# I-8024UW / I-8028UW
# I-9024U / I-9028U
# Linux API Reference Manual

**V 2.0.1 July 2020**

Written by Edward Ku

Edited by Cindy Huang

## Warranty

All products manufactured by ICP DAS are under warranty regarding defective materials for a period of one year, beginning from the date of delivery to the original purchaser.

## Warning

ICP DAS assumes no liability for any damage resulting from the use of this product. ICP DAS reserves the right to change this manual at any time without notice. The information furnished by ICP DAS is believed to be accurate and reliable. However, no responsibility is assumed by ICP DAS for its use, nor for any infringements of patents or other rights of third parties resulting from its use.

## Copyright



## Trademarks

Names are used for identification purposes only and may be registered trademarks of their respective companies.

## Contact Us

If you have any problems, please feel free to contact us.
You can count on us for a quick response.
Email: service@icpdas.com

# Table of Contents


Table of Contents ..... 3
1. Introduction ..... 5
1.1. Specifications ..... 7
1.2. Pin Assignments ..... 8
1.3. Jumper Setting ..... 10
1.4. Wire Connections ..... 11
1.5. Block Diagram ..... 12
1.6. Dimensions ..... 13
2. Quick Start ..... 14
3. Power-on Mode ..... 16
3.1. Power-on Value Mode ..... 18
3.2. Retentive Mode ..... 19
4. Watchdog ..... 20
5. API References ..... 25
5.1. i8028U_Init ..... 27
5.2. i8028U_GetFirmwareVersion ..... 28
5.3. i8028U_GetLibVersion ..... 29
5.4. i8028U_ReadAO_GainOffset ..... 30
5.5. i8028U_WriteAOHex ..... 32
5.6. i8028U_WriteAO ..... 34
5.7. i8028U_ReadAOHex ..... 36
5.8. i8028U_ReadAO ..... 38
5.9. i8028U_SetPowerOnEnStatus ..... 40
5.10. i8028U_GetPowerOnEnStatus ..... 41
5.11. i8028U_WritePowerOnHex_AO ..... 42
5.12. i8028U_WritePowerOn_AO ..... 44
5.13. i8028U_ReadPowerOnHex_AO ..... 46
5.14. i8028U_ReadPowerOn_AO ..... 48
5.15. i8028U_WriteSafeHex_AO ..... 50
5.16. i8028U_WriteSafe_AO ..... 52
5.17. i8028U_ReadSafeHex_AO ..... 54
5.18. i8028U_ReadSafe_AO ..... 56
5.19. i8028U_SetModuleWDTConfig ..... 58
5.20. i8028U_GetModuleWDTConfig ..... 60

5.21. i8028U_GetModuleWDTStatus .....62
5.22. i8028U_ResetModuleWDT .....63
5.23. i8028U_RefreshModuleWDT .....64
Appendix A. Error Code .....65
Appendix B. Revision History .....66

# 1. Introduction

The I-9024U/I-9028U is a 4-channel/8-channel Isolated analog output module that can be used on a 9000 series PAC, and allows a programmable output range on all analog output channels (0 to 5 V, ±5 V, 0 to 10 V, ±10 V, +4 to +20 mA or 0 to +20 mA). Each Analog Output channel can be configured for an individual range, providing an RF immunity level matching that defined by the IEC 61000-4-3 standard. The I-9024U/I-9028U module features channel to-channel isolation as well as 4 kV ESD protection and 1000 VDC intra-module isolation.

And the I-8024UW/ I-8028UW is also a 4-channel/8-channel Isolated analog output module that can be used on a 8000 series PAC.

### Features

| Model | I-8024UW | I-8028UW | I-9024U | I-9028U |
|---|---|---|---|---|
| Channel | 4 | 8 | 4 | 8 |
| Voltage or Current Output | Jumper | Jumper | Software | Software |
| Range | 0 ~ +5 VDC, ±5 VDC, 0 ~ +10 VDC, ±10 VDC, 0 ~ +20 mA, +4 ~ +20 mA | | | |
| Resolution | 16-bit | | | |
| Power-on Value | Yes | | | |
| Safe Value | Yes | | | |
| System LED Indicator | 1 LED as Power Indicator | | | |
| Dimension (L x W x H) | 129 mm × 31 mm × 114 mm | | 144 mm x 30.3 mm x 134 mm | |

## Applicable Platform table

The following table shows which platform the module applies to.

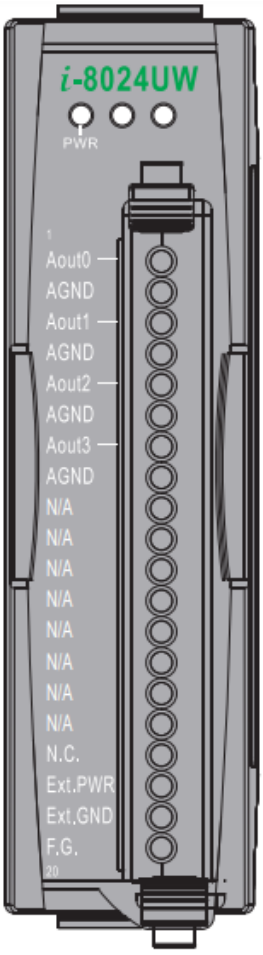
| Platform | OS | Module |
|---|---|---|
| **XPAC** | XP-8000(WES) | I-8024UW/ I-8028UW |
| | XP-8000-Atom (WES) | I-8024UW/ I-8028UW |
| | XP-8000-WES7 (WES7) | I-8024UW/ I-8028UW |
| | XP-8000-CE6 (WinCE 6.0) | I-8024UW/ I-8028UW |
| | XP-8000-Atom-CE6 (WinCE 6.0) | I-8024UW/ I-8028UW |
| | XP-9000-WES7(WES7) | I-9024U/ I-9028U |
| **WinPAC** | WP-8000 (CE 5.0/7.0) | I-8024UW/ I-8028UW |
| | WP-9000-CE7 (CE 7.0) | I-9024U/ I-9028U |
| **LinPAC** | LinPAC-8000(Linux kernel 3.2/4.4) | I-8024UW/ I-8028UW |
| | LinPAC-9000(Linux kernel 3.2/4.4) | I-9024U/ I-9028U |
| **IPAC** | iPAC-8000 (MiniOS7) | I-8024UW/ I-8028UW |
| | I-8000 (MiniOS7) | I-8024UW/ I-8028UW |

## 1.1. Specifications

| Model | I-8024UW | I-8028UW | I-9024U | I-9028U |
|---|---|---|---|---|
| **Analog Output** | | | | |
| Channels | 4 | 8 | 4 | 8 |
| Current Output Wiring | Source | | | |
| Range | 0 ~ +5 VDC, ±5 VDC, 0 ~ +10 VDC, ±10 VDC,<br>0 ~ +20 mA, +4 ~ +20 mA | | | |
| Resolution | 16-bit | | | |
| Accuracy | ±0.02% of FSR | | | |
| Zero Drift | ±0.2 μV/°C | | | |
| Span Drift | ±25 ppm/°C | | | |
| Short Circuit Protection | Yes | | | |
| Power-on Value | Yes | | | |
| Safe Value | Yes | | | |
| **External Power Requirements** | | | | |
| Reserve Polarity Protection | Yes | | | |
| Powered from Terminal Block | Yes, 15 ~ 30 VDC | | | |
| Isolation | 3000 VDC | | | |
| **LED Indicators** | | | | |
| System LED Indicator | 1 LED as Power Indicator | | | |
| **Isolation** | | | | |
| Intra-module Isolation, Field-to-Logic | 3000 VDC | | | |
| **EMS Protection** | | | | |
| ESD (IEC 61000-4-2) | ±4 kV Contact for each Terminal | | | |
| | ±8 kV Air for Random Point | | | |
| **Power** | | | | |
| Power Consumption | 2 W Max. | | | |
| **Mechanical** | | | | |
| Dimension (L x W x H) | 129 mm × 31 mm × 114 mm | | 144 mm x 30.3 mm x 134 mm | |
| **Environment** | | | | |
| Operating Temperature | -25 ~ +75°C | | | |
| Storage Temperature | -40 ~ +85°C | | | |
| Humidity | 10 ~ 90% RH, non-condensing | | | |

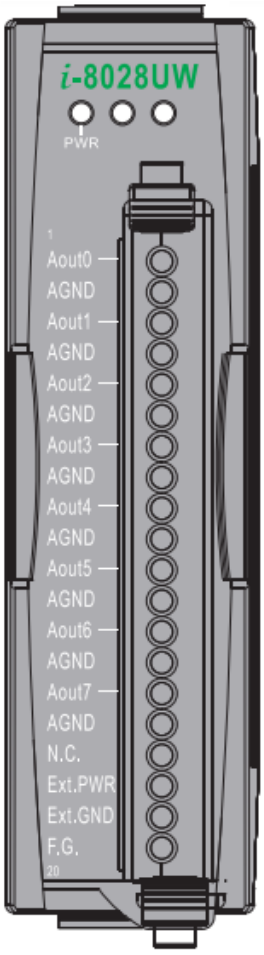
## 1.2. Pin Assignments

### I-8024UW



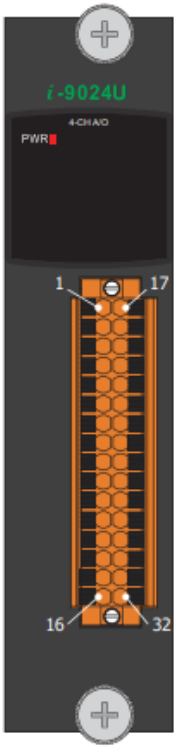

| Terminal No. | Pin Assignment |
|---|---|
| 01 | Aout0 |
| 02 | AGND |
| 03 | Aout1 |
| 04 | AGND |
| 05 | Aout2 |
| 06 | AGND |
| 07 | Aout3 |
| 08 | AGND |
| 09 | N/A |
| 10 | N/A |
| 11 | N/A |
| 12 | N/A |
| 13 | N/A |
| 14 | N/A |
| 15 | N/A |
| 16 | N/A |
| 17 | N.C. |
| 18 | Ext.PWR |
| 19 | Ext.GND |
| 20 | F.G. |

### I-8028UW



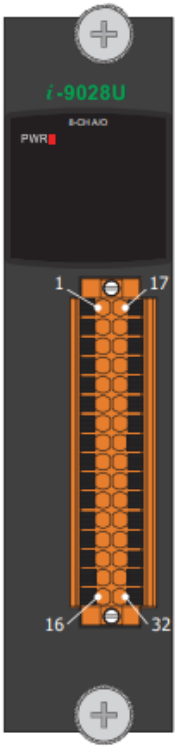

| Terminal No. | Pin Assignment |
|---|---|
| 01 | Aout0 |
| 02 | AGND |
| 03 | Aout1 |
| 04 | AGND |
| 05 | Aout2 |
| 06 | AGND |
| 07 | Aout3 |
| 08 | AGND |
| 09 | Aout4 |
| 10 | AGND |
| 11 | Aout5 |
| 12 | AGND |
| 13 | Aout6 |
| 14 | AGND |
| 15 | Aout7 |
| 16 | AGND |
| 17 | N.C. |
| 18 | Ext.PWR |
| 19 | Ext.GND |
| 20 | F.G. |

## I-9024U




| Pin Assignment | Terminal No. | | Pin Assignment |
|---|---|---|---|
| Vout0 | 01 | 17 | Vout1 |
| Iout0 | 02 | 18 | Iout1 |
| AGND | 03 | 19 | AGND |
| Vout2 | 04 | 20 | Vout3 |
| Iout2 | 05 | 21 | Iout3 |
| AGND | 06 | 22 | AGND |
| - | 07 | 23 | - |
| - | 08 | 24 | - |
| - | 09 | 25 | - |
| - | 10 | 26 | - |
| - | 11 | 27 | - |
| - | 12 | 28 | - |
| EXT.PWR | 13 | 29 | EXT.PWR |
| EXT.PWR | 14 | 30 | EXT.PWR |
| EXT.GND | 15 | 31 | EXT.GND |
| EXT.GND | 16 | 32 | EXT.GND |

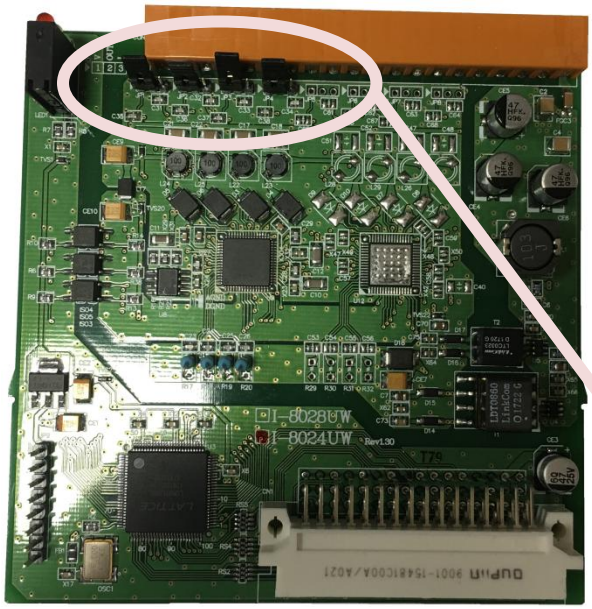
## I-9028U




| Pin Assignment | Terminal No. | | Pin Assignment |
|---|---|---|---|
| Vout0 | 01 | 17 | Vout1 |
| Iout0 | 02 | 18 | Iout1 |
| AGND | 03 | 19 | AGND |
| Vout2 | 04 | 20 | Vout3 |
| Iout2 | 05 | 21 | Iout3 |
| AGND | 06 | 22 | AGND |
| Vout4 | 07 | 23 | Vout5 |
| Iout4 | 08 | 24 | Iout5 |
| AGND | 09 | 25 | AGND |
| Vout6 | 10 | 26 | Vout7 |
| Iout6 | 11 | 27 | Iout7 |
| AGND | 12 | 28 | AGND |
| EXT.PWR | 13 | 29 | EXT.PWR |
| EXT.PWR | 14 | 30 | EXT.PWR |
| EXT.GND | 15 | 31 | EXT.GND |
| EXT.GND | 16 | 32 | EXT.GND |

## 1.3. Jumper Setting

### I-8024U



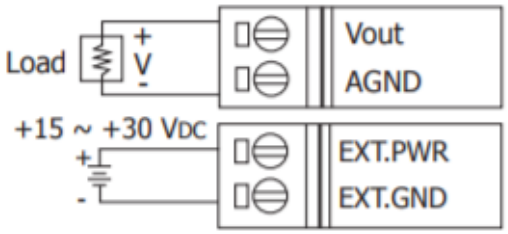
Voltage output

Current output

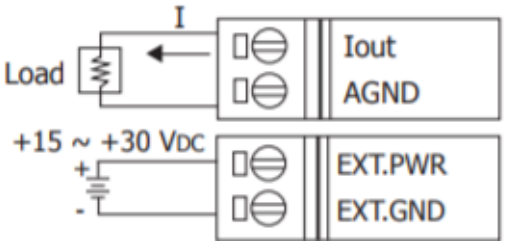
### I-8028U

## 1.4. Wire Connections

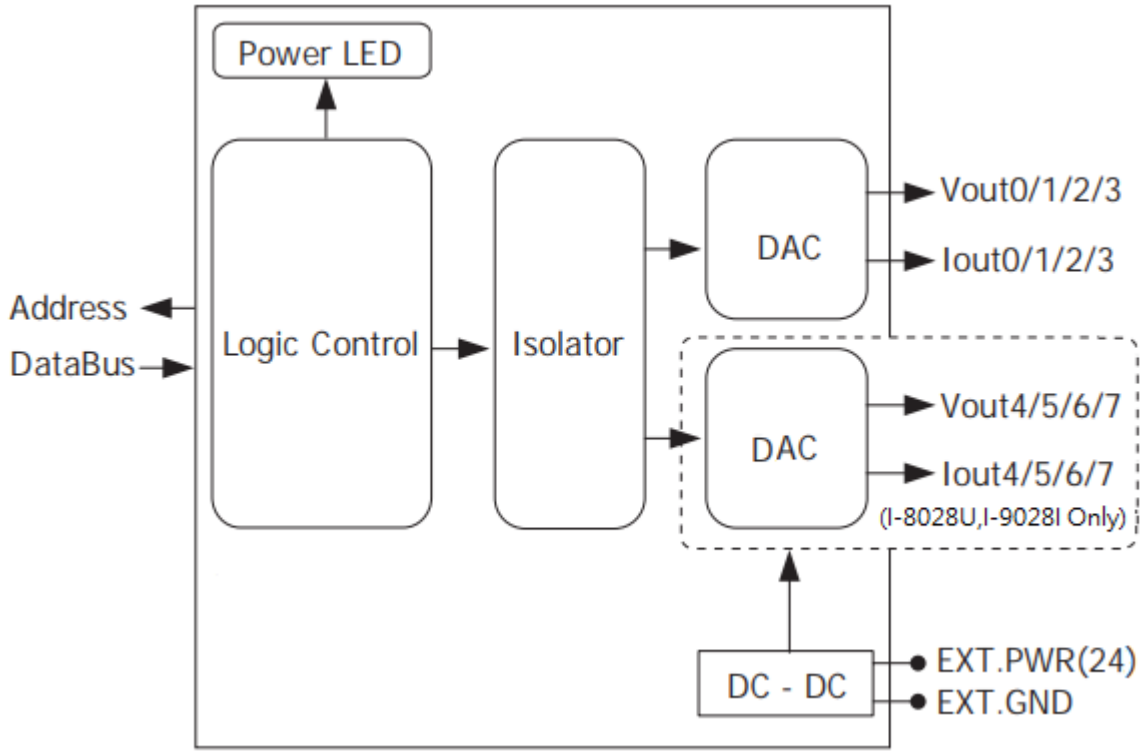
**Voltage Output Wiring**




**Current Output Wiring**



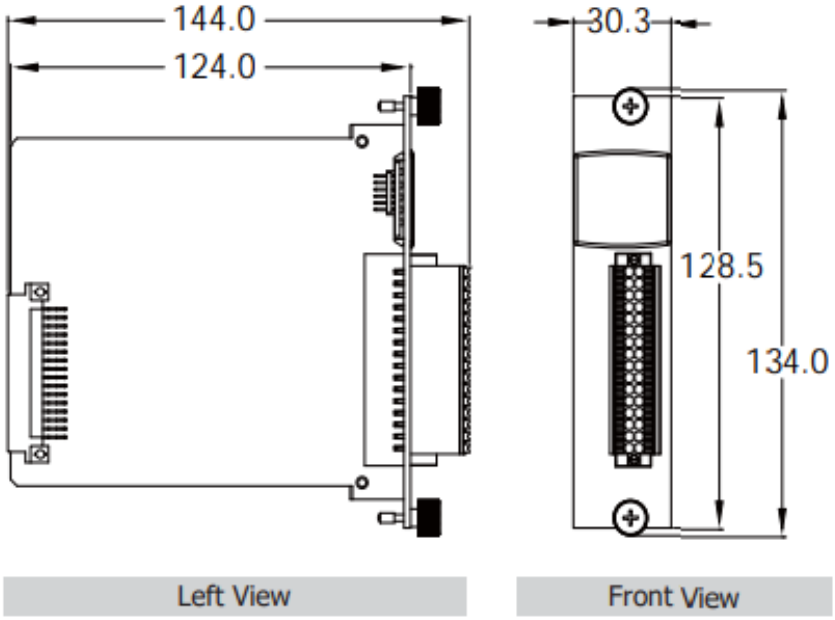
## 1.5. Block Diagram



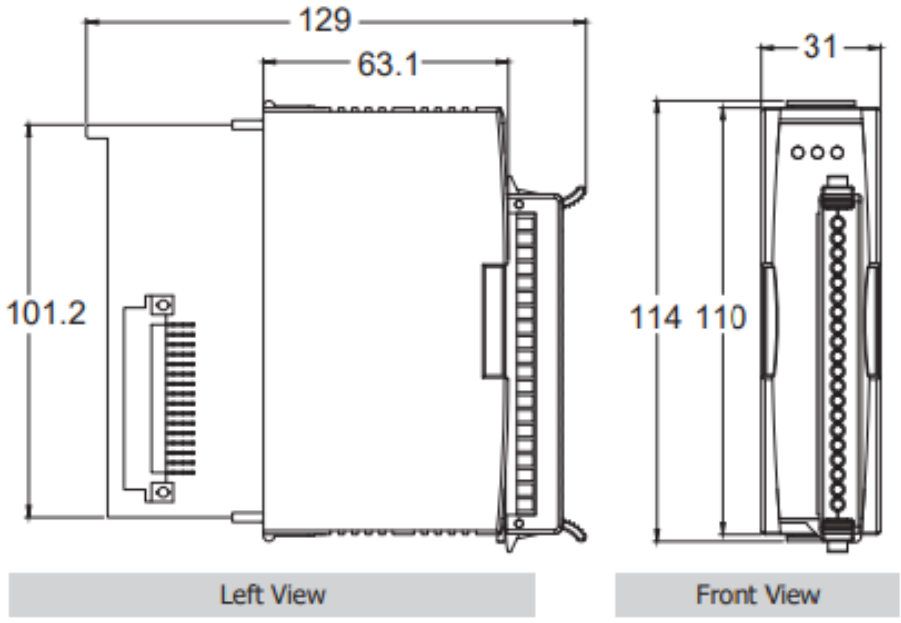
## 1.6. Dimensions

All dimensions are in millimeters.

### I-9024U /I-9028U with Spring clamp terminal connector

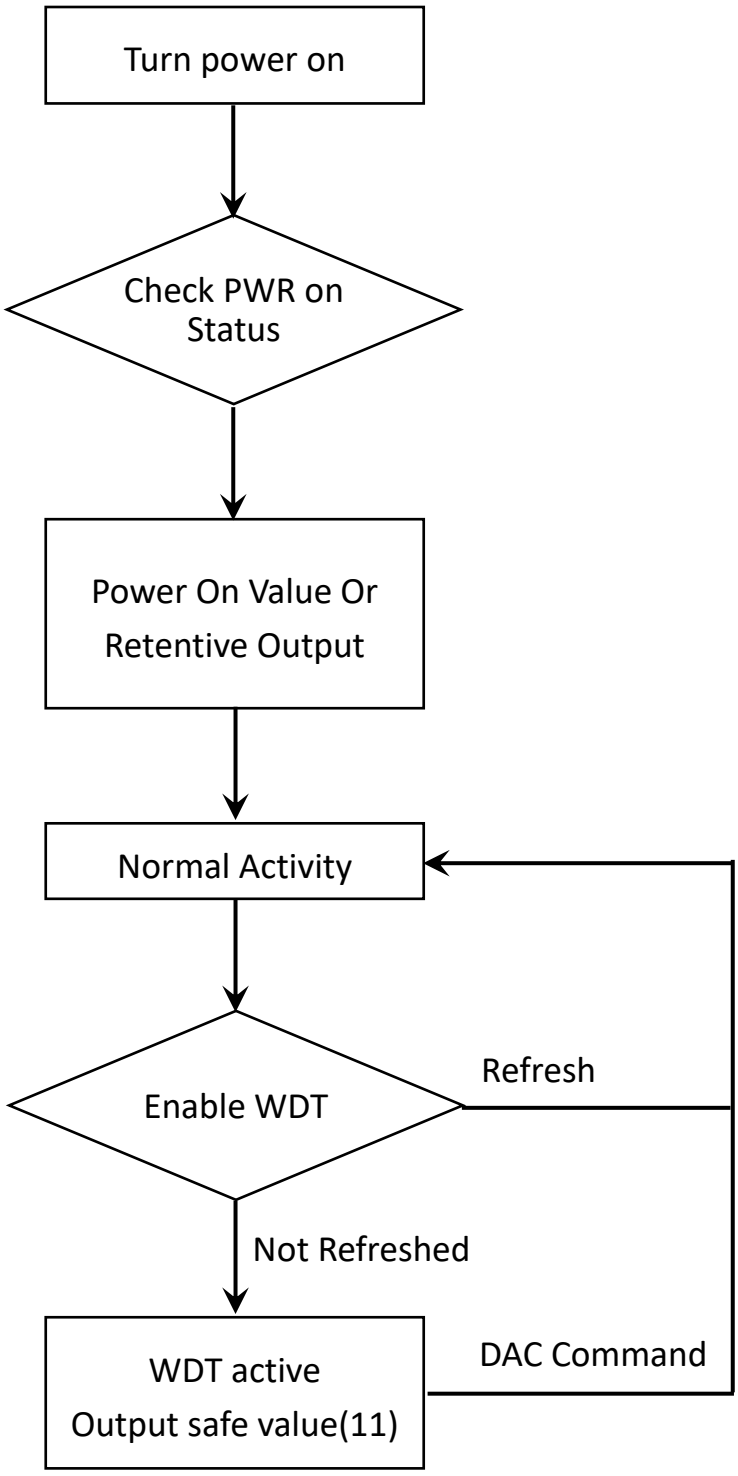



### I-8024UW /I-8028UW with Spring clamp terminal connector

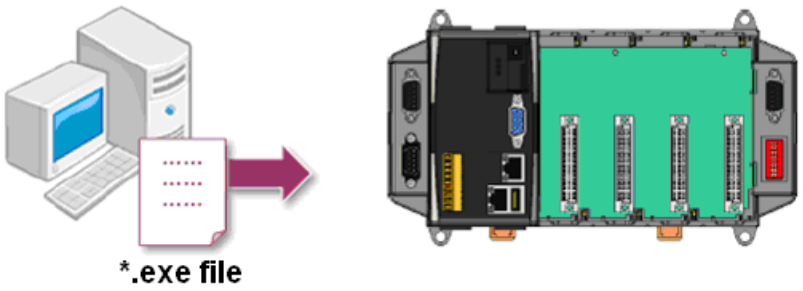


# 2. Quick Start

The following is an illustration of the operating procedure for the I-8024UW/I-8028UW/I-9024U/I-9028U module:



ICP DAS provides a range of demo programs for different platforms that can be used to verify the functions of the I-8024UW/I-8028UW/I-9024U/I-9028U. The source code contained in these programs can also be reused in your own custom programs if needed.




We need to check the following steps before running the program.

1. First, user need to download LinPAC SDK, which is includes GNU toolchain, Libraries, header, examples files, etc.
2. Check the power cable, Ethernet cable, VGA monitor, the communication cable between controller and PC has been connected well, and then check the I-8024UW/I-8028UW/I-9024U/I-9028U has been plugged in the controller.
3. Next, check the communication between controller and PC is fine, and download the demo program files to the controller.
4. The following is a list of the locations where both the demo programs and associated libraries can be found on either the ICP DAS web site or the enclosed CD, and I-8024U/I-8028U/I-9024U/I-9028U use the same library and demo.

   User can find the related files in the product CD or below website:

| PRRODUCT | CPU | DOWNLOAD LINK |
|---|---|---|
| LP-8x4x | PXA270 | http://www.icpdas.com/en/download/show.php?num=982&model=LP-8441-EN |
| LP-8x2x/9x2x | AM335x | http://www.icpdas.com/en/download/show.php?num=915&model=LP-8421 |
| LX-8000/9000 | x86/E38xx | http://www.icpdas.com/en/download/show.php?num=904&model=LX-9381 |

# 3. Power-on Mode

If the module is reset for any reason, the Analog Output will be activated in Power-on Mode so as to avoid any unknown errors that might cause the Analog Output from the module to inflict damage on other devices. This ensures that the output from the I-8024UW/I-8028UW/I-9024U/I-9028U module can be anticipated and guarantees that the output will not damage other devices when the system fail or be reset for any reason.

Two types of Power-on Mode are provided on the I-8024UW/I-8028UW/I-9024U/I-9028U module.

1. Power-on Value Mode
2. Retentive Mode

Power On mode does not support hot plug functionality. When hot plug on PAC Power On mode will not be affected, it only apply on the others cases caused by PAC power reset.

### Configuring Power-on Mode

Power-on Mode included Power-on Value Mode and Retentive Mode. Below is a description of the conditions in each mode will apply when an abnormality be encountered and causes the Power-on Mode to be activated.

| Mode | Output |
|---|---|
| Power-on Value Mode | Ser Power On as Power on Value and reset, then the module will output as configured power on value. |
| Retentive Mode | Ser Power On as Retentive and reset, then the module will output as late output. |

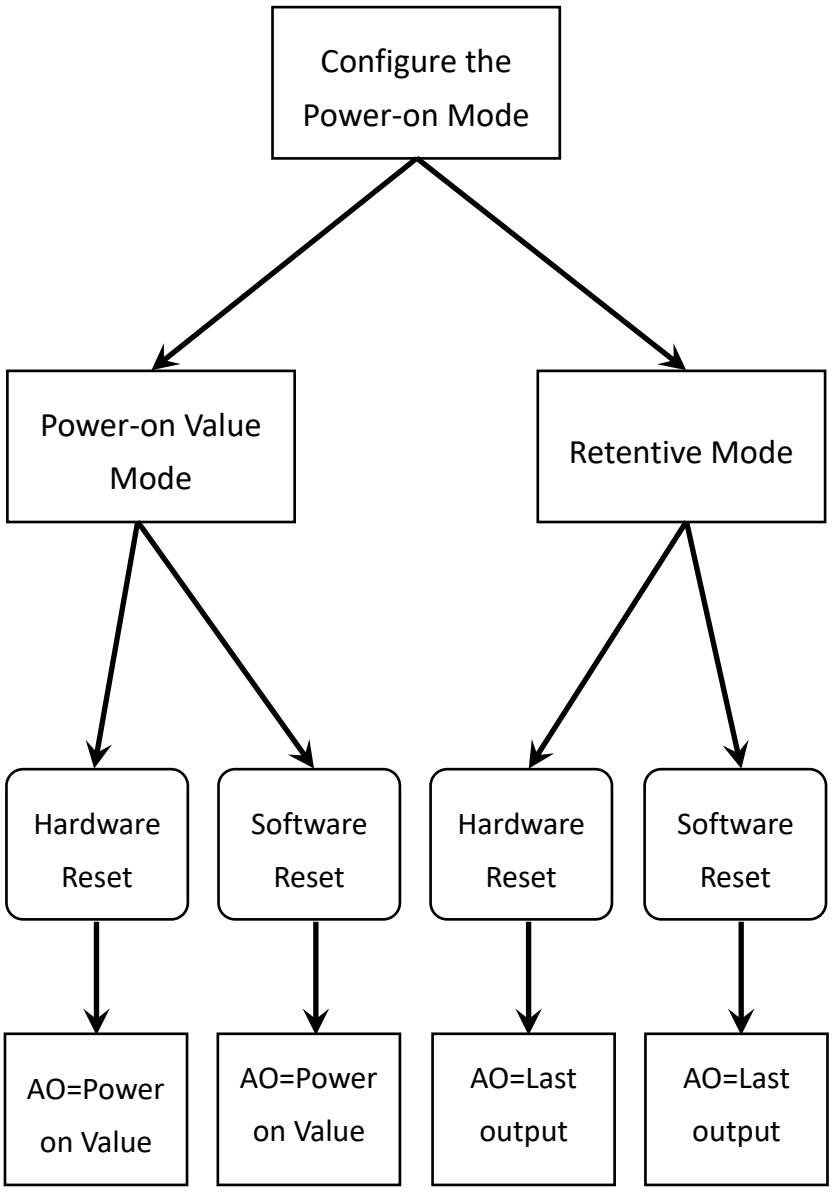
If either the PAC or the module encounters an event that causes the hardware reset or the software reset, the Output will be set to those configured for the specific mode.

The following is an overview of the Analog Output value that will be set for the I-9028U module in each of the Power-on Mode conditions:

| Mode | Hardware Reset | Software Reset |
|---|---|---|
| Power-on Value Mode | AO = Configured Power-on value | AO = Configured Power-on value |
| Retentive Mode | AO = Previous AO value | AO = Previous AO value |

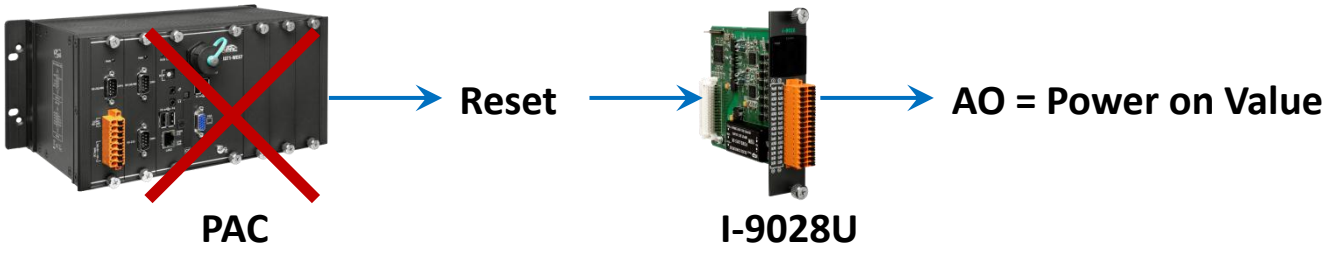
Note: Previous AO value means that after the module reset, the AO value will be retained. For example, if the Power-on Mode is set as Retentive Mode and a hardware reset happened, then the AO value will be the same as the value that existed prior to the hardware reset.

The following is a flowchart that illustrates the outcome of each Power-on Mode:



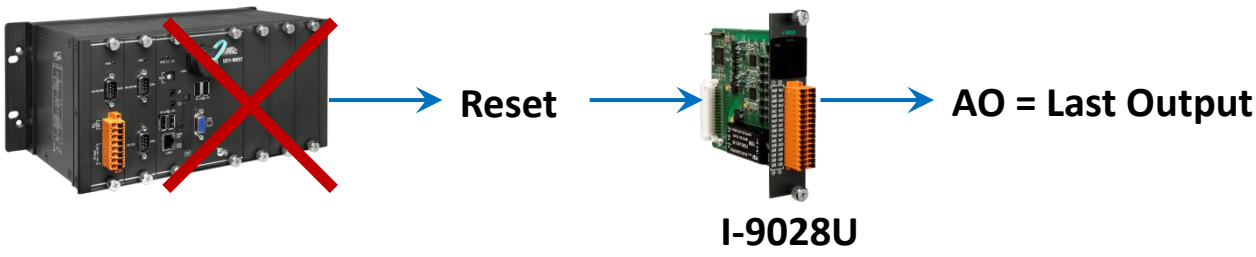
## 3.1. Power-on Value Mode

Power-on Value Mode is used to set the output value to the preconfigured Value after an unknown condition has caused the module reset.



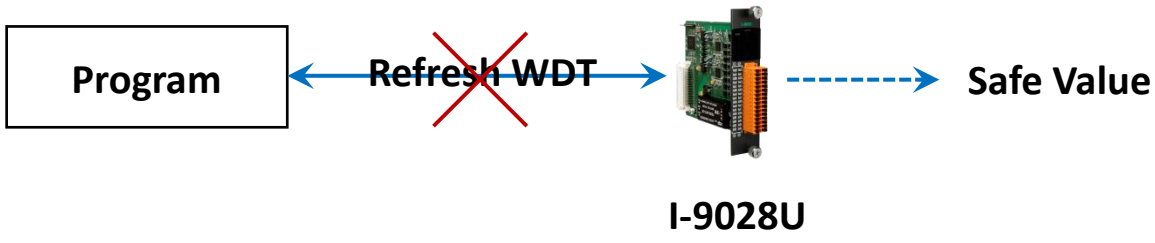
## 3.2. Retentive Mode

Retentive Mode, also known as Virtual Battery Backup Mode, is used to ensure that the previous AO value is retained if an unknown condition has caused the I-9028U module to be reset.

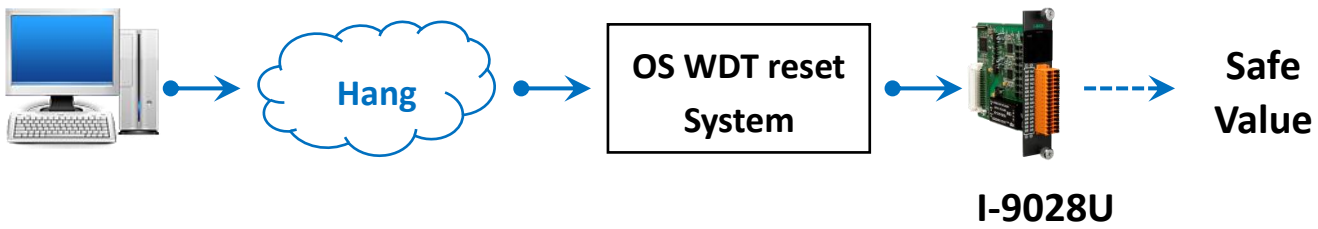


# 4. Watchdog

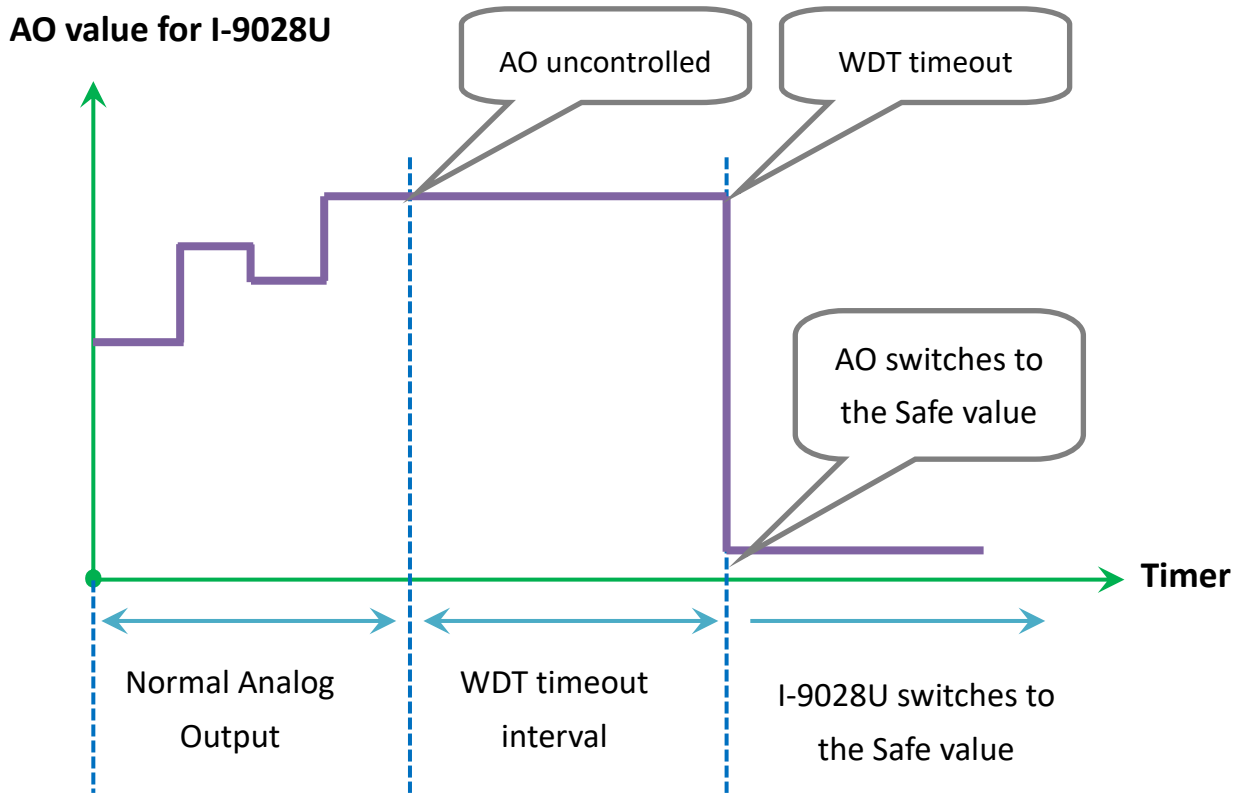
The Watchdog Timer (WDT) on the I-9028U module is a software function that monitors the operating status of the module. Its purpose is to prevent problems due to network or communication errors, or host malfunctions, such as situations where the device is affected by noise, or the program is not stable, etc. When the “refresh WDT” function fails and a Watchdog timeout happened, all output values on the module will be set to the Safe Value state in order to prevent the controlled target from performing any erroneous operations.




A common application is provided as an example below. If the system encounters an even that causes it to become unresponsive for any reason, the I-9028U WDT and the WDT provided by the operating system for the 9000-series PAC can be used in combination, thereby preventing the system from hang becoming unresponsive, which may cause the Analog Output to be uncontrolled and result in damage to the device. The WDT on the 9000-series PAC will reset the PAC to solve the unresponsiveness problem, and then the Safe Value will be set for the I-9028U module to prevent the AO from becoming uncontrolled.



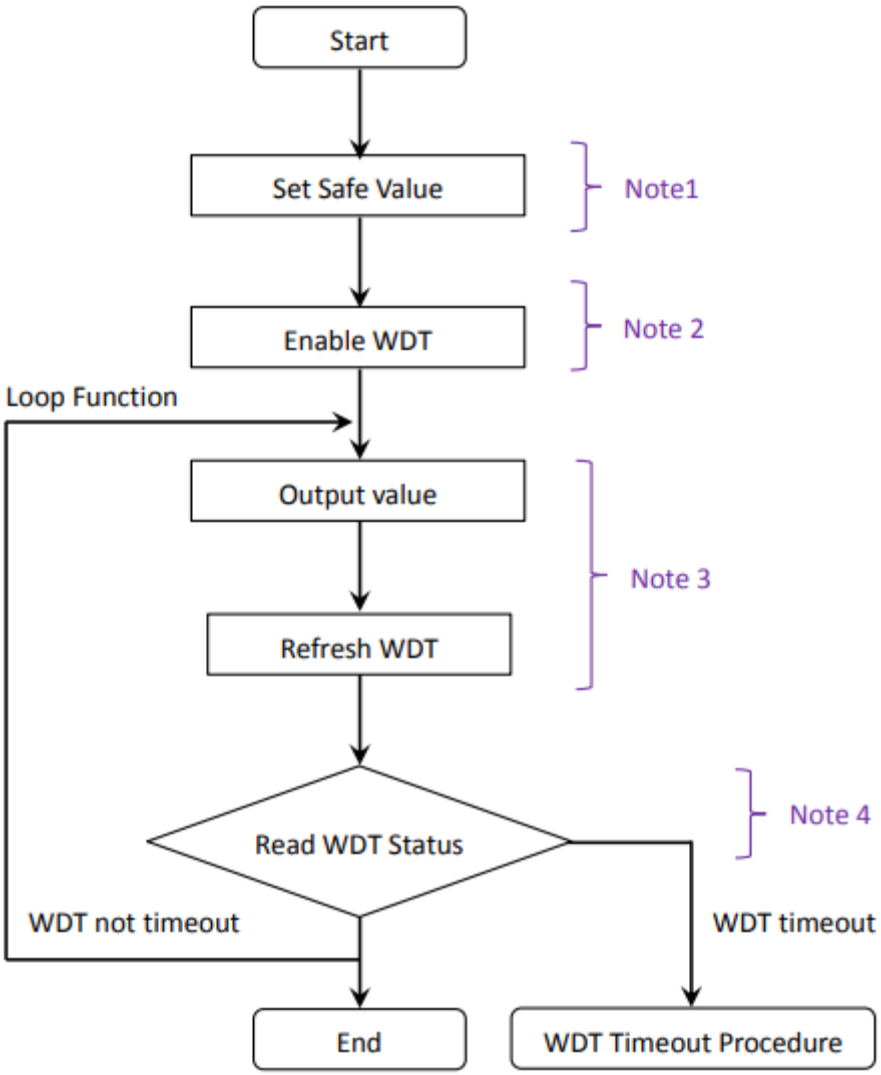
The Analog Output value for the module will be as below. When the WDT timeout is triggered, user can consider necessary to set suit WDT timeout interval and safe value to prevent the AO from becoming uncontrolled.

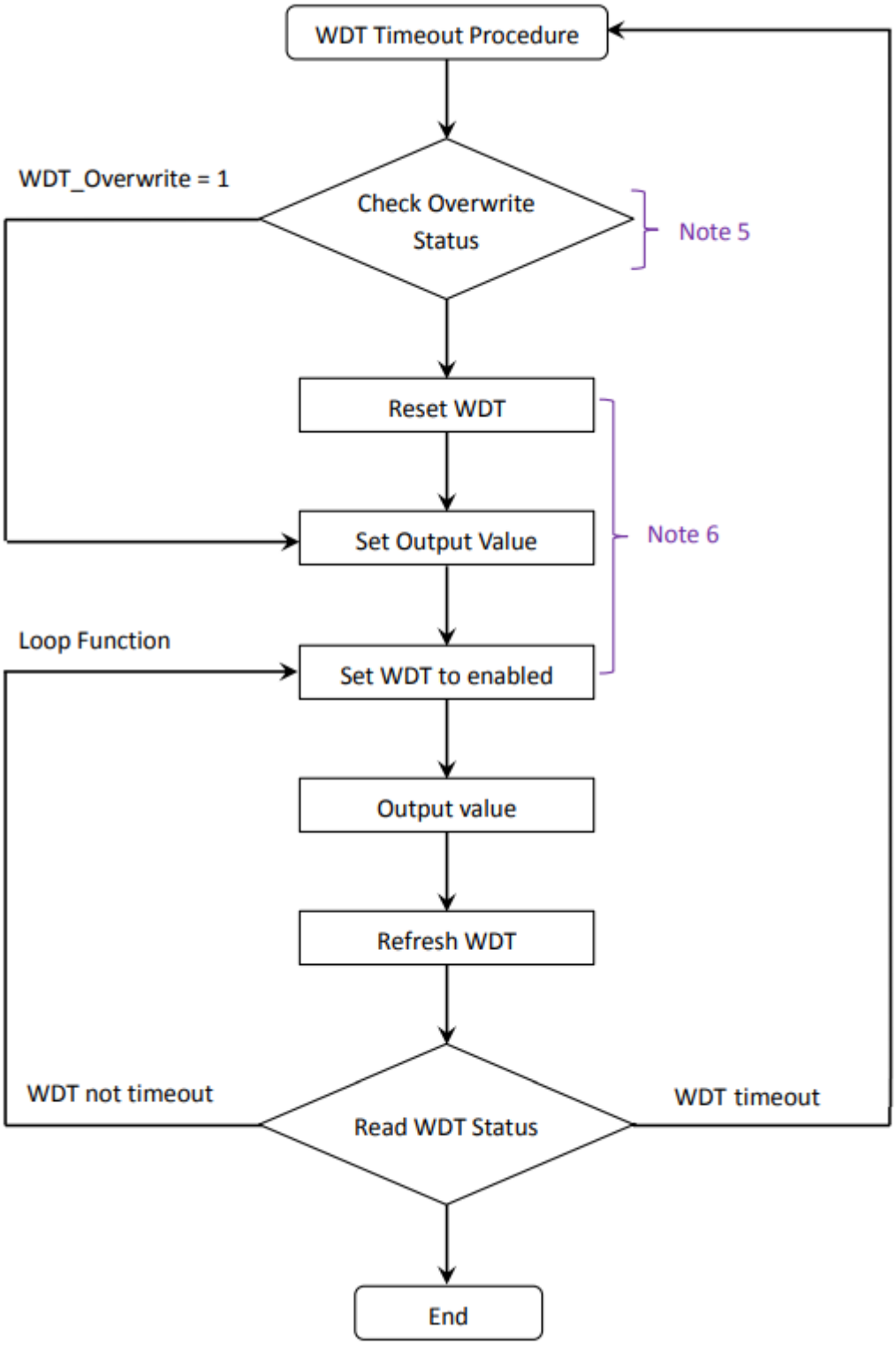



Watchdog operations include basic management operations, such as turning on and refreshing the module. The following sections provide a description of how to programmatically operate the watchdog using the watchdog functions.

## Watchdog procedure

The following is an overview of the Watchdog process. More details instructions relating to the notes indicated in the diagrams can be found below. Note that a number of API functions will be used as part of these descriptions. For a more details explanation of these functions, refer to the relevant API reference in Chapter 6.






If a Watchdog timeout occurs and the WDT_Overwrite parameter = 0, all the Analog Output values on the I-9028U will be switched to Safe Values, and any operations where an output value needs to be written will not be accepted until the Watchdog is reset. In contrast, if a timeout occurs and the WDT_Overwrite parameter = 1, any output value that is written will be accepted and will reset the Watchdog.

For more details of how to use the Watchdog function, follow the processes described below.

**Note 1: Configuring a Safe Value.**

**Note 2: Configuring the Watchdog timeout value and enabling the Watchdog**

**Note 3: Refreshing the Watchdog timer.**

**Note 4: Checking whether a Watchdog timeout has occurred**

**Note 5: Checking the communication status if a Watchdog timeout has occurred**

**Note 6: Writing the AO value when a WDT timeout has occurred and the AO is continually set to the Safe Value**

# 5. API References

ICPDAS supplies a range of C API functions for the I-8024UW/I-8028UW/I-9024U/I-9028U module. When developing a custom program, refer to either the i8028.h header file, or the API functions described in the following sections for more detailed information.

The following is an overview of the functions provided in the LinPAC library - libi8k.a. Detailed information related to individual functions can be found in the following sections.

### API naming

| Platform | Product included | API prefix characters |
|---|---|---|
| Linux | I-8028UW | "i8028U_" + function name |
| | I-9028U | "i9028U_" + function name |

| Function | Description |
|---|---|
| i8028U_Init | Used to initialize the driver and confirm the hardware ID. |
| i8028U_GetFirmwareVersion | Used to retrieve the version number for the FPGA firmware of the I-9028 module and is used for troubleshooting purposes. |
| i8028U_GetLibVersion | Used to retrieve the version number for the I-9028 currently installed on the I-9028 module and is used for troubleshooting purposes. |
| i8028U_ReadAO_GainOffset | Used to retrieve the current gain and offset values for a specified input type and channel. |
| i8028U_WriteAOHex | Used to write the Analog Output value for a specified channel in hexadecimal format. |
| i8028U_WriteAO | Used to write the Analog Output value for a specified channel in decimal format. |
| i8028U_ReadAOHex | Used to read the Analog Output value for a specified channel in hexadecimal format. |
| i8028U_ReadAO | Used to read the Analog Output value for a specified channel |
| i8028U_SetPowerOnEnStatus | Used to set the Mode for the Power-on Value. |

| | |
|---|---|
| i8028U_GetPowerOnEnStatus | Used to read the current Mode being used for the Power-on Value. |
| i8028U_WritePowerOnHex_AO | Used to write the Power -on Value for a specified channel in hexadecimal format. |
| i8028U_WritePowerOn_AO | Used to write the Power -on Value for a specified channel in decimal format. |
| i8028U_ReadPowerOnHex_AO | Used to read the current Power -on Value for a specified channel in hexadecimal format. |
| i8028U_ReadPowerOn_AO | Used to read the current Power -on Value for a specified channel in decimal format. |
| i8028U_WriteSafeHex_AO | Used to write the Safe Value for a specified channel in hexadecimal format. |
| i8028U_WriteSafe_AO | Used to write the Safe Value for a specified channel in decimal format. |
| i8028U_ReadSafeHex_AO | Used to read the current Safe Value for a specified channel in hexadecimal format. |
| i8028U_ReadSafe_AO | Used to read the current Safe Value for a specified channel in decimal format. |
| i8028U_SetModuleWDTConfig | Used to configure the attributes for the Watchdog parameter on the I-9028 module. |
| i8028U_GetModuleWDTConfig | Used to read the current attributes of the Watchdog parameter on the I-9028 module. |
| i8028U_GetModuleWDTStatus | Used to retrieve the current status of the Watchdog on the I-9028 module. |
| i8028U_ResetModuleWDT | Used to reset the status of the Watchdog on the I-9028 module. |
| i8028U_RefreshModuleWDT | Used to refresh the status of the Watchdog on the I-9028 module. |

## 5.1. i8028U_Init

This function is used to initialize the driver and confirm the hardware ID.

### Syntax

```
short i8028U_Init(
    int    slot
);
```

### Parameters

*slot:*

specifies the number of slot (1 ~ 8).

### Return Value

0 = the module in the slot is an I-8024U/I-8028U/I-9024U/I-9028U.

-1 = there is no module in this slot.

For other return values, see the Error Codes listed in Appendix A.

### Note

Before executing any functions on the I-8024UW/I-8028UW/I-9024U/I-9028U, the i8028U_Init function needs to be called once for I-8024U/I-8028U/I-9024U/I-9028U. If there are two or more I-8024UW/I-8028UW/I-9024U/I-9028U modules, you need call the i8028U_Init function for each I-8024UW/I-8028UW/I-9024U/I-9028U module individually by passing the slot number that the I-8024UW/I-8028UW/I-9024U/I-9028U module is plugged into.

### Example

**[C]**

```
int slot, err;
Open_Slot(slot);
err = i8028U_Init(slot);
```

## 5.2. i8028U_GetFirmwareVersion

This function is used to retrieve the version number of the FPGA firmware for a specified module. Note that this function is only used for troubleshooting or recording purposes.

### Syntax

```
short i8028U_GetFirmwareVersion(
    int    slot,
    short* ver
);
```

### Parameters

*slot:*

specifies the slot number (1 ~ 8).

**ver:*

[output] The version number of the primary FPGA firmware for the module..

### Return Value

please refer the Error Code.

### Example

**[C]**

```
short ver = 0, slot = 2;
Open_Slot(slot);
i8028U_GetFirmwareVersion (slot, &ver);
```

## 5.3. i8028U_GetLibVersion

This function is used to retrieve the version number of the 9028 library file currently installed on the I-9028 module. Note that this function is only used for troubleshooting or recording purposes.

### Syntax

```
short i8028U_GetLibVersion(void);
```

### Parameters

None

### Return Value

The version number of the library file currently being used on the module.

### Example

**[C]**

```
short slot=2, ver;
Open_Slot(slot);
ver= pac_i8028U_GetLibVersion();
```

## 5.4. i8028U_ReadAO_GainOffset

This function is used to retrieve the gain and offset values for a specified input type and channel on the I-9028U module.

### Syntax

```
void i8028U_ReadAO_GainOffset(
    int    slot,
    int    ch,
    short    gain,
    unsigned short    *gVal,
    short    *oVal
);
```

### Parameters

*slot:*

specifies the slot number (1 ~ 8).

*ch:*

specifies the channel number (0 ~ 7).

*gain:*

specifies the type code for the gain (0 ~ 5).

0: 0 ~ 5 V

1: 0 ~ 10 V

2: +/-5 V

3: +/-10 V

4: +/-20 mA

5: 0 ~ 20mA

**gVal:*

[output] the gain value for the input range.

**oVal:*

[output] the offset value for the input range.

## Return Value

None

## Example

### [C]

```
unsigned slot=2, short gVal = 0;
short oVal = 0;
int ch, gain;
Open_Slot(slot);
i8028U_ReadAO_GainOffset(slot, ch, gain, &gVal, &oVal);
```

## 5.5. i8028U_WriteAOHex

This function is used to write the Analog Output value for a specified channel in hexadecimal format.

### Syntax

```
short i8028U_WriteAOHex(
    int    slot,
    int    ch,
    short    gain,
    short    aoHex
);
```

### Parameters

*slot:*

specifies the slot number (1 ~ 8).

*ch:*

specifies the channel number (0 ~ 7).

*gain:*

specifies the input type (0 ~ 5), where:

0: 0 ~ 5 V

1: 0 ~ 10 V

2: +/-5 V

3: +/-10 V

4: +/-20 mA

5: 0 ~ 20 mA

*aoHex:*

specifies the data in hexadecimal format.

### Return Value

Please refer the Error Code.

## Example

### [C]

```
int slot, ch;
short gain,hVal;
Open_Slot(slot);
i8028U_WriteAOHex(slot, ch, gain, hVal);
```

## 5.6. i8028U_WriteAO

This function is used to write the Analog Output value for a specified channel in decimal format.

### Syntax

```
short i8028U_WriteAO(
    int    slot,
    int    ch,
    short    gain,
    float    aoData
);
```

### ameters

*slot:*

specifies the slot number (1 ~ 8).

*ch:*

specifies the channel number (0 ~ 7).

*gain:*

specifies the input type (0 ~ 5), where:

0: 0 ~ 5 V

1: 0 ~ 10 V

2: +/-5 V

3: +/-10 V

4: +/-20 mA

5: 0 ~ 20mA

*aoData:*

the data in decimal format.

### Return Value

0 = No Error

For other return values, see the Error Codes listed in Appendix A.

## Example

### [C]

```
int slot=2, ch, val;
short gain;
Open_Slot(slot);
i8028U_WriteAO(slot, ch, gain, val);
```

## 5.7. i8028U_ReadAOHex

This function is used to read the current Analog Output value from a specified channel in hexadecimal format.

### Syntax

```
void i8028U_ReadAOHex(
    int    slot,
    int    ch,
    short    *gain,
    short    *aoHex
);
```

### Parameters

*slot:*

specifies the slot number (1 ~ 8).

*ch:*

specifies the channel number (0 ~ 7).

**gain:*

[Output] specifies the input type (0 ~ 5), where:

0: 0 ~ 5 V

1: 0 ~ 10 V

2: +/-5V

3: +/-10 V

4: +/-20 mA

5: 0 ~ 20 mA

**aoHex:*

[Output] the data in hexadecimal format.

### Return Value

None

## Example

### [C]

```
int slot=2, ch ,gain;
shor thVal;
Open_Slot(slot);
i8028U_ReadAOHex(slot, ch, &gain, &hVal);
```

## 5.8. i8028U_ReadAO

This function is used to read the Analog Output value from a specified channel in decimal format.

### Syntax

```
void i8028U_ReadAO(
    int    slot,
    int    ch,
    short    *gain,
    float    *ao
);
```

### Parameters

*slot:*

specifies the slot number (1 ~ 8).

*ch:*

specifies the channel number (0 ~ 7).

**gain:*

[Output] specifies the input type (0 ~ 5), where:

0: 0 ~ 5 V

1: 0 ~ 10 V

2: +/-5 V

3: +/-10 V

4: +/-20 mA

5: 0 ~ 20 mA

**ao:*

[Output] the data in decimal format.

### Return Value

None

## Example

### [C]

```
int slot=2, ch, gain;
float fVal = 0.0;
Open_Slot(slot);
i8028U_ReadAO(slot, ch, &gain, &fVal);
```

## 5.9. i8028U_SetPowerOnEnStatus

This function is used to set the Mode for the Power-on Value.

### Syntax

```
short i8028U_SetPowerOnEnStatus(
    int    slot,
    short    status
);
```

### Parameters

*slot:*

specifies the slot number (1 ~ 8).

*status:*

specifies the type for the Power-on Mode (1 - 2), where:

1: Power-on Value Mode

2: Retentive Mode

### Return Value

0 = No Error

For other return values, see the Error Codes listed in Appendix A.

### Example

**[C]**

```
int slot=2, type;
Open_Slot(slot);
i8028U_SetPowerOnEnStatus(slot, type);
```

## 5.10. i8028U_GetPowerOnEnStatus

This function is used to read the current Mode being used for the Power-on Value.

### Syntax

```
short i8028U_GetPowerOnEnStatus(
    int    slot,
    short * status
);
```

### Parameters

*slot:*

specifies the slot number (1 ~ 8).

**status:*

[output] specifies the type for the Power-on Mode (1 ~ 2), where:

1: Power-on Value Mode

2: Retentive Mode

### Return Value

returns the type for Power-on Mode (1 ~ 2) where:

1: Power-on Value Mode

2: Retentive Mode

### Example

**[C]**

```
int slot=2, status;
Open_Slot(slot);
i8028U_GetPowerOnEnStatus(slot, &status);
```

## 5.11. i8028U_WritePowerOnHex_AO

This function is used to write the Power-on Value for a specified channel in hexadecimal format.

### Syntax

```
short i8028U_WritePowerOnHex_AO(
    int    slot,
    int    ch,
    short    gain,
    short    aoHex
);
```

### Parameters

*slot:*

specifies the slot number (1 ~ 8).

*ch:*

specifies the channel number (0 – 7)

*gain:*

specifies the input type (0 - 4), where:

0: 0 ~ 5 V

1: 0 ~ 10 V

2: +/-5 V

3: +/-10 V

4: +/-20 mA

5: 0 ~ 20 mA

*aoHex:*

the data in hexadecimal format.

### Return Value

0 = No Error

For other return values, see the Error Codes listed in Appendix A.

## Example

### [C]

```
int slot=2, ch;
short gain,hVal;
Open_Slot(slot);
i8028U_WritePowerOnHex_AO(slot, ch, gain, hVal);
```

## 5.12. i8028U_WritePowerOn_AO

This function is used to write the Power-on Value for a specified channel in decimal format.

### Syntax

```
short i8028U_WritePowerOn_AO(
    int    slot,
    int    ch ,
    short    gain,
    float    aoData
);
```

### Parameters

*slot:*

specifies the slot number (1 ~ 8).

*ch:*

specifies the channel number (0 – 7).

*gain:*

specifies the input type (0 - 5), where:

0: 0~ 5 V

1: 0~ 10V

2: +/- 5V

3: +/-10V

4: +/-20 mA

5: 0 ~ 20mA

*aoData:*

the data in decimal format.

### Return Value

0 = No Error,

For other return values, see the Error Codes listed in Appendix A.

## Example

### [C]

```
int slot=2, ch;
short gain ;
float fVal;
Open_Slot(slot);
i8028U_WritePowerOn_AO(slot, ch , gain, fVal);
```

## 5.13. i8028U_ReadPowerOnHex_AO

This function is used to read the Power-on Value from a specified channel in hexadecimal format.

### Syntax

```
void i8028U_ReadPowerOnHex_AO(
    int    slot,
    int    ch,
    short    *gain,
    short    *aoHex
);
```

### Parameters

*slot:*

specifies the slot number (1 ~ 8).

*ch:*

specifies the channel number (0 – 7).

**gain:*

[Output] specifies the input type (0 - 5), where:

0: 0 ~ 5 V

1: 0 ~ 10 V

2: +/-5V

3: +/-10 V

4: +/-20 mA

5: 0 ~ 20 mA

**aoHex:*

[Output] the data in hexadecimal format.

### Return Value

None

## Example

### [C]

```
int slot=2, ch;
short gain ,hVal;
Open_Slot(slot);
i8028U_ReadPowerOnHex_AO(slot, ch, &gain, &hVal);
```

## 5.14. i8028U_ReadPowerOn_AO

This function is used to read the Power-on Value from a specified channel in decimal format.

### Syntax

```
void i8028U_ReadPowerOn_AO(
    int    slot,
    int    ch,
    short    *gain,
    float    *aoData
);
```

### Parameters

*slot:*

specifies the slot number (1 ~ 8).

*ch:*

specifies the channel number (0 – 7).

**gain:*

[Output] specifies the input type (0 - 5), where:

0: 0 ~ 5 V

1: 0 ~ 10 V

2: +/-5V

3: +/-10 V

4: +/-20 mA

5: 0 ~ 20 mA

**aoData:*

[Output] the data in decimal format.

### Return Value

None

## Example

### [C]

```
int slot=2, ch;
short gain;
float fVal;
Open_Slot(slot);
i8028U_ReadPowerOn_AO(slot, ch, &gain, &fVal);
```

## 5.15. i8028U_WriteSafeHex_AO

This function is used to write the Safe Value for a specified channel in hexadecimal format.

### Syntax

```
short i8028U_WriteSafeHex_AO(
    int    slot,
    int    ch,
    short    gain,
    short    aoHex
);
```

### Parameters

*slot:*

specifies the slot number (1 ~ 8).

*ch:*

specifies the channel number (0 – 7).

*gain:*

specifies the input type (0 - 4), where:

0: 0 ~ 5 V

1: 0 ~ 10 V

2: +/-5 V

3: +/-10 V

4: +/-20 mA

5: 0 ~ 20 mA

*aoHex:*

the data in hexadecimal format.

### Return Value

0 = No Error

For other return values, see the Error Codes listed in Appendix A.

## Example

### [C]

```
int slot=2, ch;
short gain ,hVal;
Open_Slot(slot);
i8028U_WriteSafeHex_AO(slot, ch, gain, hVal);
```

## 5.16. i8028U_WriteSafe_AO

This function is used to write the Safe Value for a specified channel in decimal format.

### Syntax

```
short i8028U_WriteSafe_AO(
    int    slot,
    int    ch,
    short    gain,
    float    aoData
);
```

### Parameters

*slot:*

specifies the slot number (1 ~ 8).

*ch:*

specifies the channel number (0 ~ 7).

*gain:*

specifies the input type (0 ~ 5), where:

0: 0 ~ 5 V

1: 0 ~ 10 V

2: +/-5V

3: +/-10 V

4: +/-20 mA,

5: 0 ~ 20 mA

*aoData:*

the data in decimal format.

### Return Value

0 = No Error

For other return values, see the Error Codes listed in Appendix A.

## Example

### [C]

```
int slot=2, ch;
short gain;
float fVal;
Open_Slot(slot);
i8028U_WriteSafe_AO(slot, ch, gain, fVal);
```

## 5.17. i8028U_ReadSafeHex_AO

This function is used to read the Safe Value from a specified channel in hexadecimal format.

### Syntax

```
void i8028U_ReadSafeHex_AO(
    int    slot,
    int    ch,
    short    *gain,
    short    *aoHex
);
```

### Parameters

*slot:*

specifies the slot number (1 ~ 8).

*ch:*

specifies the channel number (0 ~ 7).

**gain:*

[Output] specifies the input type (0 ~ 5), where:

0: 0 ~ 5 V

1: 0 ~ 10 V

2: +/-5 V

3: +/-10 V

4: +/-20 mA

5: 0 ~ 20 mA

**aoHex:*

[Output] the data in hexadecimal format.

### Return Value

None

## Example

### [C]

```
int slot=2, ch;
short gain ,hVal;
Open_Slot(slot);
i8028U_ReadSafeHex_AO(slot, ch, &gain, &hVal);
```

## 5.18. i8028U_ReadSafe_AO

This function is used to read the Safe Value from a specified channel in decimal format.

### Syntax

```
void i8028U_ReadSafe_AO(
    int    slot,
    int    ch,
    short    *gain,
    float    *aoData
);
```

### Parameters

*slot:*

specifies the slot number (1 ~ 8).

*ch:*

specifies the channel number (0 ~ 7).

**gain:*

[Output] specifies the input type (0 ~ 5), where:

0: 0 ~ 5 V

1: 0 ~ 10 V

2: +/-5 V

3: +/-10 V

4: +/-20 mA

5: 0 ~ 20 mA

**aoData:*

[Output] the data in decimal format.

### Return Value

None

## Example

### [C]

```
int slot=2, ch;
short gain;
float fVal;
Open_Slot(slot);
i8028U_ReadSafe_AO(slot, ch, &gain, &fVal);
```

## 5.19. i8028U_SetModuleWDTConfig

This function is used to configure the attributes for the Watchdog parameter on the I-9028 module.

### Syntax

```
short i8028U_SetModuleWDTConfig(
    int    slot,
    short    enStatus,
    unsigned long    wdtTimeout,
    int    ifWDT_Overwrite
);
```

### Parameters

*slot:*

specifies the slot number (1 ~ 8).

*enStatus:*

sets the status of the Watchdog (0 or 1), where:

0: disables the Watchdog

1: enables the Watchdog

*wdtTimeout:*

sets duration of the Watchdog timeout in hexadecimal format (0~0xff), which is equal to 0 ~ 25.5 seconds

*ifWDT_Overwrite:*

determines whether or not an AO value can be written if a WDT timeout is currently active (0 or 1), where:

0: an AO value cannot be written if a WDT timeout is currently active

1: an AO can be written even if a WDT timeout is currently active

### Return Value

0 = No Error

For other return values, see the Error Codes listed in Appendix A.

## Example

### [C]

```
int slot=2;
short enStatus;
unsigned long wdtTimeout;
int ifWDT_Overwrite;
Open_Slot(slot);
i8028U_SetModuleWDTConfig(slot, enStatus, wdtTimeout, ifWDT_Overwrite);
```

## 5.20. i8028U_GetModuleWDTConfig

This function is used to read the current attributes of the Watchdog parameter on the I-9028 module.

### Syntax

```
short i8028U_GetModuleWDTConfig(
    int    slot,
    short    *enStatus,
    unsigned long    *wdtTimeout,
    int    *ifWDT_Overwrite
);
```

### Parameters

*slot:*

specifies the slot number (1 ~ 8).

**enStatus:*

[Output] the current status of the Watchdog (0 or 1), where:

0: Watchdog disabled

1: Watchdog enabled

**wdtTimeout:*

[Output] the current duration that is set for the Watchdog timeout in hexadecimal format (0~0xff), which is equal to 0 ~ 25.5 seconds

**ifWDT_Overwrite:*

[Output] specifies whether or not an AO value can be written if a WDT timeout is currently active (0 or 1), where:

0: an AO value cannot be written if a WDT timeout is currently active

1: an AO can be written even if a WDT timeout is currently active

### Return Value

0 = No Error

For other return values, see the Error Codes listed in Appendix A.

## Example

### [C]

```
int slot=2;
short enStatus;
unsigned long wdtTimeout;
int ifWDT_Overwrite;
Open_Slot(slot);
i8028U_GetModuleWDTConfig(slot, &enStatus, &wdtTimeout, &ifWDT_Overwrite);
```

## 5.21. i8028U_GetModuleWDTStatus

This function is used to retrieve the current status of the Watchdog on the I-9028 module.

### Syntax

```
short i8028U_GetModuleWDTStatus(
    int    slot
);
```

### Parameters

*slot:*

specifies the slot number (1 ~ 8).

### Return Value

Indicates the status of the Watchdog timeout (0 - 1), where:

0: a Watchdog timeout is not currently active

1: a Watchdog timeout is currently active

### Example

**[C]**

```
int slot=2 ,ret;
Open_Slot(slot);
ret = i8028U_GetModuleWDTStatus(slot);
```

## 5.22. i8028U_ResetModuleWDT

This function is used to reset the status of the Watchdog on the I-9028 module.

### Syntax

```
void i8028U_ResetModuleWDT(
    int    slot
);
```

### Parameters

*slot:*

specifies the slot number (1 ~ 8).

### Return Value

0 = No Error

For other return values, see the Error Codes listed in Appendix A.

### Example

**[C]**

```
int slot=2;
Open_Slot(slot);
i8028U_ResetModuleWDT(slot);
```

## 5.23. i8028U_RefreshModuleWDT

This function is used to refresh the status of the Watchdog on the I-9028 module.

### Syntax

```
void i8028U_RefreshModuleWDT(
    int    slot
);
```

### Parameters

*slot:*

specifies the slot number (1 ~ 8).

### Return Value

0 = No Error

For other return values, see the Error Codes listed in Appendix A.

### Example

**[C]**

```
int slot=2;
Open_Slot(slot);
i8028U_RefreshModuleWDT(slot);
```

# Appendix A. Error Code

| Error Code | Definition | Description |
|---|---|---|
| 0 | No Error | This error code indicates that there are no errors. |
| -1 | ID Error | This error code indicates that the ID of the module inserted into the specified slot is not for an I-9028U module. |
| -2 | FRAM_ERROR | |
| -3 | MODULE_STATUS_ERROR | |
| -7 | WDTTIMEOUT | |

# Appendix B. Revision History

This chapter provides revision history information to this document.

The table below shows the revision history.

| Revision | Date | Description |
|---|---|---|
| 1.0.1 | January 2018 | Initial issue |
| 2.0.0 | July 2018 | • Modify library , demo path<br>• Modify API |
| 2.0.1 | October 2020 | • Modify LinPAC SDK download path |