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.

Highlights of PES-2401

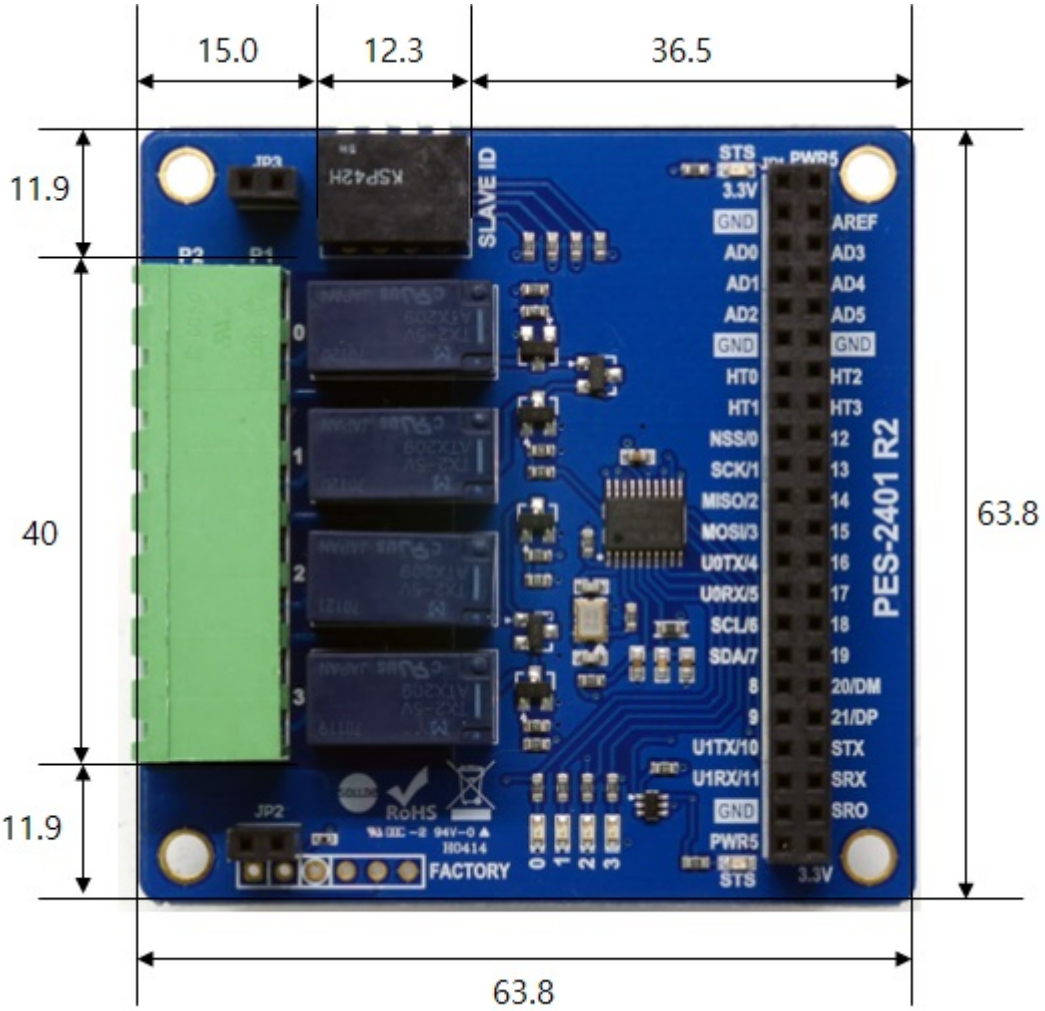
- 4 NO(Normal Open) relays
- Output interface: 8-pole terminal block(S type, T type)
- Output maximum Voltage: DC 30[V]
- Output maximum Current: 2[A]
- Current consumption: approximately 160[mA]

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

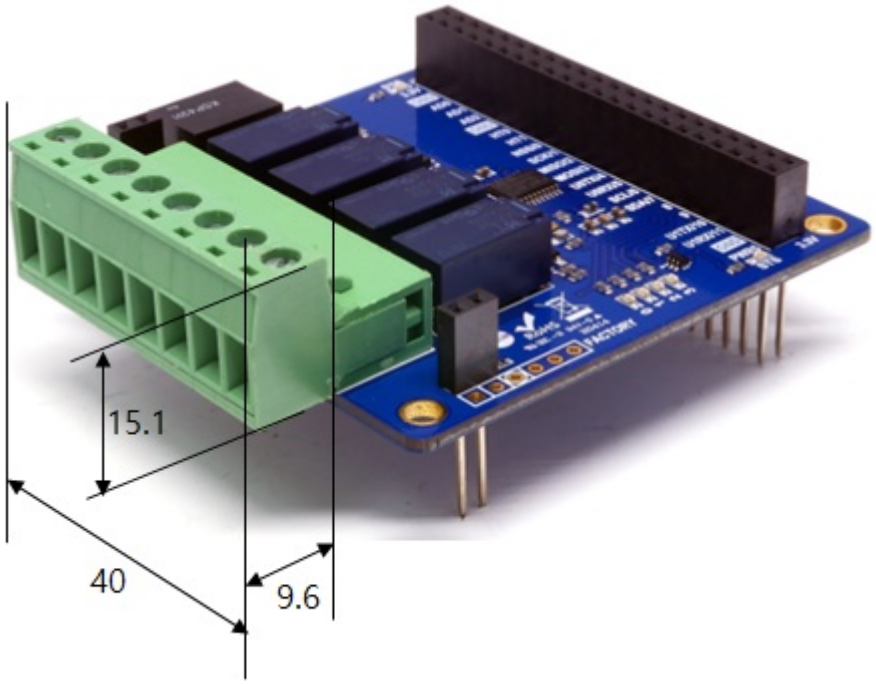
> What is the Smart Expansion Board?
>
> A smart expansion board has own devices and firmware unlike the other expansion boards. This board communicate in a master-slave protocol through the designated port. Two or more smart expansion boards can be connected to one PHPoC board and each of them required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

Dimension

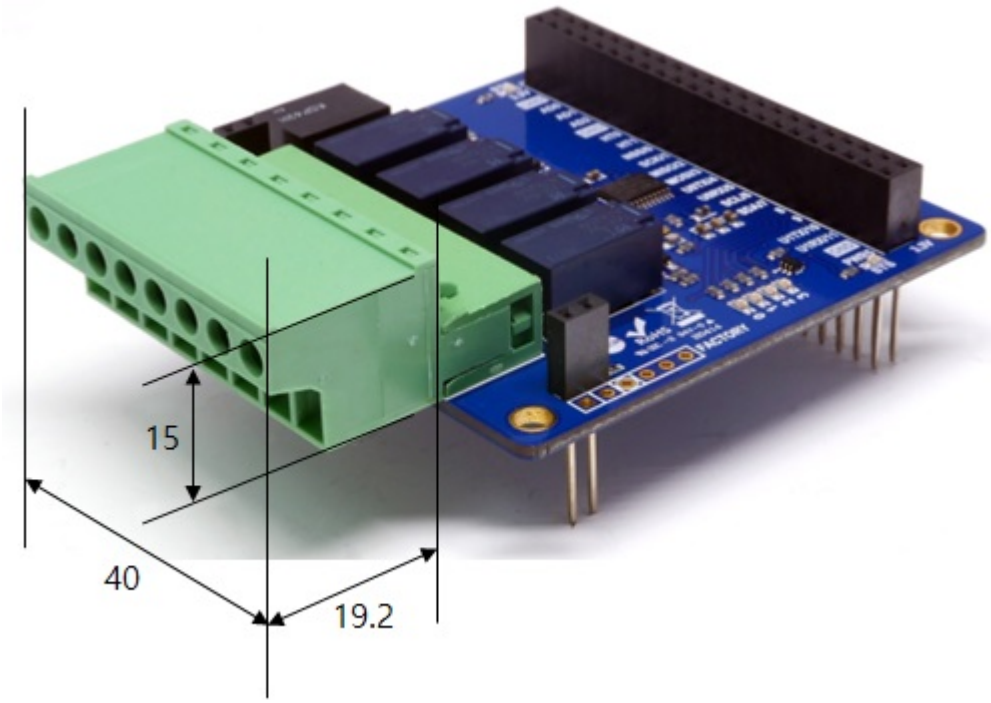
Body



with Terminal Block (T type)



with Terminal Block (S type)



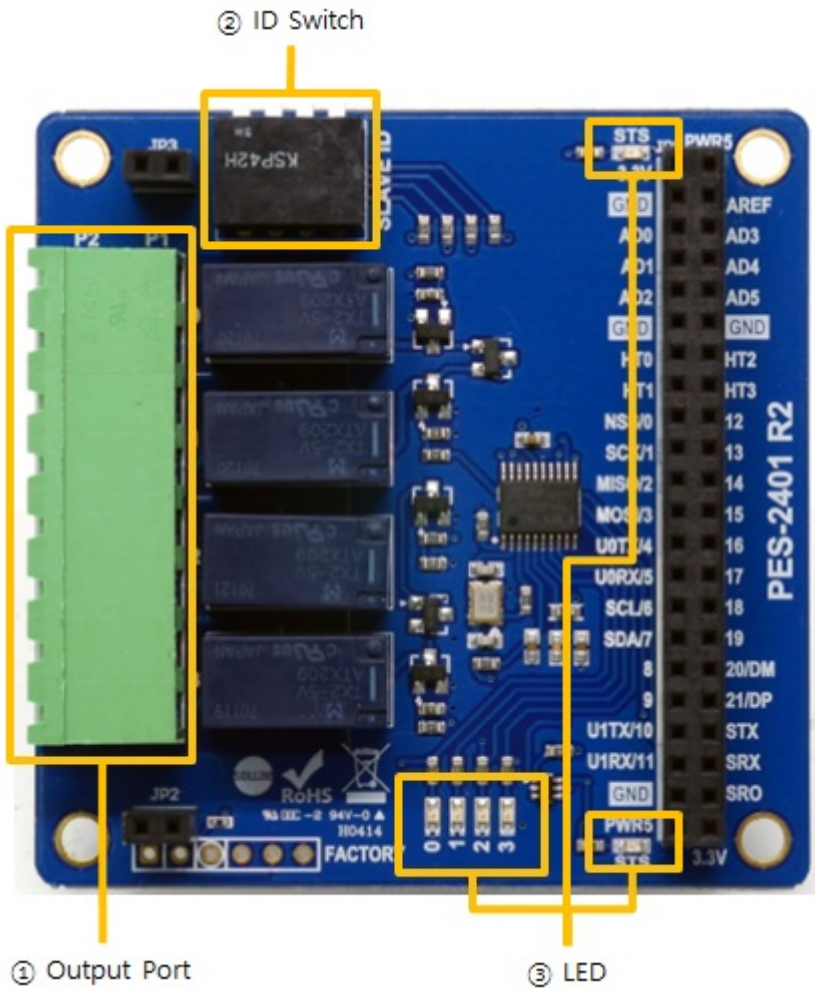
※ Dimensions(unit : mm) may vary according to a method of measurement.

Schematic

This is the schematic of PES-2401.

- PES-2401-R2-PO.pdf

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

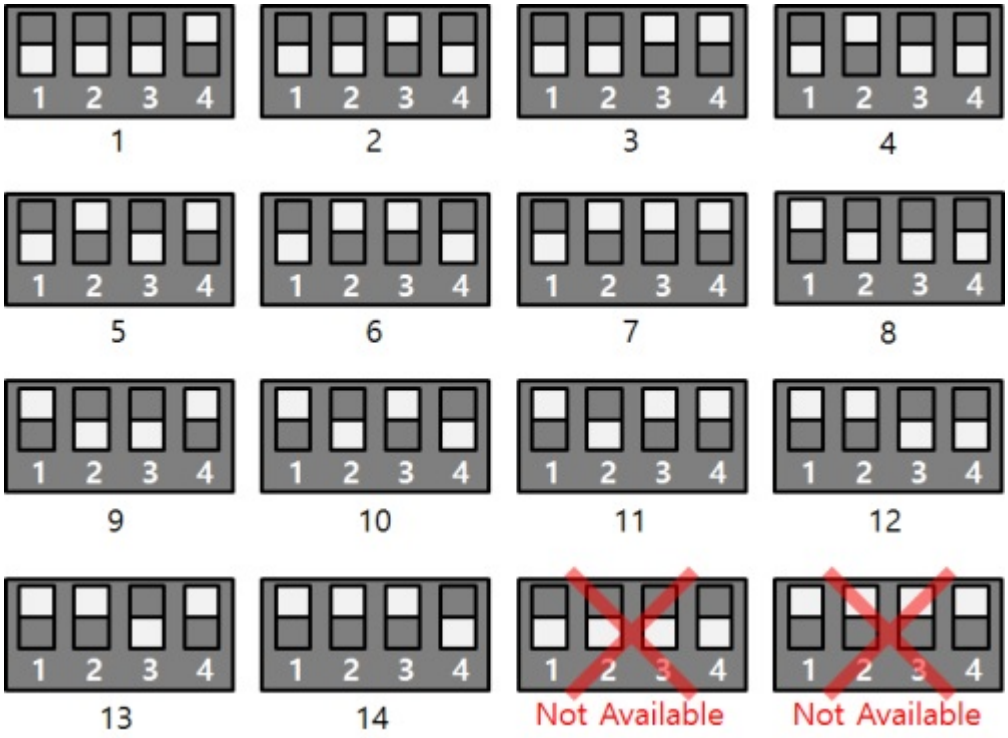
※ Caution: It may result in product malfunction to use beyond the maximum permissible current. Be sure to use it considering the peak current of a connected device.

※ Note: The maximum number of PES-2401 which can be connected to one PHPoC board is 4.

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

STS LED

PES-2401 has two STS LEDs. The one on the top of JP1 is connected to 3.3V and the other one on the bottom is connected to 5V. The operation of the two LEDs is the same as the following.

state	LED operation
normal	repeat on and off every second
invalid slave ID	blinks very quickly
fail to communicate with PHPoC	off

Digital Output Port LED

PES-2401 has 4 LEDs of digital output ports

LED	Description
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 requried to install this software on your PC because PES-2401 must be controlled by PHPoC.

- PHPoC Debugger Download Page
- PHPoC Debugger 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

You can use spc_request_dev or spc_request_sys function of the SPC library when setting or using a smart expansion board.

```
spc_request_dev($sid, $cmd)
spc_request_sys($sid, $cmd)
```

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

Common Commands of Smart Expansion Boards

The commands commonly supported by all smart expansion boards use the spc_request_sys function.
The following is a list of smart expansion board common commands.

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

PES-2401 Commands

The commands that apply to each smart expansion board use the spc_request_dev function.
The following is a list of commands supported by the PES-2401.

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(1 ~ 30,000, unit : millisecond)

Controlling Output Ports

Calling spc_request_dev function for controlling output ports

```
spc_request_dev($sid, $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();
$sid = 1;

echo "turn all output ports on₩r₩n";
spc_request_dev($sid, "set 0 output high");
spc_request_dev($sid, "set 1 output high");
spc_request_dev($sid, "set 2 output high");
spc_request_dev($sid, "set 3 output high");

sleep(1);

echo "turn all output ports off₩r₩n";
spc_request_dev($sid, "set 0 output low");
spc_request_dev($sid, "set 1 output low");
spc_request_dev($sid, "set 2 output low");
spc_request_dev($sid, "set 3 output low");
?>
```

output

```
turn all output ports on
```

```
turn all output ports off
```

Getting Status of Output

Calling spc_request_dev function for getting status of output ports

```
spc_request_dev($sid, $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

The normal return value is in string form and is as follows:

value	description
0	OFF
1	ON

Example

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

echo "turn all output ports on₩r₩n";
spc_request_dev($sid, "set 0 output high");
spc_request_dev($sid, "set 1 output high");
spc_request_dev($sid, "set 2 output high");
spc_request_dev($sid, "set 3 output high");

// get status of output ports
echo "Port 0: ", spc_request_dev($sid, "get 0 output"), "₩r₩n";
echo "Port 1: ", spc_request_dev($sid, "get 1 output"), "₩r₩n";
echo "Port 2: ", spc_request_dev($sid, "get 2 output"), "₩r₩n";
echo "Port 3: ", spc_request_dev($sid, "get 3 output"), "₩r₩n";

sleep(1);

echo "turn all output ports off₩r₩n";
```

```
spc_request_dev($sid, "set 0 output low");
spc_request_dev($sid, "set 1 output low");
spc_request_dev($sid, "set 2 output low");
spc_request_dev($sid, "set 3 output low");

// get status of output ports
echo "Port 0: ", spc_request_dev($sid, "get 0 output"), "₩r₩n";
echo "Port 1: ", spc_request_dev($sid, "get 1 output"), "₩r₩n";
echo "Port 2: ", spc_request_dev($sid, "get 2 output"), "₩r₩n";
echo "Port 3: ", spc_request_dev($sid, "get 3 output"), "₩r₩n"
?>
```

Output

```
turn all output ports on
Port 0: 1
Port 1: 1
Port 2: 1
Port 3: 1
turn all output ports off
Port 0: 0
Port 1: 0
Port 2: 0
Port 3: 0
```