I-8014(C)W/I-9014(C) Linux API Reference Manual

V 2.0.0 April 2018

Written by Martin Hsu

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

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

The I-8014(C)/I-9014(C) is a high speed isolated analog input module providing 16 single-ended or 8 differential analog input channels at 16-bit resolution. Besides including basic usage instructions and details of the SDK interface, this manual also introduces the Magic Scan function incorporated in the I-8014W that can be used for scanning multi-channel systems.

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, "Magic Scan" – This chapter introduces the attributes related to the Magic Scan function, the programming procedures, and demo programs.
- Chapter 3, "API References" – This chapter describes the functions provided in the I-8014W library together with an explanation of the differences in the naming rules used for the MiniOS7 and Windows platforms.
- Chapter 4, "Troubleshooting" – This chapter provides some troubleshooting solutions should you encounter any problems while operating the I-8014W.

1. Introduction

The I-8014(C)W/I-9014(C) are high performance analog input modules. I-8014W/I-9014is up to 16-channel single-ended or 8-channel differential inputs.I-8014CW/I-9014C isup to 8-channel differential inputs. They feature 16-bit resolution, 250Ks/s sampling rates, and 4K-sample FIFO. They provide isolation protection of2500 VRMS.

The I-8014(C)W/I-9014(C) (Hereinafter referred to as I-8014W) contain an impressive scan function called Magic Scan, which are able to improve many of the functions and meets the demands of high-end users. The Magic Scan mechanism not only scans the different input channels at vastly different rates, but also at different gains.
Even in a multi-channel scan, the sampling rates can be maintained at 250KS/s.

I-8014W contains two types of Magic Scan. One is a standardScan and the other is a virtual sample and hold function. Almost all AI Cardsare expensive if they provide a sample and hold function, but ICP DAS cannow provide you with a low-cost alternative.

I-8014W module includes a 4K onboard FIFO buffer for A/D conversion. With the Magic Scan function and 4K FIFO, the I-8014W can easily implement high-speed and time-critical data acquisition applications.

The differences between I-8014W/I-9014 and I-8014CW/I-9014C are as below:

	I-8014W/I-9014	I-8014CW/I-9014C
Input Range	+/- 10 V, +/- 5 V, +/- 2.5 V, +/- 1.25 V and +/- 20 mA	+/- 20 mA only
Select Input Type	Differential or Single-Ended Mode	Differential Mode only
Wire Connection for currentmeasurement	Need external 125 ohm resistor for currentmeasurement	Do need external 125 ohm resistor for currentmeasurement
Calibration Parameter	8 channels AI using 1 calibration parameter	8 channels AI using independent calibration parameter

1.1. Features

I-8014W/I-9014

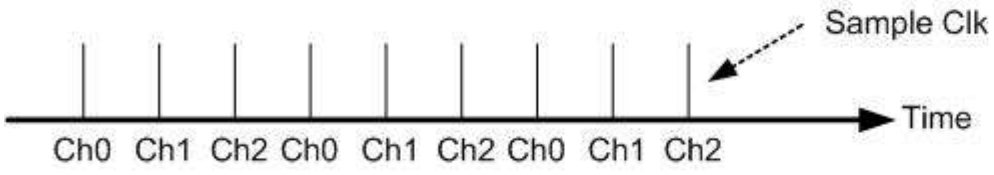
- 16 single-ended/8 differential inputs (jumper selectable)
- Input Range : +/- 10V, +/- 5V, +/- 2.5V, +/- 1.25V, +/- 20mA

I-8014CW/I-9014C

- 8 differential inputs
- Input Range : +/- 20mA
- 16-bit 250KHz ADC converter
- 4K-samples FIFO buffer
- External trigger mode : post-trigge
- Internal/external trigger start
- Magic Scan Type

Type 1: General

Each Sample clock only samples a single.



General type

When set as Standard mode,

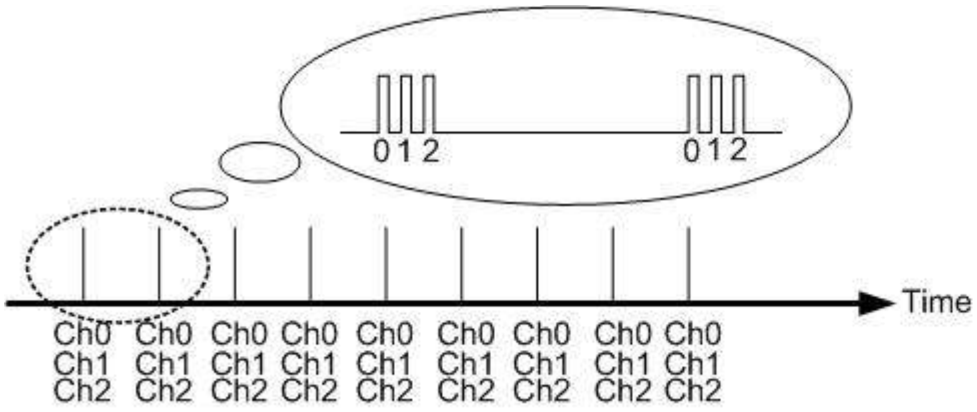
1. The maximum sample rate can set as 250 KHz.
2. If scan multi channels, the sample rate for each channel will be (Sample Rate)/(channel count)

   For example, if set sample rate as 250 KHz and scan 2 channels, the sample rate for each channel is 125 KHz.

Sample channel count	Hz/Ch
1	250KHz
2	125KHz
3	83.3KHz
4	62.5KHz

Type 2: virtual Sample and hold

Each sample clock will to sample all scan channels that have been set.



Virtual Sample and hold type

When set as Virtual Sample and Hold mode,

1. The maximum sample rate is 125 KHz.
2. It use 250 KHz (4 us) internal sample clock to scan each channel.
3. All channels are the sample rate.
4. The total sample rate for all channel must <= 125 KHz

Scanned Ch Count	Hz/Ch	Total Sample Rate
1	125KHz	125KHz
2	62.5KHz	125KHz
4	31.25 KHz	125 KHz

1.2. Specifications

Model	I-8014W/I-9014	I-8014CW/I-9014C
Analog Output		
Channels	8-ch Differential/16-Single-ended	8-ch Differential
Voltage Input Range	±1.25, ±2.5, ±5 V, ±10 V	-
Current Input Range	-20 mA ~ +20 mA(Requires OptionalExternal 125 Ω Resistor)	-20 mA ~ +20 mA
Resolution	16-bit	
Sample Rate	Single Channel Polling Mode :250K S/s	
FIFO	4 k Words	
Accuracy	0.05% of FSR	
Input Mode	Polling, Pacer (Magic Scan)	
Magic Scan Mode	Mode 1: Standard Mode Mode 2: Virtual Sample and Hold	
Overvoltage Protection	-45 V ~ +60 V	
Input Impedance	20 K, 200 K, 20 M (Jumper Select)	125 Ω
LED Indicators		
Power LED Indicator	Yes	
Isolation		
Intra-module Isolation, Field-to-Logic	2500 Vrms	
Power		
Power Consumption	2.5 W Max.	
Mechanical		
Dimension (L x W x H)	For I-8014(C)W: 102 mm x 30 mm x 115 mm For I-9014(C): 144 mm x 31 mm x 134 mm	
Environment		
Operating Temperature	-25 °C ~ +75°C	
Storage Temperature	For I-8014(C)W: -30 °C ~ +85°C For I-9014(C): -40°C ~ +85°C	
Humidity	10 % ~ 90% RH, non-condensing	

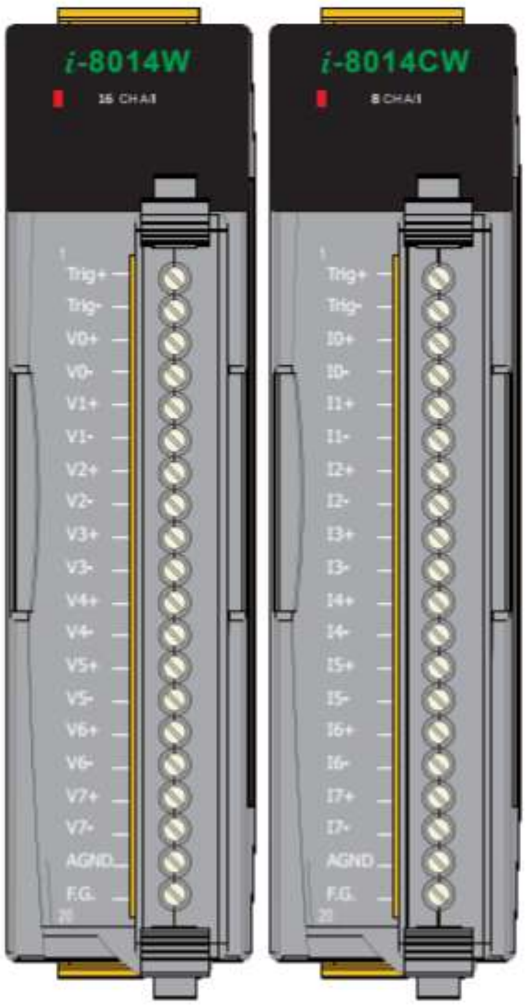
1.3. Pin Assignments

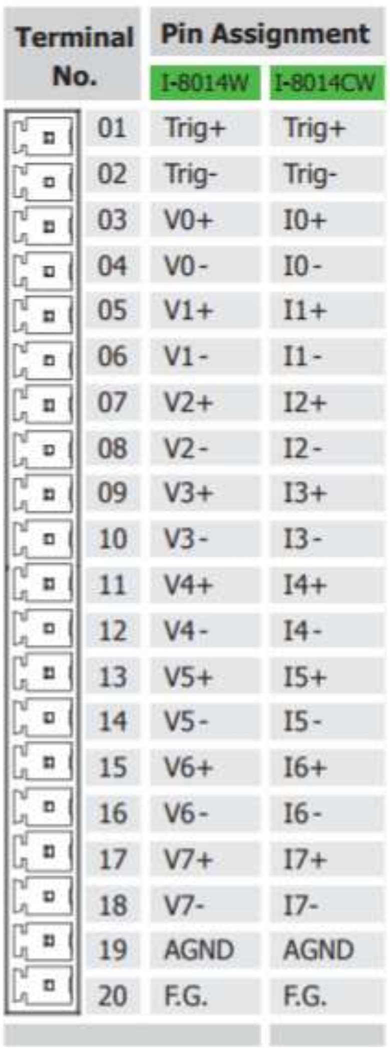
I-9014

Pin Assignment	Terminal No.		Pin Assignment
Trig+	01	02	Trig-
V0+	03	04	V0-
V1+	05	06	V1-
V2+	07	08	V2-
V3+	09	10	V3-
V4+	11	12	V4-
V5+	13	14	V5-
V6+	15	16	V6-
V7+	17	18	V7-
AGND	19	20	F.G.

I-9014C

Pin Assignment	Terminal No.		Pin Assignment
Trig+	01	02	Trig-
I0+	03	04	I0-
I1+	05	06	I1-
I2+	07	08	I2-
I3+	09	10	I3-
I4+	11	12	I4-
I5+	13	14	I5-
I6+	15	16	I6-
I7+	17	18	I7-
AGND	19	20	F.G.

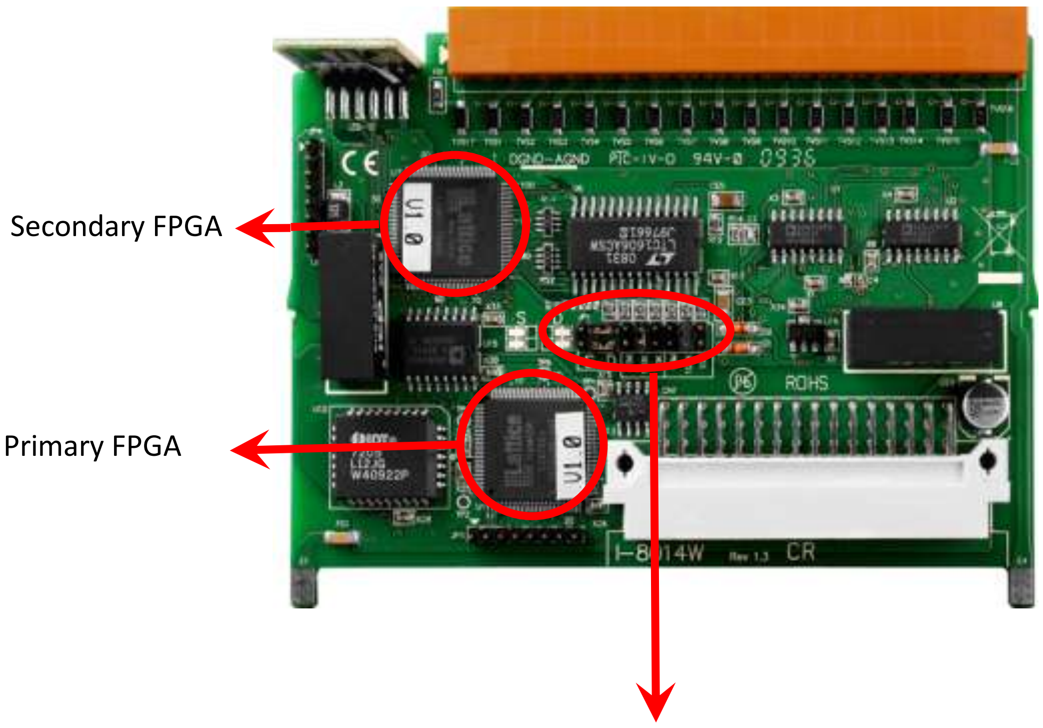




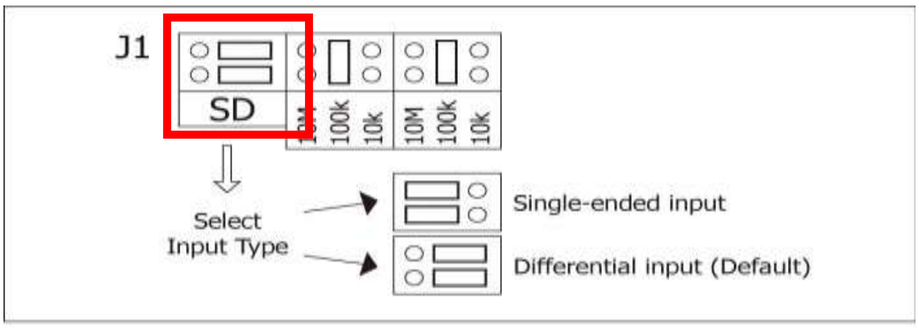
Terminal No.	Pin Assignment	
	I-8014W	I-8014CW
01	Trig+	Trig+
02	Trig-	Trig-
03	V0+	I0+
04	V0-	I0-
05	V1+	I1+
06	V1-	I1-
07	V2+	I2+
08	V2-	I2-
09	V3+	I3+
10	V3-	I3-
11	V4+	I4+
12	V4-	I4-
13	V5+	I5+
14	V5-	I5-
15	V6+	I6+
16	V6-	I6-
17	V7+	I7+
18	V7-	I7-
19	AGND	AGND
20	F.G.	F.G.

1.4. Jumper Settings

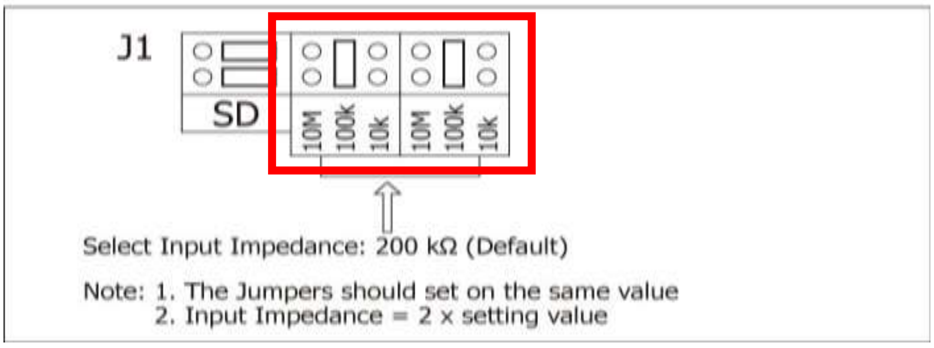
I-8014W/I-8014CW



Differential / Single-ended Jumper Selection



Input impedance Jumper Selection

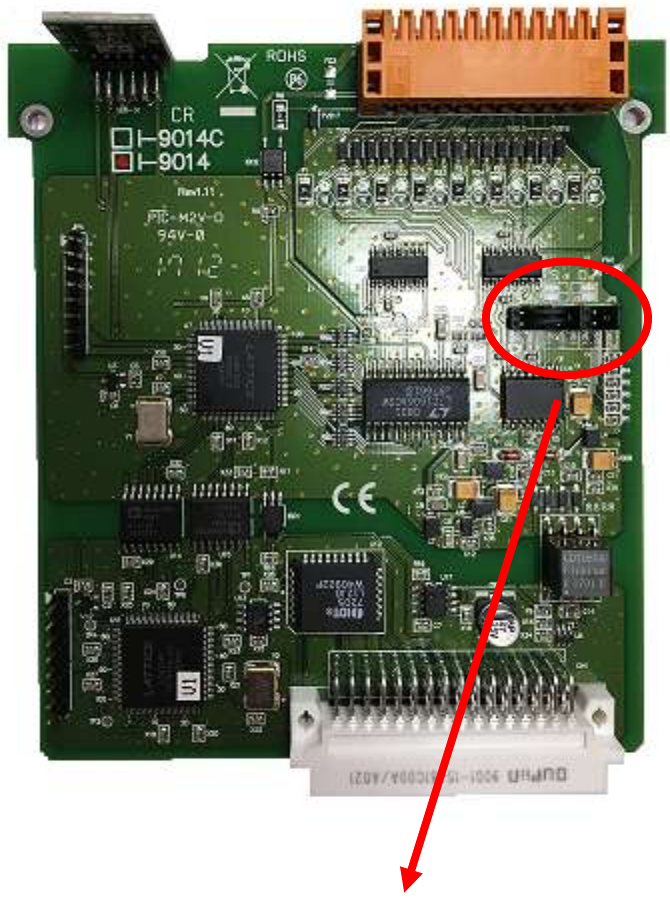


Select Input Impedance: 200 kΩ (Default)

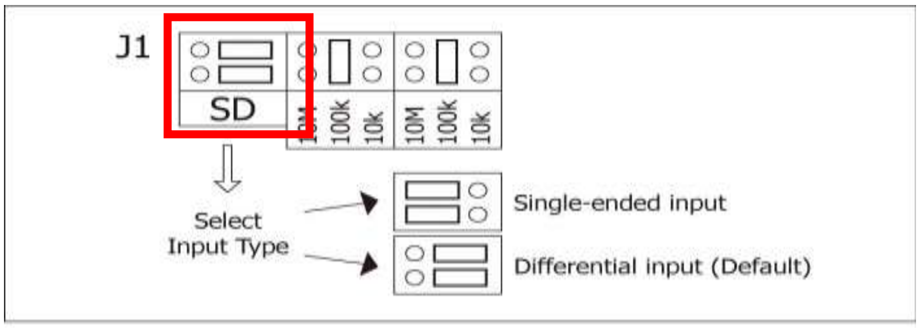
Note: 1. The Jumpers should set on the same value
2. Input Impedance = 2 x setting value

Note : I-8014CW do not have those Jumper, it is only with Differential Mode and Input impedance 20 KΩ

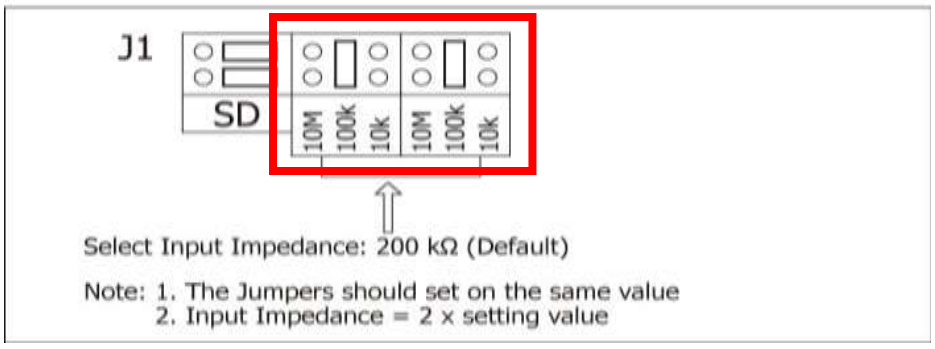
I-9014W/I-9014C



Differential / Single-ended Jumper Selection



Input impedance Jumper Selection

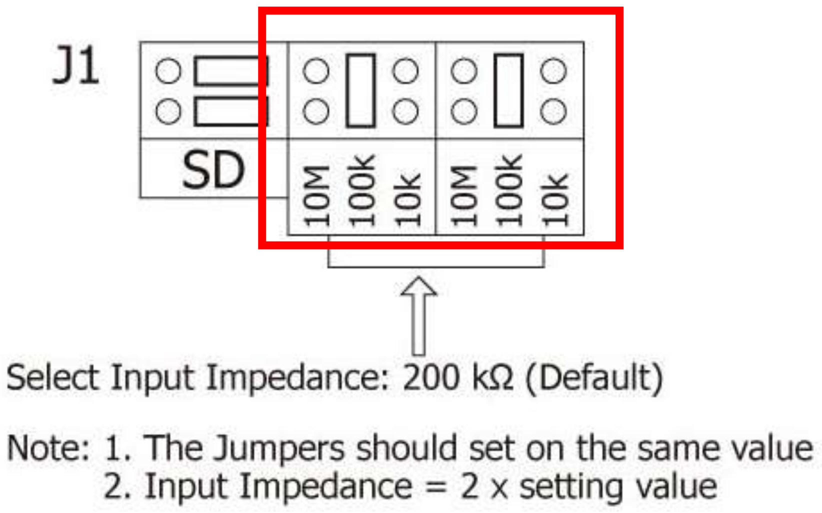

J1
SD
10M
100k
10k
10M
100k
10k
Select Input Impedance: 200 kΩ (Default)
Note: 1. The Jumpers should set on the same value
2. Input Impedance = 2 x setting value

Note : I-9014C do not have those Jumper, it is only with Differential Mode and Input impedance 20 KΩ

Adjusting the Input impedance

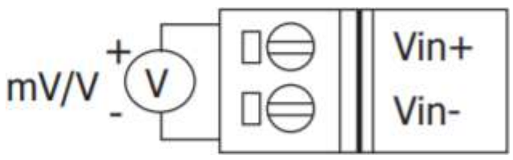
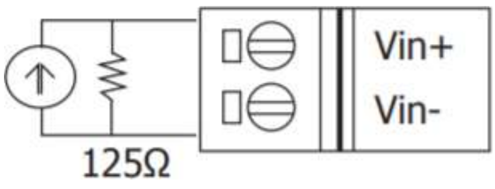
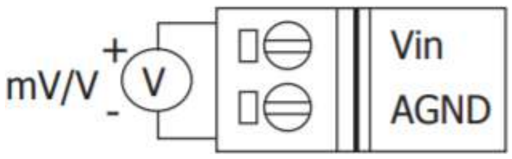
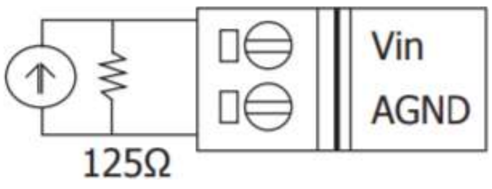
The I-8014W allows three input impedance options, including 20 kΩ, 200 kΩ (default setting) and 20 MΩ to meet system requirements. In most cases, 200 kΩ is sufficient.

Note that each time the input impedance is adjusted on a calibrated module, the module must be recalibrated. Refer to the Calibration section on page 19 if you are using an I-8000 or iPAC-8000 (MiniOS7 platform controller), or refer to page 32 for details of the calibration process if you are using a module based on the WinCE or WES platform.

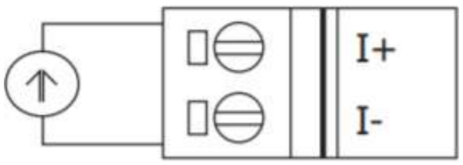

J1
SD
10M
100k
10k
10M
100k
10k
Select Input Impedance: 200 kΩ (Default)
Note: 1. The Jumpers should set on the same value
2. Input Impedance = 2 x setting value

1.5. Wire Connections

I-8014W/I-9014

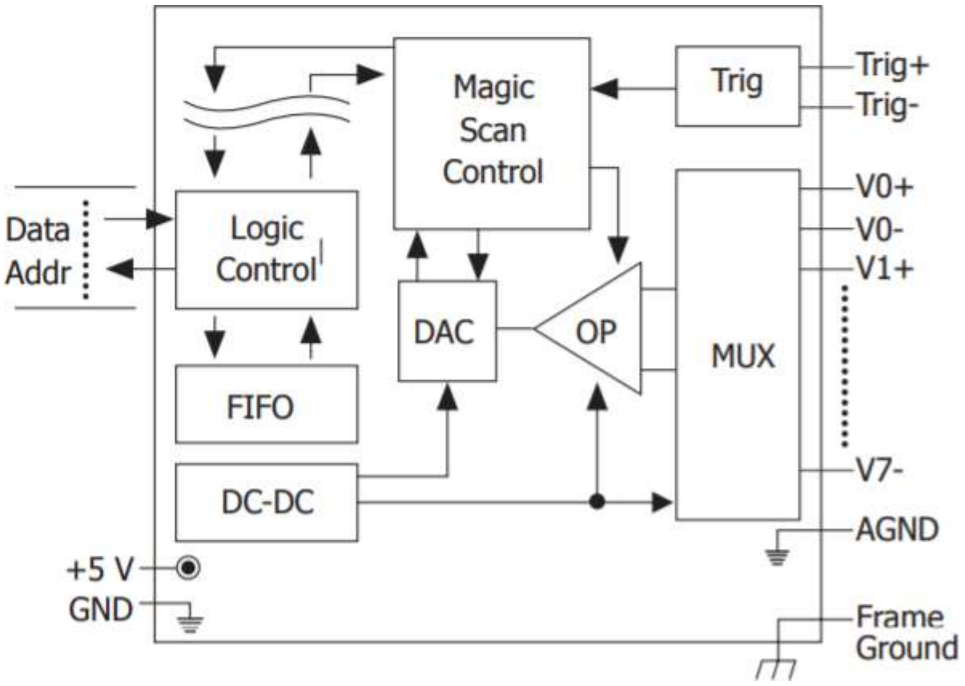
	Voltage Input Wiring	Current Input Wiring
Differential		
Single-ended		

I-8014CW/I-9014C

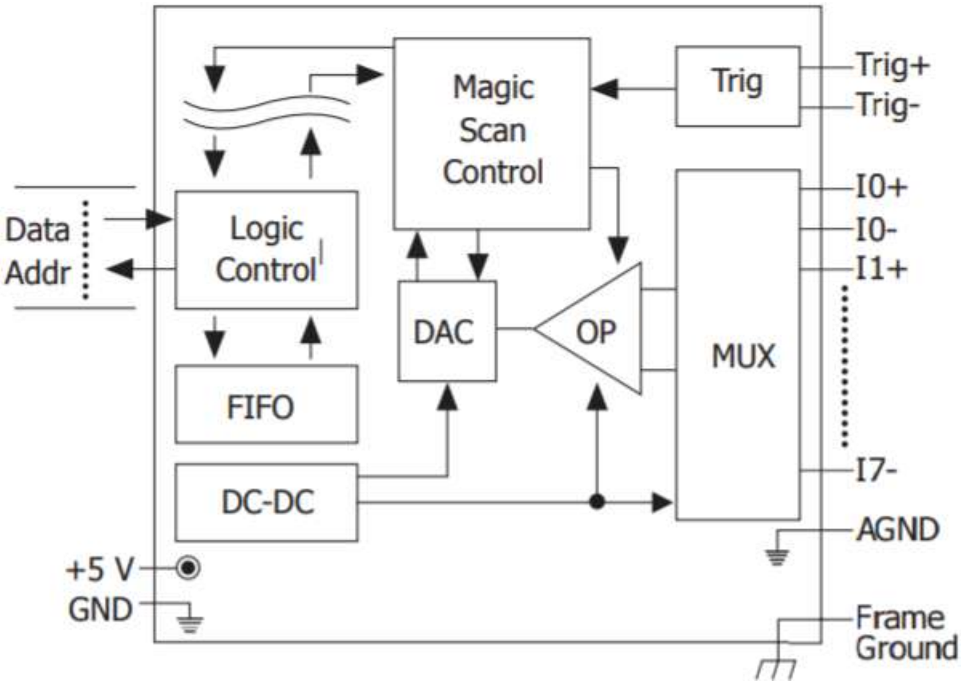
	Current Input Wiring
Differential	

1.6. Block Diagram

I-8014W/I-9014



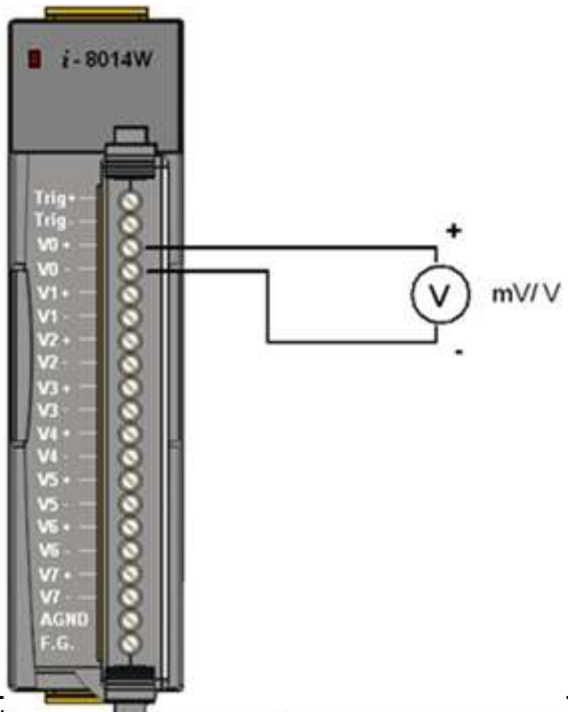
I-8014CW/I-9014C



1.7. Demo Programs

ICP DAS provides a range of demo programs for different platforms that can be used to verify the functions of the I-8014W/9014. The source code contained in these programs can also be reused in your own custom programs if needed.
Both I-8014W/9014 and I-8014CW use the same library and demo.

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-8014W/9014 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.
   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. Magic Scan

This chapter provides details related to Magic Scan, which is a key function included on the I-8014W/9014 for multi-channel analog data acquisition at high sampling rates.
Two demo programs that can be used to implement Magic Scan functionality are included at the end of this chapter. Either Magic Scan mode or the trigger method can be selected for use in the two programs, and the only difference is that Magic Scan mode uses polling to transfer data and the trigger method transfers data using interrupts.

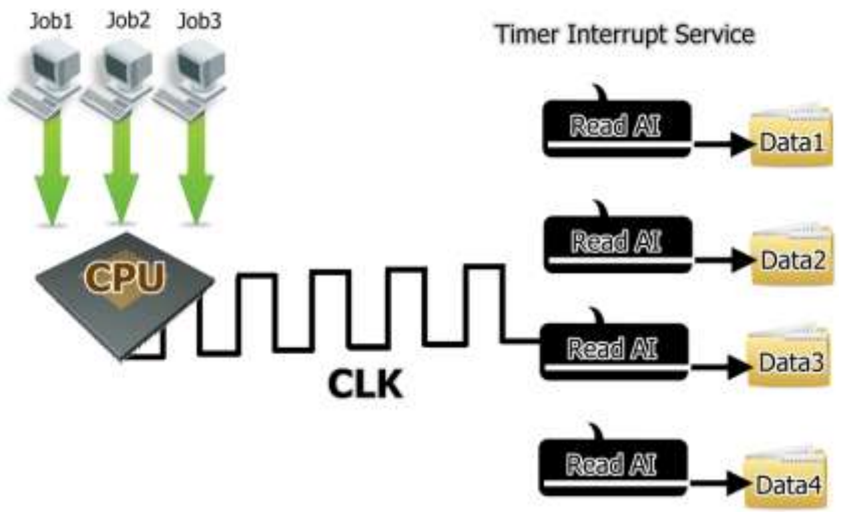
2.1. The Advantages of Magic Scan

This section contains:

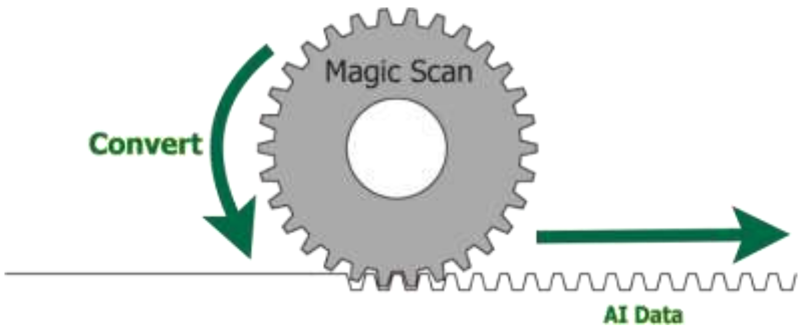
- ✓ High speed AD with high precision Timer request
- ✓ Magic Scan, read AD from 4K AI FIFO
- ✓ 4K AI FIFO with FIFO level limit interrupt to reduce the CPU loading greatly
- ✓ Application Examples

2.1.1. High speed AD with high precision Timer request

For normal AD sampling application, it needs a high precision timer to handle the sample rate, and it is very difficult to have less than 1 ms timer ISR on multi task operation system,

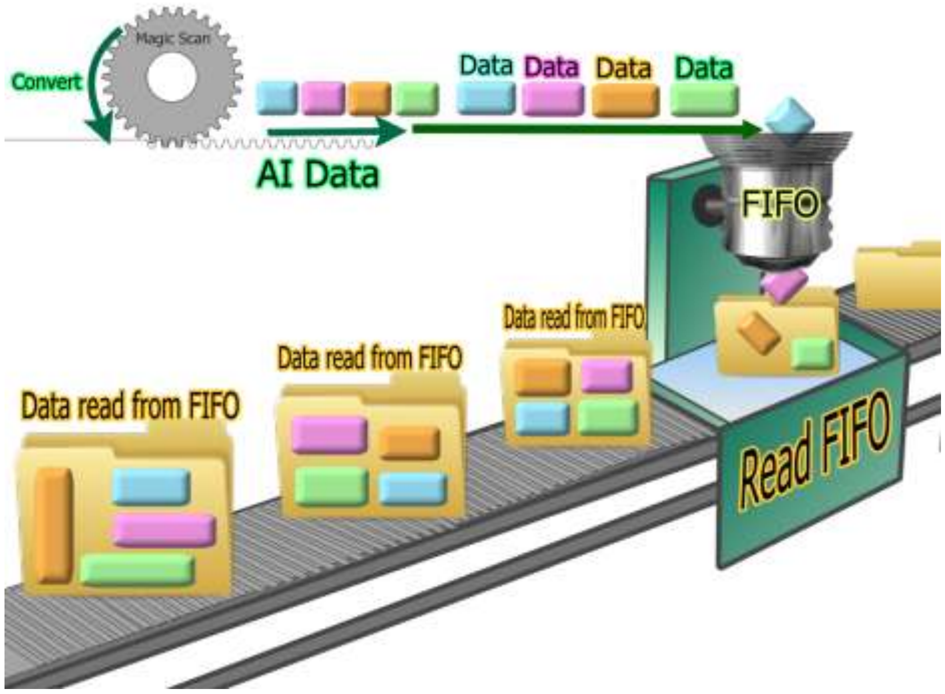


For I-8014W to sample AD, If we configure the sample rate of Magic Scan , it will use independent internal hardware clock to trigger AD, it does not rely on platform's Timer ISR



2.1.2. Magic Scan, read AD from 4K AI FIFO

Magic Scan convert AD to 4K FIFO automatically, program can read AI data from FIFO any time before FIFO full.



For normal AD modules, they need to use command to trigger the AD convert and wait time for ready signal for each sampling event.

2.1.3.4K AI FIFO with FIFO level limit interrupt to reduce the CPU loading greatly

I-8014W can set FIFO limit level for interrupt service notification. This feature can increase the performance for sampling application. Program don't need to sample data all the time, but to wait for CPU's interrupt notification if AI data count in FIFO reach the limit level.

FIFO limit level	Limit Data count to trigger interrupt
0	8
1	16
2	32
3	64
4	128
5	256
6	512
7	2048

2.1.4. Application Examples

This section contains:

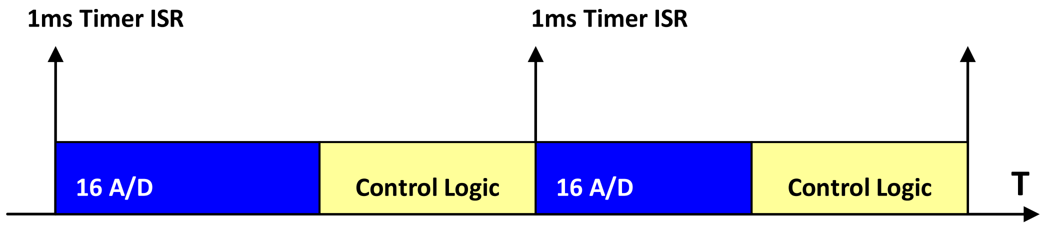
- To sample 16 channels'AI in1 ms Timer ISR
- 250KHz application
- 10KHz sample rate for two I-8014W

2.1.4.1. To sample 16 channels'AI in1 ms Timer ISR

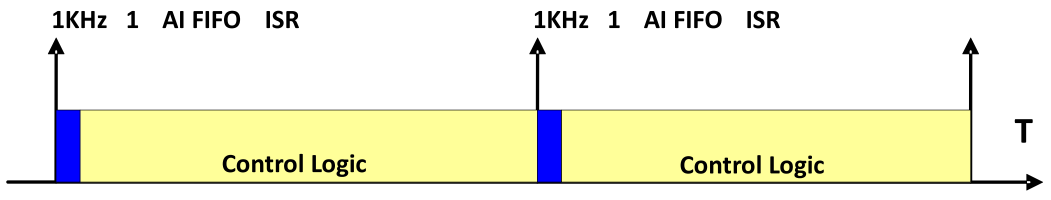
To achieve this specification

1. System must provide 1ms Timer Interrupt Service.
2. The maximum sample rate of Analog Input module must above 16KHz/sec (16 Data/ms),if application need PID control or other operation in 1 ms, it need higher sample rate.

If we take a 16-channel AI module with maximum sample rate 30KHz for example, to sample 16 data by using this AI module needs about 0.54 ms ((1000ms/30000)*16 = 0.54ms) , it means in 1ms Timer Interrupt Service Routine, it spends 540us to scan 16 channels AI data, and there will have 460us left to do other control logic.

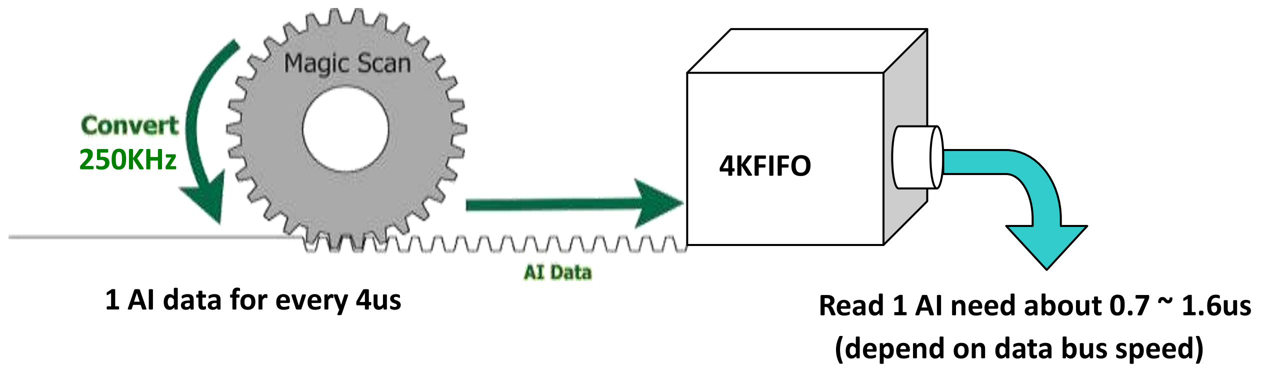


If we set I-8014W scan mode as Sample and Hold, FIFO level limit trigger as16 AI data, sample rate 1KHz. It means that there will be a FIFO ISR in every 1ms, when program receive Interrupt notification, it just needs 11us~26us to read 16 AI data from FIFO, it remains 970 us can do its control logic work.

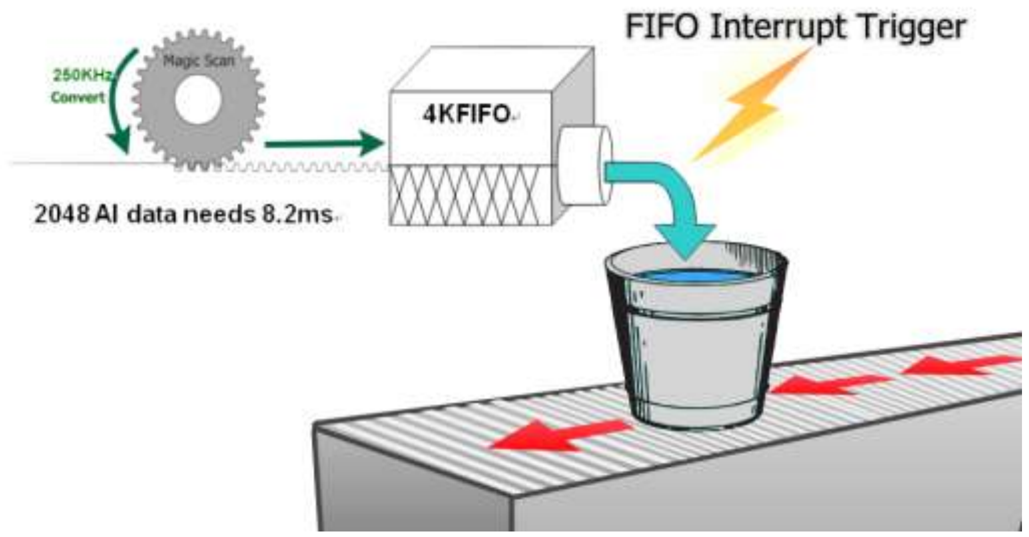


2.1.4.2. 250KHz application

I-8014W can set 250 KHz sample rate in standard mode. Below diagram shows how it works. The key feature is the speed to read1 AI data from FIFO is faster than AD convert.



If we set FIFO Limit level as 7 (2048 AI to trigger Interrupt Service), it needs 8.2 ms to convert 2048 AI data, and 3~6.5 ms to read 2048 AI data from FIFO.



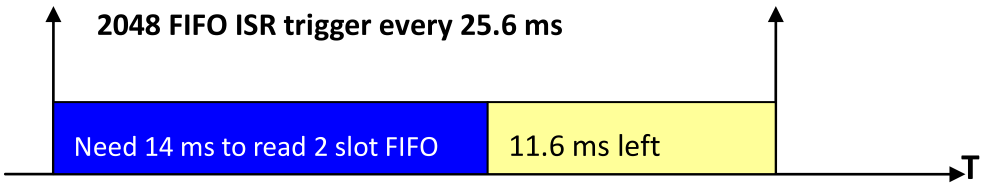
2.1.4.3. 10KHz sample rate for two I-8014W

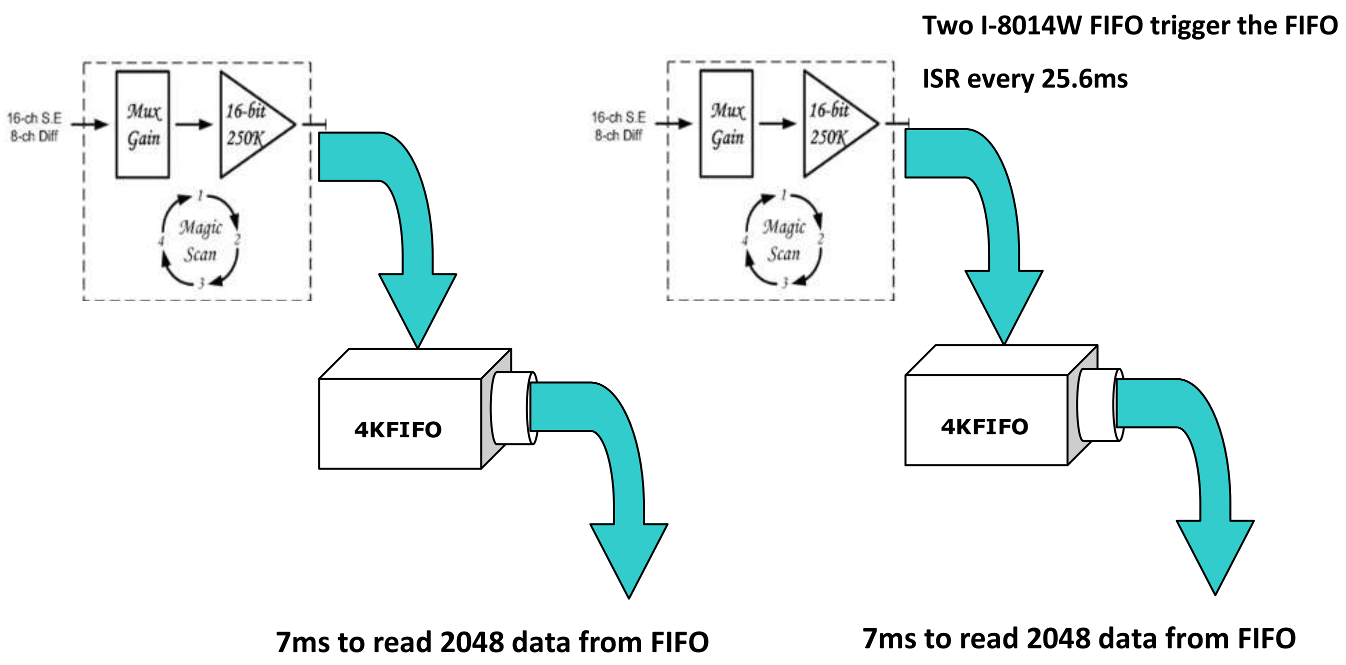
Scan parameters for each I-8014W.

Sample rate	Scan channels	Scan Mode	FIFO limit level
10KHz	8	Sample and Hold	7 (2048 AI)

Under 10KHz Sample and Hold mode,

1. It will be (80K AI data)/sec for each slot
2. It will trigger FIFO limit Interrupt every 25.6 ms, (2048*1000)/(80000)=25.6
3. There will be about 11 ms left after to get data from two slot FIFO


2048 FIFO ISR trigger every 25.6 ms
Need 14 ms to read 2 slot FIFO
11.6 ms left
T


Two I-8014W FIFO trigger the FIFO ISR every 25.6ms
16-ch S.E
8-ch Diff
Mux Gain
16-bit 250K
Magic Scan
4KFIFO
7ms to read 2048 data from FIFO
16-ch S.E
8-ch Diff
Mux Gain
16-bit 250K
Magic Scan
4KFIFO
7ms to read 2048 data from FIFO

In this application, it needs to convert 160K AI Data from two I-8014W, and this is done by I-8014W itself without using any CPU resource, program just needs to wait for FIFO ISR notification and read data from FIFO.

2.2. Magic Scan Mode

For multi-channel high speed data acquisition systems, the I-8014W provides sampling rates of up to 250 kHz and a 4k-sample FIFO that reduces the loading of the CPU and enhances the performance of your system.

The following is an overview of the Magic Scan specifications:

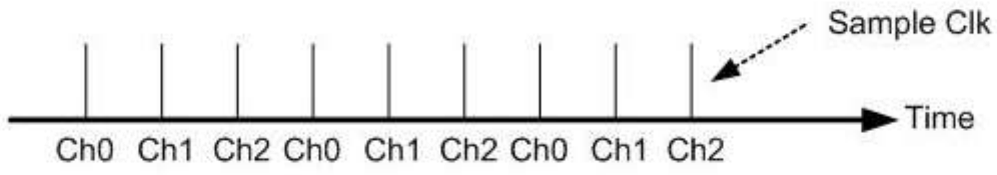
Max. Channels	16
Sampling Rate	2 Hz ~ 250 kHz
FIFO	4 k samples
Sampling Mode	- Standard - Virtual Sample and Hold
Trigger Method	- Software - Internal Hardware - External Hardware
Data Transfer Mode	- polling - Interrupt

This section describes the two Magic Scan modes that can be used on the I-8014W:

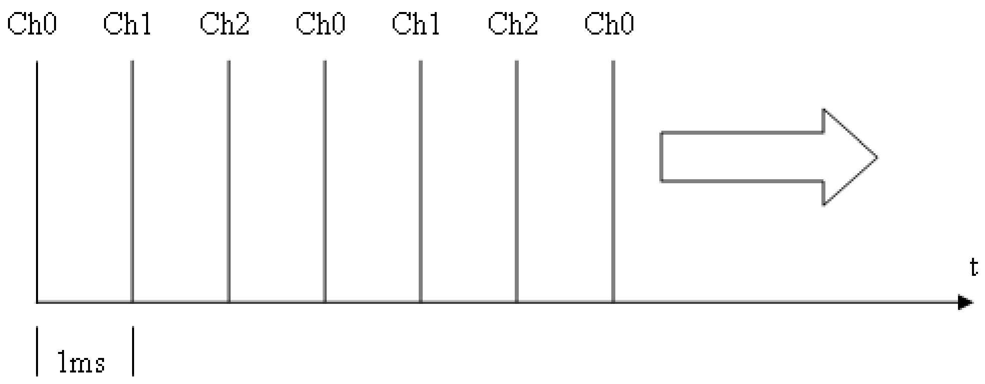
- Standard Mode
- Virtual Sample and Hold Mode

2.2.1. Standard Mode

In standard mode, the I-8014W converts data from a single channel in each sampling interval.

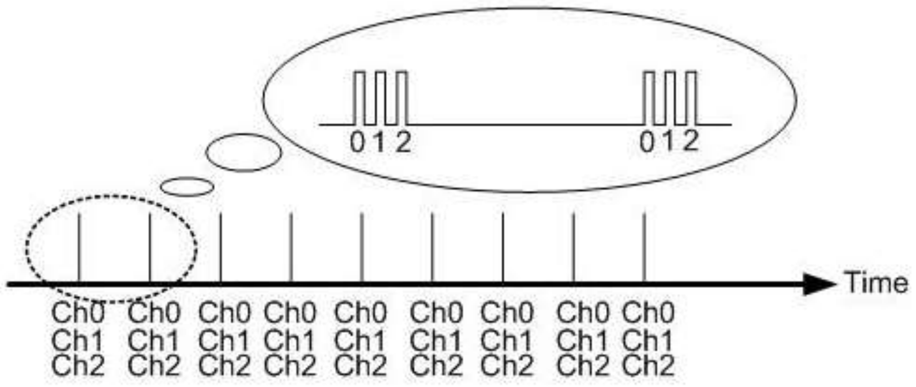


For example, if Ch0, Ch1and Ch2 are configured to perform the scan function, and the sampling rate is set to 1 kHz, the interval between each sampling operation is 1 ms, so the scanning time for a single cycle (from Ch0 to Ch1 to Ch2) is 3ms, as illustrated below:



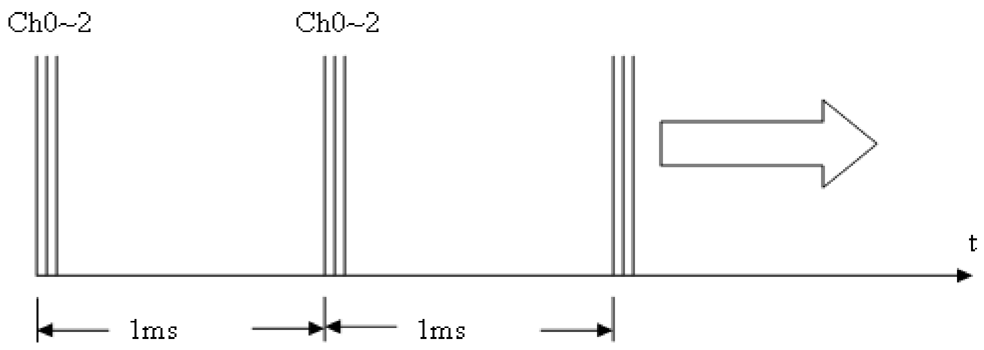
2.2.2. Virtual Sample and Hold Mode

Virtual sample and hold mode operates such that several channels can be configured to perform scanning functions and are sampled at the same time. The sampling rate is set to 250 kHz by default, and the scan cycle time is the interval that is set in the Magic Scan function.



0 1 2
0 1 2
Time
Ch0 Ch0 Ch0 Ch0 Ch0 Ch0 Ch0 Ch0 Ch0
Ch1 Ch1 Ch1 Ch1 Ch1 Ch1 Ch1 Ch1 Ch1
Ch2 Ch2 Ch2 Ch2 Ch2 Ch2 Ch2 Ch2 Ch2

For example, if the sampling rate is set to1 kHz and Ch0, Ch1, and Ch2 are configured to perform the scanning functions, the sampling rate for scanning Ch0 to Ch2 is 250 kHz, and the frequency of the scan cycle is 1 kHz, the interval between one scan cycle and the next is 1 ms.



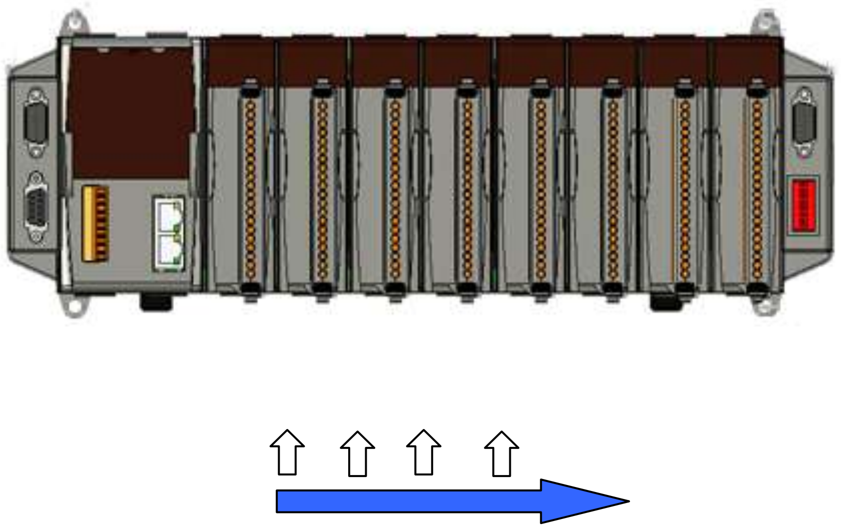
2.3. Trigger Methods

This section contains:

- Software Trigger Method
- Internal Hardware Trigger Method
- External Hardware Trigger Method

2.3.1. Software Trigger Method

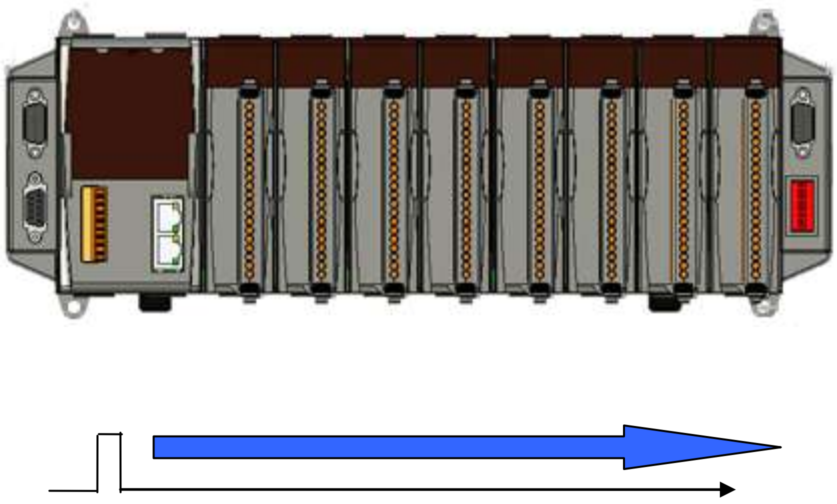
The API provides a trigger instruction that initiates Magic Scan. If you have two or more modules, you need to configure the Magic Scan parameters for each module and execute the Magic Scan instructions for the modules individually.



Execute Magic Scan on the first module and then repeat for the subsequent modules using software instructions.

2.3.2. Internal Hardware Trigger Method

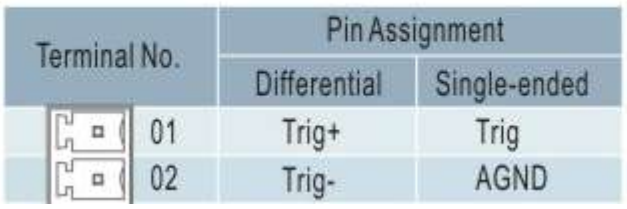
If you wish to simultaneously initiate the Magic Scan function on two or more modules, set the internal hardware signal as the trigger source in your program, and then the internal trigger signal will trigger the Magic Scan operation for the individual modules at almost the same time.



Trigger Magic Scan for each module using an internal hardware signal.

2.3.3. External Hardware Trigger Method

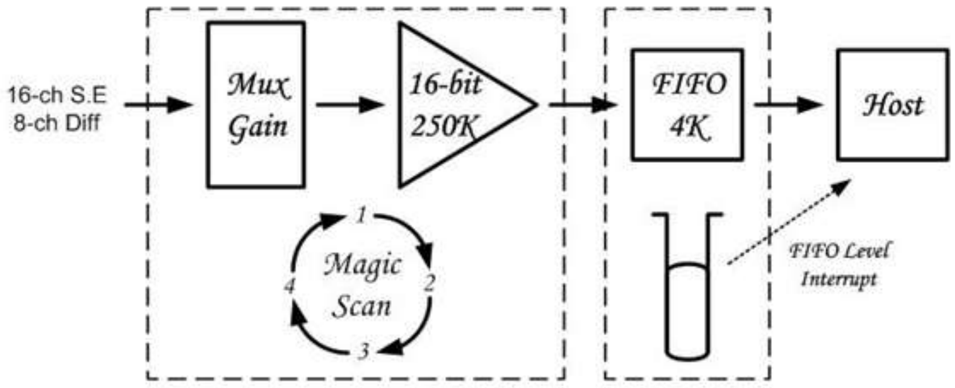
The Magic Scan function is also able to accept an external trigger source from the first two terminals, using this method, the trigger can be set as either rising edge or falling edge triggered. After setting the external trigger source and the triggering conditions, execute Magic Scan in your program. The I-8014W will wait until it receives the external signal from the Trig+ and Trig- terminals and will then execute Magic Scan.



Terminal No.	Pin Assignment	
	Differential	Single-ended
01	Trig+	Trig
02	Trig-	AGND

2.4. FIFO

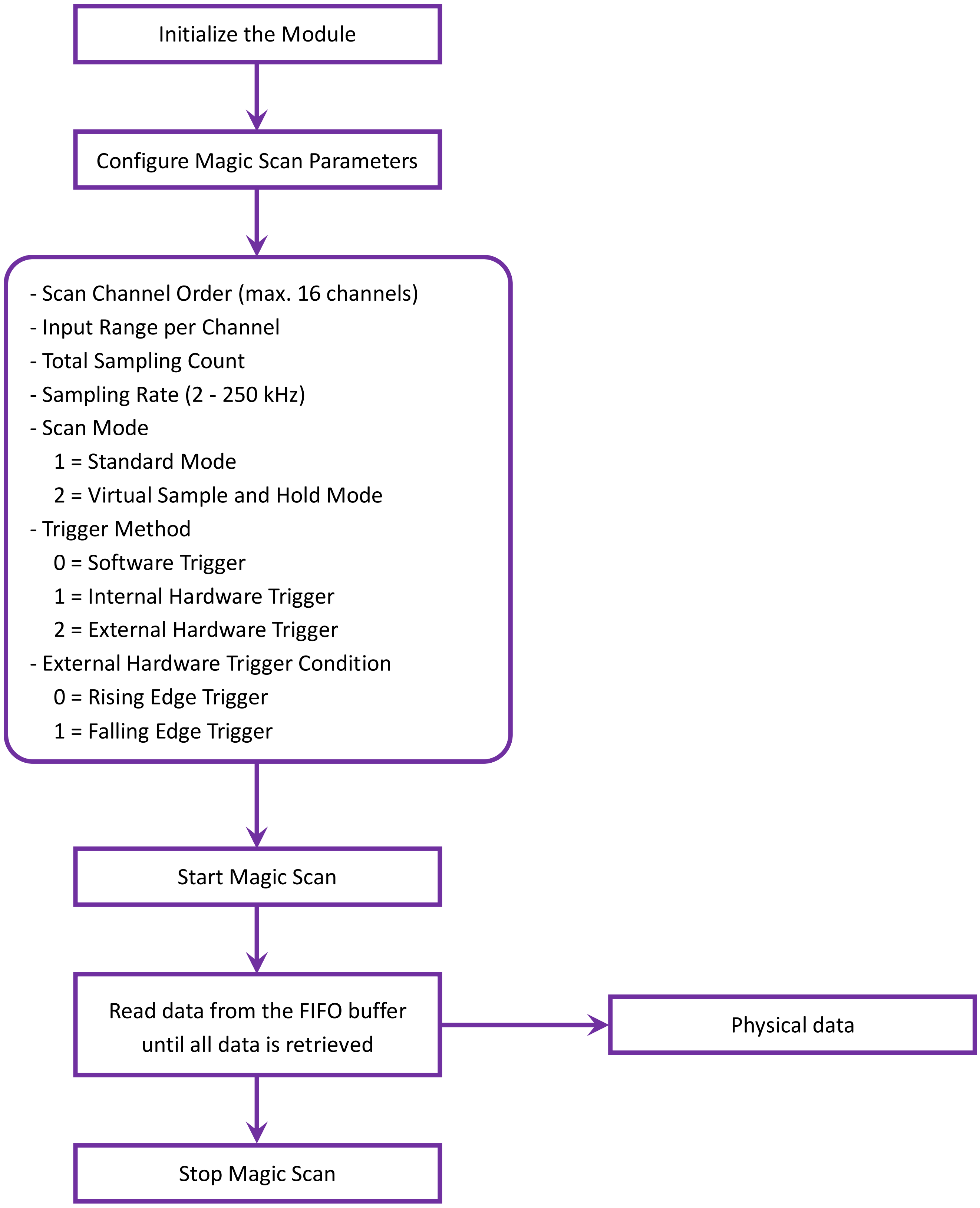
The I-8014W is equipped with a 4 k-sample FIFO buffer, which may be used to store 4096 data samples from Magic Scan to ensure that no data is lost. The acquisition data is sequentially saved to the FIFO buffer during the scan process. To prevent the FIFO buffer from being filled, the data needs to be read from the FIFO buffer within a specific timeframe. If the FIFO buffer is filled, data can no longer be saved until a command is executed that clears the FIFO buffer. In contrast, if data is read from the FIFO buffer too frequently, CPU resources will be wasted and performance will be affected. To achieve the optimum balance, two modes for transferring data from the FIFO are provided, polling mode and interrupt mode.



Note: I-8014CW only can select max 8 channels Differential Mode and +/- 20 mA Input Range

2.5. Magic Scan Procedure

The following is an illustration of the Magic Scan program procedure:

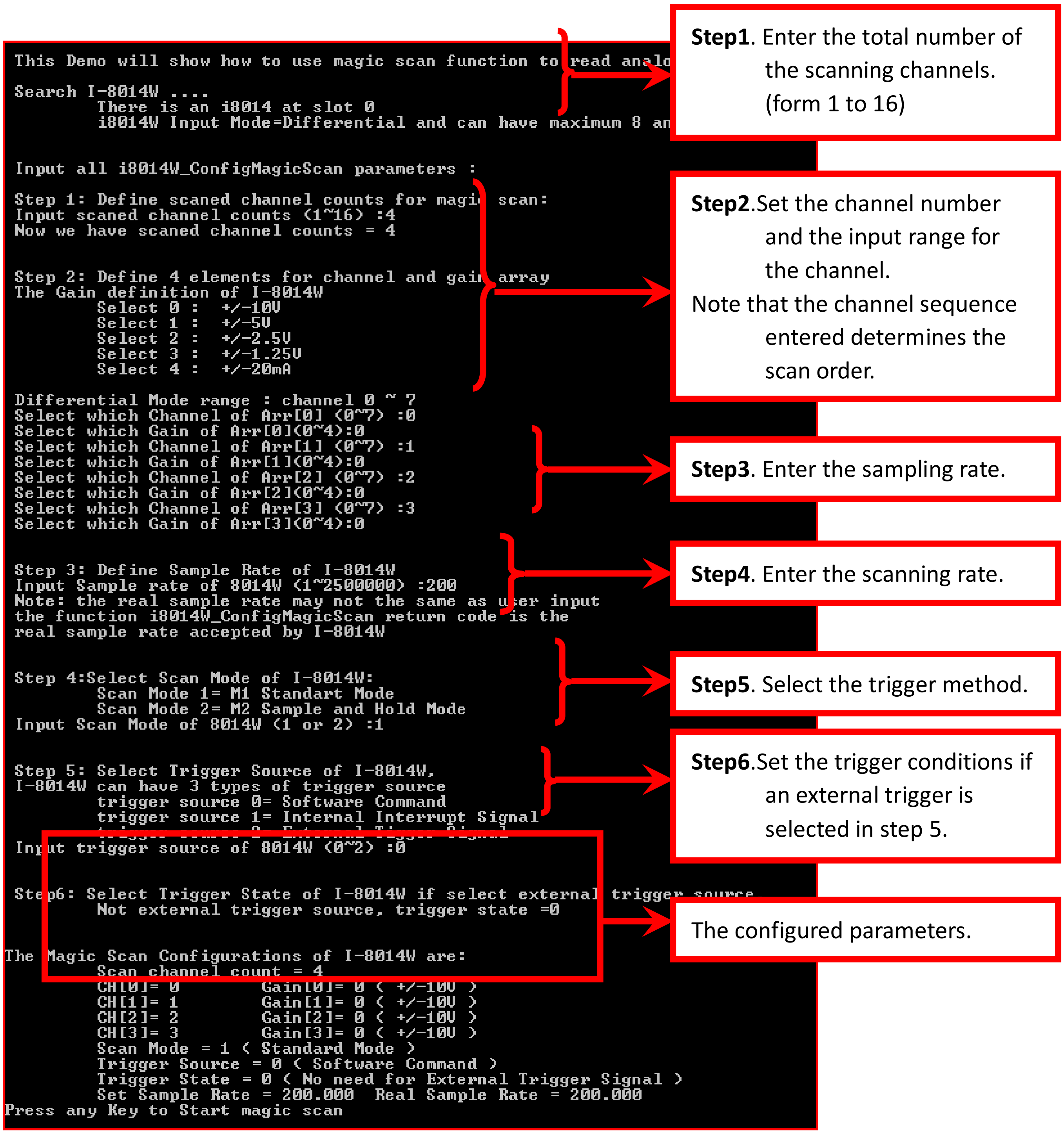

Initialize the Module
Configure Magic Scan Parameters
- Scan Channel Order (max. 16 channels)
- Input Range per Channel
- Total Sampling Count
- Sampling Rate (2 - 250 kHz)
- Scan Mode
1 = Standard Mode
2 = Virtual Sample and Hold Mode
- Trigger Method
0 = Software Trigger
1 = Internal Hardware Trigger
2 = External Hardware Trigger
- External Hardware Trigger Condition
0 = Rising Edge Trigger
1 = Falling Edge Trigger
Start Magic Scan
Read data from the FIFO buffer until all data is retrieved
Physical data
Stop Magic Scan

Note: I-8014CW only can select max 8 channels and +/- 20 mA Input Range

2.6. Magic Scan Example

This section for transferring data using the polling method.

The following figure shows the interface and parameters that should be set in Magic.exe



```
This Demo will show how to use magic scan function to read analo

Search I-8014W ....
        There is an i8014 at slot 0
        i8014W Input Mode=Differential and can have maximum 8 an


Input all i8014W_ConfigMagicScan parameters :

Step 1: Define scaned channel counts for magic scan:
Input scaned channel counts (1~16) :4
Now we have scaned channel counts = 4


Step 2: Define 4 elements for channel and gain array
The Gain definition of I-8014W
        Select 0 :  +/-10V
        Select 1 :  +/-5V
        Select 2 :  +/-2.5V
        Select 3 :  +/-1.25V
        Select 4 :  +/-20mA

Differential Mode range : channel 0 ~ 7
Select which Channel of Arr[0] (0~7) :0
Select which Gain of Arr[0](0~4):0
Select which Channel of Arr[1] (0~7) :1
Select which Gain of Arr[1](0~4):0
Select which Channel of Arr[2] (0~7) :2
Select which Gain of Arr[2](0~4):0
Select which Channel of Arr[3] (0~7) :3
Select which Gain of Arr[3](0~4):0


Step 3: Define Sample Rate of I-8014W
Input Sample rate of 8014W (1~2500000) :200
Note: the real sample rate may not the same as user input
the function i8014W_ConfigMagicScan return code is the
real sample rate accepted by I-8014W


Step 4:Select Scan Mode of I-8014W:
        Scan Mode 1= M1 Standart Mode
        Scan Mode 2= M2 Sample and Hold Mode
Input Scan Mode of 8014W (1 or 2) :1


Step 5: Select Trigger Source of I-8014W,
I-8014W can have 3 types of trigger source
        trigger source 0= Software Command
        trigger source 1= Internal Interrupt Signal
        trigger source 2= External Trigger Signal
Input trigger source of 8014W (0~2) :0


Step6: Select Trigger State of I-8014W if select external trigger source
        Not external trigger source, trigger state =0


The Magic Scan Configurations of I-8014W are:
        Scan channel count = 4
        CH[0]= 0        Gain[0]= 0 ( +/-10V )
        CH[1]= 1        Gain[1]= 0 ( +/-10V )
        CH[2]= 2        Gain[2]= 0 ( +/-10V )
        CH[3]= 3        Gain[3]= 0 ( +/-10V )
        Scan Mode = 1 ( Standard Mode )
        Trigger Source = 0 ( Software Command )
        Trigger State = 0 ( No need for External Trigger Signal )
        Set Sample Rate = 200.000  Real Sample Rate = 200.000
Press any Key to Start magic scan
```

Step1. Enter the total number of the scanning channels. (form 1 to 16)

Step2.Set the channel number and the input range for the channel.
Note that the channel sequence entered determines the scan order.

Step3. Enter the sampling rate.

Step4. Enter the scanning rate.

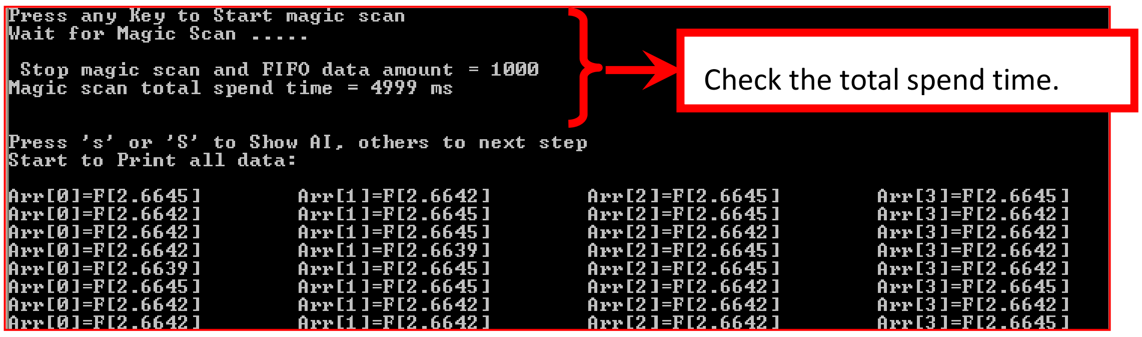
Step5. Select the trigger method.

Step6.Set the trigger conditions if an external trigger is selected in step 5.

The configured parameters.

After the Magic Scan parameters have been set, press any Key to Start Magic Scan, as shown in the figure below.

If the scan mode is set to standard mode, the total spend time will be equal to [1000] multiplied by the [sampling period]. (1000 is the total sample count defined in the demo program)



```
Press any Key to Start magic scan
Wait for Magic Scan .....

 Stop magic scan and FIFO data amount = 1000
Magic scan total spend time = 4999 ms


Press 's' or 'S' to Show AI, others to next step
Start to Print all data:

Arr[0]=F[2.6645]        Arr[1]=F[2.6642]        Arr[2]=F[2.6645]        Arr[3]=F[2.6645]
Arr[0]=F[2.6642]        Arr[1]=F[2.6645]        Arr[2]=F[2.6645]        Arr[3]=F[2.6642]
Arr[0]=F[2.6642]        Arr[1]=F[2.6645]        Arr[2]=F[2.6642]        Arr[3]=F[2.6642]
Arr[0]=F[2.6642]        Arr[1]=F[2.6639]        Arr[2]=F[2.6645]        Arr[3]=F[2.6642]
Arr[0]=F[2.6639]        Arr[1]=F[2.6645]        Arr[2]=F[2.6645]        Arr[3]=F[2.6642]
Arr[0]=F[2.6645]        Arr[1]=F[2.6645]        Arr[2]=F[2.6642]        Arr[3]=F[2.6645]
Arr[0]=F[2.6642]        Arr[1]=F[2.6642]        Arr[2]=F[2.6642]        Arr[3]=F[2.6642]
Arr[0]=F[2.6642]        Arr[1]=F[2.6642]        Arr[2]=F[2.6642]        Arr[3]=F[2.6645]
```

Note: I-8014CW only can select max 8 channels and +/- 20 mA Input Range

The following figure illustrates the interface and parameters that need be set when using Magic.exe on a Windows platform.

2.7. Magic interrupts Example

8014w_magic_isr.exe demonstrates how to transfer data using interrupts. When using this method, the Magic Scan parameter settings are identical to those used for Magic.exe. The only difference is that an interrupt service routine (ISR) must be installed before starting Magic Scan. This is achieved by adding the following code to your program:

```
i8014W_InstallMagicScanISR(slotIndex,Slot_ISR,triggerLevel);
i8014W_StartMagicScan(slotIndex);
```

The installed ISR will process any interrupt tasks when an interrupt signal is detected from the FIFO, and the parameter triggerLevel is used to configure the interrupt conditions, as indicated in the following table:

triggerLevel	Data Count
0	8
1	16
2	32
3	64
4	128
5	256
6	512
7	2048

Once the amount of data in the FIFO buffer meets the level that was set via the triggerLevel parameter, an interrupt signal will be generated, and the code in the installed ISR will be processed. Note that you need to ensure that the interrupt function in the ISR is cleared, otherwise any subsequent interrupt requests will not be processed.

Using interrupts to transfer data helps to reduce CPU usage time which could be wasted when used for polling and waiting for data from the FIFO buffer.

2.8. Case Study

The requirements in this case are:

1. Measure four differential signals ranging from -10 V to +10V.
2. The sampling rate per channel is 200 Hz, and sampling time interval from one channel to the next channel is less than 10 µs.
3. Once 2000 data samples have been collected, transfer the data via the Ethernet to a data center or a remote data storage disk.

Use the following procedure to meet the requirements:

Step 1. Set the jumper on the I-8014W to differential input mode.

Step 2. Set the input channels as ch0 - ch3, and set the input range for each channel to -10 - +10 V. (Gain = 0)

Step 3. Set the sampling rate to 200 Hz, and set the scan mode to Mode2: Virtual Sample and Hold Mode. With Virtual Sample and Hold Mode, the sampling time interval between one channel and another channel is 4 µs.

Step 4. Collect 2000 samples, which means collecting 500 samples per channel. (i.e., 2000 divided by four channels). The elapsed time will be 500 * (1/ 200 Hz) = 2500 ms.

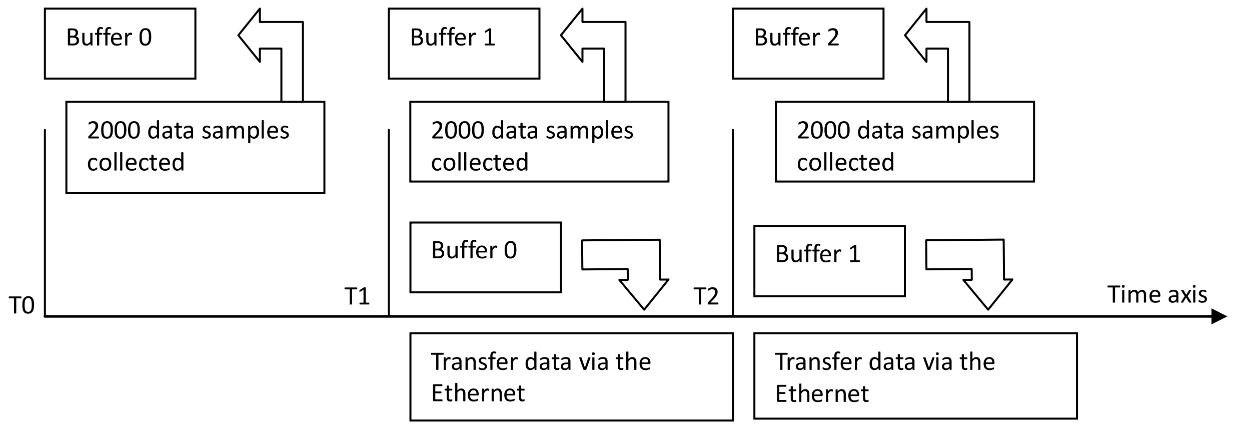
Step 5. If the system uses the MiniOS7 platform, converting the data from hexadecimal format to floating point format and then transferring it via the Ethernet will add to the CPU load. It is recommended that the hexadecimal data is first transferred to a PC client and then converted to floating point data on the PC.

Note: I-8014CW only can select max 8 channels and +/- 20 mA Input Range

If the system uses the Windows platform, converting data from hexadecimal format to floating point format will not affect the CPU load. The data can be converted to floating point format locally and then transferred via the Ethernet.

Tips & Warnings

It is recommended that several buffers are created to process the data obtained from the FIFO, which can then be reused in the processing flow, as illustrated in the figure below. This allows the system time to convert the data, and then save and transfer it.


Buffer 0
2000 data samples collected
Buffer 1
2000 data samples collected
Buffer 0
Buffer 2
2000 data samples collected
Buffer 1
T0
T1
T2
Time axis
Transfer data via the Ethernet
Transfer data via the Ethernet

3. API References

ICPDAS supplies a range of C API functions for the I-8014W/9014 module. When developing a custom program, refer to either the i8014W.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 for use with the Linux platform. Detailed information related to individual functions can be found in the following sections.

Platform	Product included	API prefix characters
Linux	I-8014W	"i8014W_" + function name
	I-9014	"i9014_" + function name

Function	Description
i8014W_Init	This function is used to initialize the driver and confirm the hardware ID.
i8014W_GetFirmwareVer_L1	This function is used to retrieve the version number of the primary FPGA firmware for a module.
i8014W_GetFirmwareVer_L2	This function is used to retrieve the version number of the secondary FPGA firmware for a module.
i8014W_GetLibVersion	This function is used to retrieve the version number of the 8014W.lib.
i8014W_GetSingleEndJumper	This function is used to retrieve the single-ended/differential jumper position settings on the I-8014(C)W/I-9014(C).
i8014W_ReadGainOffset	This function is used to obtain the gain and offset values on each input type for I-8014W/I-9014.
i8014W_Read_mA_GainOffset	This function is used to obtain the gain and offset values on each input type for I-8014CW/I-9014C.
i8014W_ReadAI	This function is used to read a floating point input (calibrated) from one specified channel.
i8014W_ReadAIHex	This function is used to read a hexadecimal input (calibrated) from a single specified channel.
i8014W_ConfigMagicScan	This function is used to configure all the parameters needed when using Magic Scan, and should be called before executing any Magic Scan instructions.

i8014W_StartMagicScan	This function is used to start Magic Scan.
i8014W_StopMagicScan	This function is used to stop Magic Scan.
i8014W_ReadFIFO	This function is used to read data from the FIFO buffer after the Magic Scan function has been triggered.
i8014W_CalibrateData	This function is used to calibrate the raw data read during the Magic Scan process and to convert the data to a floating point value.
i8014W_CalibrateDataHex	This function is used to calibrate the raw data read in Magic Scan process.
i8014W_UnLockFIFO	This function is used to unlock the FIFO buffer when it is locked after being filled.
i8014W_ClearFIFO	This function is used to clear the FIFO buffer after the UnlockFIFO function has been executed.
i8014W_InstallMagicScanISR	This function is used to install the ISR to control to control interrupt events form the FIFO buffer.
i8014W_UnInstallMagicScanISR	This function is used to uninstall the Magic Scan ISR.
i8014W_ClearInt	This function is used to clear the status of the Magic Scan interrupts.

3.1. i8014W_Init

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

Syntax

```
short i8014W_Init(int slot);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

Return Values

0 = the module in the slot is an I-8014(C)W/I-9014(C).

-1 = there is no I-8014(C)W/I-9014(C) module in this slot.

For other return values, see the Appendix A. Error Code.

Note

Before executing any functions on the I-8014(C)W/I-9014(C), the i8014W_Init function needs to be called once for each I-8014(C)W/I-9014(C). If there are two or more I-8014(C)W/I-9014(C)modules, you need call the i8014W_Init function for each I-8014(C)W/I-9014(C)module individually by passing the slot number that the I-8014(C)W/I-9014(C)module is plugged into.

Example

```
int slotIndex, err;
Open_Slot(slotIndex);
err=i8014W_Init(slotIndex);
if(err==0)
{
    printf(“There is an I-8014W module in slot %d\n”,slotIndex);
}
else
{
    printf(“There is no I-8014W module in slot %d\n”,slotIndex);
}
```

3.2. i8014W_GetFirmwareVer_L1

This function is used to retrieve the version number of the primary FPGA firmware for a module. The function is only used for troubleshooting or recording purposes.

Syntax

```
short i8014W_GetFirmwareVer_L1(int slot);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

Return Values

The version number of the primary FPGA firmware for the I-8014(C)W/I-9014(C)module.

Example

```
short ver_L1=0, slot=1;
Open_Slot(1);
ver_L1= i8014W_GetFirmwareVer_L1 (slot);
printf( "\nPrimaryFPGA Version =: %04X",i8014W_GetFirmwareVer_L1(slot) );
```

3.3. i8014W_GetFirmwareVer_L2

This function is used to retrieve the version number of the secondary FPGA firmware for a module. The function is only used for troubleshooting or recording purposes.

Syntax

```
short i8014W_GetFirmwareVer_L2(int slot);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

Return Values

The version number of the secondary FPGA firmware for the I-8014(C)W/I-9014(C)module.

Example

```
short ver_L2=0, slot=1;
Open_Slot(slot);
ver_L2= i8014W_GetFirmwareVer_L2 (slot);
printf( "\nSecondaryFPGA Version =: %04X",i8014W_GetFirmwareVer_L2(slot) );
```

3.4. i8014W_GetLibVersion

This function is used to retrieve the version number of the 8014W. The function is only used for troubleshooting or recording purposes.

Syntax

```
short i8014W_GetLibVersion(void);
```

Parameter

None

Return Values

The version number of the 8014W.

Example

```
short version, slot=1;
Open_Slot(slot);
version = i8014W_GetLibVersion();
printf("\nLibrary Version =: %04X",i8014W_GetLibVersion());
```

3.5. i8014W_GetLibDate

This function is used to retrieve the release date of the 8014W. The function is only used for troubleshooting or recording purposes.

Syntax

```
void i8014W_GetLibDate(char *LibDate);
```

Parameter

**libDate:*

[Output] the release date of the 8014W.

Return Values

None

Example

```
char slot=1, libDate [32];
Open_Slot(slot);
i8014W_GetLibDate(libDate);
printf("\nBuild Date =: %s",libDate);
```

3.6. i8014W_GetSingleEndJumper

This function is used to retrieve the single-ended/differential jumper position settings on the I-8014(C)W/I-9014(C). If you wish to use 8-channel differential input, the jumper needs to be put in differential position; similarly, the jumper needs be set to the single-ended position before 16-channel single-ended input will works correctly.

Syntax

```
short i8014W_GetSingleEndJumper(int slot);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

Return Values

0: The jumper is in the differential position.

1: The jumper is in the single-ended position.

Example

```
short slot=1, jumper=0, maxCh=0;
Open_Slot(slot);
jumper = i8014W_GetSingleEndJumper(slot);
if(jumper)
{
    maxCh=16;
    printf("i8014W Input Mode=Single-End\n\r");
}
else
{
    maxCh=8;
    printf("i8014W Input Mode=Differential\n\r");
}
```

3.7. i8014W_ReadGainOffset

This function is used to obtain the gain and offset values on each input type for I-8014W/I-9014.I-8014CW/I-9014C can use i8014W_Read_mA_GainOffset function. Please refer to section 4.1.8.

Syntax

```
void i8014W_ ReadGainOffset
(
    int slot,
    int gain,
    unsigned short* gainValue,
    short* offsetValue
);
);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

gain:

specifies the input type (0 - 4), where:0: +/-10 V, 1: +/-5 V, 2: +/-2.5 V, 3: +/-1.25 V, 4: +/-20 mA

**gainValue:*

[Output] the gain value for the input range.

**offsetValue:*

[Output] the offset value for the input range.

Return Values

None

Example

```
unsigned slot=1, short gVal=0;
short oVal=0;
Open_Slot(slot);
i8014W_ReadGainOffset(slot,gain,&gVal,&oVal);
printf("\nThe Gain and Offset values for Calibration are: Gain=%u; Offset=%d", ch, gVal, oVal);
```

3.8. i8014W_Read_mA_GainOffset

This function is used to obtain the gain and offset values on each input type for I-8014CW/I-9014C. I-8014W/I-9014 can use i8014W_ReadGainOffset function.

Syntax

```
void i8014W_Read_mA_GainOffset
(
    int slot,
    int channel,
    unsigned short* gainValue,
    short* offsetValue
);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

channel:

specifies the channel (0 - 7), for +/-20 mA

**gainValue:*

[Output] the gain value for the input range.

**offsetValue:*

[Output] the offset value for the input range.

Return Values

None

Example

```
unsigned slot=1, short gVal=0;
short oVal=0;
Open_Slot(slot);
i8014W_Read_mA_GainOffset (slot,ch,&gVal,&oVal);
printf("\nThe channel and Offset values for Calibration are: Gain=%u; Offset=%d", ch, gVal, oVal);
```

3.9. i8014W_ReadAI

This function is used to read a floating point input (calibrated) from one specified channel.

Syntax

```
short i8014W_ReadAI(
    int slot,
    intch,
    int gain,
    float* fVal
);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

ch:

specifies the channel number, 0 - 7 for differential input, or 0 - 15 for single-ended input.

gain:

specifies the input type (0 - 4), where:0: +/-10 V, 1: +/-5 V, 2: +/-2.5 V, 3: +/-1.25 V, 4: +/-20 mA

**fVal:*

[Output] the floating-point data.

Return Values

0 = No Error

For other return values, see the Appendix A. Error Code.

Example

```
int slot,ch,gain;
float fVal=0.0;

slot = 1;
gain = 0; // "+/-10V"
Open_Slot(slot);

for(ch=0;ch<8;ch++)
{
    i8014W_ReadAI( slot, ch, gain, &fVal);
    printf("\n[%02d]= [ %05.4f ]",ch,,fVal);
}
```

Note

I-8014CW/I-9014Conly can select max 8 channels and +/- 20 mA Input Range

3.10. i8014W_ReadAIHex

This function is used to read a hexadecimal input (calibrated) from a single specified channel.

Syntax

```
short i8014W_ReadAIHex(
    int slot,
    intch,
    int gain,
    short* hVal
);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

ch:

specifies the channel number, 0 - 7 for differential input, or 0 - 15 for single-ended input.

gain:

specifies the input type (0 - 4), where:0: +/-10 V, 1: +/-5 V, 2: +/-2.5 V, 3: +/-1.25 V, 4: +/-20 mA

**hVal:*

[Output] the hexadecimal data.

Return Values

0 = No Error

For other return values, see the Appendix A. Error Code.

Example

```
int slot,ch,gain;
short hVal=0.0;

slot = 1;
gain = 0; // "+/-10V"
Open_Slot(slot);

for(ch=0;ch<8;ch++)
{
   i8014W_ReadAIHex( slot, ch, gain, &hVal);
   printf("\n[%02d]= [ %04X ] ",ch,,hVal);
}
```

Note

I-8014CW/I-9014C only can select max 8 channels and +/- 20 mA Input Range

3.11. i8014W_ConfigMagicScan

This function is used to configure all the parameters needed when using Magic Scan, and should be called before executing any Magic Scan instructions.

Syntax

```
void i8014W_ConfigMagicScan
(
    int slot,
    intchArr[],
    intgainArr[],
    intscanChCount,
    floatsampleRate,
    intscanMode,
    inttriggerSource,
    inttriggerState ,
    float* realSampleRate
);
```

Parameter

slot:

specifies the slot number (1 ~ 8)

chArr[]:

creates an array that is used to set the channels to be scanned. The channel indices define the scanning order; the maximum number of channels is 16.

gainArr[]:

creates an array that is used to set the input type for the corresponding channel with the same index as that stored in chArr[], where:0: +/-10 V, 1: +/-5 V, 2: +/-2.5 V, 3: +/-1.25 V, 4: +/-20 mA

scanChCount:

a count of the channels, that have been added to chArr[].

sampleRate:

the total sampling rate, 2 - 250 kHz.

scanMode:

1: Standard mode

2: Virtual Sample and Hold mode

triggerSource:

0: Software trigger

1: Internal hardware trigger

2: External hardware trigger

triggerState:

0: Rising edge trigger. This is only valid when using an external hardware trigger.

1: Falling edge trigger. This is only valid when using an external hardware trigger.

**realSampleRate:*

[Output] the real sampling rate that was used by the I-8014W.

Return Values

None

Example

```
int slot=1, chArr[16], gainArr[16], scanChCount;
float sampleRate,realsampleRate;
int scanMode, triggerSource, triggerState;
Open_Slot(slot);
chArr[0]=0; // element 0 assigned to channel 0
chArr[1]=1;
...
chArr[15]=15; // element 15 assigned to channel 15
gainArr[0]=0; // element 0 assigned to input type 0
gainArr[1]=1; // element 1 assigned to input type 1
...
gainArr[15]=4; // element 15 assigned to input range 4
scanChCount=1; //only sample chArr[0] (channel 0 )
sampleRate=25000.0; //set the sample rate to 25 KHz
scanMode=1; // use M1 standard mode
triggerSource=1; // use internal interrupt signal Mode
triggerState=0;

realsampleRate=i8014W_ConfigMagicScan(slotIndex,chArr,gainArr,scanChCount, sampleRate,
scanMode,triggerSource,triggerState);
printf ("Set Sample Rate = %6.3f    Real Sample Rate = %6.3f \n",sampleRate, realsampleRate);

i804W_StartMagicScan(slot);
...
i8014W_ReadFIFO();
```

Note

I-8014CW/I-9014C only can select max 8 channels and +/- 20 mA Input Range

3.12. i8014W_StartMagicScan

This function is used to start Magic Scan. Once Magic scan starts, the converted data is immediately saved to the FIFO buffer. When an external hardware trigger is selected, after this function is executed, the I-8014(C)W/I-9014(C) will wait until it receives a trigger signal.
If you wish to simultaneously initial Magic Scan on two or more I-8014(C)W/I-9014(C) modules using an internal hardware trigger source, configure each module and then execute the StartMagicScan function only once. The slot number can be any of the slots that contain an I-8014(C)W/I-9014(C) modules.

Syntax

```
short i804W_StartMagicScan(int slot);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

Return Values

0 = No Error

For other return values, see the Appendix A. Error Code.

Example

```
int slot;
slot=1;
Open_Slot(slot);

i804W_StartMagicScan(slot);
```

3.13. i8014W_StopMagicScan

This function is used to stop Magic Scan. All operations for saving data to the FIFO buffer are also stopped because no further data will be converted.

Syntax

```
short i804W_StopMagicScan(int slot);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

Return Values

0 = No Error

For other return values, see the Appendix A. Error Code.

Example

```
int slot;
slot = 1;
Open_Slot(slot);

i804W_StopMagicScan (slot);
```

3.14. i8014W_ReadFIFO

This function is used to read data from the FIFO buffer after the Magic Scan function has been triggered. If the amount of data in the FIFO buffer is less than the value set using the readCount parameter, the function will read all the data and return it immediately. You will then need to reset the hexData [ ] and readCount parameters and continue to call this function until all the data required is obtained and then stop Magic Scan.

Syntax

```
short i804W_ReadFIFO
(
    int slot,
    shorthexData[],
    shortreadCount,
    short* dataCountFromFIFO
);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

hexData []:

specifies the starting address of the data array used to store the data that is read in hexadecimal format..

readCount:

specifies the amount of data required.

** dataCountFromFIFO:*

[Output] the amount of data read in this process.

Return Values

0 = No Error

For other return values, see the Appendix A. Error Code.

Example

```
int slot=1;
short hexData[8192];
long readCnt=0;
short totalScaned=0;
short TargetCnt=1000;
Open_Slot(slot);

i8014W_ReadFIFO(slot,hexData+totalScaned, TargetCnt-totalScaned,&readCnt);
if(readCnt>0)
    totalScaned+=readCnt;

if(readCnt==MAX_FIFO || totalScaned>=TargetCnt)
{
    i8014W_StopMagicScan(slot);
    i8014W_UnLockFIFO(slot);
    i8014W_ClearFIFO(slot);
}
```

3.15. i8014W_CalibrateData

This function is used to calibrate the raw data read during the Magic Scan process and to convert the data to a floating point value.

Syntax

```
void i8014W_CalibrateData
(
    int slot,
    shortiGain,
    shortdataFromFIFO,
    float* calibratedAI
);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

iGain:

specifies the input type (0 - 4), where:0: +/-10 V, 1: +/-5 V, 2: +/-2.5 V, 3: +/-1.25 V, 4: +/-20 mA

dataFromFIFO:

the raw data read from the FIFO buffer.

** calibratedAI:*

[Output] the floating point value.

Return Values

None

Example

```
int slot=1;
int i;
float calibratedAI=0;
printf("Start printing all the data:\n\n\r");
Open_Slot(slot);

for(i=0;i<totalScaned;i++);
{
   i8014W_CalibrateData(slotIndex,
   gainArr[i %scanChCount],hexData[i], &calibratedAI);
   printf("Arr[%d]=[%5.4f]\t",i%scanChCount,calibratedAI);
}
```

3.16. i8014W_CalibrateDataHex

This function is used to calibrate the raw data read in Magic Scan process.

Syntax

```
void i8014W_CalibrateDataHex
(
    int slot,
    shortiGain,
    shortdataFromFIFO,
    short* calibratedAI
);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

iGain:

specifies the input type (0 ~ 4), where:0: +/-10 V, 1: +/-5 V, 2: +/-2.5 V, 3: +/-1.25 V, 4: +/-20 mA

dataFromFIFO:

the raw data read from the FIFO buffer.

** calibratedAI :*

[Output] the calibrated hexadecimal value.

Return Values

None

Example

```
int slot=1;
int i;
float calibratedAI=0;
printf("Start printing all the data:\n\n\r");
Open_Slot(slot);

for(i=0;i<totalScaned;i++);
{
    i8014W_CalibrateDataHex (slotIndex,
    gainArr[i %scanChCount],hexData[i], &calibratedAI);
    printf("Arr[%d]=[%#x]\t",i%scanChCount,calibratedAI);
}
```

3.17. i8014W_UnLockFIFO

This function is used to unlock the FIFO buffer when it is locked after being filled. Ensure that the FIFO buffer is unlocked and cleared before starting the next Magic Scan process.

Syntax

```
void i804W_UnLockFIFO (int slot);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

Return Values

None

Example

```
int slot;
slot = 1;
Open_Slot(slot);

i804W_UnLockFIFO (slot);
```

3.18. i8014W_ClearFIFO

This function is used to clear the FIFO buffer after the UnlockFIFO function has been executed. Ensure that the FIFO buffer is unlocked and cleared before starting the next Magic Scan process.

Syntax

```
void i804W_ClearFIFO (int slot);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

Return Values

None

Example

```
int slot;
slot = 1;
Open_Slot(slot);

i804W_ClearFIFO (slot);
```

3.19. i8014W_InstallMagicScanISR

This function is used to install the ISR to control to control interrupt events form the FIFO buffer. When the amount of data in the FIFO buffer is greater than the value defined by the triggerLevel parameter (as per the table below), an interrupt event will occurs and the ISR will be executed to handle the event. In the ISR, use the ReadFIFO to transfer data from the FIFO buffer and then ClearInt to restart the status of the interrupt.

Syntax

```
short i804W_ InstallMagicScanISR
(
    int slot,
    void (*isr)
    (int slot),
    inttriggerLevel
);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

**isr (int slot):*

the function pointer passed for the ISR.

triggerLevel:

specifies the interrupt trigger condition (0 - 7) based on the amount of data in the FIFO buffer. If the value is set to greater than 7, it will be automatically forced to 7.

If the amount of data in the FIFO buffer is greater than the value defined by the triggerLevel parameter, the interrupt will be triggered and the ISR will be executed to handle the interrupt event.

The following is a definition of the triggerLevel values table lists the definition of triggerLevel and associated Data Count values:

triggerLevel	Data Count
0	8
1	16
2	32
3	64
4	128
5	256
6	512
7	2048

Return Values

0 = No Error

For other return values, see the Appendix A. Error Code.

Example

```
void main()
{
    int slot, TrgLevel;
    slot = 1;
    TrgLevel=100;
    Open_Slot(slot);

    i8014W_Install_MagicScanISR(slot,ISRFUN, TrgLevel);
    i8014W_ConfigMagicScan(…);
    i8014W_StartMagicScan(slot);
    ...
    while(1)
    {
      if(IntCnt>1)
      {
            i8014W_UnInstall_MagicScanISR(slot);
            break;
      }
    }
    ...
}

voidISRFUN(int slot);
{
    IntIntCnt=0;
    IntCnt++;
    ret=i8014W_ReadFIFO(slot, hexData+totalScaned,
    TargetCnt-totalScaned,&readCnt);
    if(readCnt>0)
    {
            totalScaned+=readCnt;
            printCom1("TotalScaned= %d\n\r",totalScaned);
            totalRead+=readCnt;
    }
    i8014W_ClearInt(slot);
}
```

3.20. i8014W_UnInstallMagicScanISR

This function is used to uninstall the Magic Scan ISR.

Syntax

```
short i804W_UnInstallMagicScanISR(int slot);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

Return Values

0 = No Error

For other return values, see the Appendix A. Error Code.

Example

```
int slot;
slot = 1;
Open_Slot(slot);

i804W_UnInstallMagicScanISR (slot);
```

3.21. i8014W_ClearInt

This function is used to clear the status of the Magic Scan interrupts. When using ISR, this function should be called to clear the status of any interrupts that have been triggered in order to continue processing future interrupt events.

Syntax

```
void i804W_ClearInt (int slot);
```

Parameter

slot:

specifies the slot number (1 ~ 8).

Return Values

None

Example

```
int slot;
slot = 1;
Open_Slot(slot);

i804W_StopMagicScan (slot);
```

4. Troubleshooting

This chapter discusses how to solve any problems you may encounter.

This chapter contains:

- Service/Request Requirements
- What to do when the data read from I-8014W seems unstable
- How to solve the FIFO LATCHED error (-6)

4.1. Service/Request Requirements

If you are using a stable signal source such as a battery to output a signal to the I-8014W module and are getting incorrect or unstable data, prepare the following three items and e-mail them to service@icpdas.com .

- The image of the physical wiring
- The file saved from the Basic Information tab
- The file saved from the AI Test tab

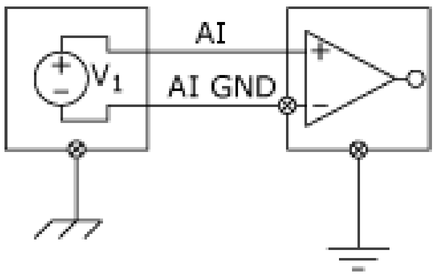
4.2. What to do when the data read from I-8014W seems unstable

If the voltage can be measured correctly when testing using a battery, but not when using the real signal source, the error may be caused by any or all of the following factors:

- A noise-corrupted signal source
- Instability in the signal source
- A floating signal source that is not referenced to a system ground (earth or building ground)

Because of the high-speed data acquisition function of the I-8014W, any noise coupled to a signal or any change in voltage on an unstable source is also captured. In this situation, signal filtering or isolation should be considered in order to enhance the quality of the signal.

It is recommended that the V- pin is connected to the AGND (system ground) pin when measuring differential signals, as shown in the figure.



4.3. How to solve the FIFO LATCHED error (-6)

After the ***StartMagicScan*** instruction is executed, it will continue scanning the channels and converting data unless the ***StopMagicScan*** command is executed. Consequently, the converted data is continuously saved to the FIFO buffer. If the Magic Scan is not stopped after obtaining the required data, or the data is not read from the FIFO buffer within the required time frame, the FIFO buffer will be filled and then locked. When the FIFO buffer is locked, the FIFO LATCHED error (-6) will occur and any new data will not be able to be saved to the FIFO buffer.

To solve this error, execute the following instructions:

1. Stop Magic Scan using the ***StopMagicScan*** function.
2. Read the remaining data in the FIFO buffer using the ***ReadFIFO*** function, or clear it using the ***ClearFIFO*** function.
3. Unlock the FIFO buffer using the ***UnLockFIFO*** function.
4. Restart Magic Scan using the ***StartMagicScan*** function.

Appendix A. Error Code

Error Code	Definition	Description
0	NoError	This indicates that there have been no errors
-1	ID_ERROR	There was a problem with the module ID
-2	SLOT_ERROR	There was a Slot index error (0 - 7)
-3	CHANNEL_ERROR	There was a Channel index error (0 - 15)
-4	GAIN_ERROR	There was a Gain error (0 - 4)
-5	FIFO_EMPTY	There is no data in the FIFO buffer
-6	FIFO_LATCHED	The FIFO buffer is full and has been latched
-7	FIFO_OVERFLOW	The FIFO buffer is full
-8	TX_NOTREADY	There was an error between the primary FPGA and the secondary FPGA

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