

**PCI1812/1814**  
**Multi-function DAS Cards**  
**User's Guide**

**V 1.01**

1 Features .....	3
2 Specifications .....	3
3 Pin Assignments.....	5
4 Operation Theory .....	7
4.1 analog input.....	7
4.2 Analog Input Range .....	8
4.3 Multichannel Scanning Considerations.....	8
5 Connecting Analog Input Signals .....	10
5.1 Single-ended Mode .....	11
5.2 Differential input mode.....	11
6 A/D Trigger Modes.....	12
6.1 Software trigger.....	12
6.2 Timer Pacer Fixed Channel Trigger.....	13
6.3 Timer Pacer Scan Channel Trigger .....	13
6.4 Input Range Select .....	13
6.5 channel select .....	13
6.6 A/D data format .....	14
6.7 sample interval .....	15
6.8 Read Operation .....	15
7 D/A output .....	16
8 Digital input and Digital output .....	16
9 Board ID.....	17
10 Programming Guide.....	17
10.1 Function List .....	17
10.2 build application.....	24
Appendix.....	28

# 1 Features

The PCI1812(PCI1814) Data Acquisition Card provides the following advanced features:

- 32-bit PCI-Bus
- analog input resolution: 12bit for PCI1812,14 bit for PCI1814
- On-board A/D 512 word FIFO memory (build in FPGA)
- Auto-scanning channel selection
- Up to 100KHz A/D sampling rates
- 16 single-ended or 8 differential analog input channels
- Bipolar or Unbipolar input signals
- Programmable Gain Control (x1, x2, x5, x10)
- Two 12-bit analog output channels
- 22 digital TTL output channels
- 22 digital TTL input channels
- Two A/D trigger modes: software trigger, programmable pacer trigger
- Integrated DC-to-DC converter for stable analog power source
- 37-pin D-type connector
- Board ID

# 2 Specifications

## Analog Input(A/D)

- Resolution : 12-bit(PCI1812) , 14-bit(PCI1814)
- Numbers of Input Channel: 16 single-ended or 8 differential
- Input Range: (Programmable)
  - Bipolar:  $\pm 10V, \pm 5V, \pm 2V, \pm 1V$
  - Unbipolar:  $+10V, +5V, +2V, +1V$
- Conversion Time: 10  $\mu$  sec
- sampling rates : 100KHz
- Analog Input Over-voltage Protection: Continuous 25V max.
- Accuracy:
  - For PCI1812
    - $\pm 10V, \pm 5V, 0\sim+10V, 0\sim+5V$  :  $\pm 2$  LSB
    - $\pm 2V, \pm 1V, 0\sim+2V, 0\sim+1V$  :  $\pm 4$ LSB
  - For PCI1814
    - $\pm 10V, \pm 5V, 0\sim+10V, 0\sim+5V$  :  $\pm 4$ LSB
    - $\pm 2V, \pm 1V, 0\sim+2V, 0\sim+1V$  :  $\pm 6$ LSB

- Input Impedance: 10 M
- Trigger Modes: Software, Timer Pacer
- Data Transfer Modes: Program polling control
- FIFO Depth: 512 words

#### Analog Output

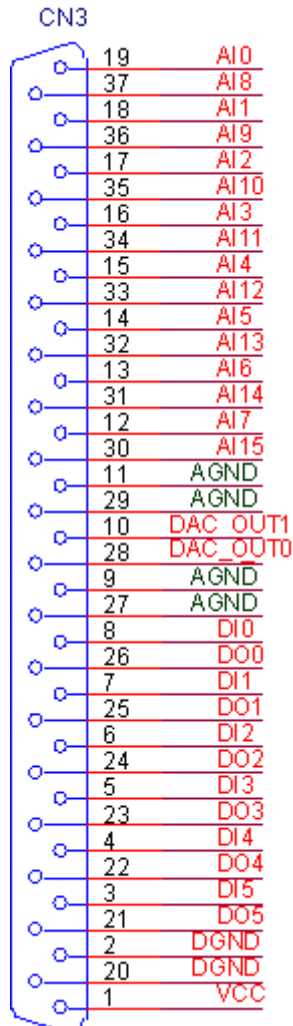
- Resolution : 12-bit for PCI1812, 14 bit for PCI1814
- Numbers of Input Channel: 2
- Output Range:  $\pm 10V$
- Conversion Time: 20 msec
- Accuracy:  $\pm 3$  LSB for PCI1812  
 $\pm 6$  LSB for PCI1814

#### Digital I/O (DIO)

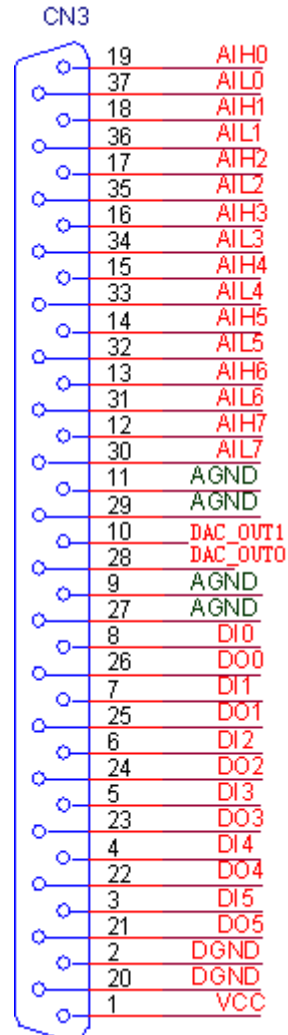
- Number of channel: 22 DI & 22 DO (TTL compatible)
- Input Voltage:
  - Low: Min. 0V; Max. 0.8V
  - High: Min. +2.0V
- Input Load:
  - Low: +0.5V @ -0.2mA max.
  - High: +2.7V @ +20 uA max.
- Output Voltage:
  - Low: Min. 0V; Max. 0.4V
  - High: Min. +2.4V
- Output Driving Capacity:
  - Low: Max. +0.5V at 8.0mA (Sink)
  - High: Min. 2.7V at 0.4mA(Source)

### 3 Pin Assignments

(For single-ended connection)

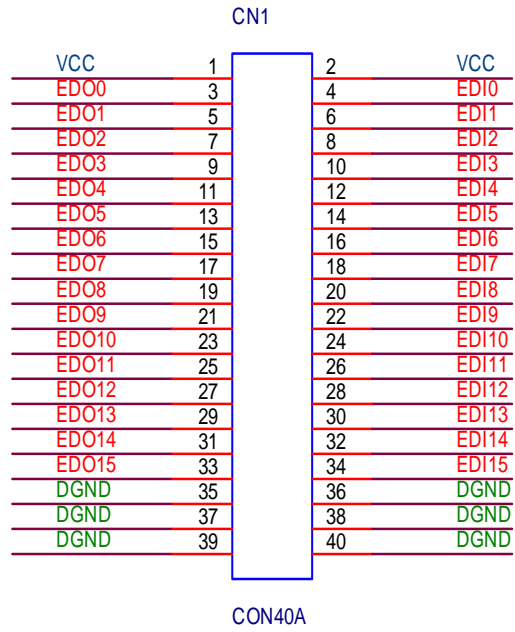


(For differential connection)



### DB37 connector

- AI n: Analog Input Channel n (single-ended)
- AIH n: Analog High Input Channel n (differential)
- AIL n: Analog Low Input Channel n (differential)
- AGND: Analog input/output ground
- DGND: Digital input/output ground
- DAC\_OUT n: Analog Output Channel
- DI\_n : Digital input Channel n
- DO\_n : Digital output Channel n



### CN1 connector

DGND: Digital input/output ground

VCC : +5V

EDOn : Digital output Channel n

EDIn : Digital input Channel n

## 4 Operation Theory

### 4.1 analog input

Figure 4-1 shows the analog input circuitry of PCI1812(PCI1814)

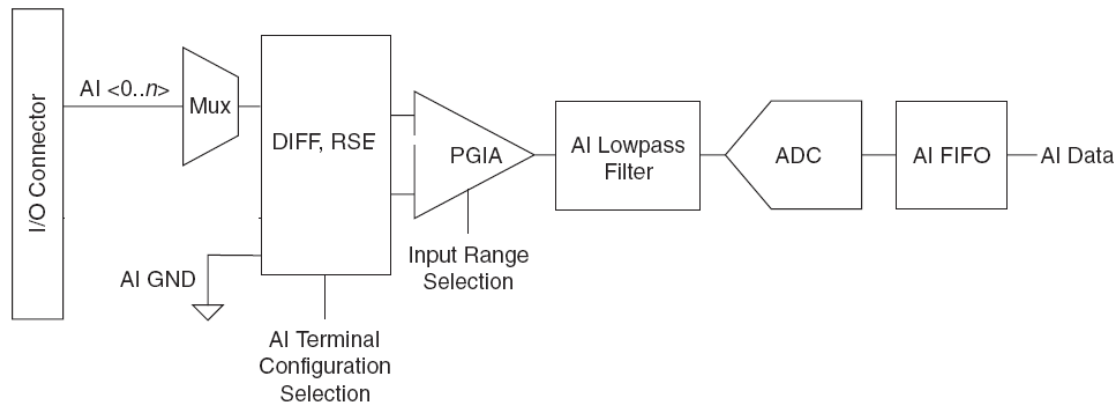


Figure 4-1

#### I/O Connector

You can connect analog input signals to the PCI1812(PCI1814) through the I/O connector. The proper way to connect analog input signals depends on the analog input ground-reference settings

#### MUX

PCI1812(PCI1814) has one analog-to-digital converter (ADC). The multiplexers (MUX) route one AI channel at a time to the ADC through PGIA.

The analog input ground-reference settings circuitry selects between differential, referenced single-ended input modes. Each AI channel can use a different mode.

#### Programmable Gain Instrumentation Amplifier (PGIA)

The programmable gain instrumentation amplifier (PGIA) is a measurement and instrument class amplifier that minimizes settling times for all input ranges. The PGIA can amplify or attenuate an AI signal to ensure that you use the maximum resolution of the ADC. PCI1812(PCI1814) use the PGIA to deliver high accuracy even when sampling multiple channels with small input ranges at fast rates.

#### A/D Converter

The analog-to-digital converter (ADC) digitizes the AI signal by converting the analog voltage into a digital number.

#### AI FIFO

PCI1812(PCI1814) can perform both single and multiple A/D conversions of a fixed or infinite number of samples. A large first-in-first-out (FIFO) buffer holds data during AI acquisitions to ensure that no data is lost.

## 4.2 Analog Input Range

Input range refers to the set of input voltages that an analog input channel can digitize with the specified accuracy. The PGIA amplifies or attenuates the AI signal depending on the input range. You can individually program the input range of each AI channel on your PCI1812(PCI1814). The input range affects the resolution of PCI1812(PCI1814) for an AI channel. Resolution refers to the voltage of one ADC code. For example, a 12-bit ADC converts analog inputs into one of 4096(=2<sup>12</sup>) codes—that is, one of 4096 possible digital values. These values are spread fairly evenly across the input range. So, for an input range of -10 V to 10 V, the voltage of each code of a 12-bit ADC is:

$$\frac{(10V - (-10V))}{2^{12}} = 4.88mV$$

PCI1812(PCI1814) use a calibration method that requires some codes (typically about 5% of the codes) to lie outside of the specified range. This calibration method improves absolute accuracy, but it increases the nominal resolution of input ranges by about 5% over what the formula shown above would indicate.

Choose an input range that matches the expected input range of your signal. A large input range can accommodate a large signal variation, but reduces the voltage resolution. Choosing a smaller input range improves the voltage resolution, but may result in the input signal going out of range. table 4-1 show the input ranges and resolutions supported of PCI1812

Input range	voltage of each code
<b>【-10V, +10V】</b>	4.88mv
<b>【-5 V , +5V】</b>	2.44mv
<b>【-2V , +2 V】</b>	0.976mv
<b>【-1V , +1V】</b>	0.488mv
<b>【0V , +10V】</b>	2.44mv
<b>【0V , +5V】</b>	1.22mv
<b>【0V , +2V】</b>	0.488mv
<b>【0V , +1V】</b>	0.224mv

table 4-2 show the input ranges and resolutions supported of PCI1814

Input range	voltage of each code
【-10V, +10V】	1.22mv
【-5 V , +5V】	0.61mv
【-2V , +2 V】	0.224mv
【-1V , +1V】	0.122mv
【0V , +10V】	0.61mv
【0V , +5V】	0.305mv
【0V , +2V】	0.122mv
【0V , +1V】	0.056mv

### 4.3 Multichannel Scanning Considerations

PCI1812(PCI1814) can scan multiple channels at high rates and digitize the signals accurately. However, you should consider several issues when designing your measurement system to ensure the high accuracy of your measurements. In multichannel scanning applications, accuracy is affected by settling time. When your PCI1812(PCI1814) switches from one AI channel to another AI channel, the device configures the PGIA with the input range of the new channel. The PGIA then amplifies the input signal. Settling time refers to the time it takes the PGIA to amplify the input signal to the desired accuracy before it is sampled by the ADC.

PCI1812(PCI1814) are designed to have fast settling times. However, several factors can increase the settling time which decreases the accuracy of your measurements. To ensure fast settling times, you should do the following (in order of importance):

- Use low impedance sources
- Use short high-quality cabling
- Avoid scanning faster than necessary

The following sections contain more information about these factors.

#### Use Low Impedance Sources

To ensure fast settling times, your signal sources should have an impedance of  $<1k \Omega$ . Large source impedances increase the settling time of the PGIA, and so decrease the accuracy at fast scanning rates. Settling times increase when scanning high-impedance signals due to a phenomenon called charge injection. Multiplexers contain switches, usually made of switched capacitors. When one of the channels, for example channel 0, is selected in a multiplexer, those capacitors accumulate charge. When the next channel, for example channel 1, is selected, the accumulated charge leaks backward through channel 1. If the output impedance of the source connected to channel 1 is high enough, the resulting reading of channel 1 can be partially affected by the voltage on channel 0. This effect is referred to as ghosting. If your source impedance is high, you can decrease the scan rate to allow the PGIA more time to settle. Another option is to use a voltage follower circuit external to your DAQ device

to decrease the impedance seen by the DAQ device.

## **Use Short High-Quality Cabling**

Using short high-quality cables can minimize several effects that degrade accuracy including crosstalk, transmission line effects, and noise. The capacitance of the cable also can increase the settling time. We recommend using individually shielded twisted-pair wires that are 2 m or less to connect AI signals to the device.

## **Avoid Scanning Faster Than Necessary**

Designing your system to scan at slower speeds gives the PGIA more time to settle to a more accurate level. Here are two examples to consider.

### **Example 1**

Averaging many AI samples can increase the accuracy of the reading by decreasing noise effects. In general, the more points you average, the more accurate the final result. However, you may choose to decrease the number of points you average and slow down the scanning rate. Suppose you want to sample 10 channels over a period of 20 ms and average the results. You could acquire 50 points from each channel at a scan rate of 25 kS/s. Another method would be to acquire 100 points from each channel at a scan rate of 50 kS/s. Both methods take the same amount of time. Doubling the number of samples averaged (from 50 to 100) decreases the effect of noise by a factor of 1.4 (the square root of 2). However, doubling the number of samples (in this example) decreases the time the PGIA has to settle from 40 $\mu$ s to 20 $\mu$ s. In some cases, the slower scan rate system returns more accurate results.

### **Example 2**

If the time relationship between channels is not critical, you can sample from the same channel multiple times and scan less frequently. For example, suppose an application requires averaging 100 points from channel 0 and averaging 100 points from channel 1. You could alternate reading between channels—that is, read one point from channel 0, then one point from channel 1, and so on. You also could read all 100 points from channel 0 then read 100 points from channel 1. The second method switches between channels much less often and is affected much less by settling time.

## **5 Connecting Analog Input Signals**

The PCM1812 provides 16 single-ended or 8 differential analog input channels. The analog signals can be converted to digital value by the A/D converter. To avoid ground loops and to obtain more accurate measurements, it is quite important to understand the signal source type and how to choose the analog input modes: signal-ended or differential. The PCM1812 device

selects 16 single-ended or 8 different analog by software.

## 5.1 Single-ended Mode

The single-ended mode has only one input relative to ground and is suitable for connecting with a floating signal source. A floating source is one that does not have any connection to ground. Figure 5-1 shows the single-ended connection. Note that when more than two floating sources are available, the source must be with common ground

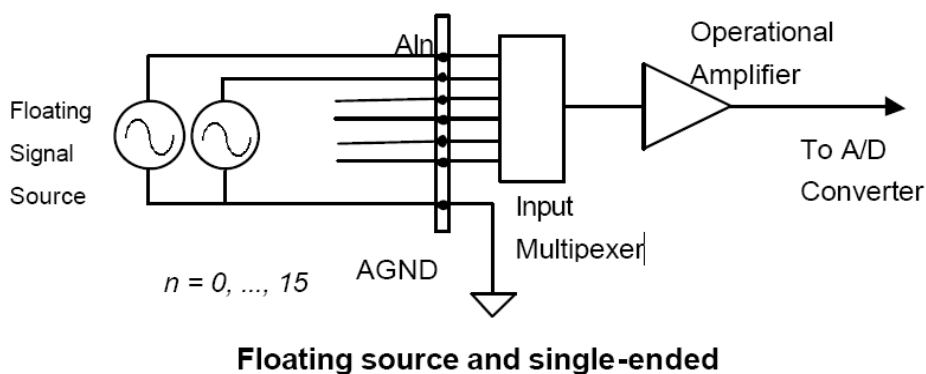


Figure5-1

## 5.2 Differential input mode

The differential input mode provides two inputs that respond to differences in signals. If the signal source has one side connected to local ground, the differential mode can be used to reduce the effect of ground loops. Figure 5-2-1 shows the connection for differential input mode. However, if the signal source is locally grounded, the single-ended mode can be used when the  $V_{cm}$  (Common Mode Voltage) is very small and the effect of ground loops is minimal.

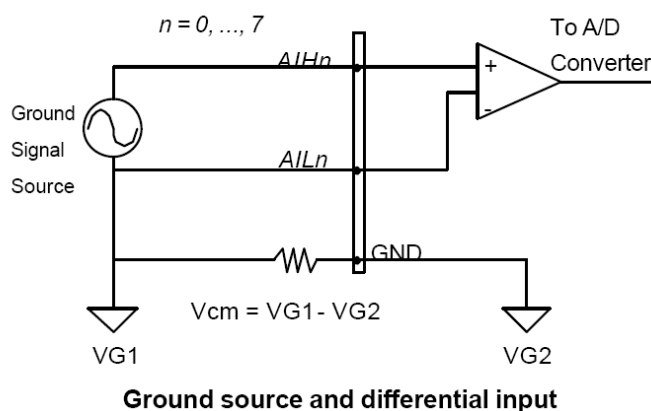


Figure 5-2-1

A differential mode must be used when the signal source is differential. A differential source means that the ends of the signal are not grounded. To avoid the danger of high voltages between the local ground of the signal and the ground of the PC system, a shorted ground path must be connected. Figure 5-2-2 shows the connection for a differential source.

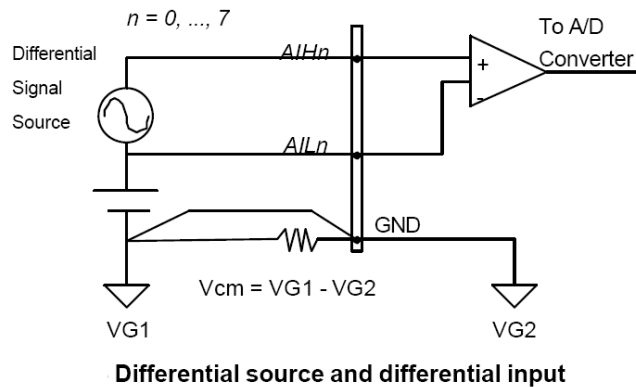


Figure 5-2-2

if the signal source are both floating, you should use the differential mode, and the floating signal source should be connected as the Figure 5-2-3

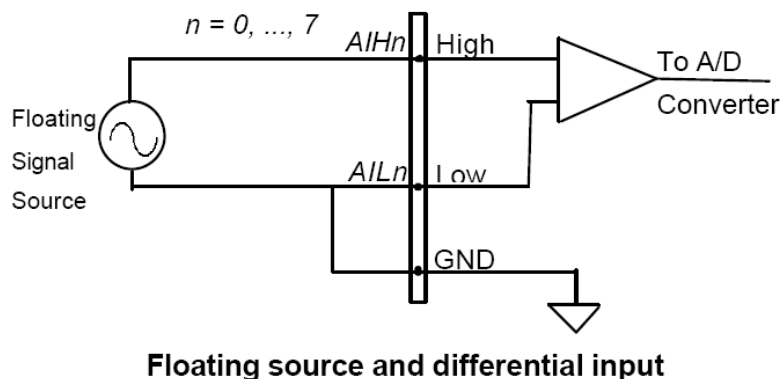


Figure 5-2-3

## 6 A/D Trigger Modes

In the PCI1812(PCI1814), A/D conversion can be triggered by three sources.

### 6.1 Software trigger

This trigger source is software controllable. That is, the A/D conversion starts when Command is written into register. This trigger mode is suitable for low speed A/D conversions. Under this mode, the timing of the A/D conversion is

fully controlled by the software. However, it is difficult to control a fixed A/D conversion rate unless another timer interrupt service routine is used to generate a fixed rate trigger.

## 6.2 Timer Pacer Fixed Channel Trigger

A counter build in FPGA is used to provide a trigger source for A/D conversion at a fixed rate and fixed channels. This mode is ideal for high speed A/D conversion. It's recommended that this mode be used if your application needs a fixed and precise A/D sampling rate.

## 6.3 Timer Pacer Scan Channel Trigger

Auto Scan Trigger command is used to provide a fixed rate and scan channel is: [Maxchannel], [Maxchannel-1],..., 3,2,1,0. [Maxchannel], [Maxchannel-1],... ,3,2,1,0. It's recommended that this mode be used if your application needs a fixed and precise A/D sampling rate and many channels.

## 6.4 Input Range Select

When you write different data to register, device can select different input range; the followed table explains relationship of data and input range

Bit3	bit1	bit0	Input range
0	0	0	[-10 , 10 ]
0	0	1	[-5 , 5 ]
0	1	0	[-2 , 2 ]
0	1	1	[-1 , 1 ]
1	0	0	[0 , 10 ]
1	0	1	[0 , 5 ]
1	1	0	[0 , 2 ]
1	1	1	[0 , 1 ]

## 6.5 channel select

After you connect analog signal to PCI1812(PCI1814) devices, you should set channel before you start A/D convert, the followed table explain the relationship of data and channel select.

data	CH	data	CH	data	CH
00000	AI0	01000	AI8	10000	AI0[H-L]
00001	AI1	01001	AI9	10001	AI1[H-L]
00010	AI2	01010	AI10	10010	AI2[H-L]
00011	AI3	01011	AI11	10011	AI3[H-L]
00100	AI4	01100	AI12	10100	AI4[H-L]
00101	AI5	01101	AI13	10101	AI5[H-L]
00110	AI6	01110	AI14	10110	AI6[H-L]
00111	AI7	01111	AI15	10111	AI7[H-L]

Note that when you need scan multiplex channels, channel set data is the max channel, PCI1812(PCI1814) devices will automatic decrease channels. For example , in Single-ended Mode , you set max channel is 5, PCI1812(PCI1814) scan channel order is [5,4,3,2,1,0,5,4,3,2,1,0] ,the same in Differential input mode. You can not mixed Single-ended Mode with Differential input mode when you scan multiplex channels. But you can mix Single-ended Mode with Differential input mode when you read ad convert data one time

## 6.6 A/D data format

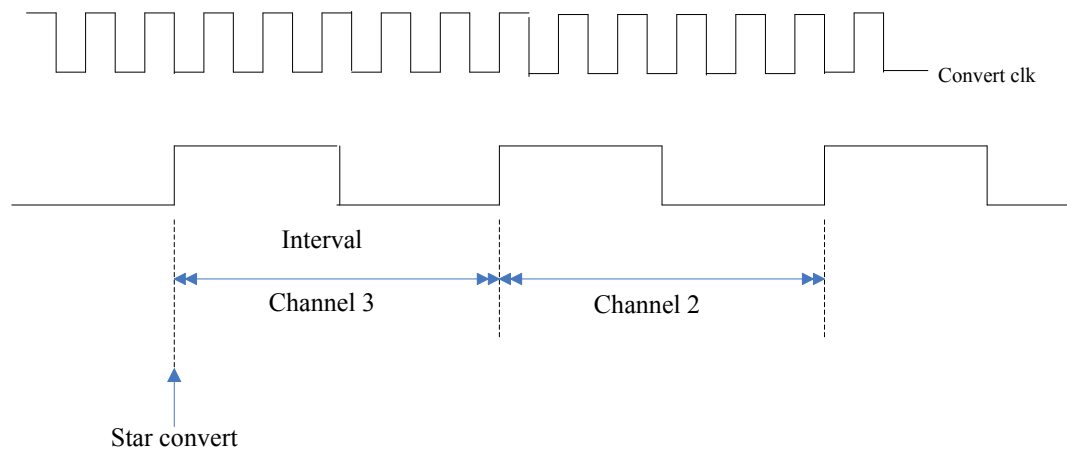
A/D data read from ad FIFO is 32 bit. Means of each bit as followed table

Bit31~Bit24	Bit23~Bit21	Bit20~Bit16	Bit15~Bit12	Bit11~Bit0
0	Input range	convert channel	0	AD convert data

When you read data from ad FIFO, the data include three parts, one is channel input rang ,one is channel select ,and the other is in the select input range and channel ,the AD convert data. For example ,if you read data from ad FIFO is 0x030047f , that means AD convert data is 0x47f,convert channel select is 0x10 which means different input channel 0, input range is 0x01 which means input range is [-5V~+5V] ,and so you can calculate analog input voltage in different input channel 0 is

$$\frac{0x47f \times (5v - (-5v))}{4096} + (-5v) = -1.908V$$

## 6.7 sample interval



PCI1812(PCI1814) have a hardware timer/counter which can control intervals to trigger AD. User can set different data to control the sample rate; the relationship between set data with sample interval time is followed:

$$data \times 400ns = Interval\_time$$

Data is 24 bit wide, and the smallest value is 0x19. max value is 0xfffff;

## 6.8 Read Operation

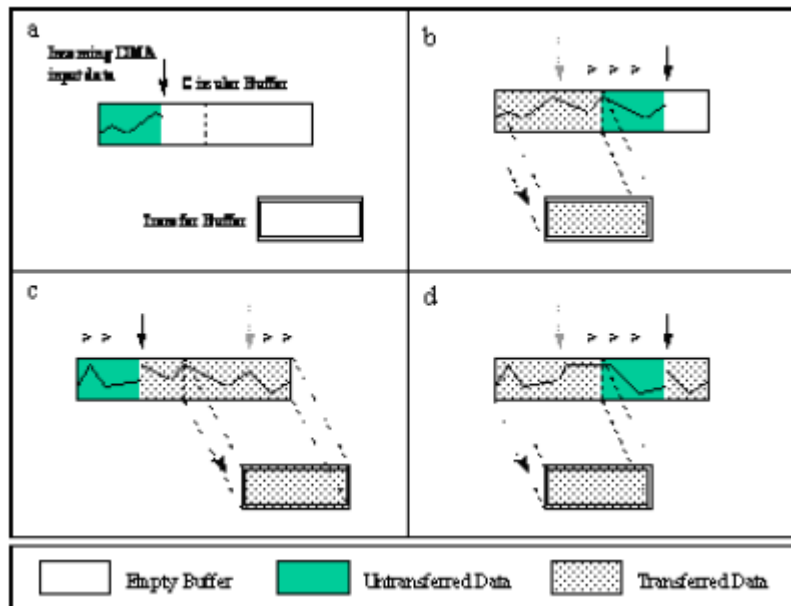
Data after A/D convert is storage in A/D FIFO. When you scan multiplex channels or continue read one channel, data first storage in ad FIFO, when ad FIFO half full (256 words), PCI1812(PCI1814) occurs interrupt to notify PC to read data from ad FIFO to PC buffer, A/D convert data can continue be written to ad FIFO until ad FIFO full, if the PC do not read the data, A/D convert data will lost. And the buffer must be time of 256. The PCI1812(PCI1814) support synchronous read model and asynchronous read model

Synchronous read model: the function will not return until data sample complete.

Asynchronous read model: the function will immediate return after call, when sample data complete, there is a flag set. In this model, you can select double buffer or buffer operation

The double-buffered input begins when the device starts writing data into the first half of the circular buffer (a). Refer to figure below. When the device starts writing to the second half of the circular buffer, the data is copied from the first half to the transfer buffer (b) also known as user buffer. You can now process the data in the transfer buffer depending on the application needs. After the board has filled the second half of the circular buffer, the board returns to the first half buffer and overwrites the old data. The data is copied from the second half of the circular buffer to the transfer buffer (c). The data in the transfer buffer is again available for process. The process may be repeated endlessly to provide a continuous stream of data to your

application (d).



## 7 D/A output

In the PCI1812(PCI1814), there are two 12bit digital to analog channels which can output analog voltage from -10V to +10V

Just like A/D, each code of D/A register represent voltage, which is

$$\frac{(10V - (-10V))}{2^{12}} = 4.88mV$$

so , the relationship between output voltage and digital code is

$$\frac{digitalcode \times 20}{4096} - 10V = vol$$

Table 7-1 list coding between digital and output voltages

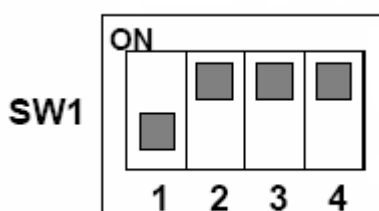
Digital code	Analog output
0x0	-10V
0x800	0v
0xffff	+10V

## 8 Digital input and Digital output

PCI1812(PCI1814) have 22 channel digital input and 22 channel digital output. There are 6 digital output channels and 6 digital input channels with 5.1K pull up resistance in DB37 connector, and there are 16 digital input channels and 16 digital output channels in CON3, all of digital input channels and digital output channels are TTL compliant.

## 9 Board ID

There are 4 bit switch on the PCI1812(PCI1814) device , you can choose board id with switches turn to "ON" or not(OFF).When the switch turn to "ON", means that the bit is '0'.else means the bit is '1'. The value is board id. For example, if the four bits of your device is as followed, and board id is  $1 \times 2^0 + 0 \times 2^1 + 0 \times 2^2 + 0 \times 2^3 = 1$



## 10 Programming Guide

### 10.1 Function List

\*when you use PCI1814 device , replace followed function PCI1812 with PCI1814

*U32 PCI1812\_Register\_Card (U32 card\_number)*

Function: get resource which will used in other functions

Arguments:

Cardnumber: board id, must be opened by Open PCI1812\_Register\_Card function first.

Note: before you use other functions ,run this function to get resource

Return Code:	
Success	API success
ErrorCardNumber	input parameter error
ErrorFailure	API fault

*U32 PCI1812\_Release(U32 cardnumber)*

Function: Release resource which get by PCI1812\_Register\_Card

Arguments:

Cardnumber: board id, must be opened by Open PCI1812\_Register\_Card function first.

Note: if you are not use the card again, use this function

Return Code:	
Success	API success
ErrorCardNumber	input parameter error

ErrorFailure	Api fault
--------------	-----------

*U32 PCI1812\_ReadChannel(U32 cardnumber,U32 channel,U32 adrange,U32 \*value)*

**Function:** Read ad data one time, triggered by software

**Arguments:**

- cardnumber: board id, must be opened by Open PCI1812\_Register\_Card function first.
- channel : channel select , see [channel select](#)
- adrange : ad range select, see [Input range select](#)
- value : address which will storage the date read from register,

<b>Return Code:</b>	
Success	API success
ErrorCardNumber	input parameter error
ErrorParameters	input parameter error
ErrorFailure	API fault

*U32 PCI1812\_AsynchContReadChanel(U32 cardnumber,U32 channel, U32 adrange, U32 \*buffer,U32 readcount,U32 Interval, U32 \*asynchflag)*

**Function:** continue read data at fix channel and fix sample rate at asynchronous read operation, more information to see [read operation](#)

**Arguments:**

- cardnumber: board id, must be opened by Open PCI1812\_Register\_Card function first.
- channel : channel select , see [channel select](#)
- adrange : ad range select, see [Input range select](#)
- buufer : address which point to where read data to storage, see [AD data format](#)
- readcount : the deep of storage , also means how many times read from ad FIFO, which must be time of 256 ,for more information see [read operation](#)
- interval : PCI1812 sample rate set. More information see [sample interval](#)
- asynchflag : flag of asynchronous read, more information see [read operation](#),the data of asynchflag is followed table

asynchflag	means
FlagFailure	Asynchronous read operation failure
FlagSuccess	Asynchronous read operation completed

Note : readcount must be time of 256 ,and you should make sure read operation will not overflow the buffer which to storage ad data

<b>Return Code:</b>	
Success	API success
ErrorCardNumber	input parameter error
ErrorParameters	input parameter error
ErrorFailure	API fault
ErrorBusy	Another thread is read ad FIFO data

*U32 PCI1812\_AsynchContScanReadChanel(U32 cardnumber,U32 channel, U32 adrange, U32 \*buffer,U32 readcount,U32 Interval, U32 \*asynchflag)*

**Function** : continue read data at fix sample rate for multiplex channels at asynchronous read operation, more information to see [read operation](#)

**Arguments:**

cardnumber: board id, must be opened by Open PCI1812\_Register\_Card function first.  
channel : channel select , see [channel select](#)  
adrange : ad range select, see [Input range select](#)  
buufer : address which point to where read data to storage, see [AD data format](#)  
readcount : the deep of storage , also means how many times read from ad fifo, which must be time of 256 ,for more information see [read operation](#)  
interval : PCI1812 sample rate set. More information see [sample interval](#)  
asynchflag : flag of asynchronous read, more information see [read operation](#)  
the data of asynchflag is followed table

asynchflag	means
FlagFailure	Asynchronous read operation failure
FlagSuccess	Asynchronous read operation completed

Note : readcount must be time of 256 ,and you should make sure read operation will not overflow the buffer which to storage ad data.

Return Code:	
Success	API success
ErrorCardNumber	input card number error
ErrorParameters	input parameter error
ErrorFailure	API fault
ErrorBusy	Another thread is read ad FIFO data

*U32 PCI1812\_AsynchContScanReadChanelDblBuffer(U32 cardnumber,U32 channel, U32 adrange, U32 \* buffer,U32 readcount,U32 Interval, U32 \*asynchflag)*

**Function** : continue read data at fix sample rate for multiplex channels at asynchronous double buffer read operation, more information to see [read operation](#)

**Arguments:**

cardnumber: board id, must be opened by Open PCI1812\_Register\_Card function first.  
channel : channel select , see [channel select](#)  
adrange : ad range select, see [Input range select](#)  
buufer : address which point to where read data to storage, see [AD data format](#)  
readcount : the deep of storage , also means how many times read from ad FIFO, which must be time of 256 ,for more information see [read operation](#)  
interval : PCI1812 sample rate set. More information see [sample interval](#)  
asynchflag : flag of asynchronous read, more information see [read operation](#) the data of asynchflag is followed table

asynchflag	
FlagFailure	Asynchronous read operation failure
FlagHalfSuccess	Asynchronous read half buffer completed
FlagSuccess	Asynchronous read operation completed

Note : readcount must be time of 512 ,and you should make sure read operation will not overflow the buffer which to storage ad data.

<b>Return Code:</b>	
Success	API success
ErrorCardNumber	input card number error
ErrorParameters	input parameter error
ErrorFailure	API fault
ErrorBusy	Another thread is read ad FIFO data

*U32 PCI1812\_AsynchContReadChanelDblBuffer(U32 cardnumber,U32 channel, U32 adrange, U32 \*buffer,U32 readcount,U32 Interval, U32 \*asynchflag)*

**Function** : continue read data at fix sample rate at fixed channel at asynchronous double buffer read operation, more information to see [read operation](#)

**Arguments:**

cardnumber: board id, must be opened by Open PCI1812\_Register\_Card function first.

channel : channel select , see [channel select](#)

adrange : ad range select, see [Input range select](#)

buufer : address which point to where read data to storage, see [AD data format](#)

readcount : the deep of storage , also means how many times read from ad fifo, which must be time of 512 ,for more information see [read operation](#)

interval : PCI1812 sample rate set. More information see [sample interval](#)

asynchflag : flag of asynchronous read, more information see [read operation](#) the data of asynchflag is followed table

asynchflag	means
FlagFailure	Asynchronous read operation failure
FlagHalfSuccess	Asynchronous read half buffer completed
FlagSuccess	Asynchronous read operation completed

Note : readcount must be time of 512 ,and you should make sure read operation will not overflow the buffer which to storage ad data.

<b>Return Code:</b>	
Success	API success
ErrorCardNumber	input card number error
ErrorParameters	input parameter error
ErrorFailure	API fault
ErrorBusy	Another thread is read ad FIFO data

*U32 PCI1812\_SynchContReadChanel(U32 cardnumber,U32 channel, U32 adrange, U32 \*  
buffer,U32 readcount,U32 Interval)*

**Function** : continue read data at fix sample rate at fixed channel at synchronour read operation, more information to see [read operation](#)

**Arguments:**

cardnumber: board id, must be opened by Open PCI1812\_Register\_Card function first.

channel : channel select , see [channel select](#)

adrange : ad range select, see [Input range select](#)

buufer : address which point to where read data to storage, see [AD data format](#)

readcount : the deep of storage , also means how many times read from ad fifo, which must be time of 256 ,for more information see [read operation](#)

interval : PCI1812 sample rate set. More information see [sample interval](#)

Note : readcount must be time of 512 ,and you should make sure read operation will not overflow the buffer which to storage ad data.

<b>Return Code:</b>	
Success	API success
ErrorCardNumber	input card number error
ErrorParameters	input parameter error
ErrorFailure	API fault
ErrorBusy	Another thread is read ad FIFO data

*U32 PCI1812\_SynchContScanReadChanel(U32 cardnumber,U32 channel, U32 adrange, U32 \*buffer,U32 readcount,U32 Interval)*

**Function** : continue read data at fix sample rate for multiplex channel at synchronous read operation, more information to see [read operation](#)

**Arguments:**

- cardnumber: board id, must be opened by Open PCI1812\_Register\_Card function first.
- channel : channel select , see [channel select](#)
- adrange : ad range select, see [Input range select](#)
- buufer : address which point to where read data to storage, see [AD data format](#)
- readcount : the deep of storage , also means how many times read from ad fifo, which must be time of 256 ,for more information see [read operation](#)
- interval : PCI1812 sample rate set. More information see [sample interval](#)

Note : readcount must be time of 256 ,and you should make sure read operation will not overflow the buffer which to storage ad data.

<b>Return Code:</b>	
Success	API success
ErrorCardNumber	input card number error
ErrorParameters	input parameter error
ErrorFailure	API fault
ErrorBusy	Another thread is read ad FIFO data

*U32 PCI1812\_DAOut(U32 cardnumber,U32 dachannel,U32 dadata)*

**Function** : PCI\_1812 DA out

**Arguments:**

- cardnumber: board id, must be opened by Open PCI1812\_Register\_Card function first.
- channel : channel select , '0' represent DAC\_OUT 0 channel, '1' DAC\_OUT1 channel
- dadata : DA write code , see [D/A output](#)

<b>Return Code:</b>	
Success	API success
ErrorCardNumber	input card number error
ErrorParameters	input parameter error
ErrorFailure	API fault

*U32 PCI1812\_DI(U32 cardnumber,U32 \*didata)*

**Function:** read pci1812 DB37 6 digital input channels

**Arguments:**

- cardnumber: board id, must be opened by Open PCI1812\_Register\_Card function first.

Didata : point data read from PCI1812  
 Note : each bit represent one channel , lowest bit for channel 0

<b>Return Code:</b>	
Success	API success
ErrorCardNumber	input card number error
ErrorParameters	input parameter error
ErrorFailure	API fault

*U32 PCI1812\_DO(U32 cardnumber,U32 writedata)*

**Function: output pci1812 DB37 6 digital output channels**

**Arguments:**

cardnumber: board id, must be opened by Open PCI1812\_Register\_Card function first.

Writedata : data to output of PCI1812 DB37 6 digital output channels

Note : each bit represent one channel , lowest bit for channel 0

<b>Return Code:</b>	
Success	API success
ErrorCardNumber	input card number error
ErrorParameters	input parameter error
ErrorFailure	API fault

*U32 PCI1812\_ExDI(U32 cardnumber,U32 \*didata)*

**Function: read pci1812 CN4 15 digital input channels**

**Arguments:**

cardnumber: board id, must be opened by Open PCI1812\_Register\_Card function first.

Didata : point data read from PCI1812

Note : each bit represent one channel , lowest bit for channel 0

<b>Return Code:</b>	
Success	API success
ErrorCardNumber	input card number error
ErrorParameters	input parameter error
ErrorFailure	API fault

*U32 PCI1812\_ExDO(U32 cardnumber,U32 writedata)*

**Function: output pci1812 CN4 15 digital output channels**

**Arguments:**

cardnumber: board id, must be opened by Open PCI1812\_Register\_Card function first.

Writedata : data to output of PCI1812 CN4 15 digital output channels

Note : each bit represent one channel , lowest bit for channel 0

<b>Return Code:</b>	
Success	API success
ErrorCardNumber	input card number error
ErrorParameters	input parameter error
ErrorFailure	API fault

U32 PCI1812libVersion(void)

**Function:** read PCI1812 lib version

Return code : lib version

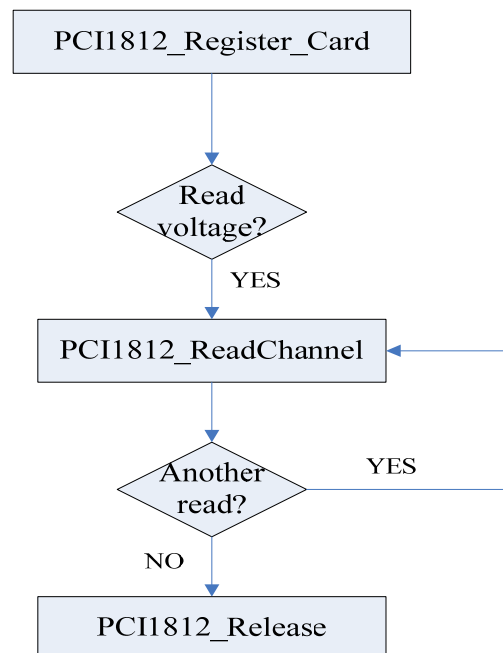
## 10.2 build application

\*when you use PCI1814 device , replace followed function PCI1812 with PCI1814

### 10.2.1 Analog input

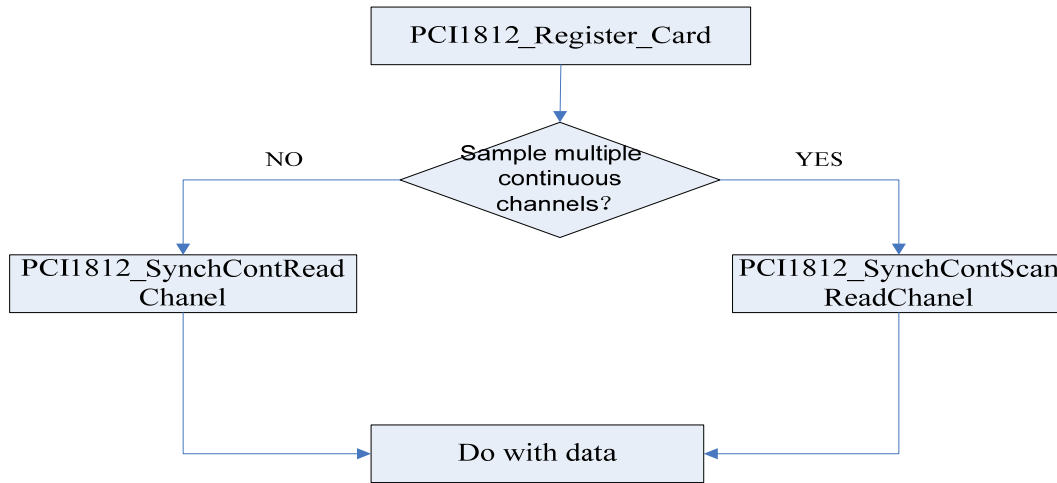
#### One-Shot Analog Input

This section describes the function flow typical of non-buffered single-point analog input readings.



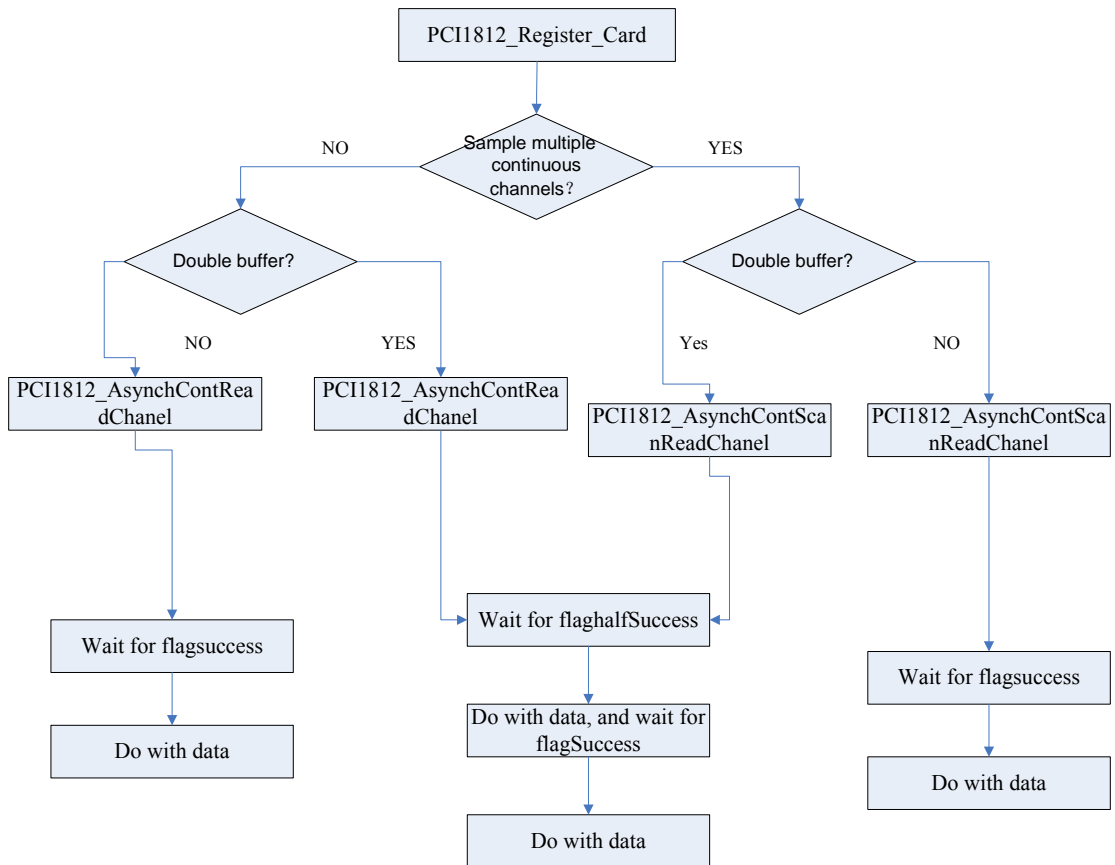
#### Synchronous Continuous Analog Input

This section describes the function flow typical of synchronous continuous analog input operation



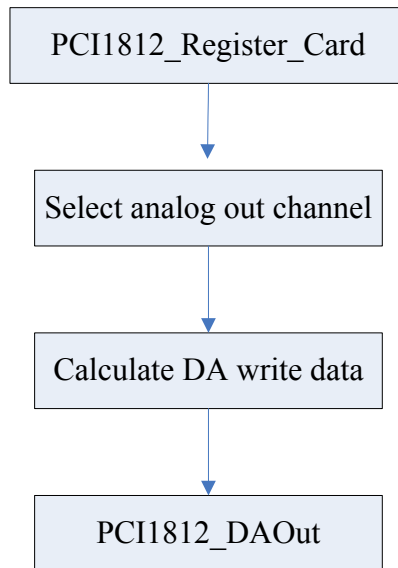
### Asynchronous Continuous Analog Input

This section describes the function flow typical of asynchronous continuous analog input operation.



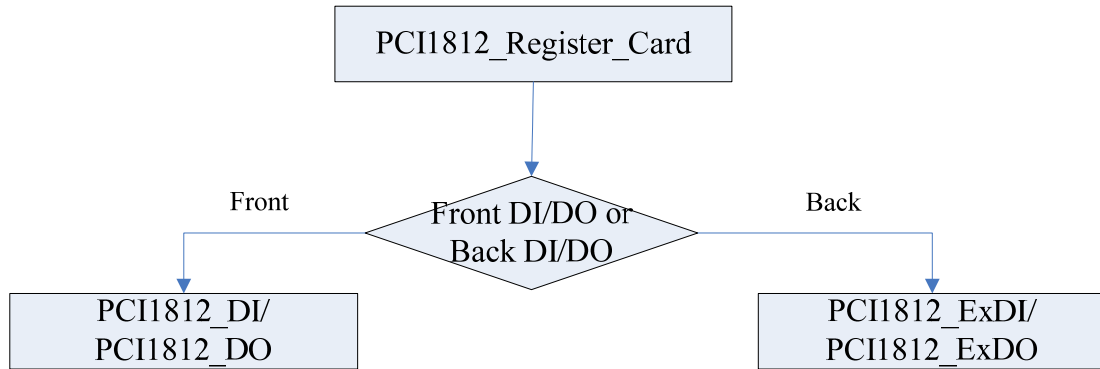
### 10.2.2 Analog output

This section describes the function flow typical of analog output operation



### 10.2.2 DI/DO

This section describes the function flow typical of digital input or output



## **Appendix**

- 1: Change DA output to 0V when system power up
- 2: Add hardware version
- 3: release lib version: 202 , support VB
- 4: manual update to version 101, change DA output channel