IPPA Provider

IMAC LED Pulse Controller IPPA series

Version 1.0.0

User's guide

September 20, 2016

[Remarks]

[Revision history]

Version	Date	Content
1.0.0	2016-09-20	First edition

[Supported models]

Model	Version	Note
IPPA-7M2		
IPPA-7M4		

Contents

1. Introduction

This document is a user's guide of IPPA provider that is CAO provider designed for the LED pulse controller "IPPA series" manufactured by IMAC. Hereafter, this provider (CaoProvIMACIPPA.dll) is called IPPA provider.

Chapter 2 describes the outline of IPPA provider and equipped method. Chapter 3 describes command reference for lighting commands and for original commands.

The support status of communication commands implemented in IPPA provider depends on the LED pulse controller (IPPA series) to be communicated.

For details about communication, refer to "m_en_IPPA-7M4(7M2).pdf" of IMAC.

2. Outline of provider

2.1. Outline

IPPA provider is a CAO provider that enables to control lighting with IMAC's LED Pulse controller (IPPA series) with commands via TCP/IP connection.

The file format is DLL (Dynamic Link Library) and it is dynamically uploaded from the CAO engine.

To use IPPA provider, you need to install ORiN2SDK, or, complete registration manually with the procedure shown below.

Table2-1 IPPA provider

File name	CaoProvIMACIPPA.dll
ProgID	CaoProv.IMAC.IPPA
Registration	regsvr32 CaoProvIMACIPPA.dll
Deregistration	regsvr32 /u CaoProvIMACIPPA.dll

2.2. Method and Properties

2.2.1. CaoWorkspace::AddController method

IPPA provider establishes communication by referring to the connection parameters for communication when AddController is executed. (This works as a TCP client.)

IPPA provider does not have network setting functions. For network setting, please use sample software for IPPA provided by IMAC website.

Syntax AddController(<bstrCtrlName:BSTR>,<bstrProvName:BSTR>,
<bstrPCName:BSTR>,<bstrOption:BSTR>))

bstrCtrlName : [in] Controller name
bstrProvName : [in] Provider name. Fixed to "CaoProv.IMAC.IPPA"
bstrPCName : [in] Computer name where provider runs
bstrOption : [in] Option character strings

The following table shows a list of option character strings.

Table 2-2 Option character strings of CaoWorkspace::AddController

Option	Description
Conn=<Communication parameter>	Required. Communication configuration and connection parameter. (See 2.2.1.1)
TimeOut[=<Timeout period>]	Timeout period at the data receiving and sending (millisecond) (Default :500)

2.2.1.1. Conn option

The following shows connection parameter strings for Conn option. Items enclosed with square brackets ("[]") are omittable. Underlined part shows the default value when the option is not specified. IPPA series equips four connection ports. Specify one port from them for connection.

(IPPA provider supports TCP/IP connection only.)

"Conn=eth:<IP Address>[:<Port No>]"

<IP Address> : IP address
Example: 196.168.0.29 (Specify an IP address from 192.168.0.1 through 192.168.255.254.)

<Port No> : TCP/IP connection port number 3100, 5006 (Specify any value from 1000 through 65532)

2.2.2. CaoController::Execute method

Directly receive and send an IPPA command. Specify a command name for the first argument and command parameter for the second argument. For details and usage of each command, refer to Chapter 3 Command reference.

Syntax Execute (<bstrCommandName:VT_BSTR>,[<vntParam : VT_VARIANT>])

bstrCommandName: [in] Command name

vntParam : [in] Parameter

2.2.3. CaoController::AddVariable method

This is a method to create a variable object.

Syntax AddVariable(<bstrVariableName:VT_BSTR>[,<bstrOption:VT_BSTR>])

<bstrVariableName>	:	[in] Variable name
<bstrOption>	:	[in] Option string (not used)

2.2.4. CaoController::get_VariableNames property

Obtain the list of variable names and system variable names that can be specified by AddVariable method.

2.2.5. CaoVariable::get_Value property

Obtain a variable name that corresponds with an object. IPPA provider does not have CaoVariable:put_Value method that sets a variable value to a variable object.

2.3. Variable list

2.3.1. CaoController class

Tabel 2-3 CaoController class system variable list

Variable	Data type	Description	Attribute	
			get	put
@MAKER_NAME	VT_BSTR	"IMAC Co., Ltd." is returned.	✓	—
@VERSION	VT_I4 | VT_ARRAY	Version information. Firmware versions of two internal devices are returned	✓	—
@ERROR_STATE	VT_I4	LED controller's error state is returned. 0 :Normal 1 : Voltage is now under-recovery; voltage dropped due to lighting. 2 : Lighting was used during voltage recovery. If you use lighting during recovery, this error is issued for 0.5 second after the lighting. 3 : The state between 1 and 2.	✓	—

2.4. Error code

In IPPA provider, original error codes described in Table 2-4 are defined. For about ORiN2 common errors, refer to the error code section of ORiN2Programming guide.

Table 2-4 Original error code

Error name	Error number	Description
Entire error response	0x80100000	This error is returned when an error response arrives from the connection destination.
Received data imperfect	0x80100001	This error is returned when analysis failed because the received data was imperfect.
Different command response	0x80100002	This error is returned if the response and the sent command mismatches.
Multiple channel error response	0x80100101 to 0x8010FF	This error is returned when one or more channel's errors are returned by the command execution that reads and writes values of multiple channels. (See 2.4.1)

2.4.1. Multiple channel error response

IPPA provider implements commands that can read and write multiple channel's setting values at the same time. (See Chapter 3.) This command uses an array that stores target channels in desired order as a parameter. (See3.1.3.) When such commands are executed, if several channel's errors are notified from the connection destination, the error code will range from 0x80100101 through 0x801001FF. The lower two digits of the error code expresses which channel's errors make up this error code. The position of error channels are treated as bit flags, and the flags are converted to hexadecimal number. (See the next page for details.)

Table 2-5 shows the correspondence between error codes and the error occurrence channel position at the multiple channel error response. Please note that the "Xth" in this table means "the order of channel in array", it does not mean the "channel number X".

If this error is returned when you execute a command that writes setting values, please keep in mind that the values of channels other than the error-occurred channel have been changed.

Table 2-5 Error code for Multiple channel error response (for 4 channels)

0x801001??	Error bit (Error: 1, Normal: 0)			
	4th	3rd	2nd	1st
01	0	0	0	1
02	0	0	1	0
03	0	0	1	1
04	0	1	0	0
05	0	1	0	1
06	0	1	1	0
07	0	1	1	1
08	1	0	0	0
09	1	0	0	1
0A	1	0	1	0
0B	1	0	1	1
0C	1	1	0	0
0D	1	1	0	1
0E	1	1	1	0
0F	1	1	1	1

Calculation formula of Multiple channel error response code "E_MULTIERR"

$$E\_MULTIERR = E\_MULTIERR\_MASK \mid ErrFlg \quad (1)$$

$$ErrFlg = (Err_1^1 + Err_2^2 + ... + Err_N^N) / 2 \quad (2)$$

E_MULTIERR_MASK : Mask for multiple channel error response = 0x80100100

ErrFlg : Error position bit flag of a command that reads/writes multiple channels simultaneously. (Hexadecimal)

Err_i : Whether the channel number specified in i-th has an error or not. (Error : 2, Normal : 0)

N : number of channels

Example: When CaoController.Execute("ReadOnOffSetting", Array(2 , 3, 1)) is executed, CH2 and CH1 have errors. (CaoController.Execute("ReadOnOffSetting") command checks the light ON/OFF status of CH2, CH3 and CH1. For details, see 3.1.3).

→ $ErrFlg = (2^1 + 0^2 + 2^3) / 2 = 0x05$

E_MULTIERR = 0x80100100 | 0x05 = 0x80100105

3. Command reference

This chapter describes each command of CaoController::Execute method

Table 3-1 CaoController::Execute command list

Command	Function	Multiple channel simultaneous reading/writing	
Lighting setting			
ReadExecutionPattern	Check the currently-selected execution pattern	−	P. 13
WriteExecutionPattern	Change the execution pattern.	−	P. 13
ReadOnOffSetting	Read the light-ON/OFF setting.	✓	P. 13
WriteOnOffSetting	Write the light-ON/OFF setting.	✓	P. 14
ReadTriggerMode	Read the trigger mode setting.	✓	P. 15
WriteTriggerMode	Write the trigger mode setting.	✓	P. 15
ReadDelayTime	Read the delay time setting.	✓	P. 16
WriteDelayTime	Write the delay time setting.	✓	P. 16
ReadPWMDuty	Read the duty cycle setting of PWM.	✓	P. 17
WritePWMDuty	Write the duty cycle setting of PWM.	✓	P. 17
ReadPWMDuration	Read the lighting duration of PWM.	✓	P. 18
WritePWMDuration	Write the lighting duration of PWM.	✓	P. 18
ReadLightingSetting	Read the lighting setting saved.	✓	P. 19
WriteLightingSetting	Save the current lighting setting.	✓	P. 19
Original expansion command			
ExecuteCommand	Execute an IPPA command.	−	P. 20
GetTimeout	Obtain the timeout period	−	P. 20
SetTimeout	Set the timeout period	−	P. 21

3.1. Lighting setting

3.1.1. CaoController::Execute ("ReadExecutionPattern ") command

Return the currently-selected execution pattern.

Syntax ReadExecutionPattern

Argument : none

Return value : Execution pattern number (1 to 8) (VT_I4)

The following sample shows how to obtain the currently-selected execution pattern.

Example

```
Dim lNum as Long
lNum = caoCtrl.Execute("ReadExecutionPattern")
' lNum : Execution pattern
```

3.1.2. CaoController::Execute ("WriteExecutionPattern") command

Change the execution pattern.

Syntax WriteExecutionPattern (<lPtnNo>)

lPtnNo : [in] Execution pattern number (1 to 8) (VT_I4)

Return value : none

The following shows how to change the execution pattern to "pattern 3".

Example

```
caoCtrl.Execute "WriteExecutionPattern", 3
```

3.1.3. CaoController::Execute ("ReadOnOffSetting") command

Read the light-ON/OFF setting of the specified channel number.

Two or more channel's settings can be read simultaneously. The maximum number of channels to be specified differs depending on the number of device channels; you can specify up to two channels for IPPA-7M2 and four channels for IPPA-7M4.

If N pieces of channel numbers {No_{ch1}, No_{ch2}, ..., No_{chN} } are specified, N pieces of ON/OFF setting values { Val_{Noch1}, Val_{Noch2}, ..., Val_{NochN} } that correspond with the specified channels will be returned with the same order as the channel numbers.

Syntax ReadOnOffSetting(<plCHNo>)

plCHNo	:	[in] Array of the channel number (1 to 4) (VT_I4 | VT_ARRAY)
Return value	:	<plOnOff> Array of light-ON/OFF setting (VT_I4 | VT_ARRAY)
		0 : Light OFF
		1 : Light ON

The following sample shows how to read the light-ON/OFF settings of CH2 and CH3.

Example

```
Dim vntRet as Variant
vntRet = caoCtrl.Execute("ReadOnOffSetting", Array(2, 3))
' vntRet(0) : ON/OFF setting of CH2
' vntRet(1) : ON/OFF setting of CH3
```

3.1.4. CaoController::Execute ("WriteOnOffSetting") command

Set the light ON/OFF to the specified channel number.

Two or more channel's settings can be set simultaneously. Specify N pieces of channel numbers {No_{ch1}, No_{ch2}, ..., No_{chN} } and N pieces of ON/OFF setting values { Val_{Noch1}, Val_{Noch2}, ..., Val_{NochN} }. The order of setting values must be the same order as the corresponding channel numbers.

Syntax WriteOnOffSetting(<plCHNo>, <plOnOff>)

plCHNo	:	[in] Array of the channel number (1 to 4) (VT_I4 | VT_ARRAY)
plOnOff	:	[in] Array of the light-ON/OFF setting (VT_I4 | VT_ARRAY)
		0 : Light OFF
		1 : Light ON
Return value	:	none

The following sample shows how to set the light of CH3 to ON (=1) and CH2 to OFF (=0).

Example

```
caoCtrl.Execute "WriteOnOffSetting", Array(Array(3, 2), Array(1, 0))
```

3.1.5. CaoController::Execute ("ReadTriggerMode") command

Read the trigger mode setting of the specified channel number. You can select either the internal trigger mode or the external trigger mode. Each trigger mode is described as follows.

・Internal trigger mode : Always light-ON in 80kHz.

・External trigger mode : Light-ON according to the external trigger

Two or more channel's settings can be read simultaneously. (See 3.1.3.)

Syntax ReadTriggerMode(<plCHNo>)

plCHNo	:	[in] Array of the channel number (1 to 4) (VT_I4 | VT_ARRAY)
Return value	:	<plTriger> : Array of trigger mode setting (VT_I4 | VT_ARRAY)
		0 : Internal trigger mode
		1 : External trigger mode

The following sample shows how to read the trigger mode settings of CH2 and CH3 lighting.

Example

```
Dim vntRet as Variant
vntRet = caoCtrl.Execute("ReadTriggerMode", Array(2, 3))
' vntRet(0) : Trigger mode of CH2
' vntRet(1) : Trigger mode of CH3
```

3.1.6. CaoController::Execute ("WriteTriggerMode") command

Set the trigger mode of the specified channel number.

Two or more channel's settings can be set simultaneously. (See 3.1.4)

Syntax WriteTriggerMode(<plCHNo>, <plOnOff>)

plCHNo	:	[in] Array of the channel number (1 to 4) (VT_I4 | VT_ARRAY)
plTriger	:	[in] Array of trigger mode setting (VT_I4 | VT_ARRAY)
		0 : Internal trigger mode
		1 : External trigger mode
Return value	:	none

The following sample shows how to set CH3 to the internal trigger mode and CH2 to the external trigger mode.

Example

```
caoCtrl.Execute "WriteTriggerMode", Array(Array(3, 2), Array(0, 1))
```

3.1.7. CaoController::Execute ("ReadDelayTime") command

Read the delay time of specified execution pattern number and channel number.

Two or more channel's settings in one execution pattern can be read simultaneously. If the pattern number No_{ptn} and *N* pieces of channel numbers {No_{ch1}, No_{ch2}, ..., No_{chN} } are specified, *N* pieces of ON/OFF setting values that correspond with the specified channels { Val_{NoPtn}, No_{ch1}, Val_{NoPtn}, No_{ch2}, ..., Val_{NoPtn}, No_{chN} } will be returned with the same order as the channel numbers.

Syntax ReadDelayTime(<lPtnNo>, <plCHNo>)

lPtnNo	:	[in] Execution pattern number (1 to 8) (VT_I4)
plCHNo	:	[in] Array of the channel number (1 to 4) (VT_I4 | VT_ARRAY)
Return value	:	<plDlyTime> : Array of delay time (VT_I4 | VT_ARRAY)
		The return value ranges from 0 through 9999 (×10 μs).

The following sample shows how to read the delay time settings of CH2 and CH3 of execution pattern 5.

Example

```
Dim vntRet as Variant
vntRet = caoCtrl.Execute("ReadDelayTime", Array(5, Array(2, 3)))
' vntRet(0) : Execution pattern 5, Delay time of CH2
' vntRet(1) : Execution pattern 5, Delay time of CH3
```

3.1.8. CaoController::Execute ("WriteDelayTime") command

Set the delay time of specified execution pattern number and channel number.

Two or more channel's settings can be set simultaneously. Specify the pattern number Noptn and N pieces of channel numbers {No_{ch1}, No_{ch2}, ..., No_{chN} } and N pieces of ON/OFF setting values { Val_{NoPtn}, No_{ch1}, Val_{NoPtn}, No_{ch2}, ..., Val_{NoPtn}, No_{chN} }. The order of setting values must be the same order as the corresponding channel numbers.

Syntax WriteDelayTime(<lPtnNo>, <plCHNo>, <plDlyTime>)

lPtnNo	:	[in] Execution pattern number (1 to 8) (VT_I4)
plCHNo	:	[in] Array of the channel number (1 to 4) (VT_I4 | VT_ARRAY)
plDlyTime	:	[in] Array of delay time (VT_I4 | VT_ARRAY)
		Specify a number from 0 through 9999 (×10 μs).
Return value	:	none

The following sample shows how to set the delay time of execution pattern 5 CH3 to 33.33 ms (3333 ×10 μs) and CH2 to 2.5 ms (250 ×10 μs).

Example

```
caoCtrl.Execute "WriteDelayTime", Array(5, Array(3, 2), Array(3333, 250))
```

3.1.9. CaoController::Execute ("ReadPWMDuty") command

Read the PWM control's duty cycle on the specified execution pattern number and channel number. In IPPA series, duty cycle (range from 0 through 100%) is expressed with 256 levels. Two or more channel's settings in one execution pattern can be read simultaneously. (See 3.1.7)

Syntax ReadPWMDuty(<lPtnNo>, <plCHNo>)

lPtnNo	:	[in] Execution pattern number (1 to 8) (VT_I4)
plCHNo	:	[in] Array of the channel number (1 to 4) (VT_I4 | VT_ARRAY)
Return value	:	<plPWMDty> : Array of PWM's Duty cycle (VT_I4 | VT_ARRAY)
		The return value ranges from 0 through 255.

The following sample shows how to read the PWM duty cycle of CH2 and CH3 of execution pattern 5.

Example

```
Dim vntRet as Variant
vntRet = caoCtrl.Execute("ReadPWMDuty", Array(5, Array(2, 3)))

' vntRet(0) : Execution pattern 5, PWM duty cycle of CH2
' vntRet(1) : Execution pattern 5, PWM duty cycle of CH3
```

3.1.10. CaoController::Execute ("WritePWMDuty") command

Set the PWM control's duty cycle on the specified execution pattern number and channel number. Two or more channel's settings can be set simultaneously. Two or more channel's settings in one execution pattern can be read simultaneously. (See 3.1.8)

Syntax WriteDelayTime(<lPtnNo>, <plCHNo>, <plDlyTime>)

lPtnNo	:	[in] Execution pattern number (1 to 8) (VT_I4)
plCHNo	:	[in] Array of the channel number (1 to 4) (VT_I4 | VT_ARRAY)
plPWMDty	:	[in] Array of PWM's duty cycle (VT_I4 | VT_ARRAY)
		Specify a number from 0 through 255.
Return value	:	none

The following sample shows how to set the Duty cycle of CH3 to 255 and CH2 to 20 in execution pattern 5.

Example

```
caoCtrl.Execute "WriteDelayTime", Array(5, Array(3, 2), Array(255, 20))
```

3.1.11. CaoController::Execute ("ReadPWMDuration") command

Read the PWM lighting duration on the specified execution pattern number and channel number. Two or more channel's settings in one execution pattern can be read simultaneously. (See 3.1.7.)

Syntax ReadPWMDuration (<lPtnNo>, <plCHNo>)

lPtnNo	:	[in] Execution pattern number (1 to 8) (VT_I4)
plCHNo	:	[in] Array of the channel number (1 to 4) (VT_I4 | VT_ARRAY)
Return value	:	<plPWMDrtn>: Array of PWM lighting duration (VT_I4 | VT_ARRAY)
		The return value ranges from 0 through 9999 (×10 μs)

The following sample shows how to read the PWM lighting duration of CH2 and CH3 of execution pattern 5.

Example

```
Dim vntRet as Variant
vntRet = caoCtrl.Execute("ReadPWMDuration", Array(5, Array(2, 3)))

' vntRet(0) : Execution pattern 5, PWM lighting duration of CH2
' vntRet(1) : Execution pattern 5, PWM lighting duration of CH3
```

3.1.12. CaoController::Execute ("WritePWMDuration") command

Set the PWM control's duty cycle on the specified execution pattern number and channel number. Two or more channel's settings can be set simultaneously. Two or more channel's settings in one execution pattern can be read simultaneously. (See 3.1.8.)

Syntax WritePWMDuration (<lPtnNo>, <plCHNo>, <plDlyTime>)

lPtnNo	:	[in] Execution pattern number (1 to 8) (VT_I4)
plCHNo	:	[in] Array of the channel number (1 to 4) (VT_I4 | VT_ARRAY)
plPWMDty	:	[in] Array of PWM's duty cycle (VT_I4 | VT_ARRAY)
		Specify a value from 0 through 9999 (×10 μs).
Return value	:	none

The following sample shows how to set the PWM lighting duration of execution pattern 5 CH3 to 55.55 ms (5555 ×10 μs) and CH2 to 60 ms (6000 ×10 μs).

Example

```
caoCtrl.Execute "WritePWMDuration", Array(5, Array(3, 2), Array(5555, 6000))
```

3.1.13. CaoController::Execute ("ReadLightingSetting") command

Read the lighting setting of the specified channel number from the saved lighting setting data, and set it to the current setting value. The following five settings are read.

・Light-ON/OFF ・Delay setting value of each pattern ・Trigger mode
・PWM Duty cycle of each pattern ・PWM lighting duration of each pattern

Two or more channel's settings can be read simultaneously.(See 3.1.3.)

Syntax ReadLightingSetting (<plCHNo>)

plCHNo : [in] Array of the channel number (1 to 4) (VT_I4 | VT_ARRAY)
Return value : none

The following sample shows how to read the lighting settings of CH2 and CH3.

Example

```
caoCtrl.Execute "ReadLightingSetting", Array(2, 3)
```

3.1.14. CaoController::Execute ("WriteLightingSetting") command

Save the lighting setting of the specified channel number. Two or more channel's settings can be set simultaneously. (See 3.1.3.) Items to be saved are the same as the ones read by 3.1.13.

Syntax WriteLightingSetting (<plCHNo>)

plCHNo : [in] Array of the channel number (1 to 4) (VT_I4 | VT_ARRAY)
Return value : none

The following sample shows how to save the lighting settings of CH2 and CH3.

Example

```
caoCtrl.Execute "WriteLightingSetting", Array(Array(3, 2))
```

3.2. Original command

3.2.1. CaoController::Execute ("ExecuteCommand") command

Execute an IPPA command directly. This command obtains the command response regardless of the command execution result (failed/succeed). For information about supported IPPA commands, refer to IPPA series user's manual.

Syntax [<vntRet> =]ExecuteCommand(< bstrCommand >)

bstrCommand	:	[in] Specify the command strings(VT_BSTR)
vntRet	:	Return the command response.(VT_BSTR)

The following sample shows how to set the light of CH3 to ON by specifying an IPPA command.

Example

```
Dim strRet as string
strRet = caoCtrl.Execute("ExecuteCommand", "W02010001")
```

3.2.2. CaoController::Execute ("SetTimeout") command

Set the timeout.

Syntax SetTimeout(<iTimeout>)

iTimeout	:	[in] Timeout (ms) (VT_UI4)
Return value	:	none

Example

```
Call caoCtrl.Execute("SetTimeout", 1000)
```

3.2.3. CaoController::Execute ("GetTimeout") command

Obtain the timeout being set.

Syntax <uiTimeout> = GetTimeout()

Argument : [in] none

<uiTimeout> : Timeout (ms) (VT_UI4)

Example

```
Dim timeout as long
timeout = caoCtrl.Execute("GetTimeout")
```