Enable/Disable<br>(X 축 : bHomMod3 [0],<br>Y 축 : bHomMod3 [1]) | 0 : Disable<br>1 : Enable |
| | BOOL bHomMod3Dir[2] | 원점 복귀 모드의 스텝 3 탐색 방향<br>(X 축 : bHomMod3Dir [0],<br>Y 축 : bHomMod3Dir [1]) | 0 : +<br>1 : - |
| | BOOL bHomMod4[2] | 원점 복귀 모드의 스텝 4<br>Enable/Disable<br>(X 축 : bHomMod4 [0],<br>Y 축 : bHomMod4 [1]) | 0 : Disable<br>1 : Enable |
| | BOOL bHomMod4Dir[2] | 원점 복귀 모드의 스텝 4 탐색 방향<br>(X 축 : bHomMod4Dir [0],<br>Y 축 : bHomMod4Dir [1]) | 0 : +<br>1 : - |
| | BOOL bHomEndPosClr[2] | 위치 카운터의 초기화 Enable/Disable<br>(X 축 : bHomEndPosClr [0],<br>Y 축 : bHomEndPosClr [1]) | 0 : Disable<br>1 : Enable |
| | BOOL bHomSig0Lev[2] | 원점 근접 신호(STOP 0) 논리 레벨<br>Low/High<br>(X 축 : bHomSig0Lev [0],<br>Y 축 : bHomSig0Lev [1]) | 0 : Low<br>1 : High |
| | BOOL bHomSig1Lev[2] | 원점 신호(STOP1) 논리 레벨<br>Low/High<br>(X 축 : bHomSig1Lev [0],<br>Y 축 : bHomSig1Lev [1]) | 0 : Low<br>1 : High |
| | BOOL bHomSig2Lev[2] | 엔코더 Z 상 신호(STOP2) 논리 레벨<br>Low/High<br>(X 축 : bHomSig2Lev [0],<br>Y 축 : bHomSig2Lev [1]) | 0 : Low<br>1 : High |

| 구조체명 | 변수 형식 | 내용 | 데이터값 |
|---|---|---|---|
| PMC_ PARADATA | int iHomLowSpd[2] | 저속 원점 복귀 속도 (X 축 : iHomLowSpd [0], Y 축 : iHomLowSpd [1]) | 1~8000 |
| | int iHomHighSpd[2] | 고속 원점 복귀 속도 (X 축 : iHomHighSpd [0], Y 축 : iHomHighSpd [1]) | 1~8000 |
| | long lHomOffset[2] | 원점 복귀 스텝 4 의 고속 오프셋 이동의 이동량 (X 축 : lHomOffset [0], Y 축 : lHomOffset [1]) | 1~8000 |

**(3) 리턴값**

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |

**(4) 사용 예**

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 PMC_PARADATA Data; //구조체 변수 선언
 PMC_PARADATA *pData = &Data;


 autpmc_GetParaHSMAll (PORTNUM, Node01, FPMC_X_Y_AXIS, pData);

 printf(" X 축 Step 1 Enable : %d\n Y 축 Step 1 Enable : %d\n", pData->bHomMod1[0],
pData->bHomMod1[1]);

 if(pData->bErrorState[0]||pData->bErrorState[1]!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }
```

```
 autpmc_Close(PORTNUM);
}
```

## 3.5 autpmc_GetLmtStopMod

autpmc_GetLmtStopMod 함수는 PMC-2HSP/2HSN 에 설정된 리미트 정지 모드를 가져옵니다.

### (1) 함수명

```
int autpmc_GetLmtStopMod(
int PortNum,
char nNodeId,
char axis,
BOOL *bStopType
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- bStopType

  즉시 정지로 설정되어 있으면 FPMC_INSTANTSTOP(0)을 감속 정지로 설정되어 있으면 FPMC_SLOWSTOP(1)을 가져옵니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
```

```
{
 int Flag=0;

 BOOL StopType=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_GetLmtStopMod (PORTNUM, Node01, FPMC_X_AXIS, &StopType);

 printf("X 축  StopType : %d\n", StopType);

 if(Flag!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }


 autpmc_Close(PORTNUM);
}
```

## 3.6 autpmc_GetLmtActLev

autpmc_GetLmtActLev 함수는 PMC-2HSP/2HSN 에 설정된 리미트 신호 논리 레벨을 가져옵니다.

### (1) 함수명

```
int autpmc_GetLmtActLev(
int PortNum,
char nNodeId,
char axis,
BOOL *bLevel
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- bLevel

  설정된 리미트 입력 신호 논리 레벨이 High 일 때는 FPMC_HIGH(1)을 Low 일 때는 FPMC_LOW(0)을 가져옵니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
```

```
{
 int Flag=0;

 BOOL bLevel=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_GetLmtActLev (PORTNUM, Node01, FPMC_X_AXIS, &bLevel);

 printf("X 축  bLevel : %d\n", bLevel);

 if(Flag!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }


 autpmc_Close(PORTNUM);
}
```

## 3.7 autpmc_GetSCurve

autpmc_GetSCurve 함수는 PMC-2HSP/2HSN 에 설정된 S 자 가감속의 사용 여부를 가져옵니다.

### (1) 함수명

```
int autpmc_GetSCurve(
int PortNum,
char nNodeId,
char axis,
BOOL *bEnable
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- bEnable

  S 자 가감속이 설정되어 있으면 FPMC_ENABLE(1)을 설정되어 있지 않으면 FPMC_DISABLE(0)을 가져옵니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
```

```
{
 int Flag=0;

 BOOL bEnable=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_GetSCurve (PORTNUM, Node01, FPMC_X_AXIS, &bEnable);

 printf("X 축  bEnable : %d\n", bEnable);

 if(Flag!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }


 autpmc_Close(PORTNUM);
}
```

## 3.8 autpmc_GetEndPEnable

autpmc_GetEndPEnable 함수는 PMC-2HSP/2HSN 에 설정된 드라이브 종료 펄스의 사용 여부를 가져옵니다.

### (1) 함수명

```
int autpmc_GetEndPEnable(
int PortNum,
char nNodeId,
char axis,
BOOL *bEnable
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- bEnable

  드라이브 종료 펄스가 설정되어 있으면 FPMC_ENABLE(1)을 설정되어 있지 않으면 FPMC_DISABLE(0)을 가져옵니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3
```

```
void main()
{
 int Flag=0;

 BOOL bEnable=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_GetEndPEnable (PORTNUM, Node01, FPMC_X_AXIS, &bEnable);

 printf("X 축  bEnable : %d\n", bEnable);

 if(Flag!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }


 autpmc_Close(PORTNUM);
}
```

## 3.9 autpmc_GetDecValue

autpmc_GetDecValue 함수는 PMC-2HSP/2HSN 에서 사다리꼴 가감속 드라이브의 대칭/비대칭 여부를 가져옵니다.

### (1) 함수명

```
int autpmc_GetDecValue(
int PortNum,
char nNodeId,
char axis,
BOOL *bEnable
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- bEnable

  가속도가 설정되어 있으면 FPMC_ACCEL(0)을 감속도가 설정되어 있으면 FPMC_DECEL(1)을 가져옵니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
```

```
{
 int Flag=0;

 BOOL bDec=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_GetDecValue (PORTNUM, Node01, FPMC_X_AXIS, &bDec);

 printf("X 축  bDec : %d\n", bDec);

 if(Flag!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }


 autpmc_Close(PORTNUM);
}
```

## 3.10 autpmc_GetSofLmtEnable

autpmc_GetSofLmtEnable 함수는 PMC-2HSP/2HSN 에서 소프트웨어 리미트의 사용 여부를 가져옵니다.

### (1) 함수명

```
int autpmc_GetSofLmtEnable(
int PortNum,
char nNodeId,
char axis,
BOOL *bEnable
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- bEnable

  소프트웨어 리미트가 설정되어 있으면 FPMC_ENABLE(0)을 설정되어 있지 않으면 FPMC_DISABLE(1)을 가져옵니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
```

```
int Flag=0;

BOOL bEnable=0;

autpmc_Open(PORTNUM, FPMC_BAUD_115200);


Flag = autpmc_GetSofLmtEnable (PORTNUM, Node01, FPMC_X_AXIS, &bEnable);

printf("X 축  bEnable : %d\n", bEnable);

if(Flag!=FPMC_OK)
{
        printf("error!\n");
        return;
}


autpmc_Close(PORTNUM);
}
```

## 3.11 autpmc_GetPowHomStart

autpmc_GetPowHomStart 함수는 PMC-2HSP/2HSN 에서 파워 온 원점 복귀 자동 스타트의 사용 여부를 가져옵니다.

### (1) 함수명

```
int autpmc_GetPowHomStart(
int PortNum,
char nNodeId,
char axis,
BOOL *bEnable
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- bEnable

  파워 온 원점 복귀 자동 스타트가 설정되어 있다면 FPMC_ENABLE(1)을 설정되어 있지 않다면 FPMC_DISABLE(0)을 가져옵니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
```

```
int Flag=0;

BOOL bEnable=0;

autpmc_Open(PORTNUM, FPMC_BAUD_115200);


Flag = autpmc_GetPowHomStart (PORTNUM, Node01, FPMC_X_AXIS, &bEnable);

printf("X 축  bEnable : %d\n", bEnable);

if(Flag!=FPMC_OK)
{
        printf("error!\n");
        return;
}


autpmc_Close(PORTNUM);
}
```

## 3.12 autpmc_GetPowPgmStart

autpmc_GetPowPgmStart 함수는 PMC-2HSP/2HSN 에서 파워 온 프로그램 자동 스타트의 사용 여부를 가져옵니다.

### (1) 함수명

```
int autpmc_GetPowPgmStart(
int PortNum,
char nNodeId,
char axis,
BOOL *bEnable
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- bEnable

  파워 온 프로그램 자동 스타트가 설정되어 있다면 FPMC_ENABLE(1)을 설정되어 있지 않다면 FPMC_DISABLE(0)을 가져옵니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
```

```
int Flag=0;

BOOL bEnable=0;

autpmc_Open(PORTNUM, FPMC_BAUD_115200);


Flag = autpmc_GetPowPgmStart (PORTNUM, Node01, FPMC_X_AXIS, &bEnable);

printf("X 축  bEnable : %d\n", bEnable);

if(Flag!=FPMC_OK)
{
        printf("error!\n");
        return;
}


autpmc_Close(PORTNUM);
}
```

## 3.13 autpmc_GetInputLev

autpmc_GetInputLev 함수는 PMC-2HSP/2HSN 에서 범용 입력 0, 1 번의 액티브 레벨을 가져옵니다.

### (1) 함수명

```
int autpmc_GetInputLev(
int PortNum,
char nNodeId,
char axis,
BOOL bInPort,
BOOL *bActLev
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- bInPort

  범용 입력 0 번을 선택하려면 FPMC_INPORT0(0)을 입력하고 1 번을 선택하려면 FPMC_INPORT1(1)을 입력합니다.

- bActLev

  액티브 레벨이 Low 로 설정되어 있다면 FPMC_LOW(0)을 High 로 설정되어 있다면 FPMC_HIGH(1)을 가져옵니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력했습니다. |

**(4) 사용 예**

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;

 BOOL bActLev=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_GetInputLev (PORTNUM, Node01, FPMC_X_AXIS, 0, &bActLev);

 printf("X 축  bActLev : %d\n", bActLev);

 if(Flag!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }


 autpmc_Close(PORTNUM);
}
```

## 3.14 autpmc_GetPulseType

autpmc_GetPulseType 함수는 PMC-2HSP/2HSN 에서 펄스 입력 방식(1PULSE/2PULSE)을 가져옵니다.

### (1) 함수명

```
int autpmc_GetPulseType(
int PortNum,
char nNodeId,
int *iPulseType
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- iPulseType

  펄스 입력 방식이 1PULSE 입력 방식일 경우에는 FPMC_1PULSETYPE(1)을 2PULSE 입력 방식이 경우에는 FPMC_2PULSETYPE(2)를 가져옵니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;

 int iPulseType=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_GetPulseType (PORTNUM, Node01, &iPulseType);

 printf("X 축  iPulseType : %d\n", iPulseType);

 if(Flag!=FPMC_OK)
```

```
{
        printf("error!\n");
        return;
}


 autpmc_Close(PORTNUM);
}
```

## 3.15 autpmc_GetSpdMul

autpmc_GetSpdMul 함수는 PMC-2HSP/2HSN 에서 속도 배율을 가져옵니다.

### (1) 함수명

```
int autpmc_GetSpdMul(
int PortNum,
char nNodeId,
char axis,
int *iSpdMul
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- iSpdMul

  설정된 속도 배율을 가져옵니다. (1~500)

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |

### (4) 사용 예

```
include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;

 int iSpdMul=0;
```

```
autpmc_Open(PORTNUM, FPMC_BAUD_115200);


Flag = autpmc_GetSpdMul (PORTNUM, Node01, FPMC_X_AXIS, &iSpdMul);

printf("X 축  iSpdMul : %d\n", iSpdMul);

if(Flag!=FPMC_OK)
{
        printf("error!\n");
        return;
}


autpmc_Close(PORTNUM);
}
```

## 3.16 autpmc_GetJrkSpd

autpmc_GetJrkSpd 함수는 PMC-2HSP/2HSN 에서 가가속도를 가져옵니다.

### (1) 함수명

```
int autpmc_GetJrkSpd(
int PortNum,
char nNodeId,
char axis,
int *iJrkSpd
);
```

### (2) 파라미터

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- iJrkSpd

  설정된 가가속도를 가져옵니다. (1~65,535)

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;

 int iJrkSpd=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);
```

```
 Flag = autpmc_GetJrkSpd (PORTNUM, Node01, FPMC_X_AXIS, &iJrkSpd);

 printf("X 축  iJrkSpd : %d\n", iJrkSpd);

 if(Flag!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }


 autpmc_Close(PORTNUM);
}
```

## 3.17 autpmc_GetAccSpdRate

autpmc_GetAccSpdRate 함수는 PMC-2HSP/2HSN 에서 가속률을 가져옵니다.

### (1) 함수명

```
int autpmc_GetAccSpdRate(
int PortNum,
char nNodeId,
char axis,
int *iAccSpdRate
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- iAccSpdRate

  설정된 가속률을 가져옵니다. (1~8,000)

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;

 int iAccSpdRate=0;
```

```
autpmc_Open(PORTNUM, FPMC_BAUD_115200);


Flag = autpmc_GetAccSpdRate (PORTNUM, Node01, FPMC_X_AXIS, &iAccSpdRate);

printf("X 축  iAccSpdRate : %d\n", iAccSpdRate);

if(Flag!=FPMC_OK)
{
        printf("error!\n");
        return;
}


autpmc_Close(PORTNUM);
}
```

## 3.18 autpmc_GetDecSpdRate

autpmc_GetDecSpdRate 함수는 PMC-2HSP/2HSN 에서 가속률을 가져옵니다.

### (1) 함수명

```
int autpmc_SetDecSpdRate(
int PortNum,
char nNodeId,
char axis,
int *iDecSpdRate
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- iDecSpdRate

  설정된 가속률을 가져옵니다. (1~8,000)

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;
```

```
int iDecSpdRate=0;

autpmc_Open(PORTNUM, FPMC_BAUD_115200);


Flag = autpmc_GetDecSpdRate (PORTNUM, Node01, FPMC_X_AXIS, &iDecSpdRate);

printf("X 축  iDecSpdRate : %d\n", iDecSpdRate);

if(Flag!=FPMC_OK)
{
        printf("error!\n");
        return;
}


autpmc_Close(PORTNUM);
}
```

## 3.19 autpmc_GetStrSpd

autpmc_GetStrSpd 함수는 PMC-2HSP/2HSN 에서 초기 속도를 가져옵니다.

### (1) 함수명

```
int autpmc_GetStrSpd(
int PortNum,
char nNodeId,
char axis,
int *iStrSpd
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- iStrSpd

  설정된 초기 속도를 가져옵니다. (1~8,000)

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;
```

```
int iStrSpd=0;

autpmc_Open(PORTNUM, FPMC_BAUD_115200);


Flag = autpmc_GetStrSpd (PORTNUM, Node01, FPMC_X_AXIS, &iStrSpd);

printf("X 축  iStrSpd : %d\n", iStrSpd);

if(Flag!=FPMC_OK)
{
        printf("error!\n");
        return;
}


autpmc_Close(PORTNUM);
}
```

## 3.20 autpmc_ GetCurDrvSpd

autpmc_GetDrvSpd 함수는 PMC-2HSP/2HSN 에서 현재 기동중인 구동 속도를 가져옵니다.

### (1) 함수명

```
int autpmc_GetCurDrvSpd(
int PortNum,
char nNodeId,
char axis,
int *iDrvSpd
);
```

### (2) 파라미터

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- iDrvSpd

  설정된 구동 속도를 가져옵니다. (1~8,000)

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;

 int iDrvSpd=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);
```

```
 Flag = autpmc_GetCurDrvSpd (PORTNUM, Node01, FPMC_X_AXIS, &iDrvSpd);

 printf("X 축  iDrvSpd : %d\n", iDrvSpd);

 if(Flag!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }


 autpmc_Close(PORTNUM);
}
```

## 3.21 autpmc_GetDrvSpdPgm

autpmc_GetDrvSpdPgm 함수는 PMC-2HSP/2HSN 에서 구동 속도를 가져오는 함수로서 프로그램 모드에서 사용됩니다.

### (1) 함수명

```
int autpmc_GetDrvSpdPgm(
int PortNum,
char nNodeId,
char axis,
int nDrvIndex,
int *iDrvSpd
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- nDrvIndex

  드라이브 속도 인덱스를 입력합니다. 이때 유효한 값은 1~4 입니다.

- iDrvSpd

  설정된 구동 속도를 가져옵니다. (1~8,000)

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력했습니다. |

**(4) 사용 예**

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;

 int iDrvSpd=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_GetDrvSpdPgm (PORTNUM, Node01, FPMC_X_AXIS, 1, &iDrvSpd);

 printf("X 축  iDrvSpd : %d\n", iDrvSpd);

 if(Flag!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }


 autpmc_Close(PORTNUM);
}
```

## 3.22 autpmc_GetTimPgm

autpmc_GetTimPgm 함수는 PMC-2HSP/2HSN 에서 포스트 타이머를 가져오는 함수로서 프로그램 모드에서 사용됩니다.

### (1) 함수명

```
int autpmc_GetTimPgm(
int PortNum,
char nNodeId,
char axis,
int nIndex,
int *iPostTim
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- nIndex

  포스트 타이머 인덱스를 입력합니다. 이때 유효한 값은 1~3 입니다.

- iPostTim

  설정된 포스트 타이머를 가져옵니다. (1~8,000)

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력하였습니다. |

**(4) 사용 예**

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;

 int iPostTim=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);



 Flag = autpmc_GetTimPgm (PORTNUM, Node01, FPMC_X_AXIS, 1, &iPostTim);

 printf("X 축  iPostTim : %d\n", iPostTim);

 if(Flag!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }



 autpmc_Close(PORTNUM);
}
```

## 3.23 autpmc_GetSofLmt

autpmc_GetSofLmt 함수는 PMC-2HSP/2HSN 에서 소프트웨어 리미트를 가져옵니다.

(펄스 스케일값에 비례합니다.)

### (1) 함수명

```
int autpmc_GetSofLmt(
int PortNum,
char nNodeId,
char axis,
int iDirection,
long *lSoftLmt
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- iDirection

  '+' 방향의 리미트를 설정하려면 FPMC_PLUS(0)을 '-' 방향의 리미트를 선택하려면 FPMC_MINUS(1)을 입력합니다.

- lSoftLmt

  설정된 소프트웨어 리미트를 가져옵니다. (-8,388,608~8,388,607)

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력했습니다. |

**(4) 사용 예**

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;

 long lSoftLmt=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);



 Flag = autpmc_GetSofLmt (PORTNUM, Node01, FPMC_X_AXIS, 0, &lSoftLmt);

 printf("X 축  lSoftLmt : %ld\n", lSoftLmt);

 if(Flag!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }



 autpmc_Close(PORTNUM);
}
```

## 3.24 autpmc_GetEndPWidth

autpmc_GetEndPWidth 함수는 PMC-2HSP/2HSN 에서 드라이브 종료 펄스의 폭을 가져옵니다.

### (1) 함수명

```
int autpmc_GetEndPWidth(
int PortNum,
char nNodeId,
char axis,
int *iEndPWidth
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- iEndPWidth

  설정된 드라이브 종료 펄스의 폭을 가져옵니다. (1~65,535)

- 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |

### (3) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;
```

```
int iEndPWidth=0;

autpmc_Open(PORTNUM, FPMC_BAUD_115200);


Flag = autpmc_GetEndPWidth (PORTNUM, Node01, FPMC_X_AXIS, &iEndPWidth);

printf("X 축  iEndPWidth : %d\n", iEndPWidth);

if(Flag!=FPMC_OK)
{
        printf("error!\n");
        return;
}


autpmc_Close(PORTNUM);
}
```

## 3.25 autpmc_GetPulSclNum

autpmc_GetPulSclNum 함수는 PMC-2HSP/2HSN 에서 펄스 스케일의 분자값을 가져옵니다.

### (1) 함수명

```
int autpmc_GetPulSclNum(
int PortNum,
char nNodeId,
char axis,
int *iPulScl
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- iPulScl

  설정된 펄스 스케일의 분자값을 가져옵니다. (1~65,535)

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;

 int iPulScl=0;
```

```
autpmc_Open(PORTNUM, FPMC_BAUD_115200);


Flag = autpmc_GetPulSclNum (PORTNUM, Node01, FPMC_X_AXIS, &iPulScl);

printf("X 축  iPulScl : %d\n", iPulScl);

if(Flag!=FPMC_OK)
{
        printf("error!\n");
        return;
}


autpmc_Close(PORTNUM);
}
```

## 3.26 autpmc_GetPulSclDen

autpmc_GetPulSclDen 함수는 PMC-2HSP/2HSN 에서 펄스 스케일의 분모값을 가져옵니다.

### (1) 함수명

```
int autpmc_GetPulSclDen(
int PortNum,
char nNodeId,
char axis,
int *iPulScl
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- iPulScl

  설정된 펄스 스케일의 분모값을 가져옵니다. (1~65,535)

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;

 int iPulScl=0;
```

```
autpmc_Open(PORTNUM, FPMC_BAUD_115200);


Flag = autpmc_GetPulSclDen (PORTNUM, Node01, FPMC_X_AXIS, &iPulScl);

printf("X 축  iPulScl : %d\n", iPulScl);

if(Flag!=FPMC_OK)
{
        printf("error!\n");
        return;
}


autpmc_Close(PORTNUM);
}
```

## 3.27 autpmc_GetHomMod

autpmc_GetHomMod 함수는 PMC-2HSP/2HSN 에서 원점 복귀 모드를 가져옵니다.

### (1) 함수명

```
struct HOMMOD *autpmc_GetHomMod(
int PortNum,
char nNodeId,
char axis,
int nStepNo,
HOMMOD *pMode
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- nStepNo

  설정 할 스텝 번호를 입력합니다. 이때 유효한 값은 1~4 입니다.

- pMode

  설정된 원점 복귀 모드의 ENABLE 여부와 방향을 가져옵니다.

| 구조체명 | 변수 형식 | 내용 | 데이터값 |
|---|---|---|---|
| HOMMODE | BOOL bEnable[2]<br>bEable[0] : X 축<br>bEable[1] : Y 축 | 원점복귀 사용 | FPMC_ENABLE(1)/<br>FPMC_DISABLE(0) |
| | BOOL bDirection[2]<br>bDirection [0] : X 축<br>bDirection [1] : Y 축 | 원점복귀 검색 방향 | FPMC_PLUS(0)/<br>FPMC_MINUS(1) |

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력했습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);



 HOMMOD Mode; //구조체 변수 선언
 HOMMOD *pMode = &Mode;


 autpmc_GetHomMod (PORTNUM, Node01, FPMC_X_Y_AXIS, 1, pMode);

 printf(" X 축 bEnable : %d\n Y 축 bEnable : %d\n", pMode->bEnable[0], pMode-
>bEnable[1]);

 if(pMode->bErrorState[0]||pMode->bErrorState[1]!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }



 autpmc_Close(PORTNUM);
}
```

## 3.28 autpmc_GetHomEndPosClr

autpmc_GetHomEndPosClr 함수는 PMC-2HSP/2HSN 에서 원점 복귀 종료 시 위치 카운터의 사용 여부를 가져옵니다.

### (1) 함수명

```
int autpmc_GetHomEndPosClr(
int PortNum,
char nNodeId,
char axis,
BOOL *bClear
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- bClear

  위치 카운터 초기화가 사용되고 있으면 FPMC_ENABLE(1)을 사용하지 않고 있으면 FPMC_DISABLE(0)을 가져옵니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
```

```
{
 int Flag=0;

 BOOL bClear=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_GetHomEndPosClr (PORTNUM, Node01, FPMC_X_AXIS, &bClear);

 printf("X 축  bClear : %d\n", bClear);

 if(Flag!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }


 autpmc_Close(PORTNUM);
}
```

## 3.29 autpmc_GetHomSigLev

autpmc_GetHomSigLev 함수는 PMC-2HSP/2HSN 에서 원점 신호의 논리 레벨을 가져옵니다.

### (1) 함수명

```
int autpmc_GetHomSigLev(
int PortNum,
char nNodeId,
char axis,
int nHomSigNo,
BOOL *bLevel
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- nHomSigNo

  원점 근접 신호를 선택하려면 FPMC_HSTOP0(0)을 원점 신호를 선택하려면 FPMC_HSTOP1(1)을 엔코더 Z 상 신호를 선택하려면 FPMC_HSTOP2(2)를 입력합니다.

- bLevel

  원점 신호의 논리 레벨이 Low 로 설정되어 있으면 FPMC_LOW(0)을 High 로 설정되어 있으면 FPMC_HIGH(1)을 가져옵니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력했습니다. |

**(4) 사용 예**

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;

 BOOL bLevel=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);



 Flag = autpmc_GetHomSigLev (PORTNUM, Node01, FPMC_X_AXIS, 0, &bLevel);

 printf("X 축  bLevel : %d\n", bLevel);

 if(Flag!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }



 autpmc_Close(PORTNUM);
}
```

## 3.30 autpmc_GetHomSpd

autpmc_GetHomSpd 함수는 PMC-2HSP/2HSN 에서 원점 복귀 속도를 가져옵니다.

### (1) 함수명

```
int autpmc_GetHomSpd(
int PortNum,
char nNodeId,
char axis,
BOOL bSpd,
int *iSpd
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- bSpd

  저속 원점 복귀 속도를 가져올려면 FPMC_LOW(0)을 고속 원점 복귀 속도를 가져올려면 FPMC_HIGH(1)을 설정합니다.

- iSpd

  설정된 원점 복귀 속도를 가져옵니다. (1~8,000)

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력했습니다. |

**(4) 사용 예**

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;

 int iSpd=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);



 Flag = autpmc_GetHomSpd (PORTNUM, Node01, FPMC_X_AXIS, 0, &iSpd);

 printf("X 축  Low HomeSpeed : %d\n", iSpd);

 if(Flag!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }



 autpmc_Close(PORTNUM);
}
```

## 3.31 autpmc_GetHomOffset

autpmc_GetHomOffset 함수는 PMC-2HSP/2HSN 에서 원점 복귀 스텝 4 의 고속 오프셋 이동의 이동량을 가져옵니다.(펄스 스케일값에 비례됩니다.)

### (1) 함수명

```
int autpmc_GetHomOffset(
int PortNum,
char nNodeId,
char axis,
long *lOffset
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- lOffset

  설정된 원점 복귀 스텝 4 의 고속 오프셋 이동량을 가져옵니다. (0~8,388,607)

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;
```

```
long lOffset=0;

autpmc_Open(PORTNUM, FPMC_BAUD_115200);


Flag = autpmc_GetHomOffset (PORTNUM, Node01, FPMC_X_AXIS, &lOffset);

printf("X 축  HomeSearchOffset : %ld\n", lOffset);

if(Flag!=FPMC_OK)
{
        printf("error!\n");
        return;
}


autpmc_Close(PORTNUM);
}
```

## 3.32 autpmc_SetLmtStopMod

autpmc_SetLmtStopMod 함수는 PMC-2HSP/2HSN 의 리미트 정지 모드를 설정합니다.

### (1) 함수명

```
int autpmc_SetLmtStopMod(
int PortNum,
char nNodeId,
char axis,
BOOL bInstant
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- bInstant

  Instant 모드일 때는 FPMC_INSTANT(0)을 Slow 모드일 때는 FPMC_SLOW(1)을 입력합니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력했습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3
```

```
void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_SetLmtStopMod(PORTNUM, Node01, FPMC_X_Y_AXIS,          0);

 if(Flag!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }


 autpmc_Close(PORTNUM);
}
```

## 3.33 autpmc_SetLmtActLev

autpmc_SetLmtActLev 함수는 PMC-2HSP/2HSN 의 리미트 신호 논리 레벨를 활성화합니다.

### (1) 함수명

```
int autpmc_SetLmtActLev(
int PortNum,
char nNodeId,
char axis,
BOOL bLmtActLev
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- bLmtActLev

  리미트 입력 신호 논리 레벨을 High 일 때 활성화시킬 경우에는 FPMC_HIGH(1)을 Low 일 때 활성화시킬 경우에는 FPMC_LOW(0)을 입력합니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력했습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3
```

```
void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_SetLmtActLev (PORTNUM, Node01, FPMC_X_Y_AXIS, 0);

 if(Flag!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }


 autpmc_Close(PORTNUM);
}
```

## 3.34 autpmc_SetSCurve

autpmc_SetSCurve 함수는 PMC-2HSP/2HSN 의 S 자 가감속의 사용 여부를 설정합니다.

### (1) 함수명

```
int autpmc_SetSCurve(
int PortNum,
char nNodeId,
char axis,
BOOL bEnable
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- bEnable

  S 자 가감속을 사용하려면 FPMC_ENABLE(1)을 사용하지 않으려면 FPMC_DISABLE(0)을 입력합니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력했습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3
```

```
void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_SetSCurve (PORTNUM, Node01, FPMC_X_Y_AXIS,      0);

 if(Flag!=FPMC_OK)
 {
              printf("error!\n");
              return;
 }


 autpmc_Close(PORTNUM);
}
```

## 3.35 autpmc_SetEndPEnable

autpmc_SetEndPEnable 함수는 PMC-2HSP/2HSN 의 드라이브 종료 펄스의 사용 여부를 설정합니다.

**(1) 함수명**

```
int autpmc_SetEndPEnable(
int PortNum,
char nNodeId,
char axis,
BOOL bEnable
);
```

**(2) 파라미터**

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

**(3)** bEnable

드라이브 종료 펄스를 사용하려면 FPMC_ENABLE(1)을 사용하지 않으려면 FPMC_DISABLE(0)을 입력합니다.

**(4) 리턴값**

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력했습니다. |

**(5) 사용 예**

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"
```

```
#define PORTNUM 3

void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_SetEndPEnable (PORTNUM, Node01, FPMC_X_Y_AXIS, 0);

 if(Flag!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }


 autpmc_Close(PORTNUM);
}
```

## 3.36 autpmc_SetDecValue

autpmc_SetDecValue 함수는 PMC-2HSP/2HSN 의 사다리꼴 가감속 드라이브의 대칭/비대칭을 설정합니다.

### (1) 함수명

```
int autpmc_SetDecValue(
int PortNum,
char nNodeId,
char axis,
BOOL bDec
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- bDec

  가속도를 사용할 시 FPMC_ACCEL(0)을 감속도를 사용할 시 FPMC_DECEL(1)을 입력합니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력했습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3
```

```
void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_SetDecValue (PORTNUM, Node01, FPMC_X_Y_AXIS, 0);

 if(Flag!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }


 autpmc_Close(PORTNUM);
}
```

## 3.37 autpmc_SetSofLmtEnable

autpmc_SetSofLmtEnable 함수는 PMC-2HSP/2HSN 의 소프트웨어 리미트의 사용 여부를 설정합니다.

### (1) 함수명

```
int autpmc_SetSofLmtEnable(
int PortNum,
char nNodeId,
char axis,
BOOL bEnable
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- bEnable

  소프트웨어 리미트를 사용하려면 FPMC_ENABLE(0)을 사용하지 않으려면 FPMC_DISABLE(1)을 입력합니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력했습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3
```

```
void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_SetSofLmtEnable (PORTNUM, Node01, FPMC_X_Y_AXIS, 0);

 if(Flag!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }


 autpmc_Close(PORTNUM);
}
```

## 3.38 autpmc_SetPowHomStart

autpmc_SetPowHomStart 함수는 PMC-2HSP/2HSN 의 파워 온 원점 복귀 자동 스타트의 사용 여부를 설정합니다.

### (1) 함수명

```
int autpmc_SetPowHomStart(
int PortNum,
char nNodeId,
char axis,
BOOL bEnable
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- bEnable

  파워 온 원점 복귀 자동 스타트를 사용하려면 FPMC_ENABLE(1)을 사용하지 않으려면 FPMC_DISABLE(0)을 입력합니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력했습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3
```

```
void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_SetPowHomStart (PORTNUM, Node01, FPMC_X_Y_AXIS, 0);

 if(Flag!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }


 autpmc_Close(PORTNUM);
}
```

## 3.39 autpmc_SetPowPgmStart

autpmc_SetPowPgmStart 함수는 PMC-2HSP/2HSN 의 파워 온 프로그램 자동 스타트의 사용 여부를 설정합니다.

### (1) 함수명

```
int autpmc_SetPowPgmStart(
int PortNum,
char nNodeId,
char axis,
BOOL bEnable
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- bEnable

  파워 온 프로그램 자동 스타트를 사용하려면 FPMC_ENABLE(1)을 사용하지 않으려면 FPMC_DISABLE(0)을 입력합니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력했습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3
```

```
void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_SetPowPgmStart (PORTNUM, Node01, FPMC_X_Y_AXIS, 0);

 if(Flag!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }


 autpmc_Close(PORTNUM);
}
```

## 3.40 autpmc_SetInputLev

autpmc_SetInputLev 함수는 PMC-2HSP/2HSN 의 범용 입력 0, 1 번의 액티브 레벨을 설정합니다.

### (1) 함수명

```
int autpmc_SetInputLev(
int PortNum,
char nNodeId,
char axis,
BOOL bInPort,
BOOL bActLev
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- bInPort

  범용 입력 0 번을 설정 할려면 FPMC_INPORT(0)을 범용 입력 1 번을 설정 할려면 FPMC_INPORT1(1)을 설정합니다.

- bActLev

  액티브 레벨을 Low 로 설정하려면 FPMC_LOW(0)을 High 로 설정하려면 FPMC_HIGH(1)을 설정합니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력했습니다. |

**(4) 사용 예**

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_SetInputLev (PORTNUM, Node01, FPMC_X_Y_AXIS, 0, 0);

 if(Flag!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }


 autpmc_Close(PORTNUM);
}
```

## 3.41 autpmc_SetPulseType

autpmc_SetPulseType 함수는 PMC-2HSP/2HSN 의 펄스 입력 방식(1PULSE/2PULSE)을 선택합니다.

### (1) 함수명

```
int autpmc_SetPulseType(
int PortNum,
char nNodeId,
int iPulseType
);
```

### (2) 파라미터

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- iPulseType

  1PULSE 입력 방식일 경우에는 FPMC_SETPULSE1(1)을 2PULSE 입력 방식일 경우에는 FPMC_SETPULSE2(2)를 입력합니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력했습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_SetPulseType (PORTNUM, Node01, 1);

 if(Flag!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }
```

```
 autpmc_Close(PORTNUM);
}
```

## 3.42 autpmc_SetSpdMul

autpmc_SetSpdMul 함수는 PMC-2HSP/2HSN 의 속도 배율를 설정합니다.

### (1) 함수명

```
int autpmc_SetSpdMul(
int PortNum,
char nNodeId,
char axis,
int iSpdMul
);
```

### (2) 파라미터

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- iSpdMul

  속도 배율을 입력합니다. 이때 유효한 값은 1~500 입니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력했습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);
```

```
 Flag = autpmc_SetSpdMul(PORTNUM, Node01, FPMC_X_Y_AXIS, 100);

 if(Flag!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }


 autpmc_Close(PORTNUM);
}
```

## 3.43 autpmc_SetJrkSpd

autpmc_SetJrkSpd 함수는 PMC-2HSP/2HSN 의 가가속도를 설정합니다.

### (1) 함수명

```
autpmc_SetJrkSpd(
int PortNum,
char nNodeId,
char axis,
int iJrkSpd
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- iJrkSpd

  가가속도를 입력합니다. 이때 유효한 값은 1~65,535 입니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력했습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
```

```
int Flag=0;

autpmc_Open(PORTNUM, FPMC_BAUD_115200);


Flag = autpmc_SetJrkSpd (PORTNUM, Node01, FPMC_X_Y_AXIS, 1000);

if(Flag!=FPMC_OK)
{
        printf("error!\n");
        return;
}


autpmc_Close(PORTNUM);
}
```

## 3.44 autpmc_SetAccSpdRate

autpmc_SetAccSpdRate 함수는 PMC-2HSP/2HSN 의 가속률을 설정합니다.

### (1) 함수명

```
int autpmc_SetAccSpdRate(
int PortNum,
char nNodeId,
char axis,
int iAccSpdRate
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- iAccSpdRate

  가속률을 입력합니다. 이때 유효한 값은 1~8,000 입니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력했습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
```

```
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_SetAccSpdRate (PORTNUM, Node01, FPMC_X_Y_AXIS, 1000);

 if(Flag!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }


 autpmc_Close(PORTNUM);
}
```

## 3.45 autpmc_SetDecSpdRate

autpmc_SetDecSpdRate 함수는 PMC-2HSP/2HSN 의 가속률을 설정합니다.

### (1) 함수명

```
int autpmc_SetDecSpdRate(
int PortNum,
char nNodeId,
char axis,
int iDecSpdRate
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- iDecSpdRate

  가속률을 입력합니다. 이때 유효한 값은 1~8,000 입니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력했습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
```

```
int Flag=0;

autpmc_Open(PORTNUM, FPMC_BAUD_115200);


Flag = autpmc_SetDecSpdRate (PORTNUM, Node01, FPMC_X_Y_AXIS, 1000);

if(Flag!=FPMC_OK)
{
        printf("error!\n");
        return;
}


autpmc_Close(PORTNUM);
}
```

## 3.46 autpmc_SetStrSpd

autpmc_SetStrSpdRate 함수는 PMC-2HSP/2HSN 의 초기 속도를 설정합니다.

### (1) 함수명

```
int autpmc_SetStrSpd(
int PortNum,
char nNodeId,
char axis,
int iStrSpd
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- iStrSpd

  초기 속도를 입력합니다. 이때 유효한 값은 1~8,000 입니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력했습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
```

```
int Flag=0;

autpmc_Open(PORTNUM, FPMC_BAUD_115200);


Flag = autpmc_SetStrSpd(PORTNUM, Node01, FPMC_X_Y_AXIS, 1000);

if(Flag!=FPMC_OK)
{
        printf("error!\n");
        return;
}


autpmc_Close(PORTNUM);
}
```

## 3.47 autpmc_ SetDrvSpd

autpmc_SetDrvSpd 함수는 PMC-2HSP/2HSN 의 구동 속도를 선택합니다.

### (1) 함수명

```
int autpmc_SetDrvSpd(
int PortNum,
char nNodeId,
char axis,
int nDrvIndex
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- nDrvIndex

  드라이브 속도 인덱스를 입력합니다. 이때 유효한 값은 1~4 입니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력했습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
```

```
int Flag=0;

autpmc_Open(PORTNUM, FPMC_BAUD_115200);



Flag = autpmc_SetDrvSpd(PORTNUM, Node01, FPMC_X_AXIS, 1); //속도 선택 1 ~ 4

autpmc_ABSMove(PORTNUM, Node01, FPMC_X_AXIS, 10000);

if(Flag!=FPMC_OK)
{
        printf("error!\n");
        return;
}



autpmc_Close(PORTNUM);
}
```

## 3.48 autpmc_SetDrvSpdPgm

autpmc_SetDrvSpdPgm 함수는 PMC-2HSP/2HSN 의 구동 속도를 설정하는 함수로서 프로그램 모드에서 사용됩니다.

### (1) 함수명

```
int autpmc_SetDrvSpdPgm(
int PortNum,
char nNodeId,
char axis,
int nDrvIndex,
int iDrvSpd
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- nDrvIndex

  드라이브 속도 인덱스를 입력합니다. 이때 유효한 값은 1~4 입니다.

- iDrvSpd

  구동 속도를 입력합니다. 이때 유효한 값은 1~8,000 입니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력했습니다. |

**(4) 사용 예**

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_SetDrvSpdPgm(PORTNUM, Node01, FPMC_X_AXIS, 1, 1000);

 if(Flag!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }


 autpmc_Close(PORTNUM);
}
```

## 3.49 autpmc_SetTimPgm

autpmc_SetTimPgm 함수는 PMC-2HSP/2HSN 의 포스트 타이머를 설정하는 함수로서 프로그램 모드에서 사용됩니다.

### (1) 함수명

```
int autpmc_SetTimPgm(
int PortNum,
char nNodeId,
char axis,
int nIndex,
int iPostTim
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- nIndex

  사용할 포스터 타이머를 선택합니다. 이때 유효한 값은 1~3 입니다.

- iPostTim

  포스트 타이머를 입력합니다. 이때 유효한 값은 1~65,535 이며, 값의 단위는 ms 입니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력했습니다. |

**(4) 사용 예**

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_SetTimPgm (PORTNUM, Node01, FPMC_X_Y_AXIS, 1, 1000);

 if(Flag!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }


 autpmc_Close(PORTNUM);
}
```

## 3.50 autpmc_SetSofLmt

autpmc_SetSofLmt 함수는 PMC-2HSP/2HSN 의 소프트웨어 리미트를 설정합니다.

펄스 스케일값에 비례됩니다.

### (1) 함수명

```
int autpmc_SetSftLmt(
int PortNum,
char nNodeId,
char axis,
int iDirection,
long lSoftLmt
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- iDirection

  각 축의 '+' 방향과 '–' 방향을 선택합니다.

  각 축의 소프트웨어 리미트 '+' 방향을 설정하실려면 FPMC_SOFLMT_PLUS(0)을 소프트웨어 리미트 '-' 방향을 설정하실려면 FPMC_SOFLMT_MINUS(1)을 입력합니다.

- lSoftLmt

  소프트웨어 리미트를 입력합니다. 이때 유효한 값은 -8,388,608~8,388,607 입니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력 에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력했습니다. |

**(4) 사용 예**

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_SetSofLmt(PORTNUM, Node01, FPMC_X_Y_AXIS, 0, 8382607);

 if(Flag!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }


 autpmc_Close(PORTNUM);
}
```

## 3.51 autpmc_SetEndPWidth

autpmc_SetEndPWidth 함수는 PMC-2HSP/2HSN 의 드라이브 종료 펄스의 폭을 설정합니다.

### (1) 함수명

```
int autpmc_SetEndPWidth(
int PortNum,
char nNodeId,
char axis,
int iEndPWidth
);
```

### (2) 파라미터

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- iEndPWidth

  드라이브 종료 펄스의 폭을 입력합니다. 이때 유효한 값은 1~65,535 이며, 값의 단위는 ms 입니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력했습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;
```

```
autpmc_Open(PORTNUM, FPMC_BAUD_115200);


Flag = autpmc_SetEndPWidth (PORTNUM, Node01, FPMC_X_Y_AXIS, 1000);

if(Flag!=FPMC_OK)
{
        printf("error!\n");
        return;
}


autpmc_Close(PORTNUM);
}
```

## 3.52 autpmc_SetPulSclNum

autpmc_SetPulScl 함수는 PMC-2HSP/2HSN 의 펄스 스케일의 분자를 설정합니다.

### (1) 함수명

```
int autpmc_SetPulSclNum(
int PortNum,
char nNodeId,
char axis,
int iPulScl
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- iPulScl

  펄스 스케일의 분자를 입력합니다. 이때 유효한 값은 1~65,535 입니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력했습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
```

```
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_SetPulSclNum (PORTNUM, Node01, FPMC_X_Y_AXIS, 1000);

 if(Flag!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }


 autpmc_Close(PORTNUM);
}
```

## 3.53 autpmc_SetPulSclDen

autpmc_SetPulScl 함수는 PMC-2HSP/2HSN 의 펄스 스케일의 분모를 설정합니다.

### (1) 함수명

```
int autpmc_SetPulSclDen(
int PortNum,
char nNodeId,
char axis,
int iPulScl
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- iPulScl

  펄스 스케일의 분자를 입력합니다. 이때 유효한 값은 이때 유효한 값은 1~65,535 입니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력했습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3
```

```
void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_SetPulSclDen (PORTNUM, Node01, FPMC_X_Y_AXIS, 1000);

 if(Flag!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }


 autpmc_Close(PORTNUM);
}
```

## 3.54 autpmc_SetHomMod

autpmc_SetHomMod 함수는 PMC-2HSP/2HSN 의 원점 복귀 모드를 설정합니다.

### (1) 함수명

```
int autpmc_SetHomMod(
int PortNum,
char nNodeId,
char axis,
int nStepNo,
BOOL bEnable,
BOOL bDirection
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- nStepNo

  설정 할 스텝 번호를 입력합니다. 이때 유효한 값은 1~4 입니다.

- bEnable

  원점 복귀를 사용하려면 FPMC_ENABLE(1)을 사용하지 않으려면 FPMC_DISABLE(0)을 입력합니다.

- bDirection

  원점 복귀 시 검색 방향을 '+'로 설정하려면 FPMC_PLUS(0)을 '-'로 설정하려면 FPMC_MINUS(1)을 입력합니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력 에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력했습니다. |

**(4) 사용 예**

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_SetHomMod(PORTNUM, Node01, FPMC_X_AXIS, 1, 1, 1);

 if(Flag!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }


 autpmc_Close(PORTNUM);
}
```

## 3.55 autpmc_HomStop

autpmc_HomStop 함수는 PMC-2HSP/SHSN 에서 원점 복귀 모드를 종료합니다.

### (1) 함수명

```
int autpmc_HomStop(
int PortNum,
char nNodeId,
char axis
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE 을 리턴합니다. 또한, Broadcast (0x80)을 입력하면 브로드캐스트 기능으로 현재 PC 와 연결된 모든 PMC-2HSP/2HSN 에 데이터를 전송합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);
```

```
autpmc_HomRun(PORTNUM, Node01, FPMC_X_AXIS); //원점 복귀 모드 실행

Flag = autpmc_HomStop(PORTNUM, Node01, FPMC_X_AXIS); //원점 복귀 모드 종료

if(Flag!=FPMC_OK)
{
        printf("error!\n");
        return;
}


autpmc_Close(PORTNUM);
}
```

## 3.56 autpmc_Step1Enable

autpmc_Step1Enable 함수는 PMC-2HSP/2HSN 의 원점 복귀 모드에서 스텝 1 의 사용 유무를 설정합니다.

### (1) 함수명

```
int autpmc_Step1Enable(
int PortNum,
char nNodeId,
char axis,
BOOL bEnable
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- bEnable

  원점 복귀를 사용하려면 FPMC_ENABLE(1)을 사용하지 않으려면 FPMC_DISABLE(0)을 입력합니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력했습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3
```

```
void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_Step1Enable (PORTNUM, Node01, FPMC_X_Y_AXIS, 0);

 if(Flag!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }


 autpmc_Close(PORTNUM);
}
```

## 3.57 autpmc_Step2Enable

autpmc_Step2Enable 함수는 PMC-2HSP/2HSN 의 원점 복귀 모드에서 스텝 2 의 사용 유무를 설정합니다.

### (1) 함수명

```
int autpmc_Step2Enable(
int PortNum,
char nNodeId,
char axis,
BOOL bEnable
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- bEnable

  원점 복귀를 사용하려면 FPMC_ENABLE(1)을 사용하지 않으려면 FPMC_DISABLE(0)을 입력합니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력했습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3
```

```
void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_Step2Enable (PORTNUM, Node01, FPMC_X_Y_AXIS, 0);

 if(Flag!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }


 autpmc_Close(PORTNUM);
}
```

## 3.58 autpmc_Step3Enable

autpmc_Step3Enable 함수는 PMC-2HSP/2HSN 의 원점 복귀 모드에서 스텝 3 의 사용 유무를 설정합니다.

### (1) 함수명

```
int autpmc_Step3Enable(
int PortNum,
char nNodeId,
char axis,
BOOL bEnable
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- bEnable

  원점 복귀를 사용하려면 FPMC_ENABLE(1)을 사용하지 않으려면 FPMC_DISABLE(0)을 입력합니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력했습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3
```

```
void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_Step3Enable (PORTNUM, Node01, FPMC_X_Y_AXIS, 0);

 if(Flag!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }


 autpmc_Close(PORTNUM);
}
```

## 3.59 autpmc_Step4Enable

autpmc_Step4Enable 함수는 PMC-2HSP/2HSN 의 원점 복귀 모드에서 스텝 4 의 사용 유무를 설정합니다.

### (1) 함수명

```
int autpmc_Step4Enable(
int PortNum,
char nNodeId,
char axis,
BOOL bEnable
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- bEnable

  원점 복귀를 사용하려면 FPMC_ENABLE(1)을 사용하지 않으려면 FPMC_DISABLE(0)을 입력합니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력했습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3
```

```
void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_Step4Enable (PORTNUM, Node01, FPMC_X_Y_AXIS, 0);

 if(Flag!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }


 autpmc_Close(PORTNUM);
}
```

## 3.60 autpmc_Step1Direction

autpmc_Step1Direction 함수는 PMC-2HSP/2HSN 의 원점 복귀 시 스텝 1 의 검색 방향을 설정합니다.

### (1) 함수명

```
int autpmc_Step1Direction(
int PortNum,
char nNodeId,
char axis,
BOOL bDirection
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- bDirection

  원점 복귀 시 검색 방향을 '+'로 설정하려면 FPMC_PLUS(0)을 '-'로 설정하려면 FPMC_MINUS(1)을 입력합니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력했습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"
```

```
#define PORTNUM 3

void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_Step1Direction (PORTNUM, Node01, FPMC_X_Y_AXIS, 1);

 if(Flag!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }


 autpmc_Close(PORTNUM);
}
```

## 3.61 autpmc_Step2Direction

autpmc_Step2Direction 함수는 PMC-2HSP/2HSN 의 원점 복귀 시 스텝 2 의 검색 방향을 설정합니다.

### (1) 함수명

```
int autpmc_Step2Direction(
int PortNum,
char nNodeId,
char axis,
BOOL bDirection
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- bDirection

  원점 복귀 시 검색 방향을 '+'로 설정하려면 FPMC_PLUS(0)을 '-'로 설정하려면 FPMC_MINUS(1)을 입력합니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력했습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3
```

```
void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_Step2Direction (PORTNUM, Node01, FPMC_X_Y_AXIS, 1);

 if(Flag!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }


 autpmc_Close(PORTNUM);
}
```

## 3.62 autpmc_Step3Direction

autpmc_Step3Direction 함수는 PMC-2HSP/2HSN 의 원점 복귀 시 스텝 3 의 검색 방향을 설정합니다.

### (1) 함수명

```
int autpmc_Step3Direction(
int PortNum,
char nNodeId,
char axis,
BOOL bDirection
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- bDirection

  원점 복귀 시 검색 방향을 '+'로 설정하려면 FPMC_PLUS(0)을 '-'로 설정하려면 FPMC_MINUS(1)을 입력합니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력했습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3
```

```
void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_Step3Direction (PORTNUM, Node01, FPMC_X_Y_AXIS, 1);

 if(Flag!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }


 autpmc_Close(PORTNUM);
}
```

## 3.63 autpmc_Step4Direction

autpmc_Step4Direction 함수는 PMC-2HSP/2HSN 의 원점 복귀 시 스텝 4 의 검색 방향을 설정합니다.

### (1) 함수명

```
int autpmc_Step4Direction(
int PortNum,
char nNodeId,
char axis,
BOOL bDirection
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- bDirection

  원점 복귀 시 검색 방향을 '+'로 설정하려면 FPMC_PLUS(0)을 '-'로 설정하려면 FPMC_MINUS(1)을 입력합니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력했습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3
```

```
void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_Step4Direction (PORTNUM, Node01, FPMC_X_Y_AXIS, 1);

 if(Flag!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }


 autpmc_Close(PORTNUM);
}
```

## 3.64 autpmc_SetHomEndPosClr

autpmc_SetHomEndPosClr 함수는 PMC-2HSP/2HSN 의 원점 복귀 종료 시 위치 카운터를 초기화합니다.

### (1) 함수명

```
int autpmc_SetHomEndPosClr(
int PortNum,
char nNodeId,
char axis,
BOOL bEnable
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- bEnable

  위치 카운터 초기화를 사용하려면 FPMC_ENABLE(1)을 사용하지 않으려면 FPMC_DISABLE(0)을 입력합니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력했습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3
```

```
void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_SetHomEndPosClr (PORTNUM, Node01, FPMC_X_Y_AXIS, 0);

 if(Flag!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }


 autpmc_Close(PORTNUM);
}
```

## 3.65 autpmc_SetHomSigLev

autpmc_SetHomSigLev 함수는 PMC-2HSP/2HSN 의 원점 신호의 논리 레벨을 설정합니다.

### (1) 함수명

```
int autpmc_SetHomSigLev(
int PortNum,
char nNodeId,
char axis,
int nHomSigNo,
BOOL bLevel
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- nHomSigNo

  원점 근접 신호를 선택하려면 FPMC_HSTOP0(0)을 원점 신호를 선택하려면 FPMC_HSTOP1(1)을 엔코더 Z 상 신호를 선택하려면 FPMC_HSTOP2(2)를 입력합니다.

- bLevel

  Low 로 설정하려면 FPMC_LOW(0)을 High 로 설정하려면 FPMC_HIGH(1)을 입력합니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력했습니다. |

**(4) 사용 예**

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_SetHomSigLev (PORTNUM, Node01, FPMC_X_Y_AXIS, 0, 0);

 if(Flag!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }


 autpmc_Close(PORTNUM);
}
```

## 3.66 autpmc_SetHomSpd

autpmc_SetHomSpd 함수는 PMC-2HSP/2HSN 의 원점 복귀 속도를 설정합니다.

### (1) 함수명

```
int autpmc_SetHomSpd(
int PortNum,
char nNodeId,
char axis,
BOOL bSpd,
int iSpd
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- bSpd

  저속 원점 복귀 속도를 설정하려면 FPMC_LOW(0)을 고속 원점 복귀 속도를 설정하려면 FPMC_HIGH(1)을 설정합니다.

- iSpd

  원점 복귀 속도를 입력합니다. 이때 유효한 값은 1~8,000 입니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력했습니다. |

**(4) 사용 예**

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_SetHomSpd(PORTNUM, Node01, FPMC_X_Y_AXIS, 0, 1000);

 if(Flag!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }


 autpmc_Close(PORTNUM);
}
```

## 3.67 autpmc_SetHomOffset

autpmc_SetHomOffset 함수는 PMC-2HSP/2HSN 의 원점 복귀 스텝 4 의 고속 오프셋 이동의 이동량을 설정합니다. (펄스 스케일값에 비례됩니다.)

### (1) 함수명

```
int autpmc_SetHomOffset(
int PortNum,
char nNodeId,
char axis,
long lOffset
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- lOffset

  원점 복귀 스텝 4 의 고속 오프셋 이동의 이동량을 입력합니다. 이때 유효한 값은 0~8,388,607 입니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력했습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3
```

```
void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_SetHomOffset(PORTNUM, Node01, FPMC_X_Y_AXIS, 1000);

 if(Flag!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }


 autpmc_Close(PORTNUM);
}
```

# 4 I/O 제어

## 4.1 autpmc_GetParallelIO

autpmc_GetParallelIO 함수는 PMC-2HSP/2HSN 의 Parallel I/F 커넥터(CN3) 입력 신호를 읽습니다.

### (1) 함수명

```
struct PARALLELSTATE *autpmc_GetParallelIO(
int PortNum,
char nNodeId,
PARALLELSTATE *pState
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- pState

  Parallel I/F 의 신호를 읽어 와서 현재 변수에 저장합니다.

| 구조체명 | 변수 형식 | 내용 | 데이터값 |
|---|---|---|---|
| PARALLEL STATE | BOOL HOME; | 원점복귀 시작 | 0 : OFF/ 1 : ON |
| | BOOL STROBE; | 드라이브 시작 | 0 : OFF/ 1 : ON |
| | BOOL X; | X 축 지정/ 조그 2 모드 Y+ | 0 : OFF/ 1 : ON |
| | BOOL Y; | Y 축 지정/ 조그 2 모드 Y+ | 0 : OFF/ 1 : ON |
| | BOOL MODE0; | 스템 지정 0/ 런+/ 조그 2 모드 X+ | 0 : OFF/ 1 : ON |
| | BOOL MODE1; | 스템 지정 1/ 런-/ 조그 2 모드 X- | 0 : OFF/ 1 : ON |
| | BOOL STEPSL0; | 스텝 지정 2/ 드라이브 속도 지정 0 | 0 : OFF/ 1 : ON |
| | BOOL STEPSL1; | 스템 지정 3/ 드라이브 속도 지정 1 | 0 : OFF/ 1 : ON |
| | BOOL STEPSL2; | 스텝 지정 4/ 조그 지정 | 0 : OFF/ 1 : ON |
| | BOOL STEPSL3; | 스텝 지정 5/ 드라이브 정지 | 0 : OFF/ 1 : ON |
| | BOOL STEPSL4; | 운전 모드 지정 0 | 0 : OFF/ 1 : ON |
| | BOOL STEPSL5; | 운전 모드 지정 1 | 0 : OFF/ 1 : ON |

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);



 PARALLELSTATE State;   //구조체 변수 선언
 PARALLELSTATE *pState = &State;

 autpmc_GetParallelIO(PORTNUM, Node01, pState);

 printf("원점 복귀 시작 : %d\n", pState->HOME);

 if(pState->bErrorState!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }



 autpmc_Close(PORTNUM);
}
```

## 4.2 autpmc_GetAxisIO

autpmc_GetAxisIO 함수는 PMC-2HSP/2HSN 의 X 축(CN4)과 Y 축(CN5)의 입/출력 커넥터 신호를 읽습니다.

**(1) 함수명**

```
struct AXISSTATE *autpmc_GetAxisIO(
int PortNum,
char nNodeId,
char axis,
AXISSTATE *pState
);
```

**(2) 파라미터**

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- pState

  CN4 번 X 축 또는 CN5 번 Y 축의 입/출력 신호를 읽어 와서 현재 변수에 저장을 합니다.

| 구조체명 | 변수 형식 | 내용 | 데이터값 |
|---|---|---|---|
| PARALLEL STATE | BOOL bHomSig0[2]; | 원점 근접 (X 축 : bHomSig0 [0], Y 축 : bHomSig0 [1]) | 0 : OFF/ 1 : ON |
| | BOOL bHomSig1[2]; | 원점 (X 축 : bHomSig1 [0], Y 축 : bHomSig1 [1]) | 0 : OFF/ 1 : ON |
| | BOOL bHomSig2[2]; | 엔코더 Z 상 (X 축 : bHomSig2 [0], Y 축 : bHomSig2 [1]) | 0 : OFF/ 1 : ON |
| | BOOL LmtP[2]; | Limit+ (X 축 : LmtP [0], Y 축 : LmtP [1]) | 0 : OFF/ 1 : ON |
| | BOOL LmtM[2]; | Limit- (X 축 : LmtM [0], Y 축 : LmtM [1]) | 0 : OFF/ 1 : ON |
| | BOOL EMG[2]; | EMG (X 축 : EMG [0], Y 축 : EMG [1]) | 0 : OFF/ 1 : ON |

| 구조체명 | 변수 형식 | 내용 | 데이터값 |
|---|---|---|---|
| PARALLEL STATE | BOOL bInput0Lev[2]; | 범용 입력 0 (X 축 : bInput0Lev [0], Y 축 : bInput0Lev [1]) | 0 : OFF/ 1 : ON |
| | BOOL bInput1Lev[2]; | 범용 입력 1 (X 축 : bInput1Lev [0], Y 축 : bInput1Lev [1]) | 0 : OFF/ 1 : ON |

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);



 AXISSTATE State;            //구조체 변수 선언
 AXISSTATE *pState = &State;

 autpmc_GetAxisIO (PORTNUM, Node01, FPMC_X_AXIS, pState);

 printf(" X 축 원점 근접 : %d\n Y 축 원점 근접 : %d\n", pState->bHomSig0[0], pState-
>bHomSig0[1]);

 if(pState->bErrorState[0]||pState->bErrorState[1]!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }



 autpmc_Close(PORTNUM);
}
```

## 4.3 autpmc_SetUserOut

autpmc_SetUserOut 함수는 PMC-2HSP/2HSN 의 CN4 커넥터의 X 축 범용 출력 0/1 핀 또는 CN5 커넥터의 Y 축 범용 출력 0/1 핀을 ON/OFF 합니다.

### (1) 함수명

```
int autpmc_SetUserOut(
int PortNum,
char nNodeId,
char axis,
BOOL bPort,
BOOL bOn
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- bPort

  범용 출력 0 핀을 설정하려면 FPMC_OUTPORT0(0)을 범용 출력 1 핀을 설정하려면 FPMC_OUTPORT1(1)을 입력합니다.

- bOn

  해당 범용 출력 포트 핀을 ON 하려면 FPMC_ON(1)을 OFF 하려면 FPMC_OFF(0)을 입력합니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
```

```
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_SetUserOut(PORTNUM, Node01, FPMC_X_Y_AXIS, 0, 0);

 if(Flag!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }


 autpmc_Close(PORTNUM);
}
```

## 4.4 autpmc_GetCurPos

autpmc_GetCurPos 함수는 PMC-2HSP/2HSN 의 현재 좌표를 읽습니다.

### (1) 함수명

```
int autpmc_GetCurPos(
int PortNum,
char nNodeId,
char axis,
long *lCurPos
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- lCurPos

  현재 좌표를 읽습니다. (-8,388,608~8,388,607)

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |

### (4) 사용 예

```
nNodeId#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;

 long lCurPos=0;
```

```
autpmc_Open(PORTNUM, FPMC_BAUD_115200);


Flag = autpmc_GetCurPos (PORTNUM, Node01, FPMC_X_AXIS, &lCurPos);

printf("현재 위치 : %ld\n", lCurPos);

if(Flag!=FPMC_OK)
{
        printf("error!\n");
        return;
}


autpmc_Close(PORTNUM);
}
```

## 4.5 autpmc_GetCurPgmNo

autpmc_GetCurPgmNo 함수는 PMC-2HSP/2HSN 의 현재 실행 중인 프로그램 스텝을 읽습니다.

### (1) 함수명

```
int autpmc_GetCurPgmNo(
int PortNum,
char nNodeId,
char axis,
int *iCurPgmNo
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- iCurPgmNo

  현재 실행 중인 프로그램 스텝을 읽습니다. (0~199)

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;
```

```
int iCurPgmNo=0;

autpmc_Open(PORTNUM, FPMC_BAUD_115200);


Flag = autpmc_GetCurPgmNo (PORTNUM, Node01, FPMC_X_AXIS, &iCurPgmNo);

printf("실행 중인 Program Step : %d\n", iCurPgmNo);

if(Flag!=FPMC_OK)
{
        printf("error!\n");
        return;
}


autpmc_Close(PORTNUM);
}
```

## 4.6 autpmc_GetErrorSt

autpmc_GetErrorSt 함수는 PMC-2HSP/2HSN 의 에러 상태를 읽습니다.

### (1) 함수명

```
struct PMC_ERRORSTATE *autpmc_GetErrorSt(
int PortNum,
char nNodeId,
char axis,
PMC_ERRORSTATE *pError
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- pError

  현재 에러 상태를 읽습니다.

| 구조체명 | 변수 형식 | 내용 | 데이터값 |
|---|---|---|---|
| PMC_ ERRORSTATE | int iErrorState | 오류 상태 확인 | FPMC_OK(0) : 함수가 정상적으로 명령을 수행하였습니다.<br>FPMC_INVALID_NODE(3) : 잘못된 노드 ID 를 입력하였습니다.<br>FPMC_INVALID_AXIS(4) : 잘못된 축을 입력하였습니다.<br>FPMC_INVALID_DATA)(5) : 잘못된 데이터를 입력하였습니다. |

| 구조체명 | 변수 형식 | 내용 | 데이터값 |
|---|---|---|---|
| PMC_<br>ERRORSTATE | BOOL bSofLmtErrP[2]<br>1. bSofLmtErrP[0] -> X 축<br>2. bSofLmtErrP[1] -> Y 축 | 소프트웨어<br>리미트 + 에러 | FPMC_ON(1)/<br>FPMC_OFF(0) |
| | BOOL bSofLmtErrM[2]<br>1. bSofLmtErrM[0] -> X 축<br>2. bSofLmtErrM[1] -> Y 축 | 소프트웨어<br>리미트 - 에러 | FPMC_ON(1)/<br>FPMC_OFF(0) |
| | BOOL bHardLmtErrP[2]<br>1. bHardLmtErrP[0] -> X 축<br>2. bHardLmtErrP[1] -> Y 축 | 하드웨어<br>리미트 + 에러 | FPMC_ON(1)/<br>FPMC_OFF(0) |
| | BOOL bHardLmtErrM[2]<br>1. bHardLmtErrM[0] -> X 축<br>2. bHardLmtErrM[1] -> Y 축 | 하드웨어<br>리미트 - 에러 | FPMC_ON(1)/<br>FPMC_OFF(0) |
| | BOOL bEmgErr[2]<br>1. bEmgErr[0] -> X 축<br>2. bEmgErr[1] -> Y 축 | 긴급정지 시<br>발생하는 에러 | FPMC_ON(1)/<br>FPMC_OFF(0) |
| | BOOL bPgmErr[2]<br>1. bPgmErr[0] -> X 축<br>2. bPgmErr[1] -> Y 축 | 프로그램 모드<br>에러 | FPMC_ON(1)/<br>FPMC_OFF(0) |
| | BOOL bHomErr[2]<br>1. bHomErr[0] -> X 축<br>2. bHomErr[1] -> Y 축 | 원점복귀 모드<br>에러 | FPMC_ON(1)/<br>FPMC_OFF(0) |
| | BOOL bInxErr[2]<br>1. bInxErr[0] -> X 축<br>2. bInxErr[1] -> Y 축 | 인덱스 에러 | FPMC_ON(1)/<br>FPMC_OFF(0) |

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
```

```
 autpmc_Open(PORTNUM, FPMC_BAUD_115200);



 PMC_ERRORSTATE   Error;        //구조체 변수 선언
 PMC_ERRORSTATE   *pError = &Error;

 autpmc_GetErrorSt (PORTNUM, Node01, FPMC_X_Y_AXIS, pError);

 printf(" X 축 소프트웨어 리미트 + 에러 : %d\n Y 축 소프트웨어 리미트 + 에러 : %d\n",
pError->bSofLmtErrP[0], pError->bSofLmtErrP[1]);

 if(pError->bErrorState[0]||pError->bErrorState[1]!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }



 autpmc_Close(PORTNUM);
}
```

## 4.7 autpmc_IsRun

autpmc_IsRun 함수는 PMC-2HSP/2HSN 의 현재 구동 상태를 읽어옵니다.

### (1) 함수명

```
struct PMC_RUNSTATE *autpmc_IsRun(
int PortNum,
char nNodeId,
char axis,
PMC_RUNSTATE *pRun
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- pRun

  현재 구동 상태를 읽습니다.

| 구조체명 | 변수 형식 | 내용 | 데이터값 |
|---|---|---|---|
| PMC_RUNSTATE | int iErrorState | 오류 상태 확인 | 0 : 함수가 정상적으로 명령을 수행하였습니다.<br>3 : 잘못된 노드 ID 를 입력하였습니다.<br>4 : 잘못된 축을 입력하였습니다.<br>5 : 잘못된 데이터를 입력하였습니다. |
| | BOOL bHomIsRun[2]<br>1. bHomIsRun[0] ->X 축<br>2. bHomIsRun[1] ->Y 축 | X 축 원점복귀 모드 구동 | 1 : ON<br>0 : OFF |
| | BOOL bJogIsRun[2]<br>1. bJogIsRun[0] ->X 축<br>2. bJogIsRun[1] ->Y 축 | X 축 조그 모드 구동 | 1 : ON<br>0 : OFF |

| 구조체명 | 변수 형식 | 내용 | 데이터값 |
|---|---|---|---|
| PMC_RU<br>NSTATE | BOOL bPgmIsRun[2]<br>1. bPgmIsRun[0] ->X 축<br>2. bPgmIsRun[1] ->Y 축 | X 축 프로그램<br>모드 구동 | 1 : ON<br>0 : OFF |

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);



 PMC_RUNSTATE Run; //구조체 변수 선언
 PMC_RUNSTATE *pRun = &Run;


 autpmc_IsRun (PORTNUM, Node01, FPMC_X_Y_AXIS, pRun);

 printf(" X 축 원점 복귀 모드 구동 : %d\n Y 축 원점 복귀 모드 구동 : %d\n", pRun-
>bHomIsRun[0], pRun->bHomIsRun[1]);

 if(pRun->bErrorState[0]||pRun->bErrorState[1]!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }



 autpmc_Close(PORTNUM);
}
```

## 4.8 autpmc_GetModName

autpmc_GetModName 함수는 PMC-2HSP/2HSN 의 모델명을 읽어옵니다.

### (1) 함수명

```
struct PMC_SOFTVERSION *autpmc_GetModName(
int PortNum,
char nNodeId,
PMC_SOFTVERSION *pVersion
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- pVersion

  모델명을 읽습니다.

| 구조체명 | 변수 형식 | 내용 | 데이터값 |
|---|---|---|---|
| PMC_SOFT VERSION | int iErrorState | 오류 상태 확인 | 0 : 함수가 정상적으로 명령을 수행하였습니다.<br>3 : 잘못된 노드 ID 를 입력하였습니다.<br>4 : 잘못된 축을 입력하였습니다.<br>5 : 잘못된 데이터를 입력하였습니다. |
| | char cModName[12]; | 모델명 읽어오기 | PMC-2HSP-USB<br>PMC-2HSP-485<br>PMC-2HSN-USB<br>PMC-2HSN-485 |

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3
```

```
void main()
{

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);



 PMC_SOFTVERSION   Version; //구조체 변수 선언
 PMC_SOFTVERSION   *pVersion = &Version;


 autpmc_GetModName (PORTNUM, Node01, pVersion);

 printf("Model Name: %s\n", pVersion->cModName);

 if(pVersion->bErrorState!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }



 autpmc_Close(PORTNUM);
}
```

## 4.9 autpmc_GetSofVer

autpmc_GetSofVer 함수는 PMC-2HSP/2HSN 의 펌웨어 버전을 읽어옵니다.

### (1) 함수명

```
struct PMC_SOFTVERSION *autpmc_GetSofVer(
int PortNum,
char nNodeId,
PMC_SOFTVERSION *pVersion
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- pVersion

  현재 펌웨어 버전을 읽습니다. 버전 구성은 연도 4 자리, 월 2 자리, 일 2 자리로 연속으로 구성되어 있습니다.

  예) 20091009  2009 년 10 월 09 일 버전

| 구조체명 | 변수 형식 | 내용 | 데이터값 |
|---|---|---|---|
| PMC_PARADATA | int iErrorState | 오류 상태 확인 | FPMC_OK(0) : 함수가 정상적으로 명령을 수행하였습니다.<br>FPMC_INVALID_NODE(3) : 잘못된 노드 ID 를 입력하였습니다.<br>FPMC_INVALID_AXIS(4) : 잘못된 축을 입력하였습니다.<br>FPMC_INVALID_DATA)(5) : 잘못된 데이터를 입력하였습니다. |
| | char cSofVer[8]; | 펌웨어 버전 | yyyymmdd |

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |

**(4) 사용 예**

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);



 PMC_SOFTVERSION Version; //구조체 변수 선언
 PMC_SOFTVERSION *pVersion = &Version;


 autpmc_GetSofVer (PORTNUM, Node01, pVersion);

 printf("Software Version: %s\n", pVersion->cSofVer);

 if(pVersion->bErrorState!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }



 autpmc_Close(PORTNUM);
}
```

# 5 동작

## 5.1 autpmc_HomRun

autpmc_HomRun 함수는 PMC-2HSP/SHSN 에서 원점 복귀 모드를 실행합니다.

**(1) 함수명**

```
int autpmc_HomRun(
int PortNum,
char nNodeId,
char axis
);
```

**(2) 파라미터**

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

  또한, Broadcast (0x80)을 입력하면 브로드캐스트 기능으로 현재 PC 와 연결된 모든 PMC-2HSP/2HSN 에 데이터를 전송합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

**(3) 리턴값**

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력하였습니다. |

**(4) 사용 예**

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3
```

```
void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_HomRun(PORTNUM, Node01, FPMC_X_Y_AXIS);

 if(Flag!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }


 autpmc_Close(PORTNUM);
}
```

## 5.2 autpmc_ABSMove

autpmc_ABSMove 함수는 PMC-2HSP/2HSN 에서 원점을 기준으로 지정된 거리를 절대 위치로 이동합니다. (펄스 스케일값에 비례됩니다.)

### (1) 함수명

```
int autpmc_ABSMove(
int PortNum,
char nNodeId,
char axis,
long lPos
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- lPos

  이동 위치를 절대값으로 입력합니다. 이때 유효 범위는 -8,388,608~8,388,607 입니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3
```

```
void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);



 autpmc_SetDrvSpd(PORTNUM, Node01, FPMC_X_AXIS, 1); //속도 선택 1 ~ 4

 Flag = autpmc_ABSMove(PORTNUM, Node01, FPMC_X_AXIS, 10000);

 if(Flag!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }



 autpmc_Close(PORTNUM);
}
```

## 5.3 autpmc_INCMove

autpmc_INCMove 함수는 PMC-2HSP/2HSN 에서 현재 위치를 기준으로 지정된 거리를 상대 위치로 이동합니다. (펄스 스케일값에 비례됩니다.)

### (1) 함수명

```
int autpmc_INCMove(
int PortNum,
char nNodeId,
char axis,
long lPos
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- lPos

  이동 위치를 상대값으로 입력합니다. 이때 유효 범위는 -8,388,608~8,388,607 입니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3
```

```
void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);



 autpmc_SetDrvSpd(PORTNUM, Node01, FPMC_X_AXIS, 1); //속도 선택 1 ~ 4

 Flag = autpmc_INCMove(PORTNUM, Node01, FPMC_X_AXIS, 10000);

 if(Flag!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }



 autpmc_Close(PORTNUM);
}
```

## 5.4 autpmc_ContMove

autpmc_ContMove 함수는 PMC-2HSP/2HSN 을 브로드캐스트 기능으로 정지 신호가 입력될 때까지 연속하여 드라이브 펄스를 출력합니다.

### (1) 함수명

```
int autpmc_ContMove(
int PortNum,
char nNodeId,
char axis,
BOOL bDirection
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE 을 리턴합니다. 또한, Broadcast (0x80)을 입력하면 브로드캐스트 기능으로 현재 PC 와 연결된 모든 PMC-2HSP/2HSN 에 데이터를 전송합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- bDirection

  '+'로 이동하려면 FPMC_PLUS(1)을 '-'로 이동하려면 FPMC_MINUS(0)을 입력합니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"
```

```
#define PORTNUM 3

void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 autpmc_SetDrvSpd(PORTNUM, Node01, FPMC_X_Y_AXIS, 1); //속도 선택 1 ~ 4

 Flag = autpmc_ContMove(PORTNUM, Broadcast, FPMC_X_Y_AXIS, 0);

 if(Flag!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }


 autpmc_Close(PORTNUM);
}
```

## 5.5 autpmc_LIDMove

autpmc_LIDMove 함수는 PMC-2HSP 에서 현재 좌표로부터 종점 좌표를 향해 2 축 직선 보간을 실행합니다.

### (1) 함수명

```
PMC_FUNC autpmc_LIDMove(
int PortNum,
BYTE nNodeId,
BOOL bFLS,
long lXEndPos,
long lYEndPos
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- bFLS

  선속 일정을 사용하려면 FPMC_ON(1)을 사용하지 않으려면 FPMC_OFF(0)을 입력합니다.

- lXEndPos

  X 축 종점 좌표를 입력합니다. 이때 유효한 범위는 -8,388,608~8,388,607 입니다

- lYEndPos

  Y 축 종점 좌표를 입력합니다. 이때 유효한 범위는 -8,388,608~8,388,607 입니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
```

```
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);

 autpmc_SetDrvSpd(PORTNUM, Node01, FPMC_X_Y_AXIS, 1); //속도  선택  1 ~ 4

 Flag = autpmc_LIDMove(PORTNUM, Node01, 0, 1000, 1000);

 if(Flag!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }


 autpmc_Close(PORTNUM);
}
```

## 5.6 autpmc_CIDMove

autpmc_CIDMove 함수는 PMC-2HSP 에서 CW 방향(시계 방향)으로 원 보간 드라이브를 실행합니다.

### (1) 함수명

```
PMC_FUNC autpmc_CIDMove(
int PortNum,
BYTE nNodeId,
BOOL bFLS,
long lRadius,
long lMDP
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- bFLS

  선속 일정을 사용하려면 FPMC_ON(1)을 사용하지 않으려면 FPMC_OFF(0)을 입력합니다.

- lRadius

  반지름을 입력합니다. 이때 유효한 범위는 0~8,388,607 입니다.

- lMDP

  매뉴얼 감속점을 입력합니다. 이때 유효한 범위는 0~268,435,455 입니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
```

```
 int Flag=0;
 long ManualDecelPoint; //ManualDecelPoint
 int Range; // Speed Multiplier
 long Acceleration; // Acceleration rate
 long Deceleration; // Deceleration rate
 long StVelocity; // Start speed
 long DrVelocity; //Drive speed
 long CtXaxis; //X axis Center position
 long CtYaxis; //Y axis Center position
 long EndXaxis; //X axis End position
 long EndYaxis; //Y axis End position
 BOOL CW=1; //Direction
 BOOL bDecValue; //Fiexd Line Speed;

 Range = 10;
 Acceleration = 400;
 Deceleration = 400;
 StVelocity = 50;
 DrVelocity = 10;
 CtXaxis = 1000;
 CtYaxis = 1000;
 EndXaxis = 1000;
 EndYaxis = 1000;
 bDecValue = 1; //Enable;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 ManualDecelPoint = CalculateManualDecelPoint(Range, Acceleration, Deceleration,
StVelocity, DrVelocity, CtXaxis, CtYaxis, EndXaxis, EndYaxis, CW, bDecValue); //매뉴얼
감속점  계산

 printf("매뉴얼  감속점  : %ld\n", ManualDecelPoint);

 Flag = autpmc_CIDMove(PORTNUM, Node01, 0, 10000, ManualDecelPoint);

 if(Flag!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }



 autpmc_Close(PORTNUM);
}
```

## 5.7 autpmc_FIDMove

autpmc_FIDMove 함수는 PMC-2HSP 에서 CW 방향(시계 방향)으로 원호 보간 드라이브를 실행합니다.

### (1) 함수명

```
PMC_FUNC autpmc_FIDMove(
int PortNum,
BYTE nNodeId,
BOOL bFLS,
long lXCenPos,
long lYCenPos,
long lXEndPos,
long lYEndPos,
long lMDP
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- bFLS

  선속 일정을 사용하려면 FPMC_ON(1)을 사용하지 않으려면 FPMC_OFF(0)을 입력합니다.

- lXCenPos

  X 축 중점 좌표를 입력합니다. 이때 유효한 범위는 -8,388,608~8,388,607 입니다.

- lYCenPos

  Y 축 중점 좌표를 입력합니다. 이때 유효한 범위는 -8,388,608~8,388,607 입니다.

- lXEndPos

  X 축 종점 좌표를 입력합니다. 이때 유효한 범위는 -8,388,608~8,388,607 입니다.

- lYEndPos

  X 축 종점 좌표를 입력합니다. 이때 유효한 범위는 -8,388,608~8,388,607 입니다.

- lMDP

  매뉴얼 감속점을 입력합니다. 이때 유효한 범위는 0~268,435,455 입니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력 에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;
 long ManualDecelPoint; //ManualDecelPoint
 int Range; // Speed Multiplier
 long Acceleration; // Acceleration rate
 long Deceleration; // Deceleration rate
 long StVelocity; // Start speed
 long DrVelocity; //Drive speed
 long CtXaxis; //X axis Center position
 long CtYaxis; //Y axis Center position
 long EndXaxis; //X axis End position
 long EndYaxis; //Y axis End position
 BOOL CW=1; //Direction
 BOOL bDecValue; //Fiexd Line Speed;

 Range = 10;
 Acceleration = 400;
 Deceleration = 400;
 StVelocity = 50;
 DrVelocity = 10;
 CtXaxis = 1000;
 CtYaxis = 1000;
 EndXaxis = 1000;
 EndYaxis = 1000;
 bDecValue = 1; //Enable;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 ManualDecelPoint = CalculateManualDecelPoint(Range, Acceleration, Deceleration,
StVelocity, DrVelocity, CtXaxis, CtYaxis, EndXaxis, EndYaxis, CW, bDecValue); //매뉴얼
감속점 계산

 printf("매뉴얼 감속점 : %ld\n", ManualDecelPoint);

 Flag = autpmc_FIDMove(PORTNUM, Node01, CW, CtXaxis, CtYaxis, EndXaxis, EndYaxis,
ManualDecelPoint);

 if(Flag!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }


 autpmc_Close(PORTNUM);
}
```

## 5.8 autpmc_RIDMove

autpmc_RIDMove 함수는 PMC-2HSP 에서 CCW 방향(반시계 방향)으로 원호 보간 드라이브를 실행합니다.

### (1) 함수명

```
PMC_FUNC autpmc_RIDMove(
int PortNum,
BYTE nNodeId,
BOOL bFLS,
long lXCenPos,
long lYCenPos,
long lXEndPos,
long lYEndPos,
long lMDP
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- bFLS

  선속 일정을 사용하려면 FPMC_ON(1)을 사용하지 않으려면 FPMC_OFF(0)을 입력합니다.

- lXCenPos

  X 축 중점 좌표를 입력합니다. 이때 유효한 범위는 -8,388,608~8,388,607 입니다.

- lYCenPos

  Y 축 중점 좌표를 입력합니다. 이때 유효한 범위는 -8,388,608~8,388,607 입니다.

- lXEndPos

  X 축 종점 좌표를 입력합니다. 이때 유효한 범위는 -8,388,608~8,388,607 입니다.

- lYEndPos

  Y 축 종점 좌표를 입력합니다. 이때 유효한 범위는 -8,388,608~8,388,607 입니다.

- lMDP

  매뉴얼 감속점을 입력합니다. 이때 유효한 범위는 0~268,435,455 입니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력 에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;
 long ManualDecelPoint; //ManualDecelPoint
 int Range; // Speed Multiplier
 long Acceleration; // Acceleration rate
 long Deceleration; // Deceleration rate
 long StVelocity; // Start speed
 long DrVelocity; //Drive speed
 long CtXaxis; //X axis Center position
 long CtYaxis; //Y axis Center position
 long EndXaxis; //X axis End position
 long EndYaxis; //Y axis End position
 BOOL CCW = 0; //Direction
 BOOL bDecValue; //Fiexd Line Speed;

 Range = 10;
 Acceleration = 1000;
 Deceleration = 1000;
 StVelocity = 500;
 DrVelocity = 1000;
 CtXaxis = 1000;
 CtYaxis = 1000;
 EndXaxis = 3000;
 EndYaxis = 3000;
 bDecValue = 1; //Enable;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);

 ManualDecelPoint = CalculateManualDecelPoint(Range, Acceleration, Deceleration,
StVelocity, DrVelocity, CtXaxis, CtYaxis, EndXaxis, EndYaxis, CCW, bDecValue); //매뉴얼
감속점 계산

 printf("매뉴얼 감속점 : %ld\n", ManualDecelPoint);

 Flag = autpmc_RIDMove(PORTNUM, Node01, bDecValue, CtXaxis, CtYaxis, EndXaxis,
EndYaxis, ManualDecelPoint);

 if(Flag!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }



 autpmc_Close(PORTNUM);
}
```

# 6 프로그램 제어

## 6.1 autpmc_PgmRun

autpmc_PgmRun 함수는 PMC-2HSP/2HSN 을 브로드캐스트 기능으로 프로그램 모드를 실행합니다.

**(1) 함수명**

```
int autpmc_PgmRun(
int PortNum,
char nNodeId,
char axis,
int iStepNo
);
```

**(2) 파라미터**

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE 를 리턴합니다. 또한, Broadcast (0x80)을 입력하면 브로드캐스트 기능으로 현재 PC 와 연결된 모든 PMC-2HSP/2HSN 에 데이터를 전송합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- iStepNo

  프로그램 시작 실행 번지를 입력합니다. 이때 유효한 범위는 0~199 입니다.

**(3) 리턴값**

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 autpmc_PgmABS (PORTNUM, Node01, FPMC_X_Y_AXIS, 0, 10000, 1, 0, 0, 0);
//프로그램 절대 위치 운전

 Flag = autpmc_PgmRun(PORTNUM, Broadcast, FPMC_X_Y_AXIS, 0); //프로그램 모드
실행(브로드캐스트)

 if(Flag!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }


 autpmc_Close(PORTNUM);
}
```

## 6.2 autpmc_PgmStepRun

autpmc_PgmStepRun 함수는 PMC-2HSP/2HSN 에서 프로그램 모드에서 한 스텝만 실행합니다.

### (1) 함수명

```
int autpmc_PgmStepRun(
int PortNum,
char nNodeId,
char axis,
int iStepNo
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE 을 리턴합니다. 또한, Broadcast (0x80)을 입력하면 브로드캐스트 기능으로 현재 PC 와 연결된 모든 PMC-2HSP/2HSN 에 데이터를 전송합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- iStepNo

  한 스텝만 실행할 프로그램 시작 번지를 입력합니다. 이때 유효한 범위는 0~199 입니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"
```

```
#define PORTNUM 3

void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 autpmc_PgmABS (PORTNUM, Node01, FPMC_X_AXIS, 0, 10000, 1, 0, 0, 0); //프로그램
절대 위치 운전

 autpmc_PgmINC (PORTNUM, Node01, FPMC_X_AXIS, 1, 1000, 1, 0, 0, 0); //프로그램
상대 위치 운전

 Flag = autpmc_PgmStepRun(PORTNUM, Node01, FPMC_X_AXIS, 0); // 시작 번지로부터
한 스텝만 실행

 if(Flag!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }


 autpmc_Close(PORTNUM);
}
```

## 6.3 autpmc_PgmPause

autpmc_PgmPause 함수는 PMC-2HSP/2HSN 을 브로드캐스트 기능으로 프로그램 모드를 일시 정지합니다.

**(1) 함수명**

```
int autpmc_PgmPause(
int PortNum,
char nNodeId,
char axis
);
```

**(2) 파라미터**

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE 을 리턴합니다. 또한, Broadcast (0x80)을 입력하면 브로드캐스트 기능으로 현재 PC 와 연결된 모든 PMC-2HSP/2HSN 에 데이터를 전송합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

**(3) 리턴값**

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |

**(4) 사용 예**

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;
```

```
 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 autpmc_PgmABS (PORTNUM, Node01, FPMC_X_Y_AXIS, 0, 10000, 1, 0, 0, 0);
//프로그램 절대 위치 운전

 autpmc_PgmRun(PORTNUM, Node01, FPMC_X_Y_AXIS, 0); //프로그램 모드 실행

 Flag = autpmc_PgmPause(PORTNUM, Node01, FPMC_X_Y_AXIS); //프로그램 모드 일시
정지

 if(Flag!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }


 autpmc_Close(PORTNUM);
}
```

## 6.4 autpmc_PgmReRun

autpmc_PgmReRun 함수는 PMC-2HSP/2HSN 을 브로드캐스트 기능으로 프로그램 모드를 재실행하는 함수로서 일시 정지에서 정지된 스텝부터 다시 시작합니다.

### (1) 함수명

```
int autpmc_PgmReRun(
int PortNum,
char nNodeId,
char axis
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE 을 리턴합니다. 또한, Broadcast (0x80)을 입력하면 브로드캐스트 기능으로 현재 PC 와 연결된 모든 PMC-2HSP/2HSN 에 데이터를 전송합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;
```

```
 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 autpmc_PgmABS (PORTNUM, Node01, FPMC_X_Y_AXIS, 0, 10000, 1, 0, 0, 0);
//프로그램 절대 위치 운전

 autpmc_PgmINC (PORTNUM, Node01, FPMC_X_Y_AXIS, 0, 5000, 1, 0, 0, 0); //프로그램
상대 위치 운전

 autpmc_PgmRun(PORTNUM, Node01, FPMC_X_Y_AXIS, 0); //프로그램 모드 실행

 autpmc_PgmPause(PORTNUM, Node01, FPMC_X_Y_AXIS); //프로그램 모드 일시 정지

 Flag = autpmc_PgmReRun (PORTNUM, Node01, FPMC_X_Y_AXIS); //프로그램 재 실행

 if(Flag!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }


 autpmc_Close(PORTNUM);
}
```

## 6.5 autpmc_PgmStop

autpmc_PgmStop 함수는 PMC-2HSP/2HSN 을 브로드캐스트 기능으로 프로그램 모드를 강제 종료합니다.

### (1) 함수명

```
int autpmc_PgmStop(
int PortNum,
char nNodeId,
char axis
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE 를 리턴합니다. 또한, Broadcast (0x80)을 입력하면 브로드캐스트 기능으로 현재 PC 와 연결된 모든 PMC-2HSP/2HSN 에 데이터를 전송합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;
```

```
 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 autpmc_PgmABS (PORTNUM, Node01, FPMC_X_Y_AXIS, 0, 10000, 1, 0, 0, 0);
//프로그램 절대 위치 운전

 autpmc_PgmRun(PORTNUM, Node01, FPMC_X_Y_AXIS, 0); //프로그램 모드 실행

 Flag = autpmc_PgmStop (PORTNUM, Node01, FPMC_X_Y_AXIS); //프로그램 종료

 if(Flag!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }


 autpmc_Close(PORTNUM);
}
```

## 6.6 autpmc_DelPgmData

autpmc_DelPgmData 함수는 PMC-2HSP/2HSN 의 프로그램 모드 내에서 지정한 스텝의 데이터를 삭제합니다.

### (1) 함수명

```
int autpmc_DelPgmData(
int PortNum,
char nNodeId,
char axis,
int nStepNo
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- nStepNo

  프로그램 스텝 번호를 입력합니다. 이때 유효한 범위는 0~199 입니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3
```

```
void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_DelPgmData (PORTNUM,Node01, FPMC_X_Y_AXIS, 0)

 if(Flag!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }


 autpmc_Close(PORTNUM);
}
```

## 6.7 autpmc_DelPgmDataAll

autpmc_DelPgmDataAll 함수는 PMC-2HSP/2HSN 의 프로그램 모드 내에서 모든 스텝의 데이터를 삭제합니다.

### (1) 함수명

```
int autpmc_DelPgmDataAll(
int PortNum,
char nNodeId,
char axis
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;
```

```
 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_DelPgmDataAll (PORTNUM, Node01, FPMC_X_Y_AXIS);

 if(Flag!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }


 autpmc_Close(PORTNUM);
}
```

## 6.8 autpmc_PgmABS

autpmc_PgmABS 함수는 PMC-2HSP/2HSN 의 프로그램 모드 내에서 원점을 기준으로 지정된 거리를 절대 위치로 이동합니다.

**(1) 함수명**

```
int autpmc_PgmABS(
int PortNum,
char nNodeId,
char axis,
int nStepNo,
long lPos,
int nSpeed,
int nTimer,
BOOL bEndP,
BOOL bBoth
);
```

**(2) 파라미터**

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- nStepNo

  프로그램 스텝 번호를 입력합니다. 이때 유효한 범위는 0~199 입니다.

- lPos

  절대 위치 좌표 (-8,388,608~8,388,607)

- nSpeed

  속도 인덱스(1~4)

- nTimer

  포스트 타이머 인덱스(1~3)

- bEndP

  드라이브 종료 펄스 유무(0~1)

- bBoth

  X,Y 축 동시 동작 유무(0~1)

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_PgmABS (PORTNUM, Node01, FPMC_X_AXIS, 0, 10000, 1, 0, 0, 0);

 if(Flag!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }


 autpmc_Close(PORTNUM);
}
```

## 6.9 autpmc_PgmINC

autpmc_PgmINC 함수는 PMC-2HSP/2HSN 의 프로그램 모드 내에서 현재 위치를 기준으로 지정된 거리를 상대 위치로 이동합니다.

### (1) 함수명

```
int autpmc_PgmINC(
int PortNum,
char nNodeId,
char axis,
int nStepNo,
long lPos,
int nSpeed,
int nTimer,
BOOL bEndP,
BOOL bBoth
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- nStepNo

  프로그램 스텝 번호를 입력합니다. 이때 유효한 범위는 0~199 입니다.

- lPos

  절대 위치 좌표(-8,388,608~8,388,607)

- nSpeed

  속도 인덱스(1~4)

- nTimer

  포스트 타이머 인덱스(1~3)

- bEndP

  드라이브 종료 펄스 유무(0~1)

- bBoth

  X,Y 축 동시 동작 유무(0~1)

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_PgmINC (PORTNUM, Node01, FPMC_X_AXIS, 0, 10000, 1, 0, 0, 0);

 if(Flag!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }


 autpmc_Close(PORTNUM);
}
```

## 6.10 autpmc_PgmHOM

autpmc_PgmHOM 함수는 PMC-2HSP/2HSN 의 프로그램 모드 내에서 원점 복귀 모드에서 설정되어 있는 순서에 따라 원점 복귀를 실행합니다.

### (1) 함수명

```
int autpmc_PgmHOM(
int PortNum,
char nNodeId,
char axis,
int nStepNo,
BOOL bEndP,
BOOL bBoth
) ;
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- nStepNo

  프로그램 스텝 번호를 입력합니다. 이때 유효한 범위는 0~199 입니다.

- bEndP

  드라이브 종료 펄스 유무(0~1)

- bBoth

  X,Y 축 동시 동작 유무(0~1)

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력 에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_PgmHOM (PORTNUM, Node01, FPMC_X_Y_AXIS, 0, 1, 1);

 if(Flag!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }


 autpmc_Close(PORTNUM);
}
```

## 6.11 autpmc_PgmLID

autpmc_PgmLID 함수는 PMC-2HSP 의 프로그램 모드 내에서 현재 좌표로부터 종점 좌표를 향해 2 축 직선 보간을 실행합니다.

### (1) 함수명

```
int autpmc_PgmLID(
int PortNum,
char nNodeId,
int nStepNo,
long lXEndPos,
long lYEndPos,
BOOL bFLS,
int nSpeed,
int nTimer,
BOOL bEndP
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- nStepNo

  프로그램 스텝 번호를 입력합니다. 이때 유효한 범위는 0~199 이며, 총 2 스텝이 할당됩니다.

- lXEndPos

  X 축 종점 좌표(-8,388,608~8,388,607)

- lYEndPos

  Y 축 종점 좌표(-8,388,608~8,388,607)

- bFLS

  선속 일정 사용 유무(0~1)

- nSpeed

  속도 인덱스(1~4)

- nTimer

  포스트 타이머 인덱스(1~3)

- bEndP

  드라이브 종료 펄스 유무(0~1)

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력 에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_PgmLID (PORTNUM, Node01, 0, 1000, 1000, 0, 1, 0, 0);

 if(Flag!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }


 autpmc_Close(PORTNUM);
}
```

## 6.12 autpmc_PgmCID

autpmc_PgmCID 함수는 PMC-2HSP 의 프로그램 모드 내에서 X, Y 축의 CW 방향(시계 방향)으로 원 보간 드라이브를 실행합니다.

### (1) 함수명

```
int autpmc_PgmCID(
int PortNum,
char nNodeId,
int nStepNo,
long lRadius,
BOOL bFLS,
int nSpeed,
int nTimer,
BOOL bEndP
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- nStepNo

  프로그램 스텝 번호를 입력합니다. 이때 유효한 범위는 0~199 이며 총 2 스텝이 할당됩니다.

- lRadius

  반지름 좌표 0~8,388,607

- bFLS

  선속 일정 사용 유무(0: 사용안함, 1: 사용)

- nSpeed

  속도 인덱스(1~4)

- nTimer

  포스트 타이머 인덱스(1~3)

- bEndP

  드라이브 종료 펄스 유무(0: 사용안함, 1: 사용)

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력 에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_PgmCID (PORTNUM, Node01, 0, 10000, 1, 1, 1, 1);

 if(Flag!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }


 autpmc_Close(PORTNUM);
}
```

## 6.13 autpmc_PgmFID

autpmc_PgmFID 함수는 PMC-2HSP 의 프로그램 모드 내에서 X, Y 축의 CW 방향(시계 방향)으로 원호 보간 드라이브를 실행합니다.

### (1) 함수명

```
int autpmc_PgmFID(
int PortNum,
char nNodeId,
int nStepNo,
long XCenPos,
long XEndPos,
long YCenPos,
long YEndPos,
BOOL bFLS,
int nSpeed,
int nTimer,
BOOL bEndP
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- nStepNo

  프로그램 스텝 번호를 입력합니다. 이때 유효한 범위는 0~199 이며, 총 2 스텝이 할당됩니다.

- XCenPos

  X 축 중점 좌표 -8,388,608~8,388,607

- XEndPos

  X 축 종점 좌표 -8,388,608~8,388,607

- YCenPos

  Y 축 중점 좌표 -8,388,608~8,388,607

- YEndPos

  Y 축 종점 좌표 -8,388,608~8,388,607

- bFLS

  선속일정 사용 유무(0: 사용안함, 1: 사용)

- nSpeed

  속도 인덱스(1~4)

- nTimer

  포스트 타이머 인덱스(1~3)

- bEndP

  드라이버 종료 펄스 유무(0: 사용안함, 1: 사용)

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_PgmFID (PORTNUM, Node01, 0, 1000, 1000, 1000, 1000, 0, 1, 0, 0);

 if(Flag!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }


 autpmc_Close(PORTNUM);
}
```

## 6.14 autpmc_PgmRID

autpmc_PgmRID 함수는 PMC-2HSP 의 프로그램 모드 내에서 X, Y 축의 CCW 방향(반시계 방향)으로 원호 보간 드라이브를 실행합니다.

### (1) 함수명

```
int autpmc_PgmRID(
int PortNum,
char nNodeId,
int nStepNo,
long XCenPos,
long XEndPos,
long YCenPos,
long YEndPos,
BOOL bFLS,
int nSpeed,
int nTimer,
BOOL bEndP
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- nStepNo

  프로그램 스텝 번호를 입력합니다. 이때 유효한 범위는 0~199 이며, 총 2 스텝이 할당됩니다.

- XCenPos
  X 축 중점 좌표 -8,388,608~8,388,607

- XEndPos
  X 축 종점 좌표 -8,388,608~8,388,607

- YCenPos
  Y 축 중점 좌표 -8,388,608~8,388,607

- YEndPos
  Y 축 종점 좌표 -8,388,608~8,388,607

- bFLS
  선속일정 사용 유무(0: 사용안함, 1: 사용)

- nSpeed
  속도 인덱스(1~4)

- nTimer
  포스트 타이머 인덱스(1~3)

- bEndP
  드라이버 종료 펄스 유무(0: 사용안함, 1: 사용)

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_PgmRID (PORTNUM, Node01, 0, 1000, 1000, 1000, 1000, 0, 1, 0, 0);

 if(Flag!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }


 autpmc_Close(PORTNUM);
}
```

## 6.15 autpmc_PgmICJ

autpmc_PgmICJ 함수는 PMC-2HSP/2HSN 의 프로그램 모드 내에서 선택한 입력 포트가 활성화 상태라면 지정된 스텝으로 점프합니다. 입력 포트가 비활성화 상태라면 바로 다음 스텝을 실행합니다.

**(1) 함수명**

```
int autpmc_PgmICJ(
int PortNum,
char nNodeId,
char axis,
int nStepNo,
int nJumpStep,
int nInputPtNo
);
```

**(2) 파라미터**

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- nStepNo

  프로그램 스텝 번호를 입력합니다. 이때 유효한 범위는 0~199 입니다.

- nJumpStep

  점프 할 스텝 번호(0~199)

- nInputPtNo

  입력 포트 번호(0~14)

**(3) 리턴값**

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력 에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_PgmICJ (PORTNUM, Node01, FPMC_X_AXIS, 0, 10, 0);

 if(Flag!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }


 autpmc_Close(PORTNUM);
}
```

## 6.16 autpmc_PgmIRD

autpmc_PgmIRD 함수는 PMC-2HSP/2HSN 의 프로그램 모드 내에서 선택한 입력 포트가 활성화 상태가 되면 다음 스텝으로 이동합니다. 입력 포트가 비활성화 상태라면 활성화 상태가 될 때까지 현재 스텝에서 대기합니다.

### (1) 함수명

```
int autpmc_PgmIRD(
int PortNum,
char nNodeId,
char axis,
int nStepNo,
int nInputPtNo
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- nStepNo

  프로그램 스텝 번호를 입력합니다. 이때 유효한 범위는 0~199 입니다.

- nInputPtNo

  입력 포트 번호(0~14)

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_PgmIRD (PORTNUM, Node01, FPMC_X_AXIS, 0, 10);

 if(Flag!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }


 autpmc_Close(PORTNUM);
}
```

## 6.17 autpmc_PgmOPC

autpmc_PgmOPC 함수는 PMC-2HSP/2HSN 의 프로그램 모드 내에서 선택한 출력 포트를 ON, OFF 합니다.

**(1) 함수명**

```
int autpmc_PgmOPC(
int PortNum,
char nNodeId,
char axis,
int nStepNo,
int nOutPtNo,
BOOL bON
);
```

**(2) 파라미터**

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- nStepNo

  프로그램 스텝 번호를 입력합니다. 이때 유효한 범위는 0~199 입니다.

- nOutPtNo

  출력 포트 번호(0~3)

- bON

  OFF(0)~ON(1)

**(3) 리턴값**

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력 에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_PgmOPC (PORTNUM, Node01, FPMC_X_Y_AXIS, 0, 3, 1);

 if(Flag!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }


 autpmc_Close(PORTNUM);
}
```

## 6.18 autpmc_PgmOPT

autpmc_PgmOPT 함수는 PMC-2HSP/2HSN 의 프로그램 모드 내에서 선택한 출력 포트를 ON Time 설정 시간 동안 ON 합니다.

**(1) 함수명**

```
int autpmc_PgmOPT(
int PortNum,
char nNodeId,
char axis,
int nStepNo,
int iOnTim,
int nOutPtNo,
BOOL bNextStep
);
```

**(2) 파라미터**

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- nStepNo

  프로그램 스텝 번호를 입력합니다. 이때 유효한 범위는 0~199 입니다.

- iOnTim

  출력 포트를 ON 시키는 시간: 0~65,535ms

- nOutPtNo

  출력 포트 번호(0~3)

- bNextStep

  ON(1) : 출력동작과 관계없이 다음 스텝으로 이동합니다.

  OFF(0) : ON 설정 시간 동안 선택한 출력 포트를 ON 시키고, 시간 종료 시 다음 스텝으로 이동합니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력 에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_PgmOPT (PORTNUM, Node01, FPMC_X_Y_AXIS, 0, 100, 1, 1);

 if(Flag!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }


 autpmc_Close(PORTNUM);
}
```

## 6.19 autpmc_PgmJMP

autpmc_PgmJMP 함수는 PMC-2HSP/2HSN 의 프로그램 모드 내에서 지정된 스텝으로 점프합니다.

### (1) 함수명

```
int autpmc_PgmJMP(
int PortNum,
char nNodeId,
char axis,
int nStepNo,
int nJumpStep
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- nStepNo

  프로그램 스텝 번호를 입력합니다. 이때 유효한 범위는 0~199 입니다.

- nJumpStep

  점프 할 스텝 번호(0~199)

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_PgmJMP (PORTNUM, Node01, FPMC_X_Y_AXIS, 0, 199);

 if(Flag!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }


 autpmc_Close(PORTNUM);
}
```

## 6.20 autpmc_PgmREP

autpmc_PgmREP 함수는 PMC-2HSP/2HSN 의 프로그램 모드 내에서 이 명령의 다음 스텝 부터 autpmc_PgmRPE 함수(반복 종료)를 만날 때까지 지정 횟수만큼 반복 실행합니다.

### (1) 함수명

```
int autpmc_PgmREP(
int PortNum,
char nNodeId,
char axis,
int nStepNo,
int nRepCnt
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- nStepNo

  프로그램 스텝 번호를 입력합니다. 이때 유효한 범위는 0~199 입니다.

- nRepCnt

  반복 횟수(1~255)

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);



 Flag = autpmc_PgmREP(PORTNUM, Node01, FPMC_X_AXIS, 0, 10); //반복 시작

 autpmc_PgmRPE(PORTNUM, Node01, FPMC_X_AXIS, 3); // 반복 종료

 if(Flag!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }



 autpmc_Close(PORTNUM);
}
```

## 6.21 autpmc_PgmRPE

autpmc_PgmREP 함수는 PMC-2HSP/2HSN 의 프로그램 모드 내에서 autpmc_PgmREP(반복시작) 함수의 종료 함수입니다.

### (1) 함수명

```
int autpmc_PgmRPE(
int PortNum,
char nNodeId,
char axis,
int nStepNo
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- nStepNo

  프로그램 스텝 번호를 입력합니다. 이때 유효한 범위는 0~199 입니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3
```

```
void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);

autpmc_PgmREP(PORTNUM, Node01, FPMC_X_AXIS, 0, 10); //반복 시작

Flag = autpmc_PgmRPE(PORTNUM, Node01, FPMC_X_AXIS, 3); // 반복 종료

 if(Flag!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }


 autpmc_Close(PORTNUM);
}
```

## 6.22 autpmc_PgmEND

autpmc_PgmEND 함수는 PMC-2HSP/2HSN 의 프로그램 모드 내에서 프로그램을 종료합니다. 프로그램의 마지막에 반드시 입력해야 합니다.

**(1) 함수명**

```
int autpmc_PgmEND(
int PortNum,
char nNodeId,
char axis,
int nStepNo
);
```

**(2) 파라미터**

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- nStepNo

  프로그램 스텝 번호를 입력합니다. 이때 유효한 범위는 0~199 입니다.

**(3) 리턴값**

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력하였습니다. |

**(4) 사용 예**

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3
```

```
void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_PgmEND(PORTNUM, Node01, FPMC_X_Y_AXIS, 0);

 if(Flag!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }


 autpmc_Close(PORTNUM);
}
```

## 6.23 autpmc_PgmTIM

autpmc_PgmTIM 함수는 PMC-2HSP/2HSN 의 프로그램 모드 내에서 지정 시간만큼 대기 명령을 수행합니다.

### (1) 함수명

```
int autpmc_PgmTIM(
int PortNum,
char nNodeId,
char axis,
int nStepNo,
int nOnTim
);
```

### (2) 파라미터

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- nStepNo

  프로그램 스텝 번호를 입력합니다. 이때 유효한 범위는 0~199 입니다.

- nOnTim

  대기시간을 입력합니다. 이때 유효한 범위는 0~65,535ms 입니다.

### (3) 리턴값

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력하였습니다. |

### (4) 사용 예

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3

void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_PgmTIM (PORTNUM, Node01, FPMC_X_Y_AXIS, 0, 100);

 if(Flag!=FPMC_OK)
 {
        printf("error!\n");
        return;
 }


 autpmc_Close(PORTNUM);
}
```

## 6.24 autpmc_PgmNOP

autpmc_PgmNOP 함수는 PMC-2HSP/2HSN 의 프로그램 모드 내에서 아무 것도 처리하지 않습니다.

**(1) 함수명**

```
int autpmc_PgmNOP(
int PortNum,
char nNodeId,
char axis,
int nStepNo
);
```

**(2) 파라미터**

- PortNum

  명령어를 수행 할 Serial Port 를 입력합니다.

- nNodeId

  설정 할 노드 ID 를 선택합니다. 노드 ID 의 범위는 1~16 까지이며, ID 의 범위를 벗어났을 경우에는 FPMC_INVALID_NODE(3)을 리턴합니다.

- axis

  제어 할 축(X 축, Y 축)을 선택합니다. 잘못된 축을 입력하였을 경우 FPMC_INVALID_AXIS(4)를 리턴합니다.

| 구분 | 정의 | 내용 | 상수값 |
|---|---|---|---|
| PMC_AXIS | FPMC_X_AXIS | X 축 | 0 |
| | FPMC_Y_AXIS | Y 축 | 1 |
| | FPMC_X_Y_AXIS | X, Y 축 | 2 |

- nStepNo

  프로그램 스텝 번호를 입력합니다. 이때 유효한 범위는 0~199 입니다.

**(3) 리턴값**

| 구분 | 정의 | 리턴값 | 내용 |
|---|---|---|---|
| 정상 | FPMC_OK | 0 | 함수가 정상적으로 명령을 수행하였습니다. |
| 입력에러 | FPMC_INVALID_NODE | 3 | 잘못된 노드 ID 를 입력하였습니다. |
| | FPMC_INVALID_AXIS | 4 | 잘못된 축을 입력하였습니다. |
| | FPMC_INVALID_DATA | 5 | 잘못된 데이터를 입력하였습니다. |

**(4) 사용 예**

```
#include <stdio.h>
#include <windows.h>
#include "Library.h"

#define PORTNUM 3
```

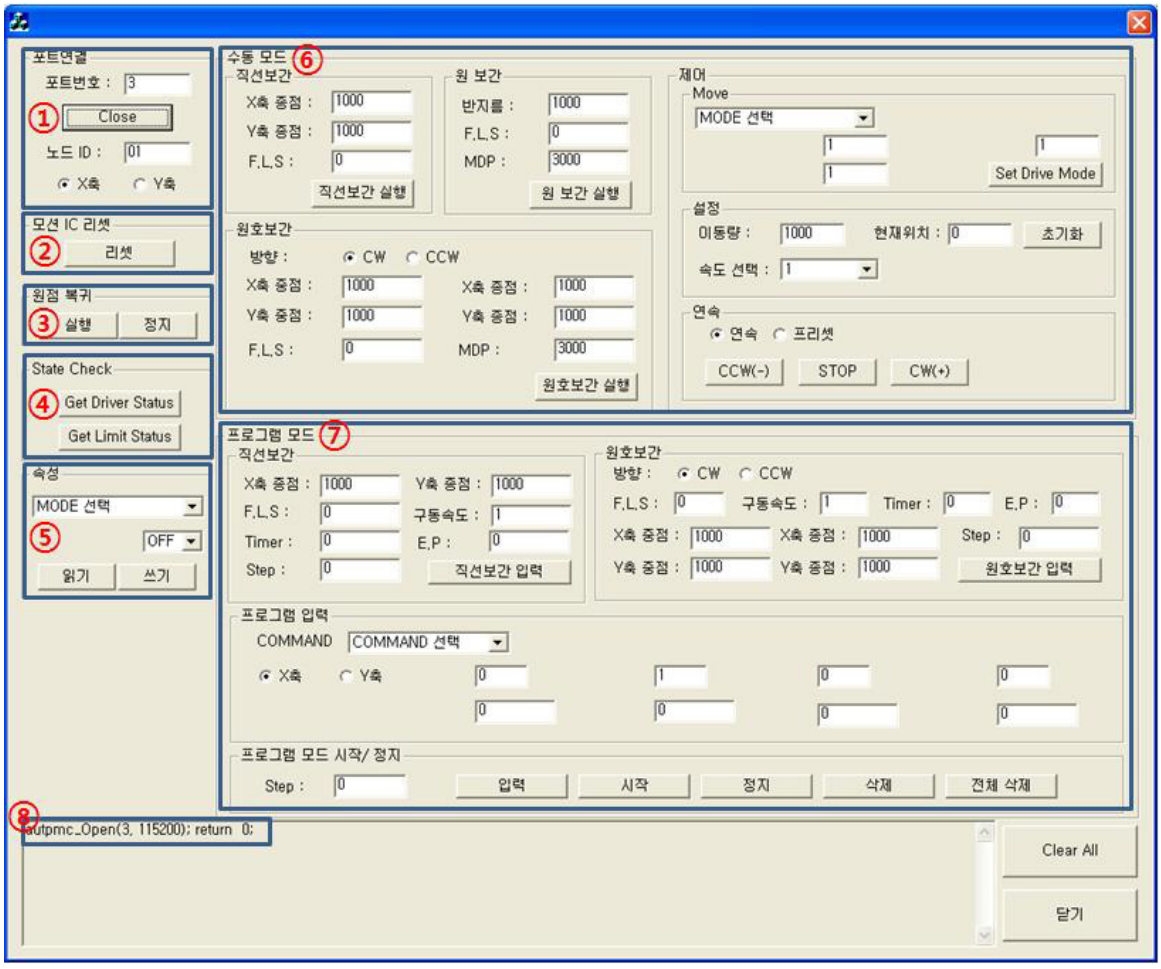
 

```
void main()
{
 int Flag=0;

 autpmc_Open(PORTNUM, FPMC_BAUD_115200);


 Flag = autpmc_PgmNOP(PORTNUM, Node01, FPMC_X_Y_AXIS, 190);

 if(Flag!=FPMC_OK)
 {
            printf("error!\n");
            return;
 }


 autpmc_Close(PORTNUM);
}
```

# 7 라이브러리로 활용 가능한 MFC 예제 프로그램

## (1) MFC 예제 프로그램

라이브러리 함수를 활용한 MFC 예제 프로그램입니다. 소스를 직접 확인 할 수 있으므로, 필요한 동작을 쉽게 이해하고 사용 할 수 있습니다.




## (2) 설명

| 번호 | 구분 | 내용 |
|---|---|---|
| ① | 포트연결 | 포트 연결 라이브러리 함수를 호출하여 통신 연결 및 해제를 수행합니다. |
| ② | 모션 IC 리셋 | 현재 연결되어 있는 PMC 모션 IC 를 리셋합니다. |
| ③ | 원점 복귀 | 원점 복귀 라이브러리 함수를 호출하여 원점 복귀 실행 또는 정지합니다. |
| ④ | 드라이버 상태 체크 | 현재 연결되어 있는 PMC 모션 드라이버의 상태나 오류를 확인합니다.<br>(각 축의 원점 복귀모드, 조그 모드, 프로그램 모드가 구동하고 있는지 또는 오류 확인이 가능합니다.) |

| 번호 | 구분 | 내용 |
|---|---|---|
| ⑤ | 파라미터 속성 읽기/ 쓰기 | 현재 설정되어 있는 PMC 모션 드라이버의 파라미터값을 읽거나 쓸 수 있습니다. |
| ⑥ | 수동 모드 | 통신을 통하여 패킷을 보내어 하나의 동작을 수행하는 모드입니다. |
| ⑦ | 프로그램 모드 | EEPROM에 미리 프로그램 스텝을 저장해 놓고, 그 후 통신이 불가능한 환경에서 사용하거나 연속적인 동작을 수행하기 위한 모드입니다. |
| ⑧ | 로그 | 실행한 기능에 대한 라이브러리 함수명과 세부 속성값을 확인 할 수 있습니다. |

ISO-9001

**Autonics**
Sensors & Controllers

www.autonics.co.kr

## Distributor

■ 주요생산품목

포토센서 · 광화이버센서 · 도어센서 · 도어사이드센서 · 에리어센서 · 근접센서 · 압력센서 · 로터리 엔코더 · 커넥터/소켓 · 온도조절기 · 온/습도 센서 · SSR/전력조정기 · 카운터 · 타이머 · 판넬메타 · 타코/스피드/펄스메타 · 디스플레이 유닛 · 센서 컨트롤러 · 스위칭 모드 파워 서플라이 · 제어용 스위치/램프/부저 · I/O 단자대/케이블 · 스테핑 모터/드라이버/컨트롤러 · 그래픽/로직 패널 · 필드 네트워크 기기 · 레이저 마킹 시스템($CO_2$, Nd:YAG) · 레이저 웰딩/솔더링 시스템


## Autonics Corporation

■ 본사
TEL : 051-519-3151 / FAX : 051-519-4432
■ 서울사무소
TEL : 032-610-2700 / FAX : 032-323-3008
■ 대구사무소
TEL : 053-383-7673 / FAX : 053-383-7674
■ 광주
TEL : 062-521-6716~7, 010-9277-3023 / FAX : 062-521-6717
■ 기술 상담 센터
제품기술상담 : 1588-2333 (전국)
A / S 상담 : 080-529-3333 (수도권/충청/강원)
080-519-3333 (영남/호남/제주)




MTC-PMC2HSL1-V1.0-1503KO