I-8093W/I-9093

Linux API Reference Manual

V2.0.0 July 2018

Written by Edward Ku

Edited by Anna 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

Preface

I-8093W/I-9093 are 3-axis encoder counter board. I-8093W/I-9093 encoder card has 32 bits counter and high counting rate 10Mpps. The application of I-8093W/I-9093 board is position/distance measurement, velocity measurement, feedback for motor control, hard wheel input and so on.

The information contained in this manual is divided into the following topics:

- Chapter 1, "Introduction" – This chapter provides information related to the hardware, such as the specifications, the jumper settings details and wiring information.
- Chapter 2, "Quick Start" – This chapter provides information on how to get started, an overview of the location of the demo programs, a "Getting Started Guide", and an outline of the calibration process.
- Chapter 3, "Compare Trig Out" – This chapter introduces the attributes related to the Compare Trig Out function, the programming procedures, and demo programs.
- Chapter 4, "API References" – This chapter provides some troubleshooting solutions should you encounter any problems while operating the I-8093W/9093.

1. Introduction

I-8093W is a 3-axis high speed encoder module. Its each axis can be independently configured as one of A/B Phase, Pulse/Direction and CW/CCW input mode.

I-9093 is also a 3-axis encoder which included A/B Phase, Pulse/Direction and CW/CCW input mode with compare trigger output function.
It can generate a periodic trigger signal when the motor reaches a specified position.
The specified position is called a breakpoint and is similar to a switch that is triggered after the motor passes a certain position.

The high-end specifications of I-8093W/I-9093 and complete software support make it ideal for wide range applications in position measurement of motion systems for industrial and laboratory environment.

Applicable Platform table

The following table shows which platform the module applies to.

Platform	OS	Module
XPAC	XP-8000(WES)	I-8093W
	XP-8000-Atom (WES)	I-8093W
	XP-8000-WES7 (WES7)	I-8093W
	XP-8000-CE6 (WinCE 6.0)	I-8093W
	XP-8000-Atom-CE6 (WinCE 6.0)	I-8093W
	XP-9000-WES7(WES7)	I-9093
WinPAC	WP-8000 (CE 5.0/7.0)	I-8093W
	WP-9000-CE7 (CE 7.0)	I-9093
LinPAC	LinPAC-8000(Linux kernel 3.2/4.4)	I-8093W
	LinPAC-9000(Linux kernel 3.2/4.4)	I-9093
IPAC	iPAC-8000 (MiniOS7)	I-8093W
	I-8000 (MiniOS7)	I-8093W

Features

Model	I-8093W	I-9093
Encoder Axis	3-axis	
Encoder Counter	32-bit	
Encoder counting mode	A/B Phase , Dir/Pulse , CW/CCW	
Input Level	5V/12V/24V	
Max. Speed	4M hz	6M hz
System LED Indicator	1 LED as Power Indicator 9 LED as Status Indicator	1 LED as Power Indicator 12 LED as Status Indicator
Dimension (L x W x H)	102 mm x 30 mm x 115 mm	144 mm x 30.3 mm x 134 mm

The difference between I-8093W and I-9093

Functions	I-8093W	I-9093
CW/CCW	✓	✓
Dir/Pulse	✓	✓
A/B Phase	✓	✓
Read Encoder	✓	✓
XOR Bit	✓	✓
Index Latched Status	✓	×
Read Frequency	✓	×
External Trig Mode	×	✓
Compared Trig Out	×	✓
Previous Trig Steps	×	✓
Low Pass Filter	×	✓

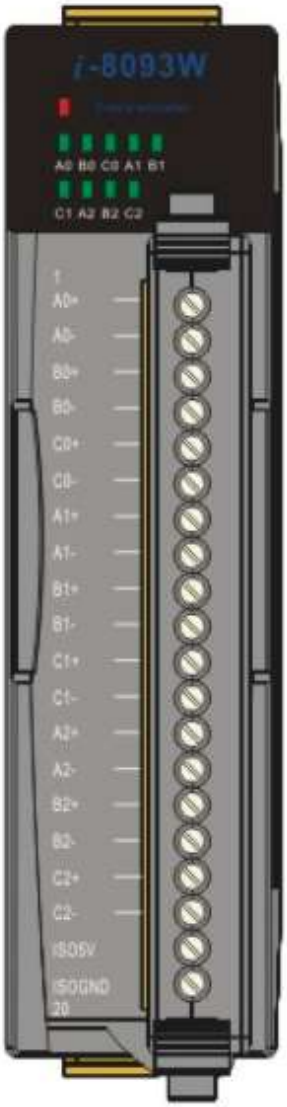
1.1. Specifications

Model		I-8093W	I-9093
Encoder Input			
Encoder Axis		3-axis	
Encoder Counter		32-bit	
Encoder Mode		A/B Phase , CW/CCW , Pulse/Dir	
Input Level	5V	Logic High: 4 V ~ 5 V Logic Low: 0 V ~ 2 V (Jumper Select)	Logic High:3.5 V ~ 5 V Logic Low:0.8V Max (Jumper Select)
	12V	Logic High: 5 V ~ 12 V Logic Low: 0 V ~ 2 V (external resistor 1 K ohm)	Logic High: 5 V ~ 12 V Logic Low: 0 V ~ 2 V (external resistor 1 K ohm)
	24V	Logic High: 7 V ~ 24 V. Logic Low: 0 V ~ 2 V (Jumper Select)	Logic High:10 V ~ 24 V Logic Low:0.8V Max (Jumper Select)
Max. Speed	Quadrant	1 MHz	2 MHz Max.
	CW/CCW	4 MHz	6 MHz
	Pulse/Dir	4 MHz	6 MHz
Programmable Digital Filter		--	1 ~ 250 µs
A/B/C signal isolation		2500 VDC	2500 VDC
External Input			
Channel		--	3
ON Voltage Level		--	+3.5 VDC ~ +5 VDC Or 10 VDC ~ 24 VDC(Jumper Select)
OFF Voltage Level		--	+0.8 VDC Max.
Trigger Output			
Channel		--	3
Trigger Pulse Width		--	10 uS~ 128 uS
Load Voltage		--	5 ~ 48 V
Max Load Current		--	100 mA
LED Indicators			
System LED Indicator		1 LED as Power Indicator 9 LED as Status Indicator	1 LED as Power Indicator 12 LED as Status Indicator
Isolation			
Intra-module Isolation, Field-to-Logic		2500 Vrms	3000 VDC

EMS Protection		
ESD(IEC 61000-4-2)	±4 kV Contact for Each Terminal	±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)	102 mm x 30 mm x 115 mm	144 mm x 30.3 mm x 134 mm
Environment		
Operating Temperature	-25 °C ~ +75°C	
Storage Temperature	-30 ~ 85 °C	-40 ~ +85°C
Humidity	5 ~ 95 % RH, Non-condensing	10 ~ 90% RH, non-condensing

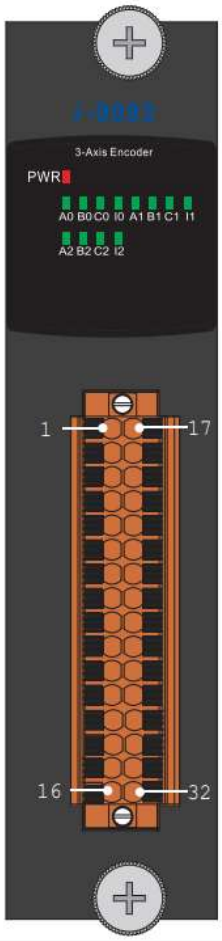
1.2. Pin Assignments

I-8093W



Terminal No.	Pin Assignment
01	A0+
02	A0-
03	B0+
04	B0-
05	C0+
06	C0-
07	A1+
08	A1-
09	B1+
10	B1-
11	C1+
12	C1-
13	A2+
14	A2-
15	B2+
16	B2-
17	C2+
18	C2-
19	ISO5V
20	ISOGND

I-9093

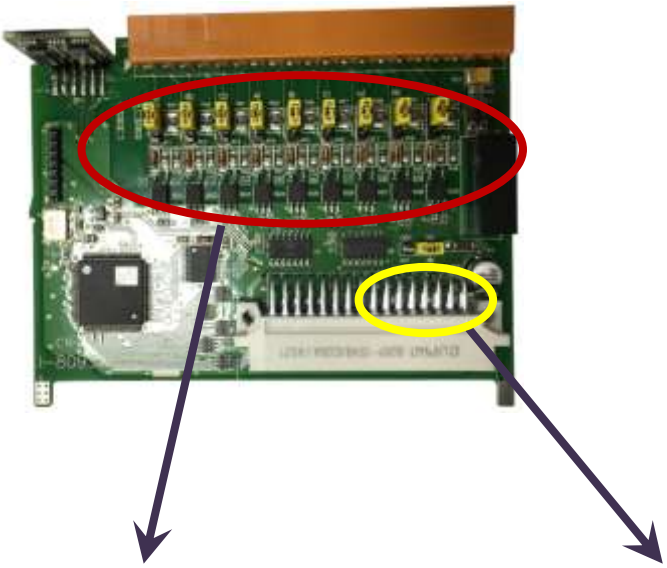


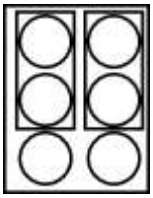
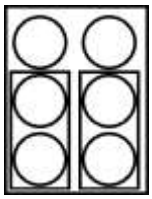
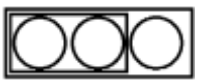
Pin Assignment	Terminal No.		Pin Assignment
A0+	01	17	A0-
B0+	02	18	B0-
C0+	03	19	C0-
I0+	04	20	I0-
Trig0	05	21	ISO.GND
A1+	06	22	A1-
B1+	07	23	B1-
C1+	08	24	C1-
I1+	09	25	I1-
Trig1	10	26	ISO.GND
A2+	11	27	A2-
B2+	12	28	B2-
C2+	13	29	C2-
I2+	14	30	I2-
Trig2	15	31	ISO.GND
ISO5V	16	32	ISO.GND

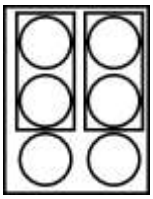
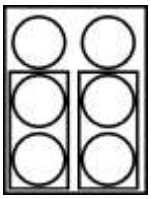
32-pin Connector

1.3. Jumper Settings

I-8093W

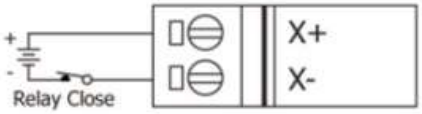
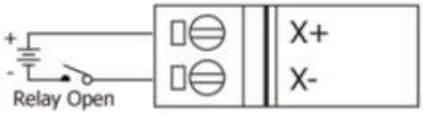
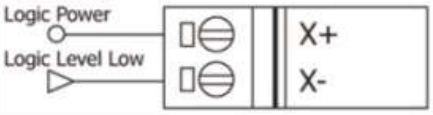
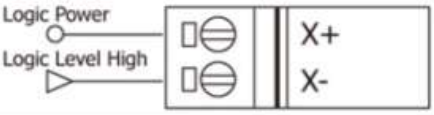
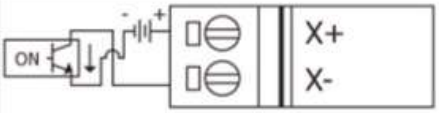
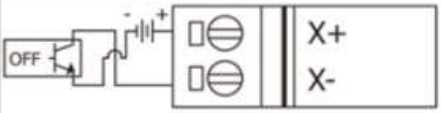
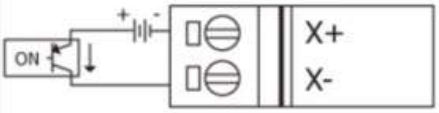
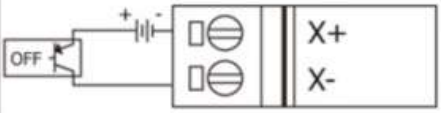


5V / 24V Input Level	Enable / Disable 5V power supply
5V	Enable
	
24V	Disable
	

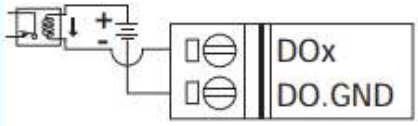
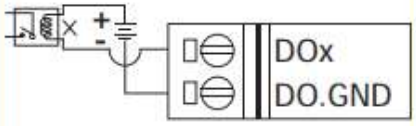
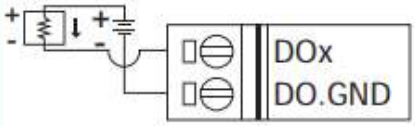
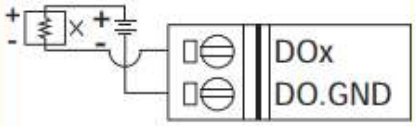
5V / 24V Input Level
5V

24V


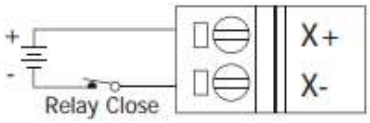
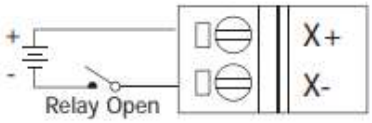
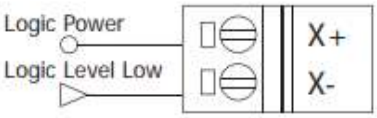
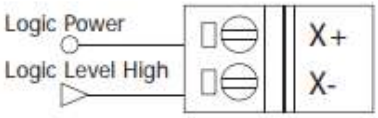
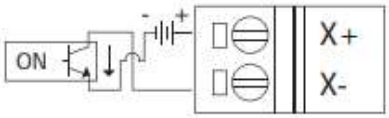
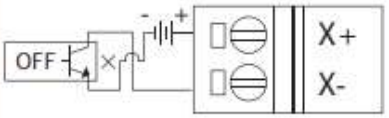
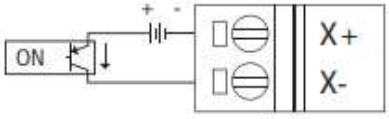
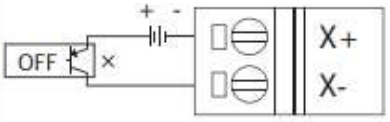
1.4. Wire Connections

I-8093W

Input Type	ON State LED ON Readback as 0	OFF State LED OFF Readback as 1
Relay Contact	Relay ON	Relay Off
		
TTL/CMOS Logic	Voltage > 4V	Voltage < 0.8V
		
NPN Output	Open Collector On	Open Collector Off
		
PNP Output	Open Collector On	Open Collector Off
		

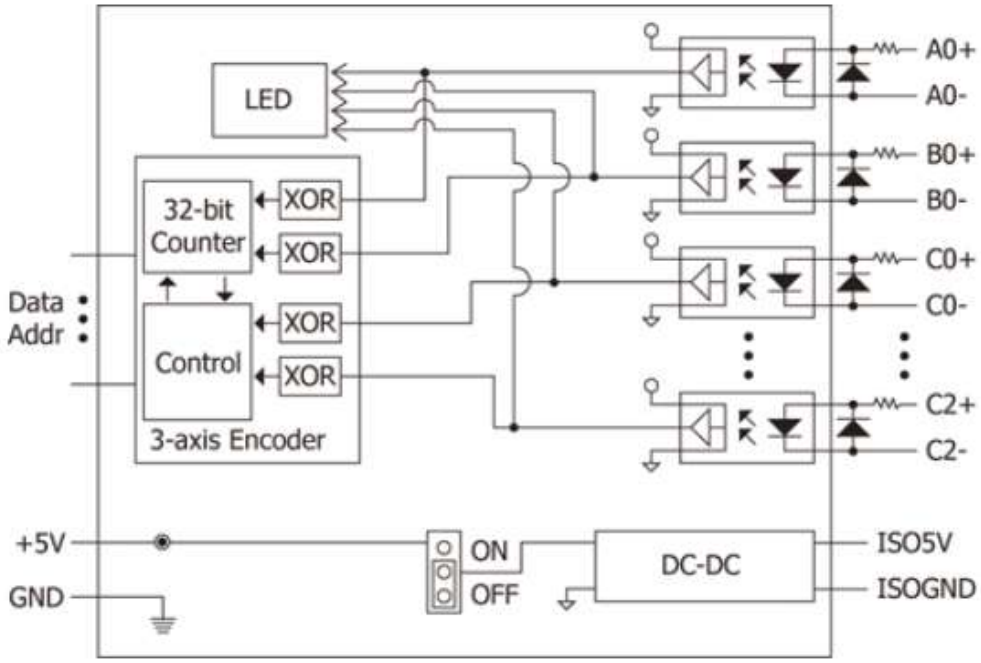
I-9093

Output Type	ON State Readback as 1	OFF State Readback as 0
Drive Relay	Relay ON	Relay OFF
		
Resistance Load		
		

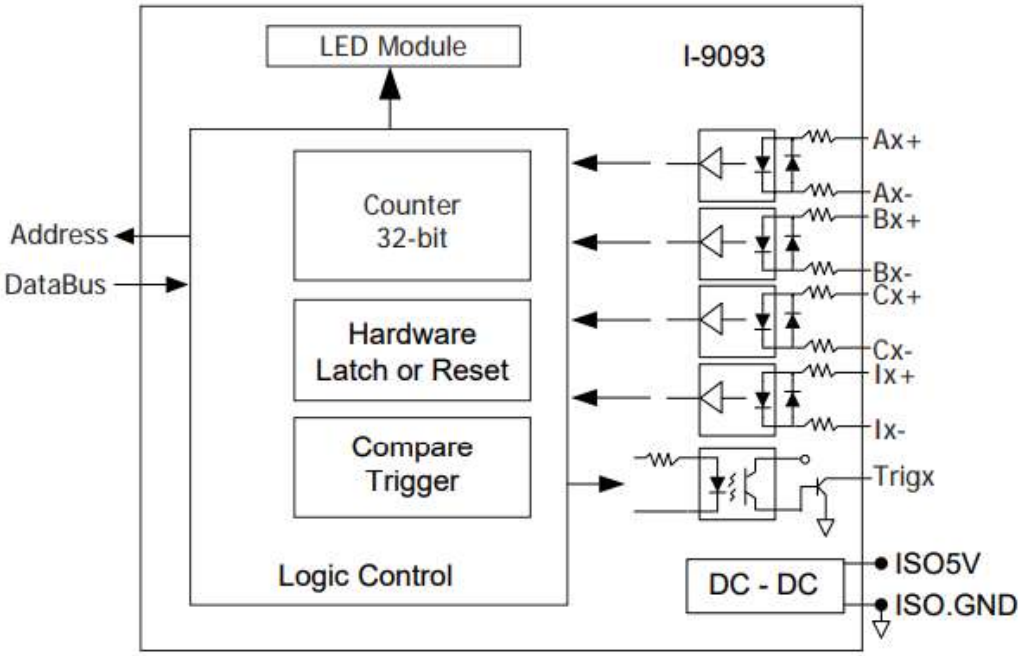
Input Type	ON State LED ON Readback as 1	OFF State LED OFF Readback as 0
Relay Contact	Relay ON	Relay OFF
		
TTL/CMOS Logic	Voltage > 4 V	Voltage < 0.8 V
		
NPN Output	Open Collector ON	Open Collector OFF
		
PNP Output	Open Collector ON	Open Collector OFF
		

1.5. Block Diagram

I-8093W



I-8014CW/I-9014C



2. Quick Start

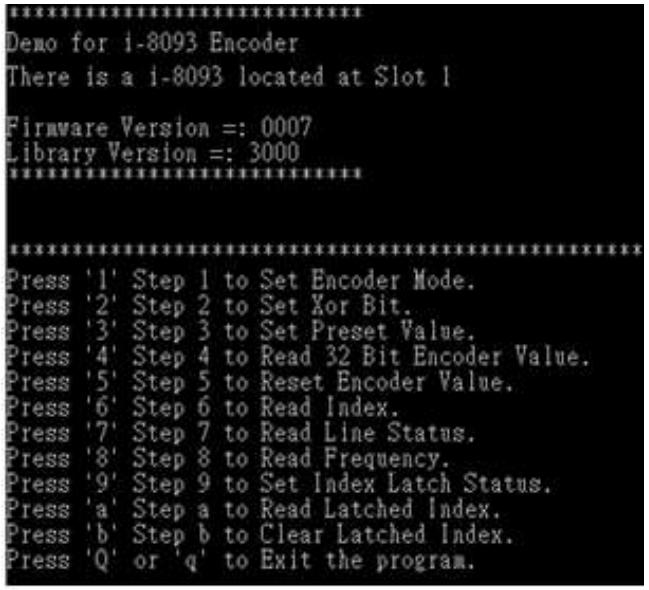
ICP DAS provides a range of demo programs for different platforms that can be used to verify the functions of the I-8093W/9093. The source code contained in these programs can also be reused in your own custom programs if needed. The executable file, can be used to retrieve the basic configuration information related to the module and to verify the functions.

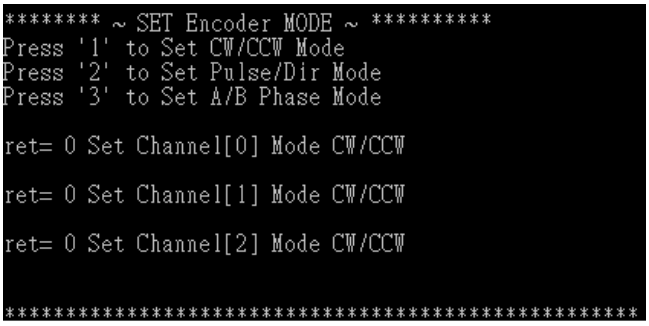
1. First, user need to download LinPAC SDK, which is includes GNU toolchain, Libraries, header, examples files, etc.
2. Check the power cable, Ethernet cable, VGA monitor, the communication cable between controller and PC has been connected well, and then check the I-8093W/9093 has been plugged in the controller.
3. Next, check the communication between controller and PC is fine, and download the demo program files to the controller.
4. User can find the related files in the product CD or below website: http://www.icpdas.com/root/product/solutions/pac/linpac/linpac-8000_download.html

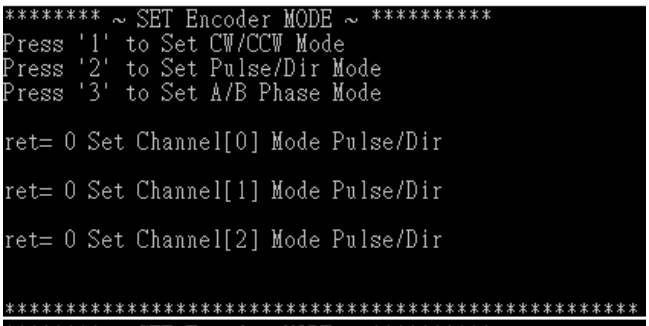
2.1. Getting start on LinPAC Controllers

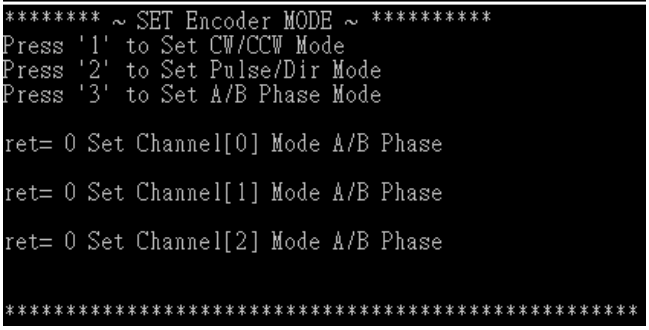
This part will show the functions of I-8093W with 'demo8093' file.

After execute 'demo8093W.exe' file you can see the functions as following picture:







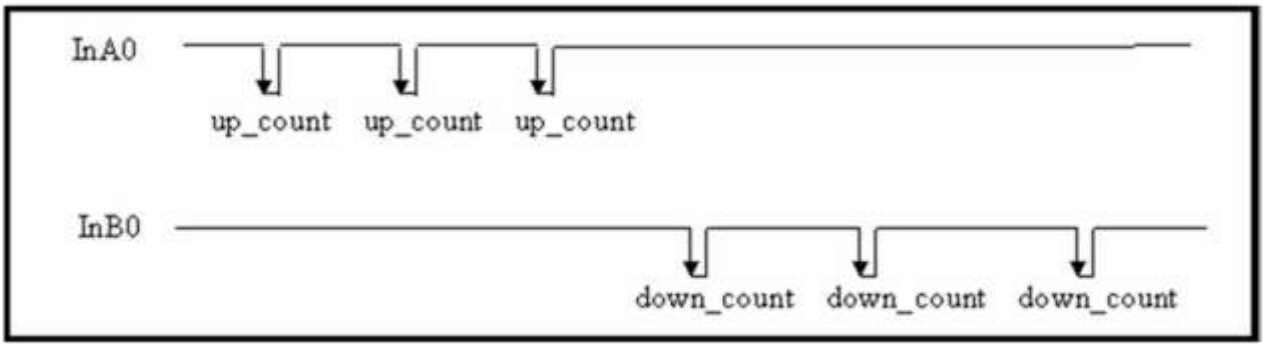


There are 4 options you can choose for each channel.

(1) (STOP , CW/CCW , Dir/Pulse , A/B Phase)

CW/CCWCounting

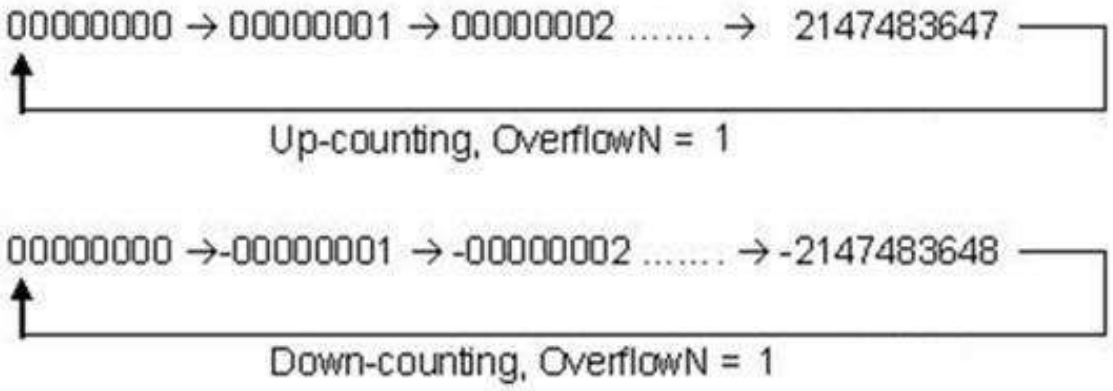
The counter operation for Up/Down mode is as follows:



When InA0 is used as an UP_clock and InB0 is used as a DOWN_clock, counter_0 will be increased by one for every falling edge of InA0 and decreased by one for every falling edge of InB0.

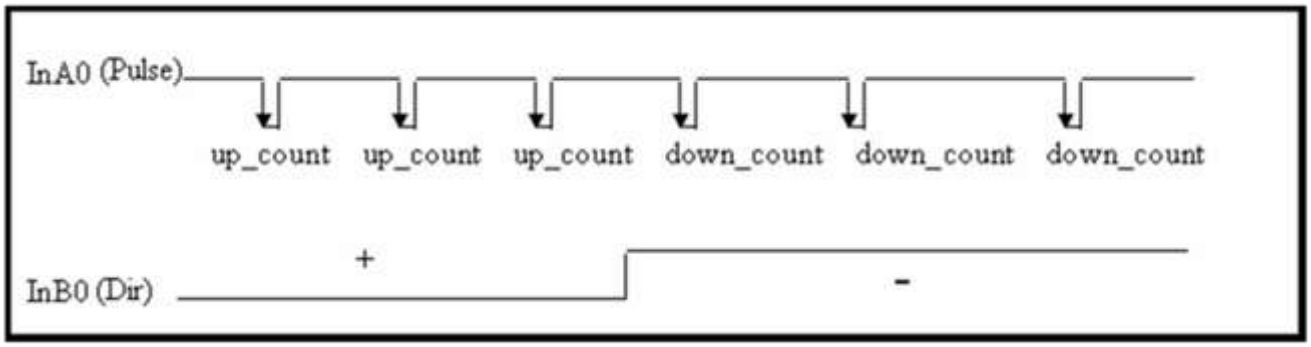
CountN	the current counter value for channel N, 32 bits wide, from -2147483648 to 2147483647
OverflowN	0 = no overflow 1 = overflow

This gives the following:



Dir/PulseCounting

The counter operation for Dir/Pulse mode is as follows:

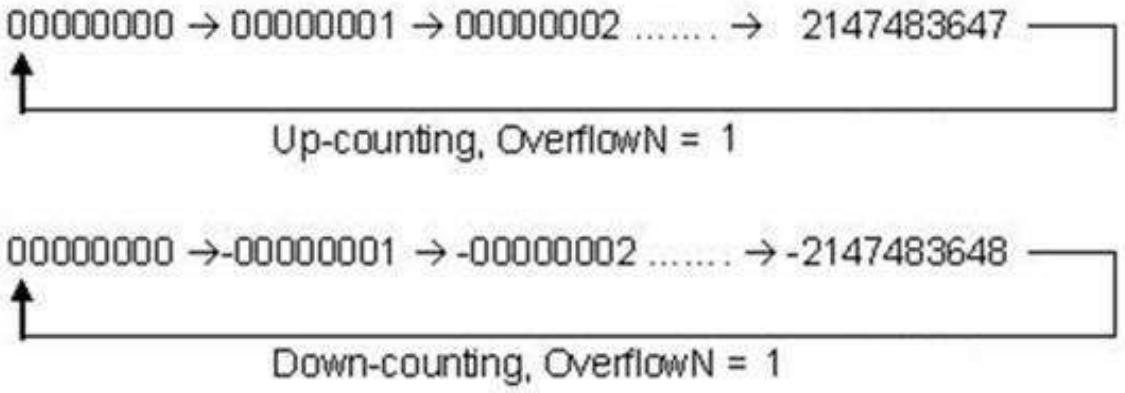


When InB0 is used as Dir:

- If InB0 is High, counter_0 will be increased by one for every falling edge of InA0.
- If InB0 is Low, counter_0 will be decreased by one for every falling edge of InA0.

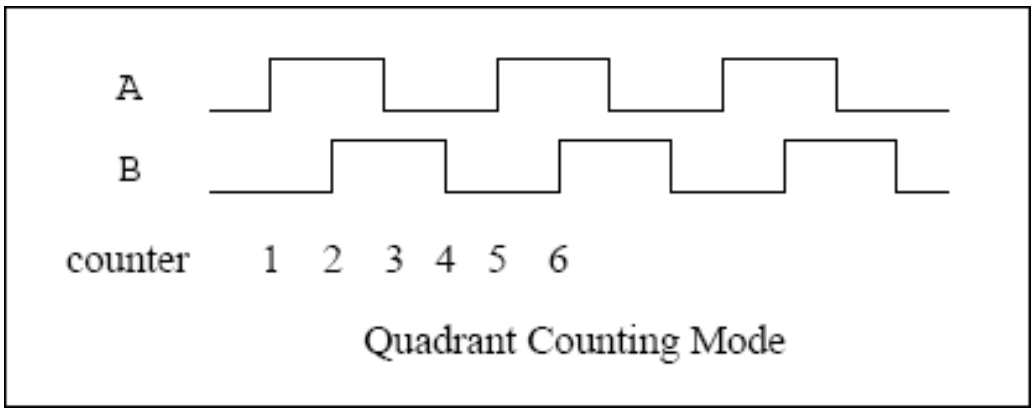
CountN	the current counter value for channel N, 32 bits wide, from -2147483648 to 2147483647
OverflowN	0 = no overflow 1 = overflow

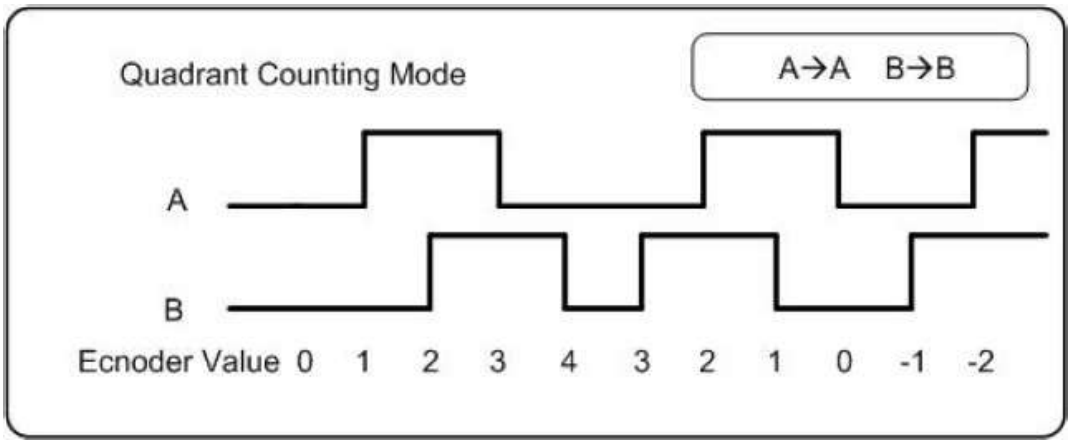
This gives the following:



A/B PhaseCounting

The counter operation for A/B Phase is as follows:





When InA0 is used as an A signal and InB0 is used as a B signal:

- Counter_0 will be increased by one when the InA0 phase leads by 90 degrees to InB0.
- Counter_0 will be decreased by one when the InA0 phase lags by 90 degrees to InB0.

(2) XOR Bit is used to change the status of Z_index.

The status of Z index is 1 and the XOR Bit is disable when shipment, enable XOR bit and the status of Z index will become 0.

*When the status of Z index = 1, the encoder value will be latched.

```
******** ~ SET XOR ~ **********
Press '0' to Set XOR bit = 0
Press '1' to Set XOR bit = 1

ret= 0 Set Channel[0] Xor 1

ret= 0 Set Channel[1] Xor 1

ret= 0 Set Channel[2] Xor 1


*****************************************************
```

(3) Preset value is used to set the set the starting position of the count.

```
******** ~ SET PRESET VALUE  ~ **********
Input CH:0 Hex Data Range 0 ~ 0xffffffff
1
Set CH:0 Preset 1
Input CH:1 Hex Data Range 0 ~ 0xffffffff
10
Set CH:1 Preset 10
Input CH:2 Hex Data Range 0 ~ 0xffffffff
100
Set CH:2 Preset 100


*****************************************************
```

```
******** ~ READ 32 BIT ENCODER VALUE ~ **********
Press 'Q' or 'q' to Go Back Main Menu.

S[1]C[0] 0000000001 00000001    S[1]C[1] 0000000016 00000010    S[1]C[2] 0000000256 00000100

S[1]C[0] 0000000001 00000001    S[1]C[1] 0000000016 00000010    S[1]C[2] 0000000256 00000100

S[1]C[0] 0000000001 00000001    S[1]C[1] 0000000016 00000010    S[1]C[2] 0000000256 00000100

S[1]C[0] 0000000001 00000001    S[1]C[1] 0000000016 00000010    S[1]C[2] 0000000256 00000100


*****************************************************
```

(4) Read Frequency

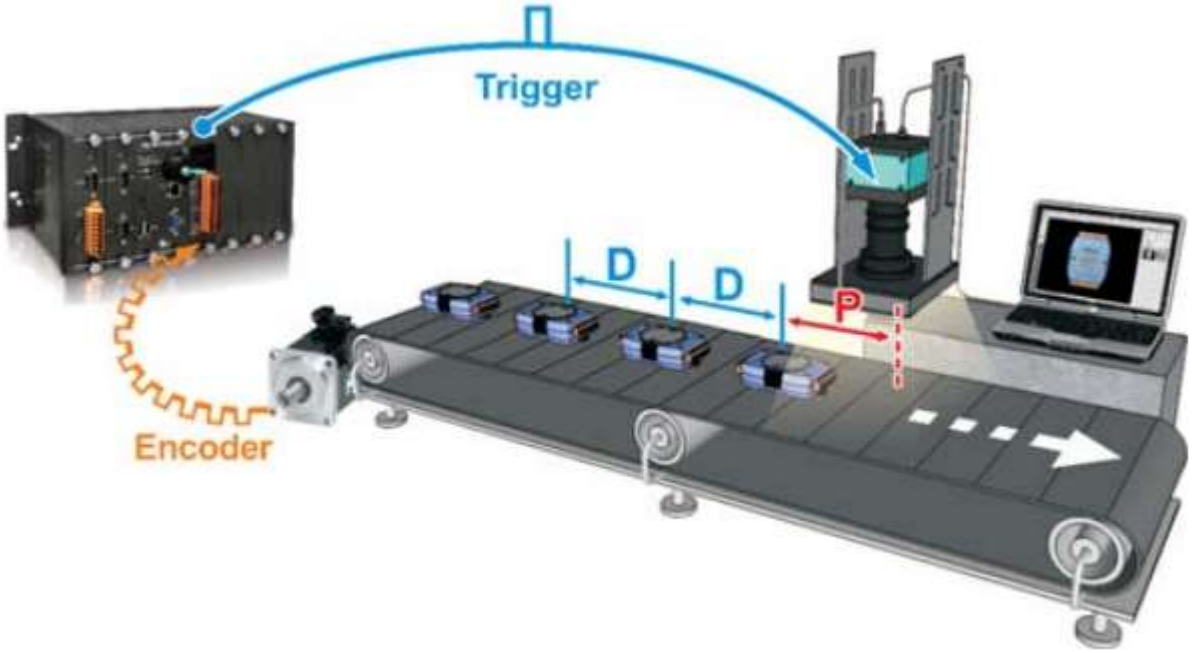
I-8093W can read frequency of input signal.

*If there is no encoder input, the frequency will be 0.093 not 0

```
******** ~ READ Frequency ~ **********
S[1]C[0] Freq= 0.093    S[1]C[1] Freq= 0.093    S[1]C[2] Freq= 0.093
S[1]C[0] Freq= 0.093    S[1]C[1] Freq= 0.093    S[1]C[2] Freq= 0.093
S[1]C[0] Freq= 0.093    S[1]C[1] Freq= 0.093    S[1]C[2] Freq= 0.093
S[1]C[0] Freq= 0.093    S[1]C[1] Freq= 0.093    S[1]C[2] Freq= 0.093
S[1]C[0] Freq= 0.093    S[1]C[1] Freq= 0.093    S[1]C[2] Freq= 0.093
S[1]C[0] Freq= 0.093    S[1]C[1] Freq= 0.093    S[1]C[2] Freq= 0.093


*****************************************************
```

3. Compare Trig Out



To use the compare trigger output function, you have to set an initial point (First Trig Position) and a trigger period of the following points (Distance).

The trigger signal is an I/O line that can be used to fire another device.

For example, when a motor reaches a certain position, the trigger signal can be used to fire the shutter of a camera to capture an image for the defect detection.

If the action of device takes time to prepare, user can use pac_i8093W_SetPreTriggerSteps to adjust the delay that caused by the machine.

For example, if first trig position is 5000, and set pac_i8093W_SetPreTriggerSteps is 100, I-9093 will output a pulse when encoder value is 4900.

All operations of position compare and trigger pulse output are automatically done by the hardware circuit.

There is no software calculation effort when the system is operating.

I-9093 makes the system design simpler, and significantly increases the system performance.

4. API References

ICPDAS supplies a range of C API functions for the I-8093W/I-9093 module. When developing a custom program, refer to either the i8093W.h header file, or the API functions described in the following sections for more detailed information.

API naming table

The following table describes the platforms and in which the product series included and the different part of function name.

The following is an overview of the functions provided in the LinPAC library-libi8k.a for use with the Linux platform.

Platform	Product included	API prefix characters
Linux	I-8093W	"i8093W_" + function name
	I-9093	"i9093_" + function name

API for both I-8093W and I-9093

Function	Description
i8093W_Init	This function can initial the I-8093W/I-9093 and can check the hardware ID.
i8093W_GetFirmwareVersion	This function use to get the firmware version of I-8093W/I-9093 hardware.
i8093W_GetLibVersion	This function use to get the library version of 8093W.lib/i8093W.dll.
i8093W_SetMode	This function use to set the operation mode of I-8093W/I-9093.
i8093W_GetMode	This function use to get the operation mode of I-8093W/I-9093.
i8093W_SetXOR	This function use to set the xor of I-8093W/I-9093 for each channel.
i8093W_GetXOR	This function use to get the xor of I-8093W/I-9093 for each channel.
i8093W_GetLineStatus	This function use to get A,B and Z status of I-8093W/I-9093.
i8093W_GetIndex	This function use to get Z index status of I-8093W/I-9093.
i8093W_Read32BitEncoder	This function use to get 32-Bit Encoder value of I-8093W/I-9093.
i8093W_ResetEncoder	This function use to reset 32-Bit Encoder value to zero.
i8093W_SetPresetValue	This function use to set 32-Bit preset value of I-8093W/I-9093.
i8093W_GetPresetValue	This function use to get 32-Bit preset value of I-8093W/I-9093.
i8093W_ClearLatchedIndex	This function use to clear the index latched status.

API for I-8093W

Function	Description
i8093W_ReadFreq (For I-8093W only)	This function use to read frequency value of encoder input signal.
i8093W_SetIndexLatchStatus (For I-8093W only)	This function use to enable/disable the index latch function of I-8093W.
i8093W_GetIndexLatchStatus (For I-8093W only)	This function use to get the index latch function status of I-8093W.

API for I-9093

Function for Windows	Description
i9093_SetExTrigMode (For I-9093 only)	This function use to set External latch mode of I-9093.
i8093_GetExTrigMode (For I-9093 only)	This function use to get External latch mode of I-9093.
i9093_SetExSignal (For I-9093 only)	This function use to set External latch signal of I-9093.
i9093_GetExSignal (For I-9093 only)	This function use to get External latch signal of I-9093.
i9093_SetPreTriggerSteps (For I-9093 only)	This function use to set steps to adjust the delay that caused by the machine.
i9093_GetPreTriggerSteps (For I-9093 only)	This function use to get the steps.
i9093_SetFirstTrigPosition (For I-9093 only)	This function use to set first trig position of I-9093.
i9093_GetFirstTrigPosition (For I-9093 only)	This function use to get first trig position of I-9093.
i9093_SetTrigDistance (For I-9093 only)	This function use to set distance of I-9093.
i9093_GetTrigDistance (For I-9093 only)	This function use to get distance of I-9093.
i9093_ReadNextPosition (For I-9093 only)	This function use to get next trig position of I-9093.
i9093_ClearNextPosition (For I-9093 only)	This function use to clear next trig position of I-9093.
i9093_ConfigTriggerOut (For I-9093 only)	This function use to enable/disable the compared trig out function of I-9093.

i9093_GetTriggerOutConfig (For I-9093 only)	This function use to get the compared trig out function status of I-9093.
i9093_SetLowPassFilter (For I-9093 only)	This function use to set the low pass filter of I-9093.
i9093_GetLowPassFilter (For I-9093 only)	This function use to get the Low Pass Filter of I-9093.
i9093_ReadIndexLatchedPosition (For I-9093 only)	This function use to get the index latched position of I-9093.
i9093_ReadExTrigLatchedPosition (For I-9093 only)	This function use to get the external trig latched position of I-9093.

4.1.i8093W_Init

This function can initial the I-8093W/I-9093 and can check the hardware ID.

Syntax

```
int i8093W_Init(
    int slot
);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

Return Values

0 = the module in the slot is an I-8093W/I-9093.

-1 = there is no I-8093W/I-9093 module in this slot.

For other return values, please refer the Error Code.

Note

Before executing any functions on the I-8093W/I-9093, the i8093W_Init function needs to be called once for I-8093W/I-9093. If there are two or more I-8093W/I-9093 modules, you need call the i8093W_Init function for each I-8093W/I-9093 module individually by passing the slot number that the I-8093W/I-9093 module is plugged into.

Example

```
int slot;
Open_Slot(slot);
i8014W_Init(slot);
```

4.2.i8093W_GetFirmwareVersion

This function use to get the firmware version of I-8093W hardware.

Syntax

```
short i8093W_GetFirmwareVersion(int slot);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

Return Values

The version number of the primary FPGA firmware for the I-8093W/I-9093 module.

Example

```
int ver, slot;
Open_Slot(slot);
ver= i8093W_GetFirmwareVersion(slot);
```

4.3.i8093W_GetLibVersion

This function use to get the library version of 8093W.lib/i8093W.dll.

Syntax

```
short i8093W_GetLibVersion(void);
```

Parameter

None

Return Values

The version number of the 8093W.lib/i8093W.dll.

Example

```
int ver, slot;
Open_Slot(slot);
ver = i8093W_GetLibVersion(void);
```

4.4.i8093W_SetMode

This function use to set the operation mode of I-8093W/I-9093.

Syntax

```
int i8093W_SetMode(
    int slot,
    int ch,
    int Mode
);
```

For Windows (CE and WES)

Parameter

slot:

specifies the slot number (1 ~ 8).

ch:

specifies the channel (0 ~ 2).

Mode:

set encoder counting mode

1 ==> CW/CCW counting mode

2 ==> Pulse/Direction counting mode

3 ==> Quadrant counting mode

Return Values

Please refer the Error Code.

Example

```
int slot , ch , mode;
Open_Slot(slot);
i8093W_SetMode(lot , ch , mode);
```

4.5.i8093W_GetMode

This function use to get the operation mode of I-8093W/I-9093.

Syntax

```
int i8093W_GetMode(
    int slot,
    int ch,
    int* Mode
);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

ch:

specifies the channel (0 ~ 2).

**Mode:*

[output] get encoder counting mode

1 ==> CW/CCW counting mode

2 ==> Pulse/Direction counting mode

3 ==> Quadrant counting mode

Return Values

Please refer the Error Code.

Example

```
int slot , ch , mode;
Open_Slot(slot);
i8093W_GetMode (lot , ch , *mode);
```

4.6.i8093W_SetXOR

This function use to set the xor of I-8093W/I-9093 for each channel.

Syntax

```
int i8093W_SetXOR(
    int slot,
    int ch,
    int Xor
);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

ch:

specifies the channel (0 ~ 2).

Xor:

to change the status of Z_index

0 : not activated

1 : activated

Return Values

Please refer the Error Code.

Example

```
int slot , ch , xor;
Open_Slot(slot);
i8093W_SetXOR(lot , ch , xor);
```

4.7.i8093W_GetXOR

This function use to get the xor of I-8093W/I-9093 for each channel.

Syntax

```
int i8093W_GetXOR(
    int slot,
    int ch,
    int* Xor
);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

ch:

specifies the channel (0 ~ 2).

**Xor:*

[output] to change the status of Z_index

0 : not activated

1 : activated

Return Values

Please refer the Error Code.

Example

```
int slot , ch , xor;
Open_Slot(slot);
i8093W_GetXOR (lot , ch , &xor);
```

4.8.i8093W_GetLineStatus

This function use to get A,B and Z status of I-8093W/I-9093.

Syntax

```
int i8093W_GetLineStatus(
    int slot,
    int ch,
    int* A_Status,
    int* B_Status,
    int* C_Status
);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

ch:

specifies the channel (0 ~ 2).

[Output] A_Status:*

[output]

0: not activated

1: activated

**B_Status:*

[output]

0: not activated

1: activated

**C_Status:*

[output]

0: not activated

1: activated

Return Values

Please refer the Error Code.

Example

```
int slot , ch , A_Status , B_Status , C_Status;
Open_Slot(slot);
i8093W_GetLineStatus(slot , ch , & A_Status , & B_Status , & C_Status);
```

4.9.i8093W_GetIndex

This function use to get Z index status of I-8093W/I-9093.

Syntax

```
int i8093W_GetIndex(
    int slot,
    int ch,
    int* index
);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

ch:

specifies the channel (0 ~ 2).

** index:*

[output]

0: not activated

1: activated

Return Values

Please refer the Error Code.

Example

```
int slot , ch , index;
Open_Slot(slot);
i8093W_GetIndex(slot , ch , &index);
```

4.10. i8093W_Read32BitEncoder

This function use to get 32-Bit Encoder value of I-8093W/I-9093.

Syntax

```
int i8093W_Read32BitEncoder( int slot , int ch , long* EnCode32);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

ch:

specifies the channel (0 ~ 2).

** EnCode32:*

[output] get encoder value.

Return Values

return Index enable/disable and Index latched status of I-8093W

0: Disable Index Latch (0), Index Not Latched (0)

1: Enable Index Latch (1), Index Not Latched (0)

2: Disable Index Latch (0), Index Latched (1)

3: Enable Index Latch (1), Index Latched (1)

For I-9093 please refer the Error Code.

Example

```
int slot , ch ;
long EnCode;
Open_Slot(slot);
i8093W_Read32BitEncoder(slot , ch , &EnCode);
```

4.11. i8093W_ResetEncoder

This function use to reset 32-Bit Encoder value to zero.

Syntax

```
int i8093W_ResetEncoder(
    int slot,
    int ch
);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

ch:

specifies the channel (0 ~ 2).

Return Values

Please refer the Error Code.

Example

```
int slot , ch ;
Open_Slot(slot);
i8093W_Read32BitEncoder(slot , ch , &EnCode);
```

4.12. i8093W_SetPresetValue

This function use to set 32-Bit preset value of I-8093W/I-9093.

Syntax

```
int i8093W_SetPresetValue(
    int slot,
    int ch,
    long presetVal
);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

ch:

specifies the channel (0 ~ 2).

presetVal:

Set preset value.

Return Values

Please refer the Error Code.

Note

Preset value is supported at lib version 0x1007 or above

Example

```
int slot , ch ;
long presetVal;
Open_Slot(slot);
i8093W_SetPresetValue(int slot, int ch, long presetVal);
```

4.13. i8093W_GetPresetValue

This function use to get 32-Bit preset value of I-8093W/I-9093.

Syntax

```
int i8093W_GetPresetValue(
    int slot,
    int ch,
    long* presetVal
);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

ch:

specifies the channel (0 ~ 2).

presetVal:

[output] get preset value.

Return Values

Please refer the Error Code.

Note

Preset value is supported at lib version 0x1007 or above

Example

```
int slot , ch ;
long presetVal;
Open_Slot(slot);
i8093W_GetPresetValue(slot , ch , &presetVal);
```

4.14. i8093W_ReadFreq

This function use to read frequency value of encoder input signal.

Syntax (For I-8093W only)

```
int i8093W_ReadFreq(
    int slot,
    int ch,
    float* freq
);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

ch:

specifies the channel (0 ~ 2).

** freq:*

[output] get frequency value.

Return Values

Please refer the Error Code.

Note

1: read frequency function supported by firmware version 3 or later version.

2: if there is no encoder input, the frequency will be 0.093 not 0

Example

```
int slot , ch ;
float freq;
Open_Slot(slot);
i8093W_ReadFreq(slot , ch , &freq);
```

4.15. i8093W_SetIndexLatchStatus

This function use to enable/disable the index latch function of I-8093W.

Syntax (For I-8093W only)

```
int i8093W_SetIndexLatchStatus(
    int slot,
    int ch,
    int ifEnableLatch
);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

ch:

specifies the channel (0 - 2).

ifEnableLatch:

to enable or disable the index latch function of i-8093.

Return Values

Please refer the Error Code.

Note

1. normally we can use i8093W_GetIndex to read the Z_index status, but sometimes it is not easy to let the encoder to match the right position, so 8093W add this function to latch the Z_index status
2. if enable the latch index function and the Z_index latched, the encoder value will also be latched at the position where Z_index latched point, so we can adjust the encoder position to the latched point.
3. disable the index latch function, i8093W_Read32BitEncoder can read the normal encoder value as usual.

Example

```
int slot, ch ;
int ifEnableLatch;
Open_Slot(slot);
i8093W_SetIndexLatchStatus(slot, ch , ifEnableLatch);
```

4.16. i8093W_GetIndexLatchStatus

This function use to get the index latch function status of I-8093W.

Syntax (For i-8093W only)

```
int i8093W_GetIndexLatchStatus(
    int slot,
    int ch,
    int* latchedStatus,
    int* ifEnableLatch
);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

ch:

specifies the channel (0 - 2).

** latchedStatus:*

[output] check z index latched or not.

**ifEnableLatch:*

[output] read enable latched of i-8093.

Return Values

Please refer the Error Code.

Example

```
int slot , ch ;
int latchedStatus ,ifEnableLatch;
Open_Slot(slot);
i8093W_GetIndexLatchStatus(slot, ch, &latchedStatus, &ifEnableLatch);
```

4.17. i8093W_ClearLatchedIndex

This function use to clear the index latched status.

Syntax

```
int i8093W_ClearLatchedIndex(
    int slot,
    int ch
);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

ch:

specifies the channel (0 - 2).

Return Values

Please refer the Error Code.

Example

```
int slot , ch ;
Open_Slot(slot);
i8093W_ClearLatchedIndex(slot, ch);
```

4.18. i9093_SetExTrigMode

This function use to set External latch mode of I-9093.

Syntax (for I-9093 only)

```
int i9093_SetExTrigMode(
    int slot,
    int ch,
    unsigned char Mode
);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

ch:

specifies the channel (0 - 2).

Mode:

0 : disable external latched mode

1 : reset encoder

2 : external latch

Return Values

Please refer the Error Code.

Example

```
int slot , ch ;
unsigned char mode;
Open_Slot(slot);
i9093_SetExTrigMode(slot, ch, mode);
```

4.19. i9093_GetExTrigMode

This function use to get External latch mode of I-9093.

Syntax (for I-9093 only)

For Windows (CE and WES)

```
int i9093_GetExTrigMode(
    int slot,
    int ch,
    unsigned char *Mode
);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

ch:

specifies the channel (0 ~ 2).

**Mode:*

[output] get status of external latched mode

0 : disable external latched mode

1 : reset encoder

2 : external latch

Return Values

Please refer the Error Code.

Example

```
int slot , ch ;
unsigned char mode;
Open_Slot(slot);
i9093_GetExTrigMode (slot, ch, &mode);
```

4.20. i9093_SetExSignal

This function use to set External latch signal of I-9093.

Syntax (for I-9093 only)

```
int i9093_SetExSignal(
    int slot,
    unsigned char Edge
);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

Edge:

0 : Falling edge

1 : rising edge

Return Values

Please refer the Error Code.

Example

```
int slot ;
unsigned char edge;
Open_Slot(slot);
i9093_SetExSignal (slot, edge);
```

4.21. i8093_GetExSignal

This function use to get External latch signal of I-9093.

Syntax (for I-9093 only)

```
int i9093_GetExSignal(
    int slot,
    unsigned char *Edge
);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

**Edge:*

[output]

0 : Falling edge

1 : rising edge

Return Values

Please refer the Error Code.

Example

```
int slot ;
unsigned char edge;
Open_Slot(slot);
i9093_GetExSignal (slot, &edge);
```

4.22. i9093_SetPreTriggerSteps

This function use to set steps to adjust the delay that caused by the machine.

Syntax (for I-9093 only)

```
int i9093_SetPreTriggerSteps(
    int slot,
    int ch,
    unsigned char Steps
);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

ch:

specifies the channel (0 ~ 2).

Steps:

Set the steps 0~255.

Return Values

Please refer the Error Code.

Example

```
int slot , ch ;
unsigned char step;
Open_Slot(slot);
i9093_SetPreTriggerSteps (slot, ch, step);
```

4.23. i9093_GetPreTriggerSteps

This function use to get the steps.

Syntax (for I-9093 only)

```
int i90093_GetPreTriggerSteps(
    int slot,
    int ch,
    unsigned char *Steps
);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

ch:

specifies the channel (0 ~ 2).

**Steps:*

[output] get steps 0~255.

Return Values

Please refer the Error Code.

Example

```
int slot , ch ;
unsigned char step;
Open_Slot(slot);
i9093_GetPreTriggerSteps (slot, ch, &step);
```

4.24. i9093_SetFirstTrigPosition

This function use to set first trig position of I-9093.

Syntax (for I-9093 only)

```
int i9093_SetFirstTrigPosition(
    int slot,
    int ch,
    unsigned long Position
);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

ch:

specifies the channel (0 ~ 2).

Position:

Set first trig position

Return Values

Please refer the Error Code.

Example

```
int slot , ch ;
unsigned long position;
Open_Slot(slot);
i8093_SetFirstTrigPosition(slot, ch , position);
```

4.25. i9093_GetFirstTrigPosition

This function use to get first trig position of I-9093.

Syntax (for I-9093 only)

```
int i9093_GetFirstTrigPosition(
    int slot,
    int ch,
    unsigned long* Position
);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

ch:

specifies the channel (0 ~ 2).

**Position:*

[output] get first trig position

Return Values

Please refer the Error Code.

Example

```
int slot , ch ;
unsigned long position;
Open_Slot(slot);
i9093_GetFirstTrigPosition (slot , ch , &position);
```

4.26. i9093_SetTrigDistance

This function use to set distance of I-9093.

Syntax (for I-9093 only)

```
int i9093_SetTrigDistance(
    int slot,
    int ch,
    int dir,
    unsigned long Distance
);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

ch:

specifies the channel (0 ~ 2).

dir:

Direction count

Distance:

set Distance

Return Values

Please refer the Error Code.

Example

```
int slot , ch, dir;
unsigned long distance;
Open_Slot(slot);
i9093_SetTrigDistance (slot , ch, dir, distance);
```

4.27. i9093_GetTrigDistance

This function use to get distance of I-9093.

Syntax (for I-9093 only)

```
int i9093_GetTrigDistance(
    int slot,
    int ch,
    int dir,
    unsigned long* Distance
);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

ch:

specifies the channel (0 ~ 2).

dir:

Direction count

**Distance:*

[output] get Distance

Return Values

Please refer the Error Code.

Example

```
int slot , ch, dir;
long *distance;
Open_Slot(slot);
i9093_GetTrigDistance (slot, ch, dir, & distance);
```

4.28. i8093_ReadNextPosition

This function use to get next trig position of I-9093.

Syntax (for I-9093 only)

```
int pac_i8093W_ReadNextPosition(
    int slot,
    int ch,
    unsigned long* Data32
);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

ch:

specifies the channel (0 ~ 2).

** Data32:*

[output] get next position

Return Values

Please refer the Error Code.

Example

```
int slot , ch ;
long Data32;
Open_Slot(slot);
i9093_ReadNextPosition (slot, ch, &Data32);
```

4.29. i9093_ClearNextPosition

This function use to clear next trig position of I-9093.

Syntax (for I-9093 only)

```
int i9093_ClearNextPosition(
    int slot,
    int ch
);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

ch:

specifies the channel (0 ~ 2).

Return Values

Please refer the Error Code.

Example

```
int slot , ch ;
Open_Slot(slot);
i9093_ClearNextPosition (slot, ch);
```

4.30. i8093_ConfigTriggerOut

This function use to enable/disable the compared trig out function of I-9093.

Syntax (for I-9093 only)

```
int i9093_ConfigTriggerOut(
    int slot,
    int ch,
    unsigned char enStatus
);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

ch:

specifies the channel (0 ~ 2).

enStatus:

set compare trig out status

0 : disable

1 : enable

Return Values

Please refer the Error Code.

Example

```
int slot , ch ,status;
Open_Slot(slot);
i9093_ConfigTriggerOut (slot, ch, status);
```

4.31. i9093_GetTriggerOutConfig

This function use to get the compared trig out function status of I-9093.

Syntax (for I-9093 only)

```
int i9093_GetTriggerOutConfig(
    int slot,
    int ch,
    unsigned char *enStatus
);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

ch:

specifies the channel (0 ~ 2).

**enStatus:*

[output] get compare trig out status

0 : disable

1 : enable

Return Values

Please refer the Error Code.

Example

```
int slot , ch ,status;
Open_Slot(slot);
i9093_GetTriggerOutConfig (slot, ch, &status);
```

4.32. i9093_SetLowPassFilter

This function use to set the low pass filter of I-9093.

Syntax (for I-9093 only)

```
int i9093_SetLowPassFilter(
    int slot,
    int ch,
    int filter
);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

ch:

specifies the channel (0 ~ 2).

filter:

	Max Input frequency	
Value	CW/CCW,DIR/Pulse	A/B Phase
0	4MHz (No low pass filter)	6MHz(No low pass filter)
1	4MHz	1MHz
2	2MHz	500KHz
3	1MHz	250KHz
4	640KHz	160 KHz
5	320KHz	80KHz
6	160KHz	40KHz
7	80KHz	20KHz
8	40KHz	10KHz

Return Values

Please refer the Error Code.

Example

```
int slot , ch , filter;
Open_Slot(slot);
i9093_SetLowPassFilter(slot, ch, filter);
```

4.33. i9093_GetLowPassFilter

This function use to get the Low Pass Filter of I-9093.

Syntax (for I-9093 only)

```
int i9093_GetLowPassFilter(
    int slot,
    int ch,
    int* filter
);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

ch:

specifies the channel (0 ~ 2).

**filter:*

[output]

	Max Input frequency	
Value	CW/CCW,DIR/Pulse	A/B Phase
0	4MHz (No low pass filter)	6MHz(No low pass filter)
1	4MHz	1MHz
2	2MHz	500KHz
3	1MHz	250KHz
4	640KHz	160 KHz
5	320KHz	80KHz
6	160KHz	40KHz
7	80KHz	20KHz
8	40KHz	10KHz

Return Values

Please refer the Error Code.

Example

```
int slot , ch , filter;
Open_Slot(slot);
i9093_GetLowPassFilter (slot, ch, &filter);
```

4.34. i9093_ReadIndexLatchedPosition

This function use to get the index latched position of I-9093.

With this function user can find the origin of the motor easier.

Syntax (for I-9093 only)

For Windows (CE and WES)

```
int i9093_ReadIndexLatchedPosition(
    int slot,
    int ch,
    long* Data32
);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

ch:

specifies the channel (0 ~ 2).

**Data32:*

[output] get the index latched position.

Return Values

Please refer the Error Code.

Example

```
int slot , ch ;
long Data32;
Open_Slot(slot);
i9093_ReadIndexLatchedPosition(slot, ch, *Data32);
```

4.35. i9093_ReadExTrigLatchedPosition

This function use to get the external trig latched position of I-9093.

Syntax (for I-9093 only)

```
int i9093_ReadExTrigLatchedPosition(
    int slot,
    int ch,
    long* Data32
);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

ch:

specifies the channel (0 ~ 2).

**Data32:*

[output] get the external trig latched position.

Return Values

Please refer the Error Code.

Example

```
int slot , ch ;
long Data32;
Open_Slot(slot);
i9093_ReadExTrigLatchedPosition(slot, ch,&Data32);
```

Appendix A. Error Code

Error Code	Definition	Description
0	OK	This indicates that there have been no errors
-1	ID_ERROR	There was a problem with the module ID
-2	SLOT_OUT_RANGE	There was a Slot index error (1 ~ 8)
-3	CHANNEL_OUT_RANGE	There was a Channel index error (0 - 15)
-4	MODE_ERROR	There was a Mode error(CW/CCW, DIR/Pulse, A/B Phase)
-5	FIRMWARENOTSUPPORT	The Firmware is not support.

Appendix B. Revision History

This chapter provides revision history information to this document.

The table below shows the revision history.

Revision	Date	Description
1.0.0	January 2018	Initial issue
2.0.0	July 2018	Added content for the I-9093 modules Added 2.Quick start Added 3.Compared Trig Out Added API naming table Added Compare table Added Applicable Platform table Modify library , demo path Modify API