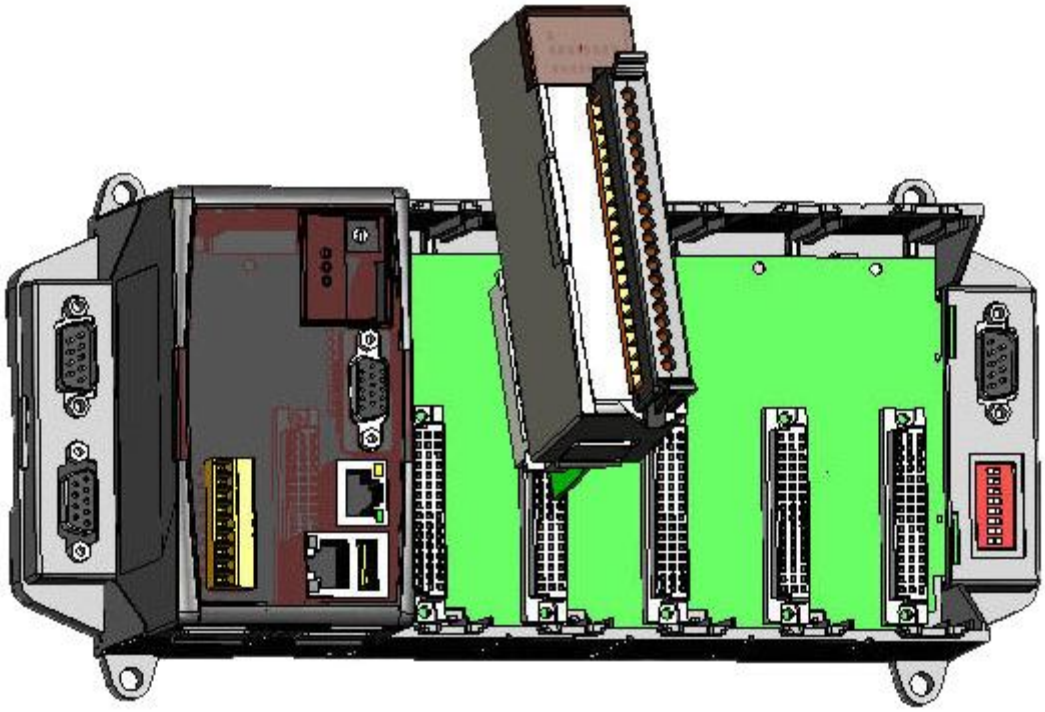
I8017HW
Reference Manual

Version 1.0 beta, October 2008

Service and usage information for WinPAC 8000 Series and iPAC 8000 Series



Written by Kyon Huang

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, no for any infringements of patents or other rights of third parties resulting from its use.

Copyright

Trademarks

The names used in this manual are for identification purpose only and may be registered trademarks of their respective companies.

Table of Contents

1. Introduction to the I8017HW

1.1. Specification

Terminal No.	Pin Assignment Name	
	Differential	Single-ended
01	Trig	Trig
02	AGND	AGND
03	Vin0 +	Vin0
04	Vin0 -	Vin8
05	Vin1 +	Vin1
06	Vin1 -	Vin9
07	Vin2 +	Vin2
08	Vin2 -	Vin10
09	Vin3 +	Vin3
10	Vin3 -	Vin11
11	Vin4 +	Vin4
12	Vin4 -	Vin12
13	Vin5 +	Vin5
14	Vin5 -	Vin13
15	Vin6 +	Vin6
16	Vin6 -	Vin14
17	Vin7 +	Vin7
18	Vin7 -	Vin15
19	AGND	AGND
20	AGND	AGND

1.2. Jumper Settings

This DI jumper sets the discrete input circuits as either “Single-ended” or “Differential” inputs.

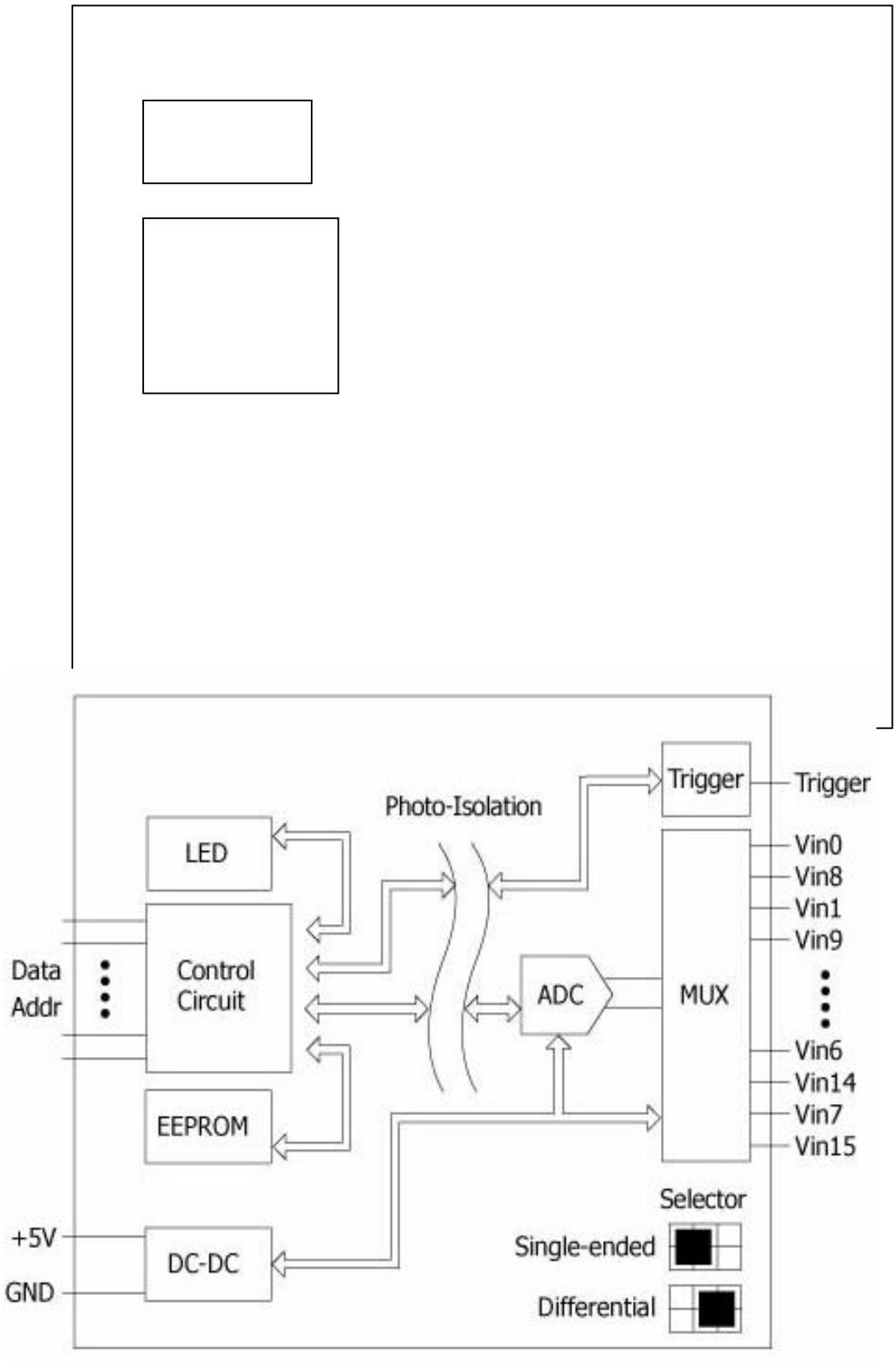
Single-ended Inputs

Differential Inputs

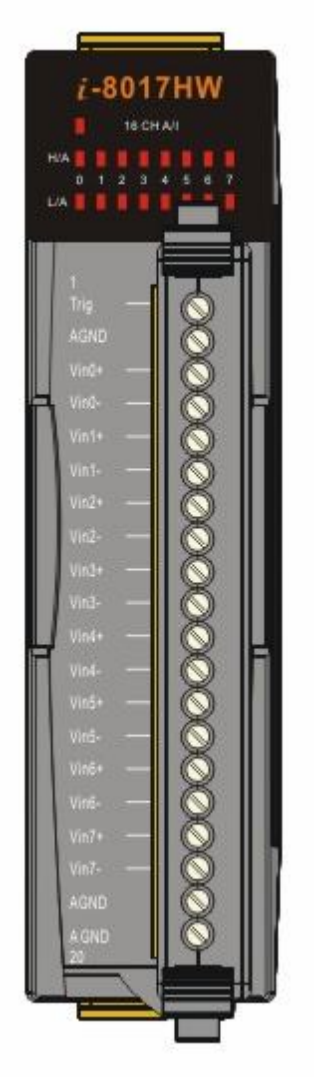
DI Jumper Location

Determine “Single-ended” or “Differential” inputs

1.3. I/O Structure

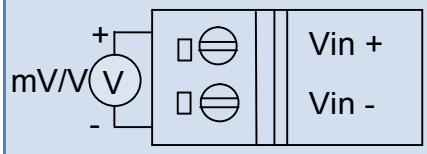
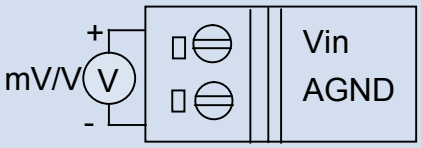
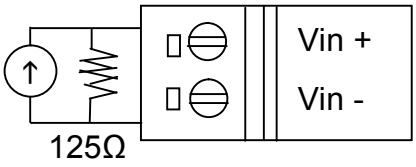
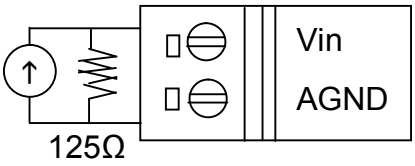


1.4. Pin Assignment



Terminal No.	Pin Assignment Name	
	Differential	**Single-ended**
01	Trig	Trig
02	AGND	AGND
03	Vin0 +	Vin0
04	Vin0 -	Vin8
05	Vin1 +	Vin1
06	Vin1 -	Vin9
07	Vin2 +	Vin2
08	Vin2 -	Vin10
09	Vin3 +	Vin3
10	Vin3 -	Vin11
11	Vin4 +	Vin4
12	Vin4 -	Vin12
13	Vin5 +	Vin5
14	Vin5 -	Vin13
15	Vin6 +	Vin6
16	Vin6 -	Vin14
17	Vin7 +	Vin7
18	Vin7 -	Vin15
19	AGND	AGND
20	AGND	AGND

1.5. Wire Connection

Input Type	Differential	Single-ended
Voltage Input Wiring		
Current Input Wiring		

Note:

When connecting to a current source, an optional external 125-Ohm resistor is required.

2. The API on the WinPAC-8000 for I8017HW

2.1. pac_i8017HW_GetLibVersion

This function is used to initialize the I-8017HW module (Analog input module) which is inserted into the specified slot. Users must execute this function once before trying to use other functions of I-8017HW.

Syntax

short pac_i8017HW_GetLibVersion(void);

Parameter

None

Return Values

Return (short):Library version of i-8017HW.

Examples

[C++]

short version = pac_i8017HW_GetLibVersion();

[C#]

Int16 version = pac8017HW.LibVersion();

2.2. pac_i8017HW_GetLibDate

This function is used to get the built date of I-8017HW library.

Syntax

void pac_i8017HW_GetLibDate (char libDate[]);

Parameter

libDate:

[out] The built date string with null terminal.

Return Values

None

Examples

[C++]

char* builtDate;

pac_ pac_i8017HW_GetLibDate (builtDate);

[C#]

string builtDate;;

builtDate = pac_i8017HW_GetLibDate ();

2.3. pac_i8017HW_GetFirmwareVersion

This function is used to get the lattice version of i-8017HW at specific slot.

Syntax

short pac_i8017HW_GetFirmwareVersion (int slot, short* version);

Parameter

slot:

[in] specified slot of WinPac system (Range: 0 to 7)

version:

[out] The firmware version of i-8017HW

Return Values

Please refer to Appendix A: Error code definition.

Examples

[C++]

```
int slot;
short ret, firmware;
ret = pac_i8017HW_GetFirmwareVersion (slot &firmware);
```

[C#]

```
int slot;
Int16 firmware = 0;
Int16 ret = pac_i8017HW_GetFirmwareVersion (slot, ref firmware);
```

2.4. pac_i8017HW_Init

This function is used to initialize the I-8017HW module (Analog input module) which is inserted into the specified slot. Users must execute this function once before trying to use other functions of I-8017HW.

Syntax

short pac_i8017HW_Init(int slot);

Parameter

slot:

[in] specified slot of WinPac system (Range: 0 to 7)

Return Values

Please refer to Appendix A: Error code definition.

Examples

[C++]

```
int slot;
short ret = pac_i8017HW_Init(slot);
```

[C#]

```
int slot;
pac8017HW.Init(slot);
```

2.5. pac_i8017HW_SetLED

Turns the I-8017HW modules LED's on/off. They can be used to act as an alarm.

Syntax

short pac_i8017HW_SetLED(int slot,unsigned short led);

Parameter

slot:

[in] specified slot of WinPac system (Range: 0 to 7)

led:

[in] range from 0 to 0xffff

Return Values

Please refer to Appendix A: Error code definition.

Examples

[C++]

```
int slot;
unsigned short led;
short ret = pac_i8017HW_Init(int slot);
if (ret == 0)
pac_i8017HW_SetLED( slot, led);
```

[C#]

```
int slot;
UInt16 led;
Int16ret = pac8017HW.Init(slot);
If (ret == 0)
pac8017HW.SetLED( slot, led);
```

2.6. pac_i8017HW_GetSingleEndJumper

This function is used to get the mode of input channels, single-end or differential.

Syntax

short pac_i8017HW_GetSingleEndJumper(int slot);

Parameter

slot:

[in] Specify the slot of WinPac system (Range: 0 to 7)

Return Values

1: Single-Ended Mode.

0: Differential Mode.

Examples

[C++]

```
int slot=1;
short Jumper = pac_i8017HW_GetSingleEndJumper(slot);
```

[C#]

```
int slot;
Int16 Jumper = pac8017HW. SingleEndJumper (slot);
```

2.7. pac_i8017HW_ReadAI

Obtain the analog input value from the analog input module in the float format at specific slot and channel according to given Gain.

Syntax

float pac_i8017HW_ReadAI(int iSlot,int iChannel,int iGain);

Parameter

iSlot: 0 ~ 7

iChannel: 0 ~ 7 if Differential mode; 0 ~ 15 if Single Ended mode

iGain:

0: +/- 10.0V

1: +/- 5.0V

2: +/- 2.5V

3: +/- 1.25V

4: +/- 20mA

Return Values

Return (float): The analog input value.

Examples

[C++]

int slot=1, ch=0, gain=1;

```
float data;
pac_i8017HW_Init(slot);
data = pac_i8017HW_ReadAI (slot,ch,gain);
```

[C#]

```
int slot=1, ch=0, gain=1;
float data;
pac8017HW.Init (slot);
data= pac8017HW. ReadAI (slot, ch, gain);
```

2.8. pac_i8017HW_ReadAIHex

Obtain the analog input value from the analog input module in the 16 bit hex format at specific slot and channel according to given Gain.

Syntax

short pac_i8017HW_ReadAIHex(int iSlot,int iChannel,int iGain);

Parameter

iSlot: 0 ~ 7

iChannel: 0 ~ 7 if Differential mode; 0 ~ 15 if Single Ended mode

iGain:

0: +/- 10.0V

1: +/- 5.0V

2: +/- 2.5V

3: +/- 1.25V

4: +/- 20mA

Return Values

Return (short): The analog input value.

Examples

[C++]

int slot=1, ch=0, gain=1;

```
short data;
pac_i8017HW_Init(slot);
data = pac_i8017HW_ReadAIHex(slot,ch,gain);
```

[C#]

```
int slot=1, ch=0, gain=1;
float data;
pac8017HW.Init (slot);
data= pac8017HW. ReadAIHex(slot, ch, gain);
```

Note: i-8017HW uses 14 bit AD chip, it is more convenient for pac_i8017HW_ReadHex return 16 bit data when user need to scale the hex data.

so it will have least two bit invalid data.

user can use following statement to convert 16 bit data to 14 bit

```
short Convert16To14( short bit16Data)
{
    short bit14Data= bit16Data >> 2;
    bit14Data &= 0x3fff;
    return bit14Data;
}
```

2.9. pac_i8017HW_SetChannelGainMode

This function is used to configure the range and mode of the analog input channel for the module I-8017HW in the specified slot before using ADC (analog to digital converter).

It has to call i8017HW_SetChannelGainMode before getting analog input if channel index or gain is different previous.

Syntax

void pac_i8017HW_SetChannelGainMode(int slot,int ch,int gain,int mode);

Parameter

slot:

[in] Specify the slot of i-8000 system (Range: 0 to 7)

ch:

[in] Specify the I-8017HW channel (Range: 0 to 7)

gain:

[in] input range:

0: +/- 10.0V

1: +/- 5.0V

2: +/- 2.5V

3: +/- 1.25V

4: +/- 20mA

mode:

[in] 0: normal mode (polling) ; 1: interrupt mode

Return Values

None

Examples

[C/C++]

```
int slot=1, ch=0, gain=0;

pac_i8017HW_SetChannelGainMode( slot, ch, gain, 0);
```

2.10. pac_i8017HW_GetADFValue

To get the analog input value from the analog input module in the float format.

It has to call i8017HW_SetChannelGainMode before using i8017HW_GetADFValue to

get analog input if channel index or gain is different from previous.

Syntax

float pac_i8017HW_GetADFValue(int gain);

Parameter

gain:

[in] input range

0: +/- 10.0V

1: +/- 5.0V

2: +/- 2.5V

3: +/- 1.25V

4: +/- 20mA

Return Values

Return (float): The analog input value.

Examples

[C++]

```
int slot=1, ch=0, gain=1;
float data;
i8017HW_Init(slot);
pac_i8017HW_SetChannelGainMode( slot, ch, gain, 0);
data = pac_i8017HW_GetADFValue(gain);
```

2.11. pac_i8017HW_GetADHValue

To get the analog input value from the analog input module in the float format.

It has to call i8017HW_SetChannelGainMode before using i8017HW_GetADHValue to

get analog input if channel index or gain is different from previous.

Syntax

short pac_i8017HW_GetADHValue(void);

Parameter

None

Return Values

The voltage analog input value.

Examples

[C++]

```
int slot=1, ch=0, gain=1;
short data;
pac_i8017HW_Init(slot);
pac_i8017HW_SetChannelGainMode( slot, ch, gain, 0);
data = pac_ i8017HW_GetADHValue(gain);
```

2.12. pac_i8017HW_ADPolling

This function is used to get the analog input values of the specific channel from an analog input module and convert the value in HEX format according to the configuration of the slot, the gain and the data number.

Note:

Those data get from i8017HW_ADPolling are 16-bit values and not calibrated. It can use i8017HW_HexArrayToFloat to calibrate the data and convert them to float values.

Syntax

int pac_i8017HW_ADPolling(int slot,int ch,int gain,int datacount,int *DataPtr);

Parameter

slot:

[in] Specified slot in I-8000 system (Range: 1 to 7)

ch:

[in] Specified channel for I-8017HW (Range: 0 to 7)

gain:

[in] Input range:

0: +/- 10.0V

1: +/- 5.0V

2: +/- 2.5V

3: +/- 1.25V

4: +/- 20mA

datacount:

[in] Range from 1 to 8192, total ADCs number

DataPtr:

[out] The starting address of data array[ ] and the array size must be equal to or bigger than the datacount.

Return Values

0: indicates success.

1: indicates failure.

Examples

[C++]

```
int slot=1, ch=0, gain=0, count=10, idata[10];
pac_i8017HW_Init(slot);
pac_i8017HW_ADPolling(slot, ch, gain, datacount, idata);
for(int i=0;i<10;i++)
Print("Data[%d]= %04X\n", i, idata[i]); // Display polling results
// Those data in this example are not calibrated, it can use i8017HW_HexArrayToFloat
// to calibrate and convert them to float format
float fdata[10];
pac_i8017HW_HexArrayToFloat (idata,fdata,slot,gain,datacount);
for(i=0;i<iDataCount;i++)
{
   Print (" %7.3f \n",fdata[i]);
}
```

2.13. pac_i8017HW_HexArrayToFloat

This function is used to convert the I-8017HW AD chip data (not calibrated) from hex to float values based on the configuration of the slot, gain and data length.

Syntax

void pac_i8017HW_HexArrayToFloat(int *HexValue,float *Ivalue,int slot,int gain,int len);

Parameter

HexValue:

[in] data array in integer type before converting.

Ivalue:

[out] Converted data array in float type (voltage or current).

slot:

[in] Specify the slot of I-8000 system (Range: 0 to 7)

gain:

[in] input range

len:

[in] ADC data length.

Return Values

None

Examples

[C++]

```
int slot=1, ch=0, gain=0, datacount=10, idata[10];
float fdata[10];
pac_i8017HW_Init(slot);
pac_i8017HW_ADPolling(slot, ch, gain, datacount, idata);
pac_i8017HW_HexArrayToFloat(idata, fdata, slot, gain, datacount);
// You gain ten record HEX values to change ten records float values
```

2.14. pac_i8017HW_AD_TimerINT

Uses timer interrupt to sample A/D values for specific channel of I-8017HW.

Note:

Those data get from i8017HW_AD_TimerINT are 16-bit values and not calibrated. It can

use i8017HW_HexArrayToFloat to calibrate the data and convert them to float values.

Syntax

int pac_i8017HW_AD_TimerINT(int slot,int ch,int gain,int datacount,unsigned int

sampleCLK,int *iDataPtr);

Parameter

slot:

[in] Specified slot of I-8000 system (Range: 1 to 7)

ch:

[in] Specified channel for I-8017HW (Range: 0 to 7)

gain:

[in] Input range:

0: +/- 10.0V

1: +/- 5.0V

2: +/- 2.5V

3: +/- 1.25V

4: +/- 20mA

datacount:

[in] Range from 1 to 8192, total ADCs number

sampleCLK:

[in] Sampling rate 200 ~ 50K (unit: Hz)

DataPtr:

[out] The starting address of data array[ ] and the array size must be equal to or bigger than the datacount.

Return Values

0: successful.

1: failed.

Examples

[C++]

```
int slot=1, ch=0, gain=0, count=10, idata[10];
unsigned int samplerate;
pac_i8017HW_Init(slot);
pac_i8017HW_AD_TimerINT(slot, ch, gain, datacount, samplerate, idata);
for(int i=0;i<10;i++)
Print("Data[%d]= %04X\n", i, idata[i]); // Display interrupt sampling results
//Those data in this example are not calibrated, it can use i8017HW_HexArrayToFloat
// to calibrate and convert them to float format
float fdata[10];
pac_i8017HW_HexArrayToFloat (idata,fdata,slot,gain,datacount);
for(i=0;i<iDataCount;i++)
```

```
{
   Print (" %7.3f \n",fdata[i]);
}
```

2.15. pac_i8017HW_AD_TimerINT_Scan

Uses timer interrupt to sample A/D values for specific slot of I-8017HW.

Note:

Those data get from i8017HW_AD_TimerINT_Scan are 16-bit values and not alibrated. It can use i8017HW_HexArrayToFloat to calibrate the data and convert them to float values.

Syntax

int pac_i8017HW_AD_TimerINT_Scan(int slot,,int gain,int datacount,unsigned int sampleCLK,int *iDataPtr);

Parameter

slot:

[in] Specified slot of I-8000 system (Range: 1 to 7)

gain:

[in] Input range:

0: +/- 10.0V

1: +/- 5.0V

2: +/- 2.5V

3: +/- 1.25V

4: +/- 20mA

datacount:

[in] Range from 1 to 8192, total ADCs number

sampleCLK:

[in] Sampling rate 200 ~ 50K (unit: Hz)

DataPtr:

[out] The starting address of data array[ ] and the array size must be equal to or bigger than the datacount.

data in the array[] buffer is ranged as

Differential Input Mode:

ch0 ch1 ch2...ch7 ch0 ch1....ch7 ch0 ch1 ...

Single ended Input Mode:

ch0 ch1 ch2...ch7 ch0 ch1....ch15 ch0 ch1 ...

Return Values

0: successful.

1: failed.

Examples

[C++]

```
int slot=1, gain=0, datacount =32, idata[32];
unsigned int samplerate;
pac_i8017HW_Init(slot);
pac_i8017HW_AD_TimerINT_Scan(slot, gain, datacount, samplerate, idata);
for(int i=0;i< datacount;i++)
Print("Data[%d]= %04X\n", i, idata[i]); // Display interrupt sampling results
// Those data in this example are not calibrated, it can use i8017HW_HexArrayToFloat
// to calibrate and convert them to float format.
```

```
float fdata[32];
pac_i8017HW_HexArrayToFloat (idata,fdata,slot,gain,datacount);
for(i=0;i< datacount;i++)
{
   Print (" %7.3f \n",fdata[i]);
}
```

3. The API on the iPAC-8000 for I8017HW h

3.1. i8017H_GetLibVersion

function to get the library version of i-8017h

return: version number.
for example: 0x106; = Rev:1.0.6

short i8017H_GetLibVersion(void);

3.2. i8017H_GetLibDate

function to get the library built date

return: none

void i8017H_GetLibDate(char libDate[]); //unsigned char -> char

3.3. i8017H_GetFirmwareVersion

function to get the lattice version of i-8017h at specific slot

iSlot: 0 ~ 7
return: Error code .

short i8017H_GetFirmwareVersion(int iSlot,short* firmware);

3.4. i8017H_ReadGainOffset_Info

function to get the calibrated gain and offset value for i-8017h at specific slot for certain input range

iSlot: 0 ~ 7
iGain:
0: +/- 10.0V
1: +/- 5.0V
2: +/- 2.5V
3: +/- 1.25V
4: +/- 20mA
iGainValue: the calibated gain value
iOffsetValue: the calibated offset value

return: Error code

short i8017H_ReadGainOffset_Info(int slot,int Gain,unsigned short* GainValue, short* offsetValue);

3.5. i8017H_GetSingleEndJumper

function to get the single ended/differential jumper status for i-8017h at specific slot

selectJumper: 1 Single Ended; 0 Differential.
iSlot: 0 ~ 7
return: Error code

short i8017H_GetSingleEndJumper(int iSlot,short* selectJumper);

3.6. i8017H_Init

function to initialize i-8017h at specific slot

return: None.
iSlot: 0 ~ 7

void i8017H_Init(int iSlot);

3.7. i8017H_SetLED

function to have programmable LED control for i-8017h at specific slot

return: None.
iSlot: 0 ~ 7
iLedValue: 0 ~ 0xffff

void i8017H_SetLED(int iSlot,unsigned short iLedValue);

3.8. i8017H_ReadAI

function to read float format analog input data from i-8017h at specific slot

iSlot: 0 ~ 7
iChannel: 0 ~ 7 if Differential mode; 0 ~ 15 if Single Ended mode
iGain:
0: +/- 10.0V
1: +/- 5.0V
2: +/- 2.5V
3: +/- 1.25V
4: +/- 20mA
fValue: float format analog input data.

return: Error code

short i8017H_ReadAI(int iSlot,int iChannel,int iGain,float* fValue);

3.9. i8017H_ReadAIHex

function to read 16 bit hex format analog input data from i-8017h at specific slot

iSlot: 0 ~ 7
iChannel: 0 ~ 7 if Differential mode; 0 ~ 15 if Single Ended mode
iGain:
0: +/- 10.0V
1: +/- 5.0V
2: +/- 2.5V
3: +/- 1.25V
4: +/- 20mA
iValue: 16 bit hex format analog input data.

return: Error code

```
short i8017H_ReadAIHex(int iSlot,int iChannel,int iGain,short* iValue);
```

3.10. i8017H_Set_ChannelGainMode

Sets 8017 MUX configuration.

iSlot: 0~3 or 0~7
iChannel: 0~7
iGain : 0 ==> +/- 10.0 V
1 ==> +/- 5.0 V
2 ==> +/- 2.5 V
3 ==> +/- 1.25 V
4 ==> +/- 20.0 mA
iMode: 0 ==> polling mode
(for i8017_Get_AD_Value and i8017_Get_AD_Value_Hex)
1 ==> timer interrupt mode

[English comment]
If the channel, range or mode you want to sample is different
to previous sampling, you must call the function first to change
the MUX configuration.
Note: You need to consider to call the function only when using
1. i8017H_Get_AD_FValue
2. i8017H_Get_AD_FValue
Because other functions that to sample multi-data already
include function to set MUX.

```
void i8017H_Set_ChannelGainMode(int iSlot,int iChannel,int iGain,int iMode);
```

3.11. i8017H_Get_AD_FValue

Get A/D value (both voltage and current) in floating format
iGain : 0 ==> +/- 10.0 V
1 ==> +/- 5.0 V
2 ==> +/- 2.5 V
3 ==> +/- 1.25 V
4 ==> +/- 20.0 mA

```
float i8017H_Get_AD_FValue(int iGain);
```

3.12. i8017H_Get_AD_HValue

Get A/D value in (both voltage and current) hexdecimal format
8017H is 14-bit resolution. It uses 14-bit 2's format to store values.
When A/D value between 0000 ~ 1FFF ==> 0 ~ + max value
between 2000 ~ 3FFF ==> - max value ~ 0 (a little less than 0)
Algorithm to convert 14-bit integer (iValue) to float (fValue)
fValue=iValue/8192*span ,fValue>=0, iValue between 0000~1FFF
fValue=(8192-iValue)/8192*span ,fValue<0, iValue between 2000~3FFF

int i8017H_Get_AD_HValue(void);

3.13. i8017H_AD_Polling

Uses polling mode to sample A/D values of one chnnel.
iSlot: 0~3 or 0~7
iChannel: 0~7
iGain: 0 ==> +/- 10.0 V
1 ==> +/- 5.0 V
2 ==> +/- 2.5 V
3 ==> +/- 1.25 V
4 ==> +/- 20.0 mA
iDataCount: 1~8192
iDataPointer: pointer to the data buffer

Note: The data is an uncalibrated 16-bit value.
You can use ARRAY_HEX_TO_FLOAT or HEX_TO_FLOAT
to calibrate the data and convert to float value.

int i8017H_AD_Polling(int iSlot,int iChannel,int iGain,int iDataCount,int *iDataPointer);

3.14. i8017H_Array_HexToFloat

void i8017H_Array_HexToFloat(int *HexValue,float *FloatValue,int iSlot,int iGain,int iDataCount);

3.15. i8017H_HexToFloat

float i8017H_HexToFloat(int HexValue,int iSlot,int iGain);

3.16. i8017H_AD_TimerINT

Uses timer interrupt to sample A/D values of one chnnel.
iSlot: 0~3 or 0~7
iChannel: 0~7
iGain: 0 ==> +/- 10.0 V
1 ==> +/- 5.0 V
2 ==> +/- 2.5 V
3 ==> +/- 1.25 V
4 ==> +/- 20.0 mA
iDataCount: 1~8192
iSampleCLK: 200 ~ 50K (unit: Hz)
iDataPointer: pointer to the data buffer

Note: The data is an uncalibrated 16-bit value.
You can use ARRAY_HEX_TO_FLOAT or HEX_TO_FLOAT
to calibrate the data and convert to float value.

int i8017H_AD_TimerINT(int iSlot,int iChannel,int iGain,int iDataCount,unsigned int iSampleCLK,int *iDataPointer);

3.17. i8017H_AD_TimerINT_Scan

Uses timer interrupt to sample A/D values of 8 chnnels.

iSlot: 0~3 or 0~7

iChannel: 0~7

iGain: 0 ==> +/- 10.0 V
1 ==> +/- 5.0 V
2 ==> +/- 2.5 V
3 ==> +/- 1.25 V
4 ==> +/- 20.0 mA

iDataCount: 1~8192

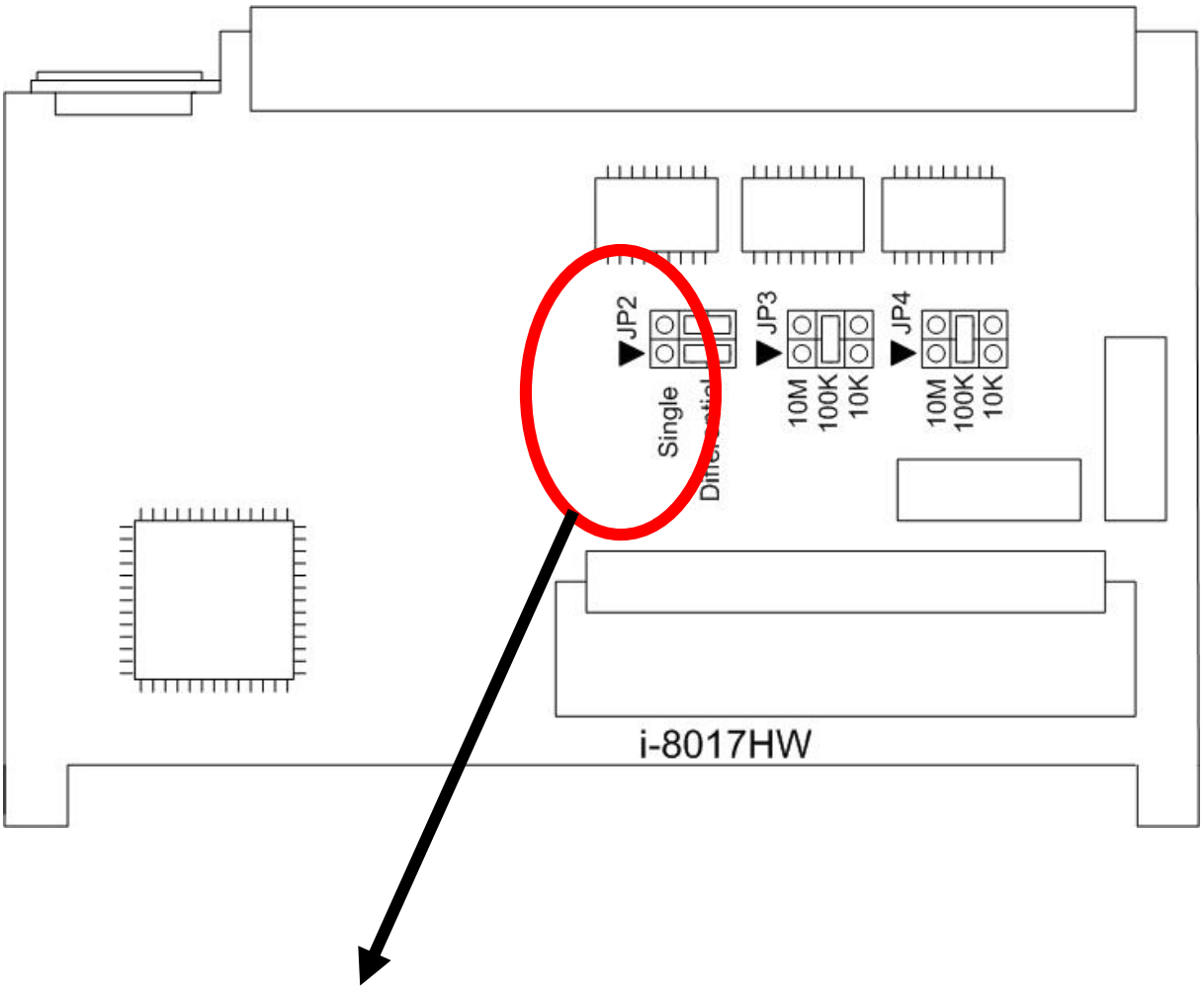
iSampleCLK: 200 ~ 50K (unit: Hz)

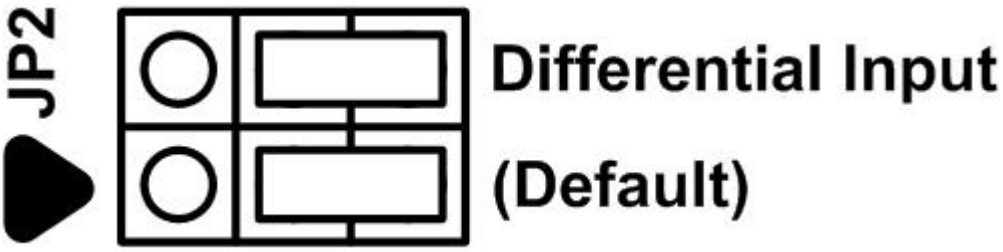
iDataPointer: pointer to the data buffer
data in the buffer is ranged as
Diff Input Mode:
ch0 ch1 ch2...ch7 ch0 ch1....ch7 ch0 ch1 ...
Single end Input Mode:
ch0 ch1 ch2...ch7 ch0 ch1....ch15 ch0 ch1 ...

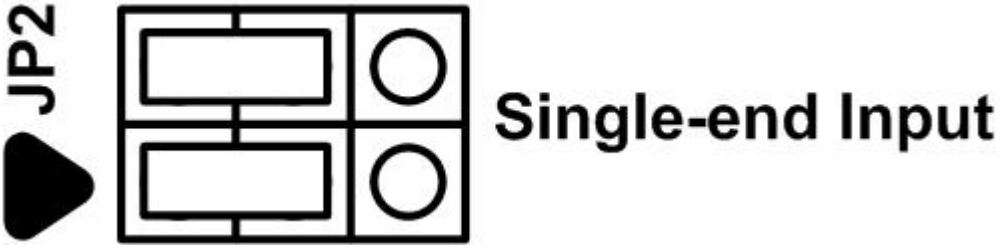
Note: The data is an uncalibrated 16-bit value.
You can use ARRAY_HEX_TO_FLOAT or HEX_TO_FLOAT
to calibrate the data and convert to float value.

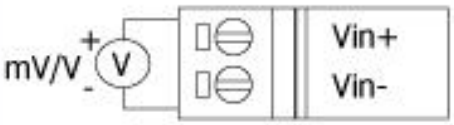
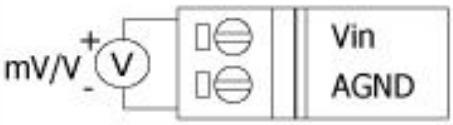
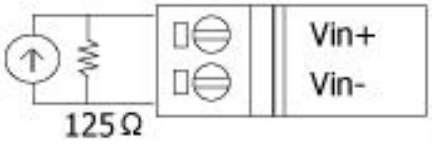
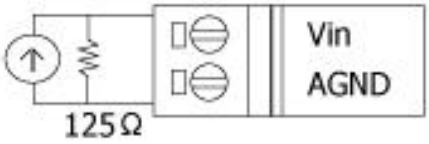
int i8017H_AD_TimerINT_Scan(int iSlot,int iGain,int iDataCount,unsigned int iSampleCLK,int *iDataPointer);

4. Using the I8017HW

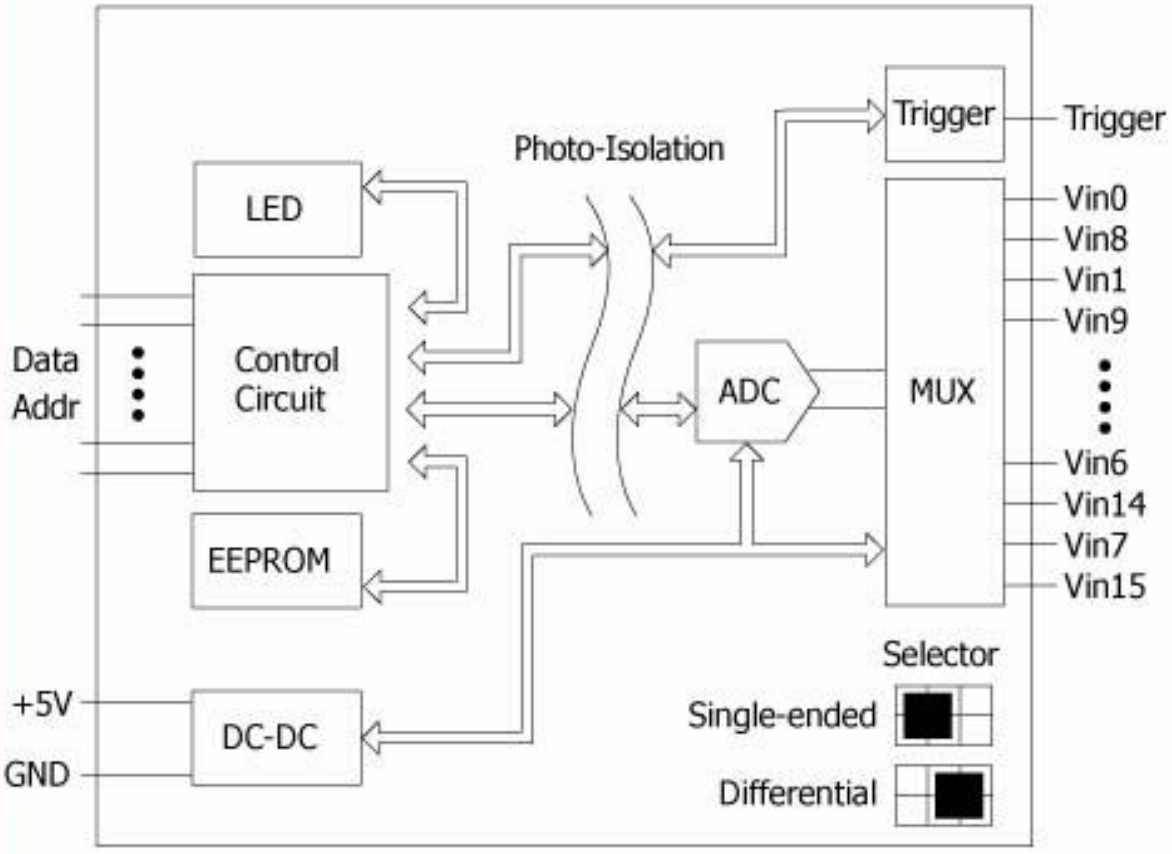

JP2
JP3
JP4
Single
10M
100K
10K
10M
100K
10K
i-8017HW


JP2
Differential Input
(Default)


JP2
Single-end Input

Input Type	Differential	Single-ended
Voltage Input Wiring		
Current Input Wiring		
Note: When connecting to a current source, an optional external 125-Ohm resistor is required.		

I-8017HW Wire Connection



I-8017HW Internal I/O Structure