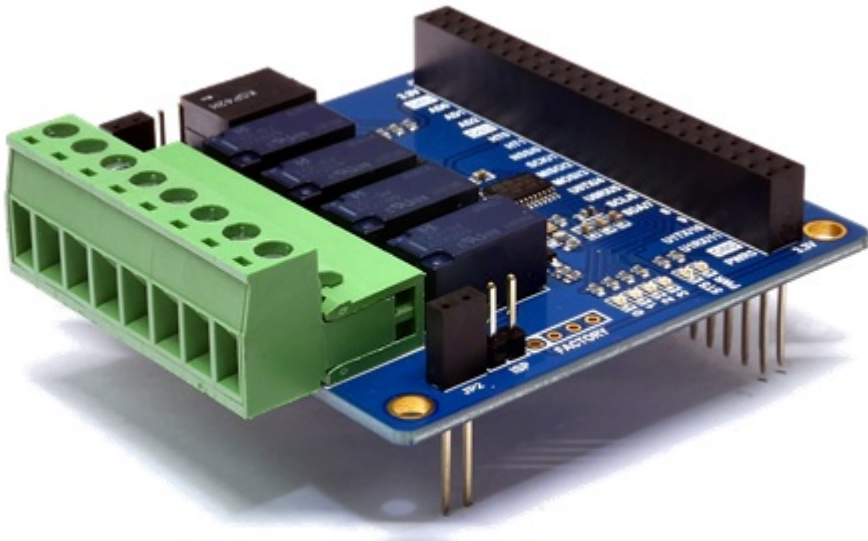


# Introduction



## PES-2401

PES-2401, 4-Port Relay Output board, is one of smart expansion boards for PHPoC boards. You can turn some devices on or off by using this board.

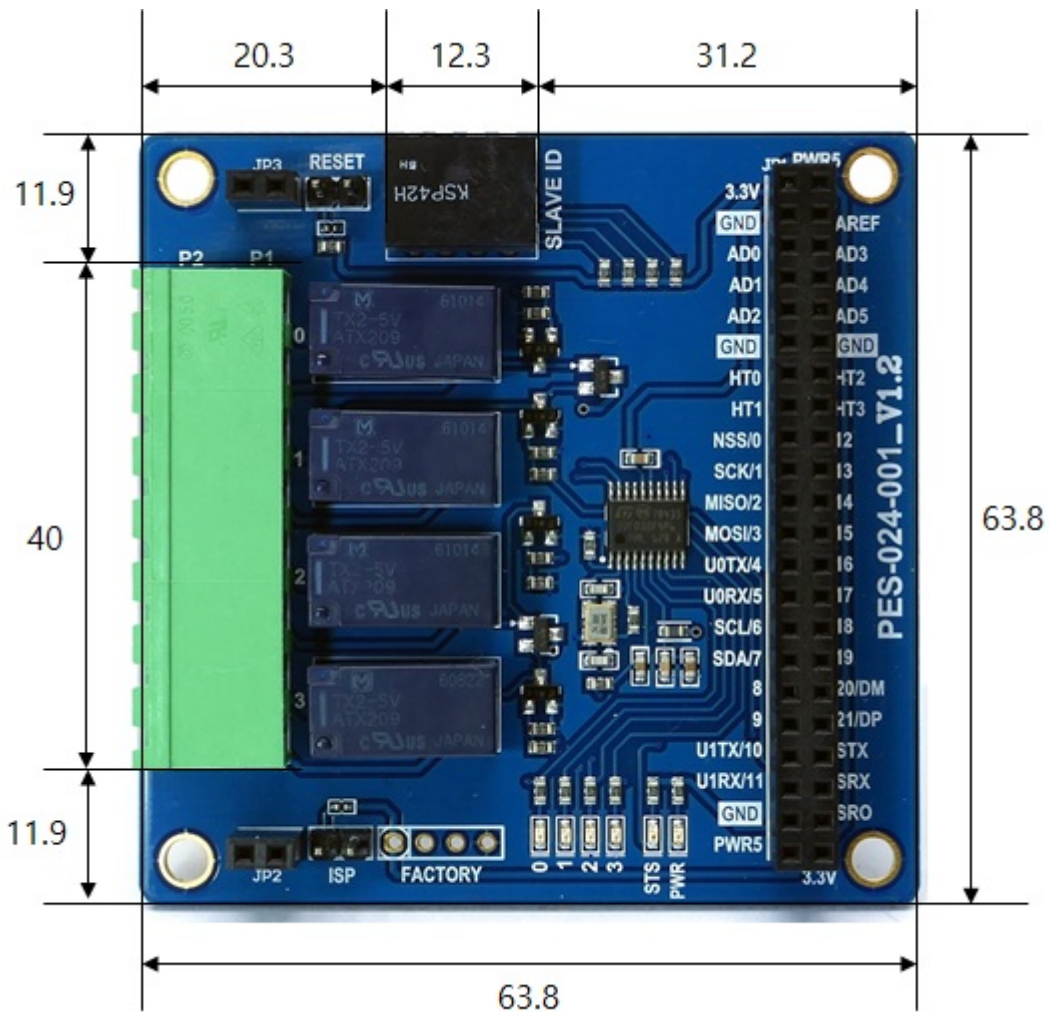
※ Caution: A PHPoC Board is required to use this PES-2401 board!

### What is the Smart Expansion Board?

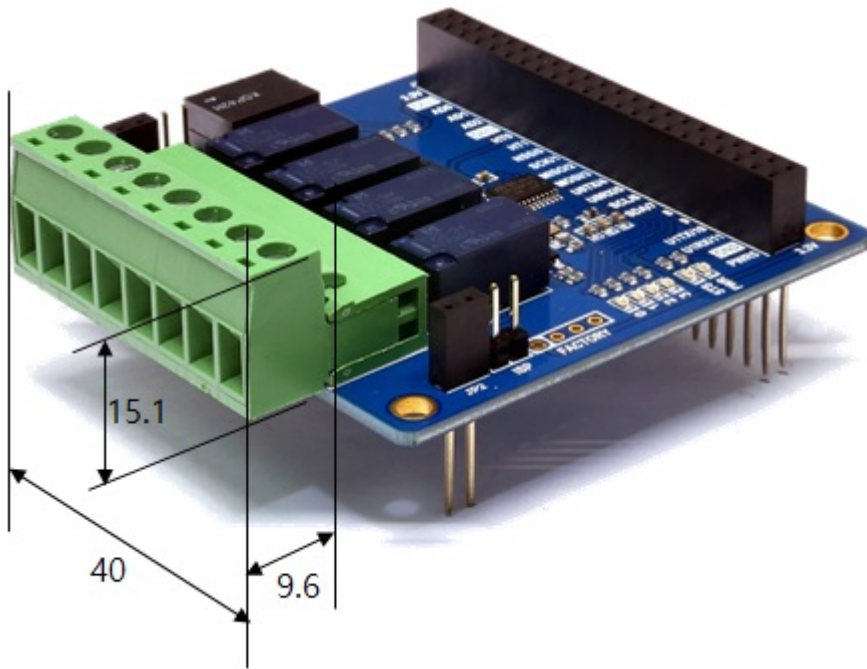
A smart expansion board has own devices and firmware unlike the other expansion boards. This board communicate in a master-slave protocol through the designated port. Two or more smart expansion boards can be connected to one PHPoC board and each of them required to be setting a slave id.

# Dimension

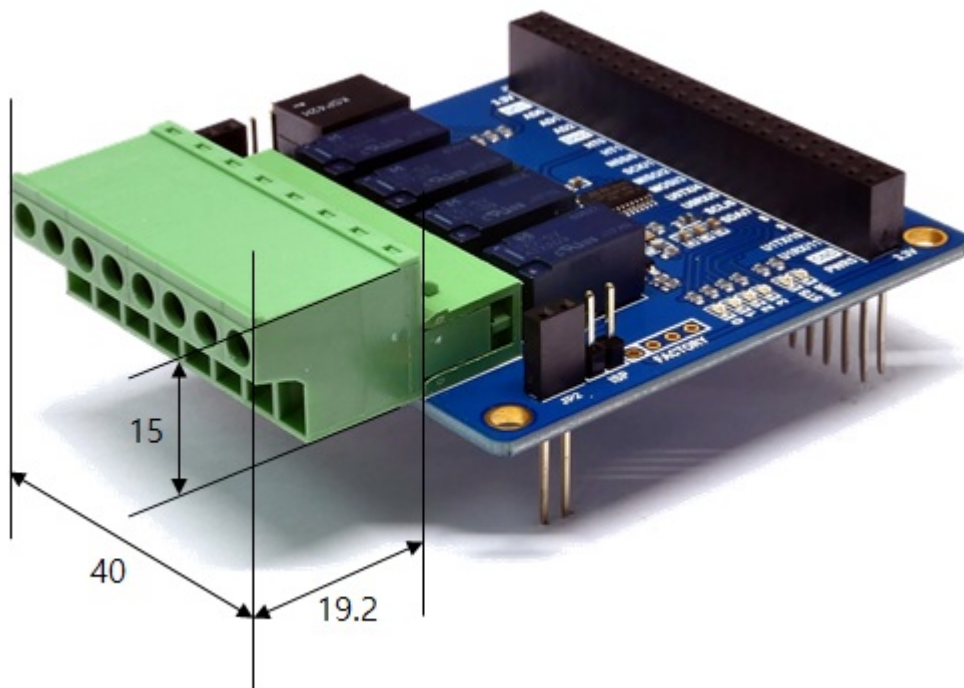
## Body



with Terminal Block (T type)

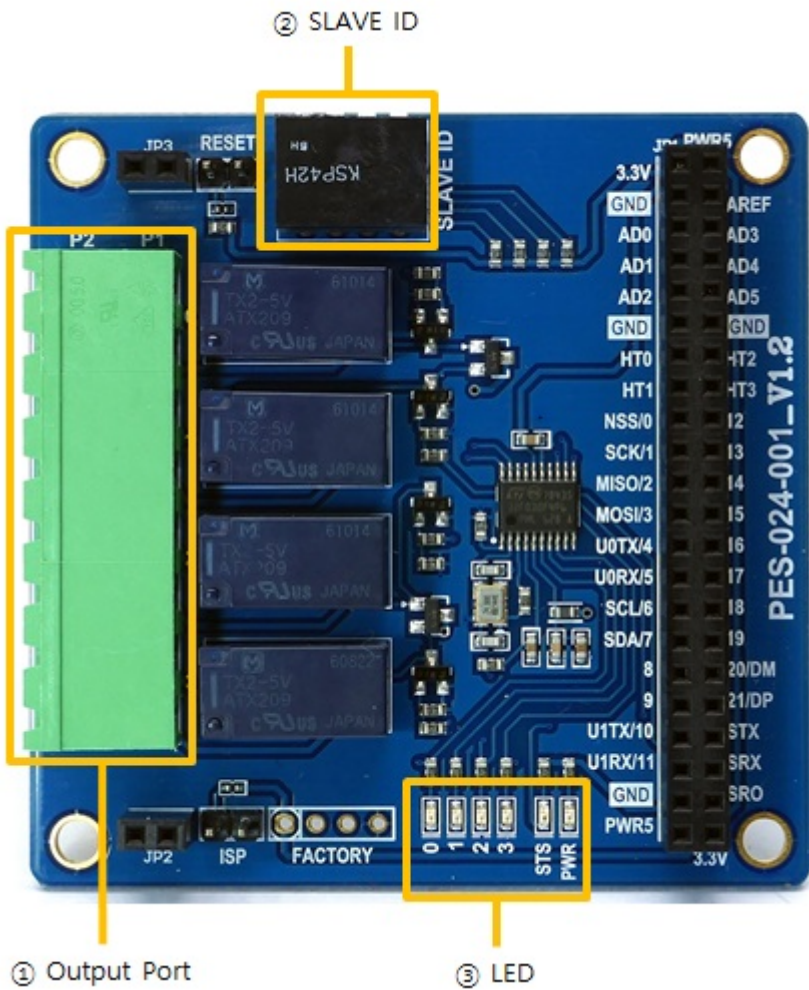


with Terminal Block (S type)



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

# Layout



## 1. Output Ports

Output ports are interfaced with a 5mm spaced terminal block which has 8 terminals. Every output port is connected to a relay which is NO (Normal Open) type.

※ Normal Open: This means the default state of output port is OFF.

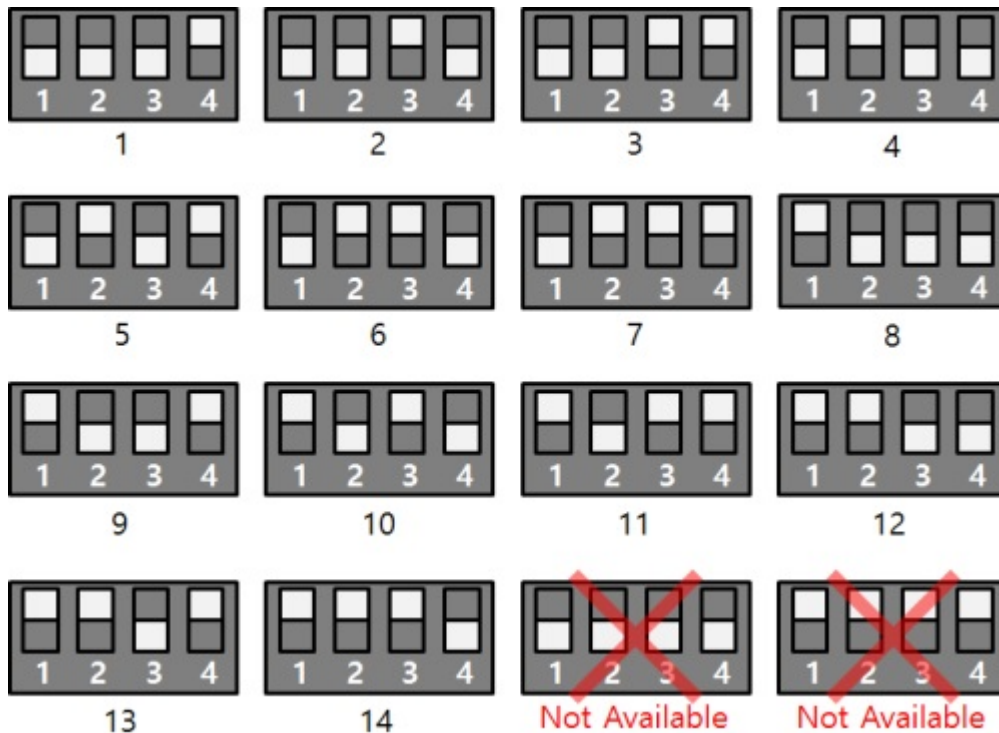
Output ports' range of use is as follows:

Voltage (DC)	Max. Permissible Current
30V	2A

※ Caution: It may result in product malfunction to use beyond the maximum permissible current. Be sure to use it considering the peak current of a connected device.

## 2. SLAVE ID Switch

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



## 3. LED

PES-2401 has 6 on-board LEDs.

LED	Description
PWR	turned ON with stable power supply
STS	setting a valid ID > repeat On/Off in every second setting an invalid ID > blinks fast without communication with PHPoC > Off
0	turned ON with output 0 is ON
1	turned ON with output 1 is ON
2	turned ON with output 2 is ON
3	turned ON with output 3 is ON

---

# How to Use

---

PES-2401 can be used by steps as follows.

## 1. Connect to a PHPoC board

It is not possible to use PES-2401 alone. Please be sure that connection to a PHPoC board is required.

## 2. Install Software (IDE)

PHPoC Debugger is a software which is used for configuring PHPoC products and developing PHPoC script. It is required to install this software on your PC because PES-2401 must be controlled by PHPoC.

- [Download PHPoC Debugger](#)
- [PHPoC Manual Page](#)

## 3. Use SPC Library and Sample Codes

The SPC library is for smart expansion boards such as PES-2401. This library makes it easy for you to use smart expansion boards. Refer to the manual page of SPC library for more information.

- [SPC Library Manual Page](#)

# Commands and Responses

You can use `spc_request` or `spc_request_csv` function when setting or using a smart expansion board.

## Common Commands of Smart Expansion Boards

A common command list of `spc_request` function for all smart expansion boards is as follows:

Command	Option	Description
get	did	get a device ID
get	uid	get a unique ID

## PES-2401 Commands

A command list of `spc_request` function only for PES-2401 is as follows:

Command	Option	Description
set	\$port output \$level	turn a specified port ON(high) or OFF(low)
set	\$port delay \$time	set a delay on a specified port
get	\$port output	get status of a specified port

- \$port : an output port(0 ~ 3)
- \$level : signal level to output(high or low)
- \$time : delay time(unit : millisecond)

## Response of PES-2401

### 1. Response Codes

Response Code	Description
200	command ok
300	unknown command
301	invalid argument

### 2. Response of `spc_request`

A response of `spc_request` from smart expansion boards is a string in CSV(comma-separated values) format.

e.g. "200,0,1,..."

#### ※ Structure of Response Frame(String)

Name	Size	Example(ASCII)
response code	3 bytes	200
comma	1 byte	,
value1	variable	0
comma	1 byte	,
value2	variable	1
...	...	...

### 3. Response of spc\_request\_csv

A response of spc\_request\_csv from smart expansion boards is an array.

e.g. array(200,0,1,...)

#### ※ Structure of Response Frame(Array)

Name	Index #	Exmample
response code	0	200
value1	1	0
value2	2	1
...	...	...



# Controlling Output Ports

## Calling spc\_request for controlling output ports

```
spc_request($sid, 4, $cmd);
```

- \$sid : a slave ID
- \$cmd : a command string

Structure of a command string is as follows:

```
"set $port output $value"
```

- \$port : an index number of an output port, 4 numbers from 0 to 3 are available
- \$value : "high" to turn it on, "low" to turn it off

## Example

```
<?php
include "/lib/sd_spc.php";
spc_reset();
spc_sync_baud(115200);
$sid = 1;

echo "turn all output ports onWrWn";
spc_request($sid, 4, "set 0 output high");
spc_request($sid, 4, "set 1 output high");
spc_request($sid, 4, "set 2 output high");
spc_request($sid, 4, "set 3 output high");

sleep(1);

echo "turn all output ports offWrWn";
spc_request($sid, 4, "set 0 output low");
spc_request($sid, 4, "set 1 output low");
spc_request($sid, 4, "set 2 output low");
spc_request($sid, 4, "set 3 output low");
?>
```

# Monitoring Output Ports

## Calling spc\_request for monitoring output ports

```
spc_request($sid, 4, $cmd);
```

- \$sid : a slave ID
- \$cmd : a command string

Structure of a command string is as follows:

```
"get $port output"
```

- \$port : an index number of an output port, 4 numbers from 0 to 3 are available

## Return Value

A normal response is as follows:

```
"200,$state"
```

- \$state: 0 on LOW, 1 on HIGH

## Example

```
<?php
include "/lib/sd_spc.php";
spc_reset();
spc_sync_baud(115200);
$sid = 1;

echo "turn all output ports onWrWn";
spc_request($sid, 4, "set 0 output high");
spc_request($sid, 4, "set 1 output high");
spc_request($sid, 4, "set 2 output high");
spc_request($sid, 4, "set 3 output high");

// get status of input ports
echo "Port 0: ", spc_request($sid, 4, "get 0 output"), "WrWn";
echo "Port 1: ", spc_request($sid, 4, "get 1 output"), "WrWn";
echo "Port 2: ", spc_request($sid, 4, "get 2 output"), "WrWn";
echo "Port 3: ", spc_request($sid, 4, "get 3 output"), "WrWn";

sleep(1);
```

```
echo "turn all output ports offWrWn";
spc_request($sid, 4, "set 0 output low");
spc_request($sid, 4, "set 1 output low");
spc_request($sid, 4, "set 2 output low");
spc_request($sid, 4, "set 3 output low");

// get status of input ports
echo "Port 0: ", spc_request($sid, 4, "get 0 output"), "WrWn";
echo "Port 1: ", spc_request($sid, 4, "get 1 output"), "WrWn";
echo "Port 2: ", spc_request($sid, 4, "get 2 output"), "WrWn";
echo "Port 3: ", spc_request($sid, 4, "get 3 output"), "WrWn"
?>
```