I-8084W

Linux API Reference Manual

Version 2.0.0 , June 2014

Service and usage information

for

LinPAC-8000 Series

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 I-8084W

I-8084W is a 4/8-channel Counter/Frequency Module.

1.1. Specification

Digital Input	
Mode	4-ch Up/Down Counter (Up/Down)
	4-ch Dir/Pulse Counter (Bi-direction)
	4-ch Quadrant Counting
	8-ch Up Counter
	8-ch Frequency
	Programmable Built-in gate time: 0.33 sec (Default)
	Programmable Digital Noise Filter: 1 ~ 32737 µs
Isolated Input Level	Logic Level 0: +1 V max
	Logic Level 1: +4.5 ~ 30 V
TTL Input Level	Logic Level 0: 0 ~ 0.8 V
	Logic Level 1: 2 ~ 5 V
Input Frequency	0 ~ 450 kHz (Frequency Mode)
	450 kHz (Counter Mode)
Minimum Pulse Width	1 µs (Frequency Mode)
	1 µs (Counter Mode)
EEPROM	128 KB
Isolated Voltage	1000 Vrms
ESD Protection	2 kV (Contact for each channel)

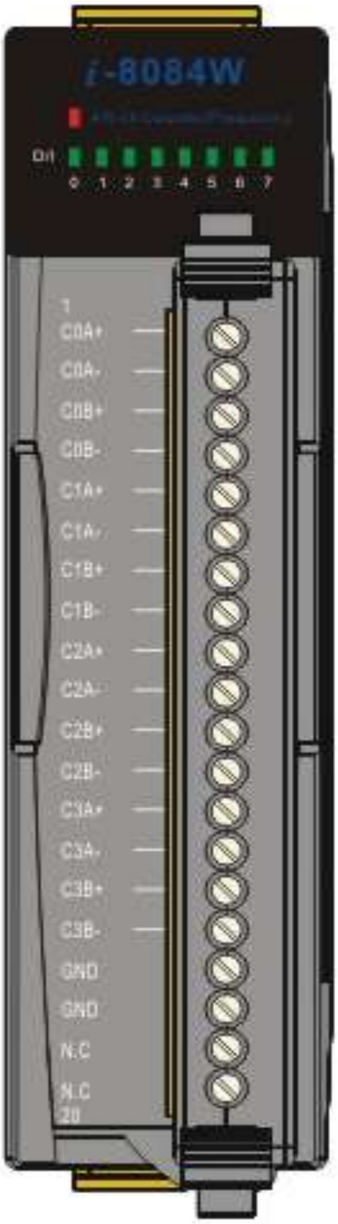
LED Display
1 LED as Power Indicator
8 LEDs as Digital Input Indicators

Power	
Power Consumption	1 W

Environment	
Operating Temperature	-25 ~ 75 ℃
Storage Temperature	-30 ~ 85 ℃
Humidity	5 ~ 95 % RH, Non-condensing

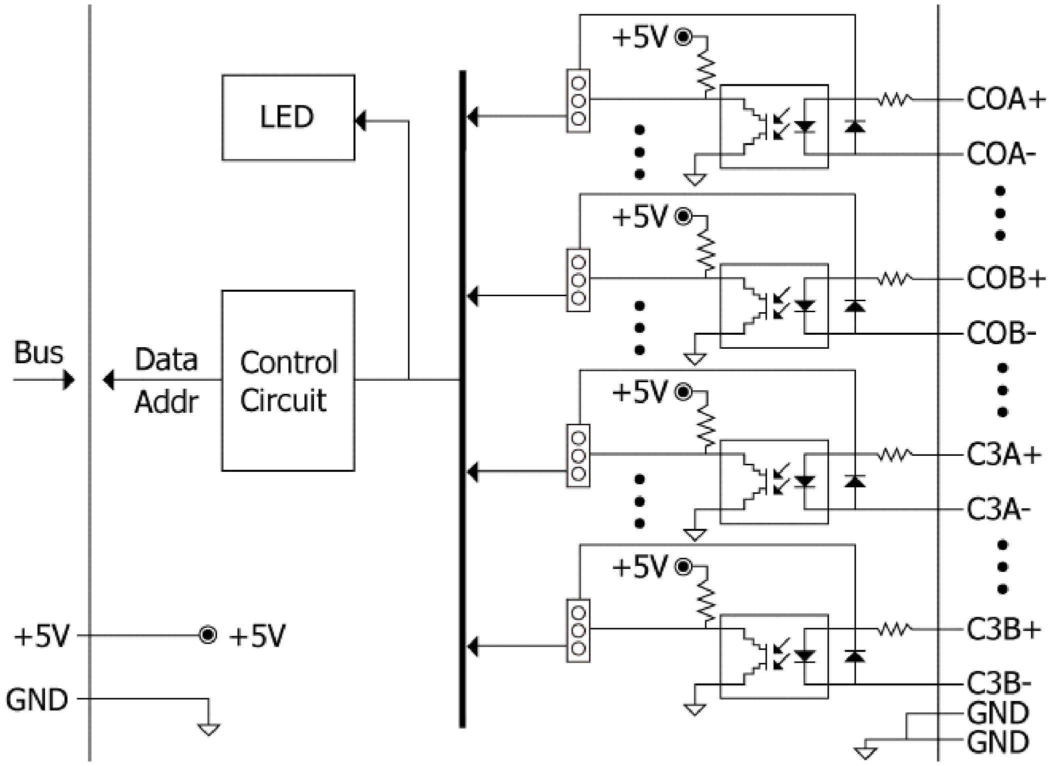
Dimensions
30 mm x 85 mm x 114 mm (W x L x H)

1.2. Pin Assignment

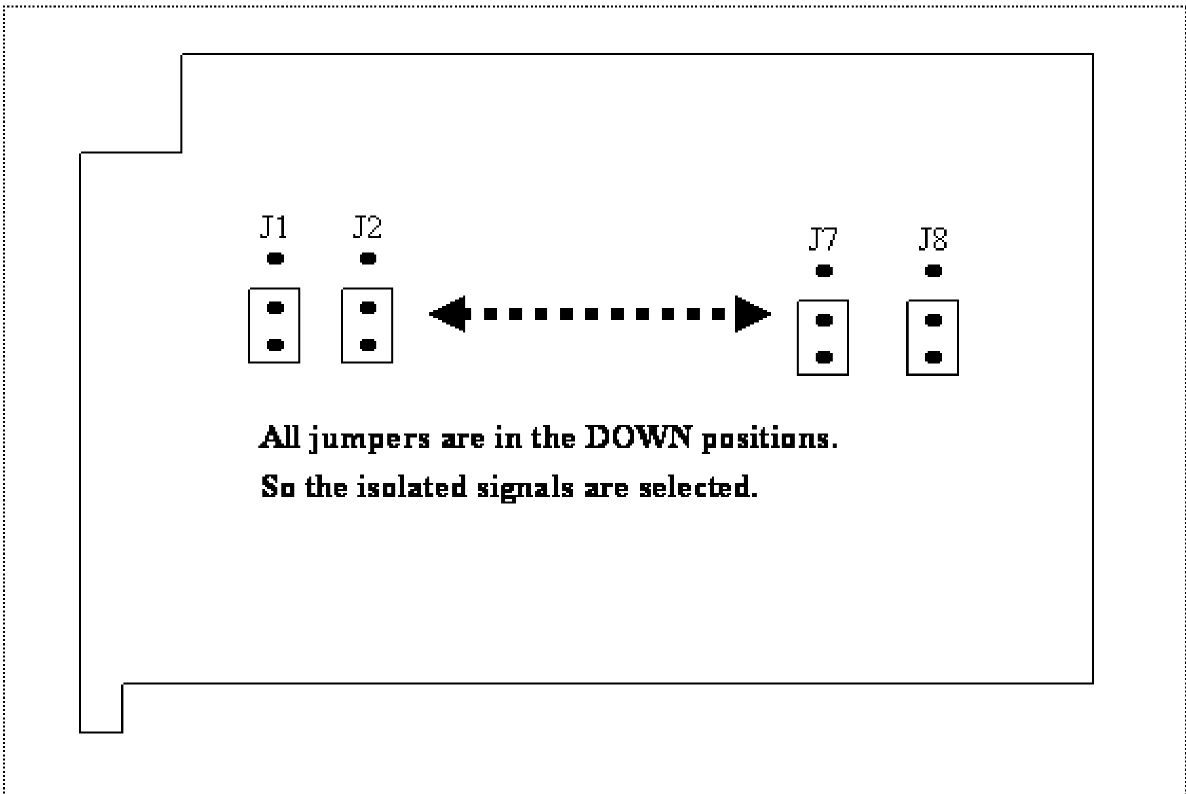


Terminal No.	Pin Assignment
01	C0A+
02	C0A-
03	C0B+
04	C0B-
05	C1A+
06	C1A-
07	C1B+
08	C1B-
09	C2A+
10	C2A-
11	C2B+
12	C2B-
13	C3A+
14	C3A-
15	C3B+
16	C3B-
17	GND
18	GND
19	N.C
20	N.C

1.3. I/O Structure

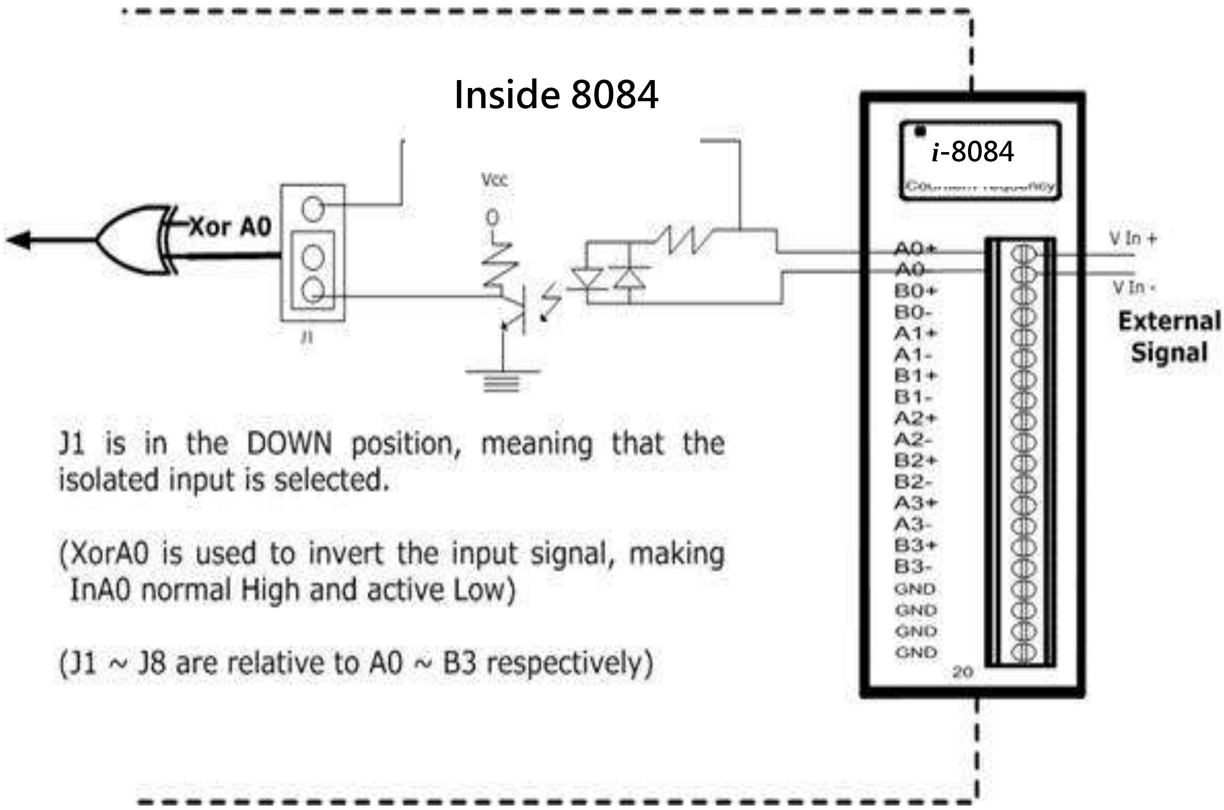


The default jumper settings are as follows:



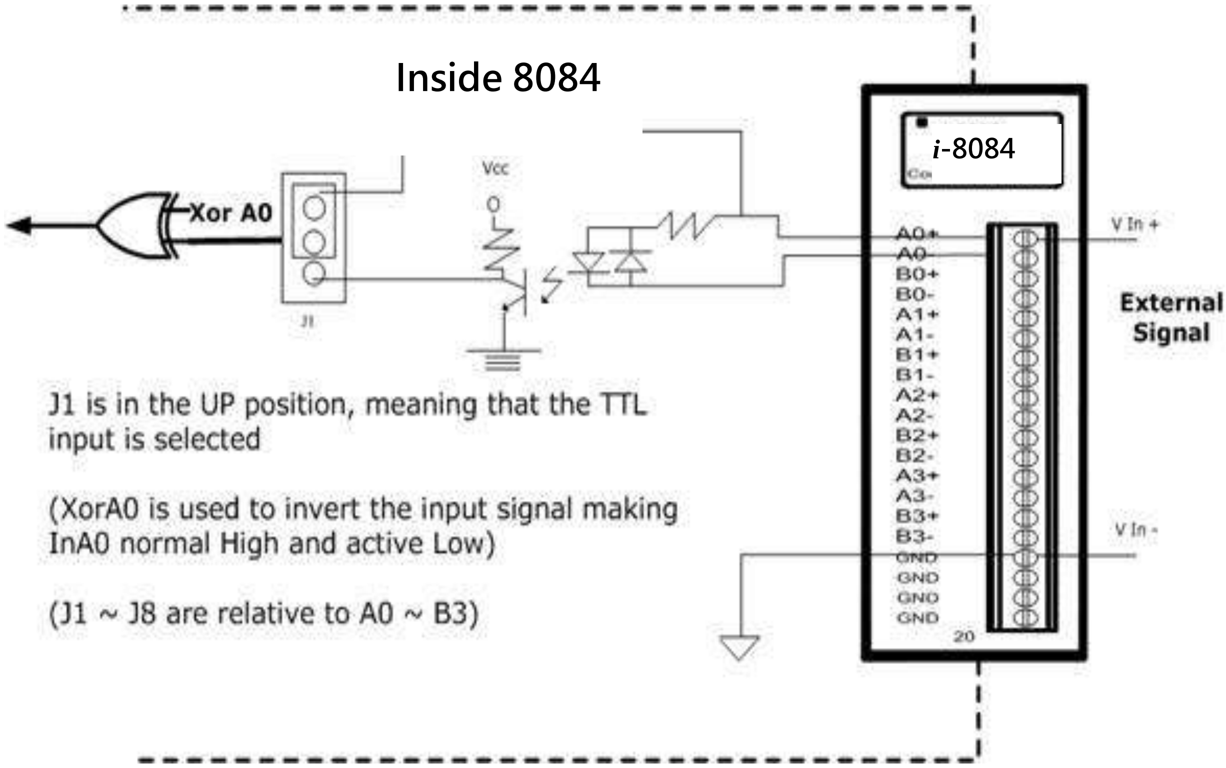
Isolated

Input:



TTL

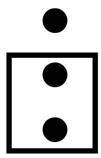
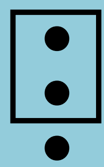
Input:



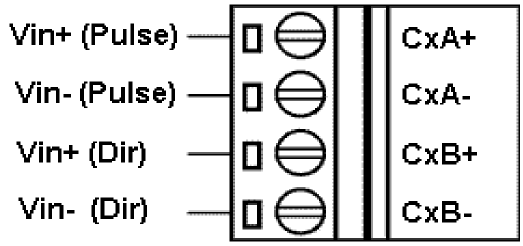
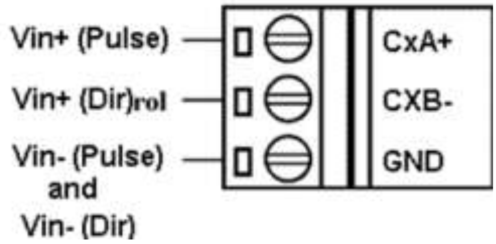
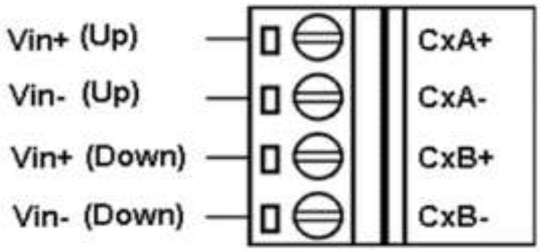
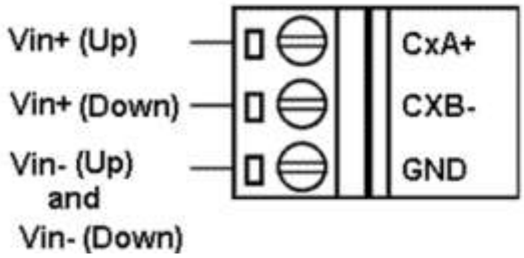
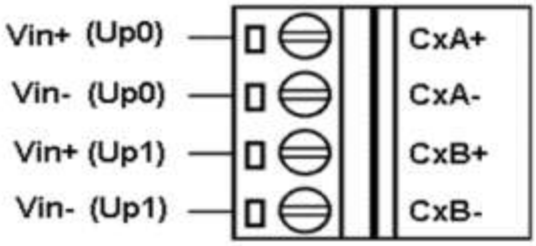
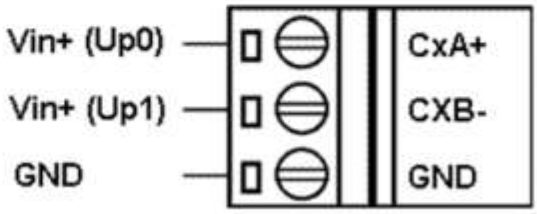
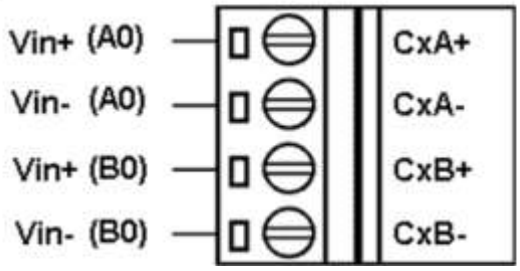
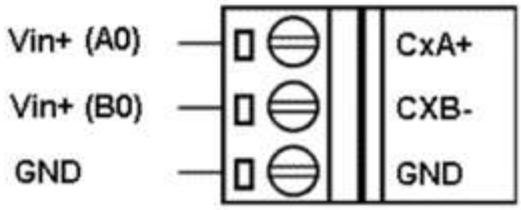
Inside 8084
i-8084
Xor A0
Vcc
J1
V In +
V In -
External Signal
A0+
A0-
B0+
B0-
A1+
A1-
B1+
B1-
A2+
A2-
B2+
B2-
A3+
A3-
B3+
B3-
GND
GND
GND
GND
20
J1 is in the UP position, meaning that the TTL input is selected
(XorA0 is used to invert the input signal making InA0 normal High and active Low)
(J1 ~ J8 are relative to A0 ~ B3)

Isolated or TTL input is selected by using JP1 to JP3 as indicated below:

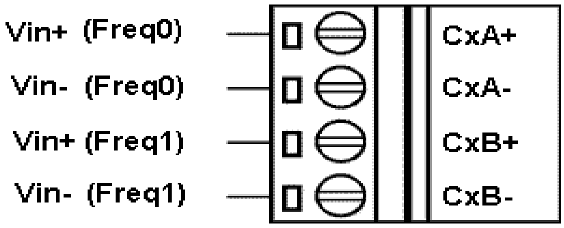
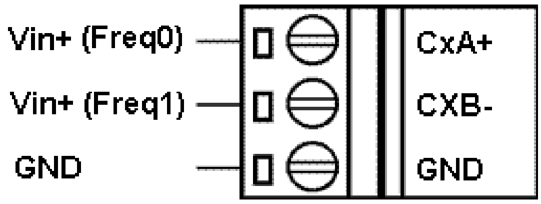
J1	**Select A0**
J2	Select B0
J3	Select A1
J4	Select B1
J5	Select A2
J6	Select B2
J7	Select A3
J8	Select B3

J1/2/3/4/5/6/7/8	**J1/2/3/4/5/6/7/8**
	
Isolated input (Default Setting)	TTL input

1.4. Wiring Connection

Counter Type		
Mode	Isolation	No-n-Isolation
Dir/Pulse		
Up/Down		
Up		
Quadrant		

Frequency Type		
Mode	Isolation	No-n-Isolation

Frequency	Vin+ (Freq0) Vin- (Freq0) Vin+ (Freq1) Vin- (Freq1) CxA+ CxA- CxB+ CxB-	Vin+ (Freq0) Vin+ (Freq1) GND CxA+ CXB- GND

2. Hardware Operation Principle

2.1. Input Signal Model

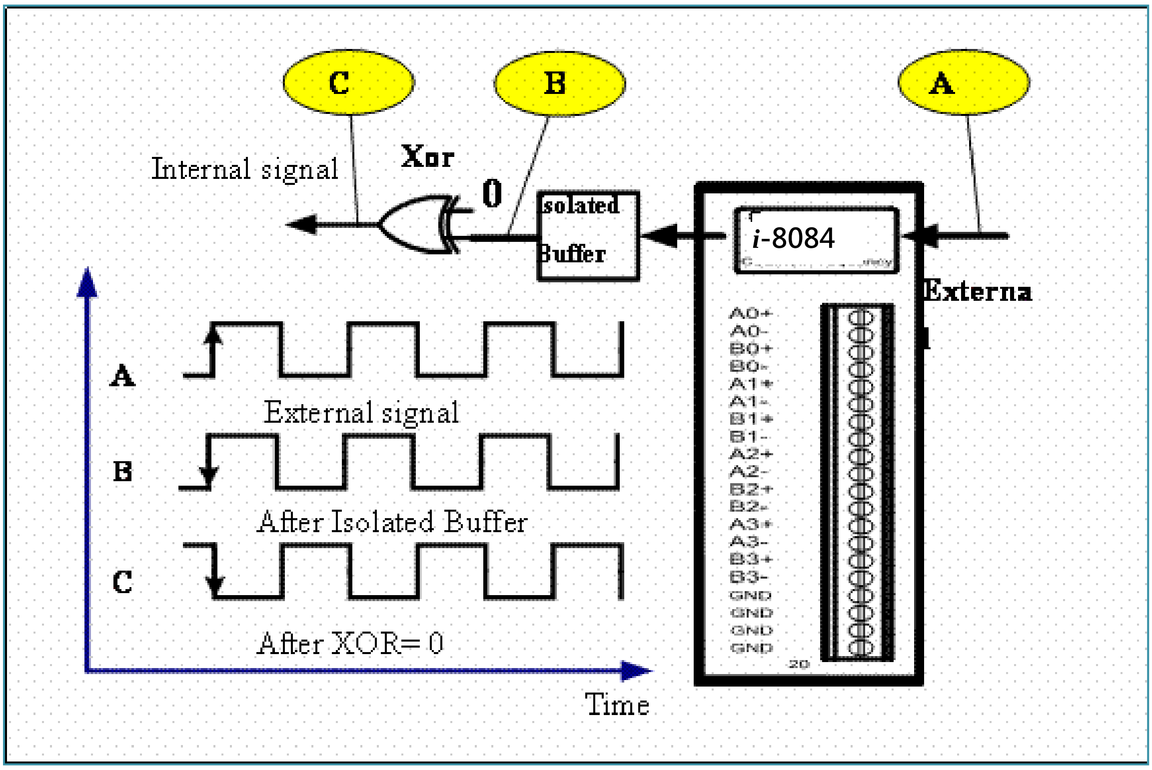
1. Isolated Input (XOR=0)

The operational logic applied on the 8084 modules is the falling edge trigger. (Normal High and Active Low)

The external signal is input into an 8084 module through the isolated mechanism, with the signal being reversed from the external signal.

This internal signal is the suggested waveform, as it doesn't need to execute the XOR operation (XOR=0).

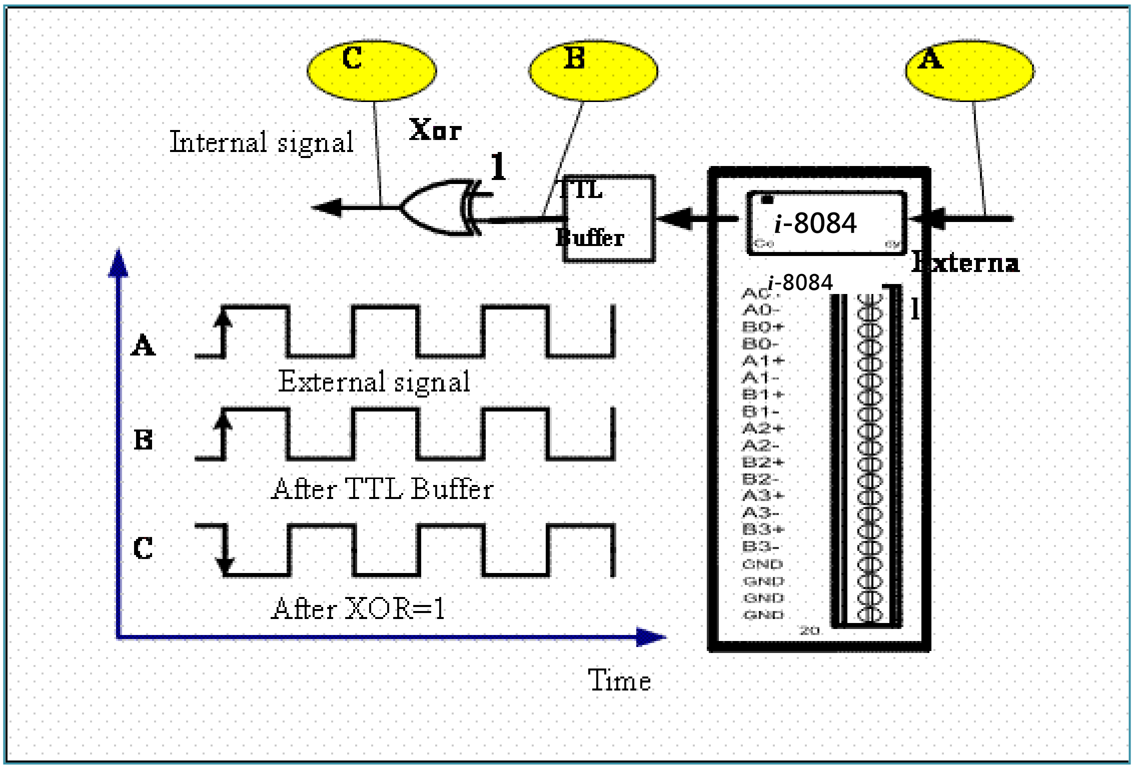
The solution is shown below.



2. TTL Input (XOR=1)

When an external TTL signal is input into an 8084 module through the TTL mechanism, the signal will be the same as the external signal. This internal signal isn't the recommended waveform as it must execute the exclusive OR (XOR=1) operation.

The solution is shown below.



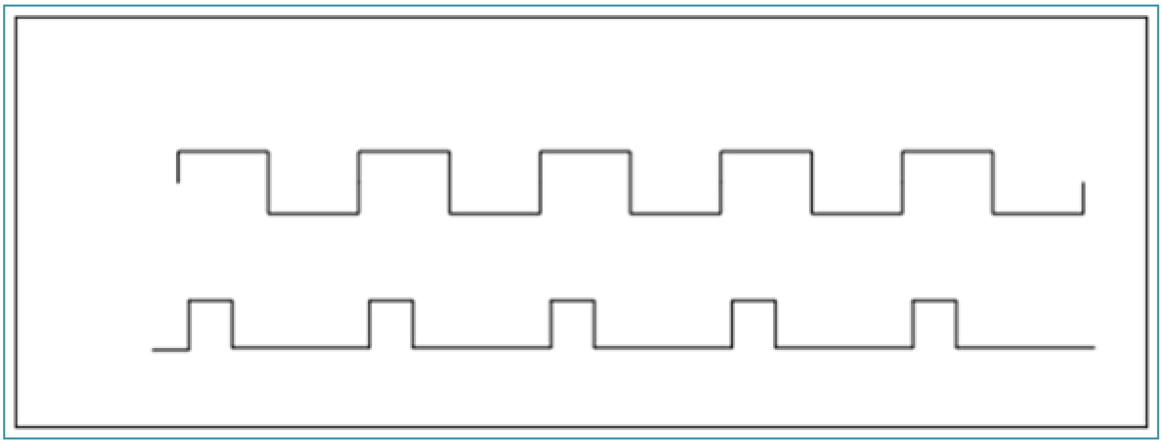
3. Always XOR=0

Regardless of whether the input signal is TTL or isolated, XOR is always set to 0, and the maximum count error can only be 1. XOR=0 can be used for all cases, if a 1-count error is acceptable.

Note:

- When XOR=0 and the 8084 module status is OPEN status (i.e. no signals on the input terminal) , regardless of whether you select the TTL or Isolated mode, the signal at the C point will always be 1. Similarly, if XOR=1 and the status is OPEN, then the signal at the C point will always be 0.

- If the input signal is a pulse rather than a 50/50 duty cycle square waveform, then the 1-count error will not occur as the pulse width is shorter..



2.2. Digital Low Pass Filter

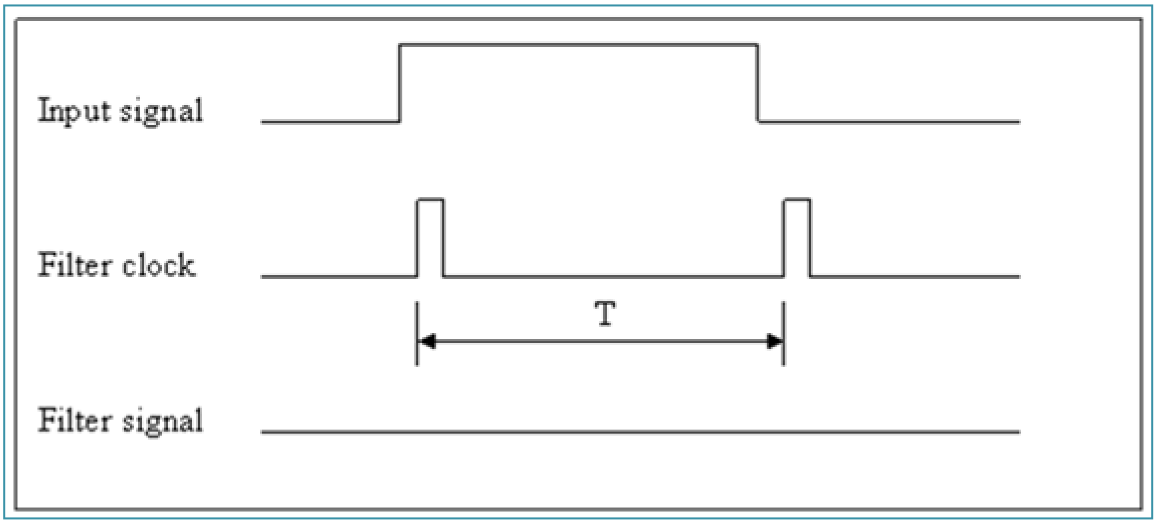
The 8084 has three independent 2nd-order digital noise filters, LP0, LP1 & LP2, to

remove

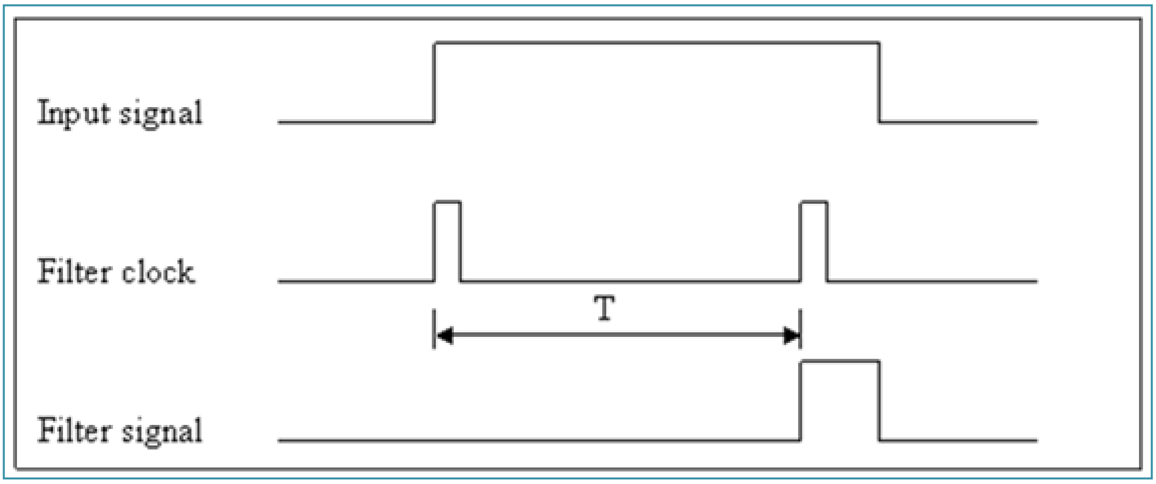
noises as follows:

Channel	Low Pass Filter
A0	Low Pass Filter 0
B0	Low Pass Filter 0
A1	Low Pass Filter 1
B1	Low Pass Filter 1
A2	Low Pass Filter 2
B2	Low Pass Filter 2
A3	Low Pass Filter 2
B3	Low Pass Filter 2

- The Low Pass Filter can be either disabled or programmable from 2 µs to 65535 µs.
- The Low Pass Filter will apply to all working modes, counter or frequency.
- These 3 Low Pass Filters are disabled status in the default shipping. User defined program can be used to issue a command to enable or disable the filters.
- Assume that the filter clock of the Low Pass Filter is set to T, this clock is used to sample the input signal.
- If one of the adjacent 2 samples is low, then the input signal will be removed as follows:

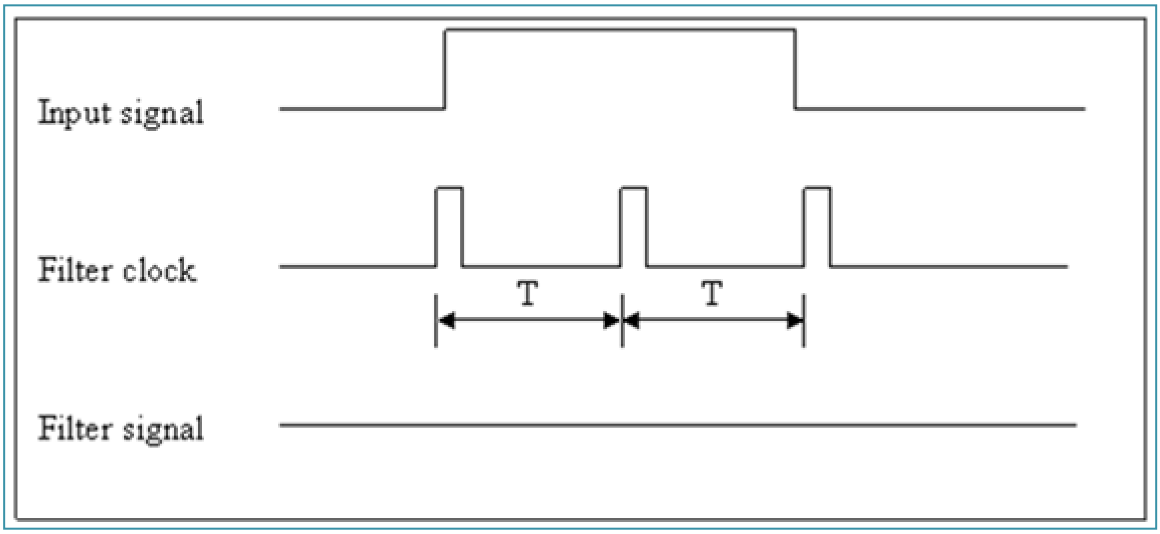


- If the high width of the input signal is shorter then T, it will be filtered.
- If the adjacent 2 samples are all HIGH, the input signal can pass as indicated below:



Note: the filter signal is shorter than the original input signal.

- If the input signal is shorter than 2T, it may be filtered in the following manner:



- The relationship between the input signal and the filter signal is as follows:

  if (2T<input signal), it will pass

  if (T<=input signal<=2T), it may be filtered or passed

  if (input signal <T), it will be filtered

- The software driver, i8084_SetLowPassUs (int Slot, int Channel, unsigned int Us), provides an parameter, Us which can be used to set the Low Pass Filter as follows:

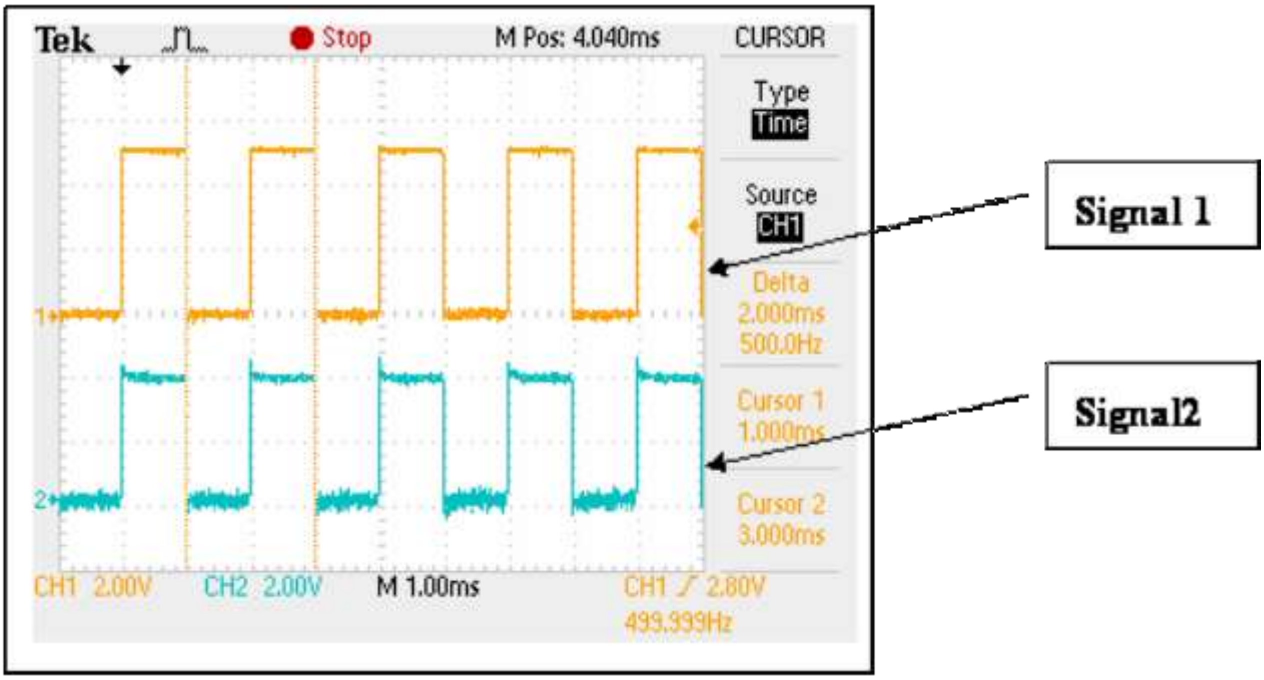
  if Us=1 and 2T =1µs then T = 0.5µs and signal<=0.5µs will be removed

  if Us=2 and 2T= 2µs then T = 1 µs and signal<=1µs will be removed

  if Us=N, N from 1 to 0x7fff and 2T = N µs then signal<=(N/2) µs will be removed

- The Low Pass Filter range can be configured from 1µs to 32767µs. The high width of the signal < (Us/2) will be removed.

For example, if you use a function generator as signal source, the 500Hz signal & 50/50 duty cycle will generate a 1000 μs high & 1000 μs low as follows:



Input signal=500Hz & Low Pass Filter Disable

Signal 1 =

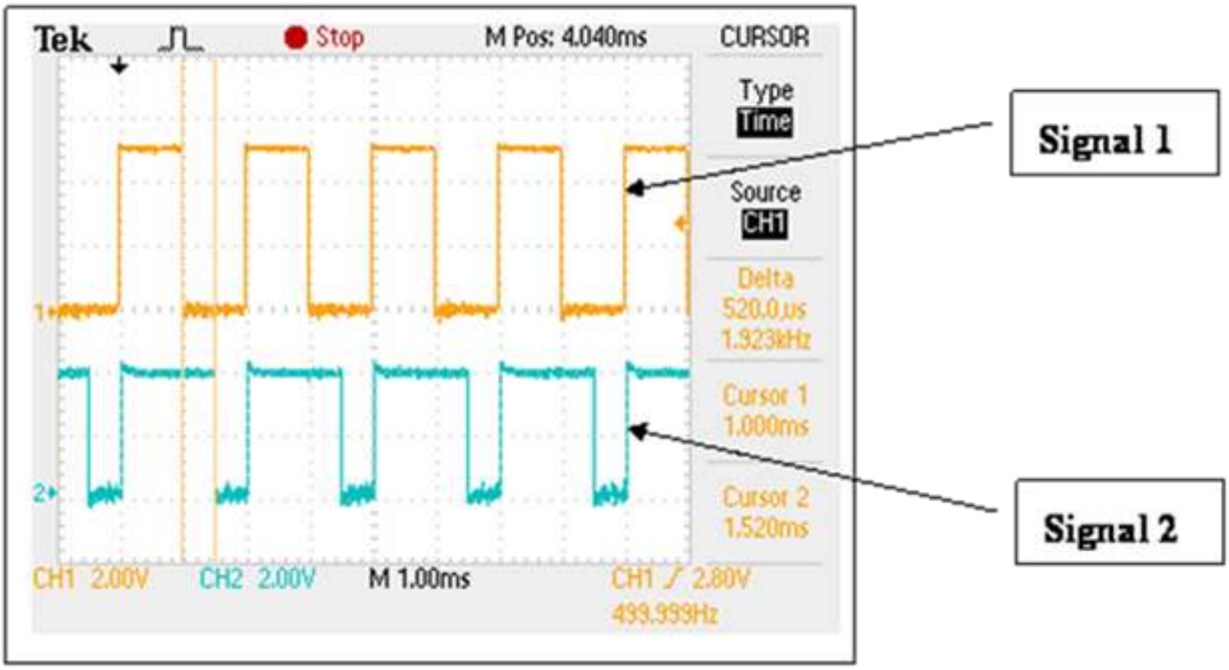
input signal=500Hz, 50/50 duty cycle

Signal 2 =

input signal after Xor and Low Pass Filter, now Xor=0 and Low Pass Filter is disable.

If the Low Pass Filter is disabled, signal 2 will be the same as signal 1 in the above diagram.

If the Low Pass Filter is enabled, signal 2 will be shorter than signal 1 as shown below:



Input signal=500Hz & Low Pass Filter Enabled=1µs

Signal 1 =

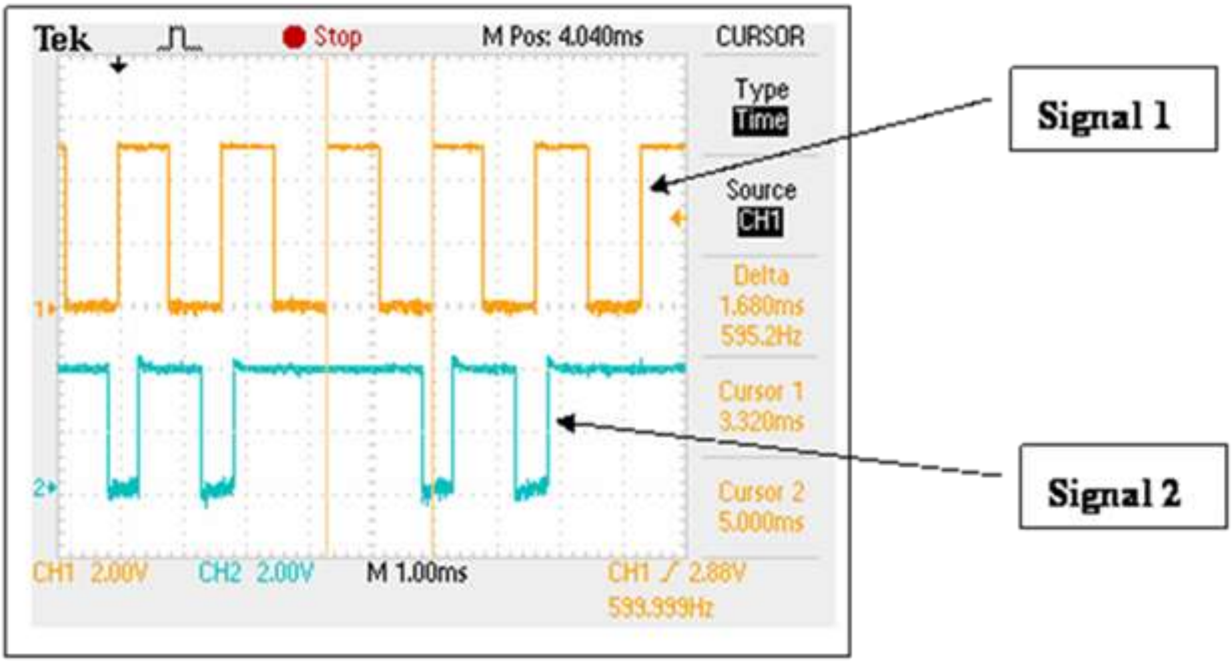
input signal=500Hz, 50/50 duty cycle

Signal 2 =

input signal after Xor and Low Pass Filter, now Xor=0 and the Low Pass Filter is enabled.

Nearly all pulses are passed.

Now you can find that nearly all pulses are passed. If the input signal is increased to 600Hz, then some of the pulses are filtered as follows:



Input signal=600Hz & Low Pass Filter Enabled=1μs

Signal 1 =

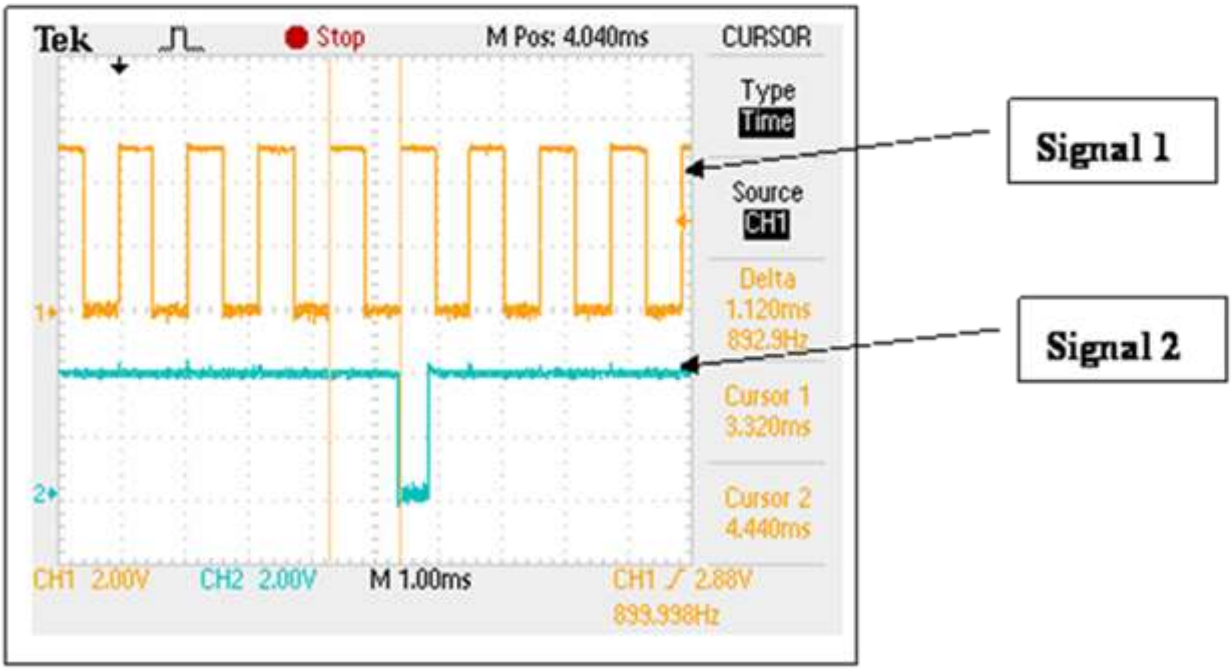
input signal=600Hz, 50/50 duty cycle.

Signal 2 =

input signal after Xor and Low Pass Filter, now Xor =0 and Low Pass Filter is enabled.

Some pulses are filtered.

If the input signal is increased to 900Hz, then nearly all pulses are filtered as illustrated below:



Input signal=900Hz & Low Pass Filter Enabled=1μs

Signal 1 =

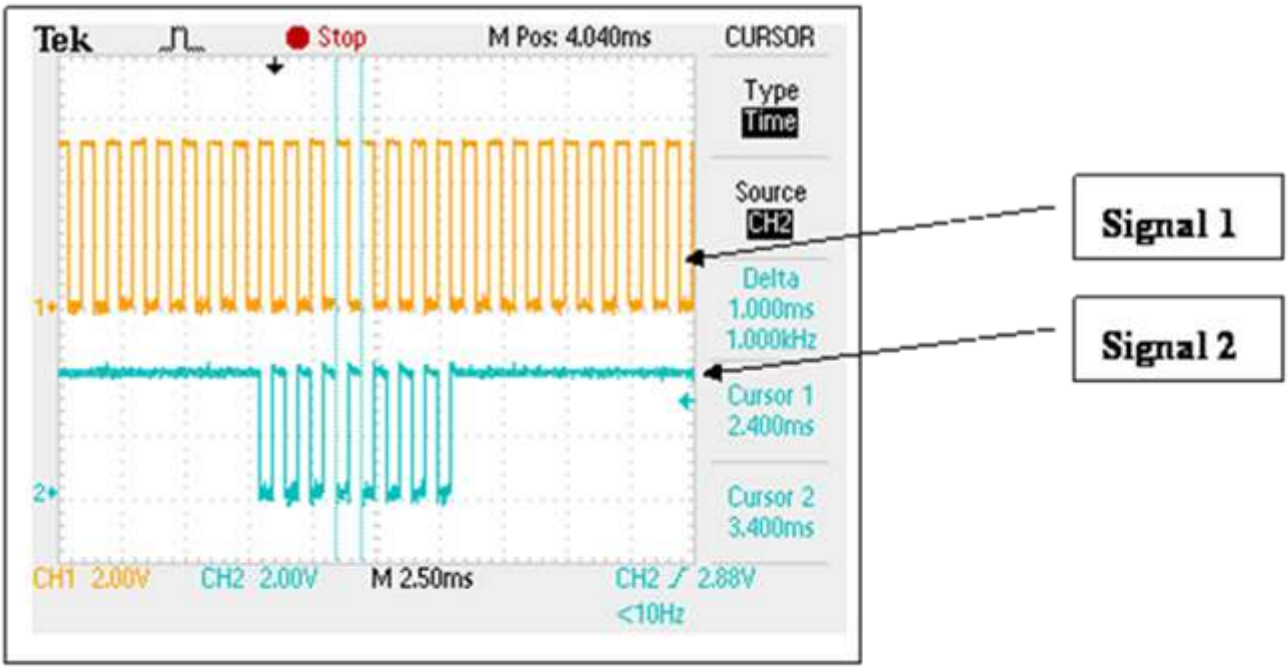
input signal=900Hz, 50/50 duty cycle

Signal 2 =

input signal after Xor and Low Pass Filter, now Xor=0 and the Low Pass Filter is enabled.

Nearly all pulses are filtered.

Because there are some frequency offset errors in the internal crystal, there may be some noises when the input signal width = Low Pass Filter/2 as follows:



Input signal=1000Hz & Low Pass Filter Enabled=1μs

Signal 1 =

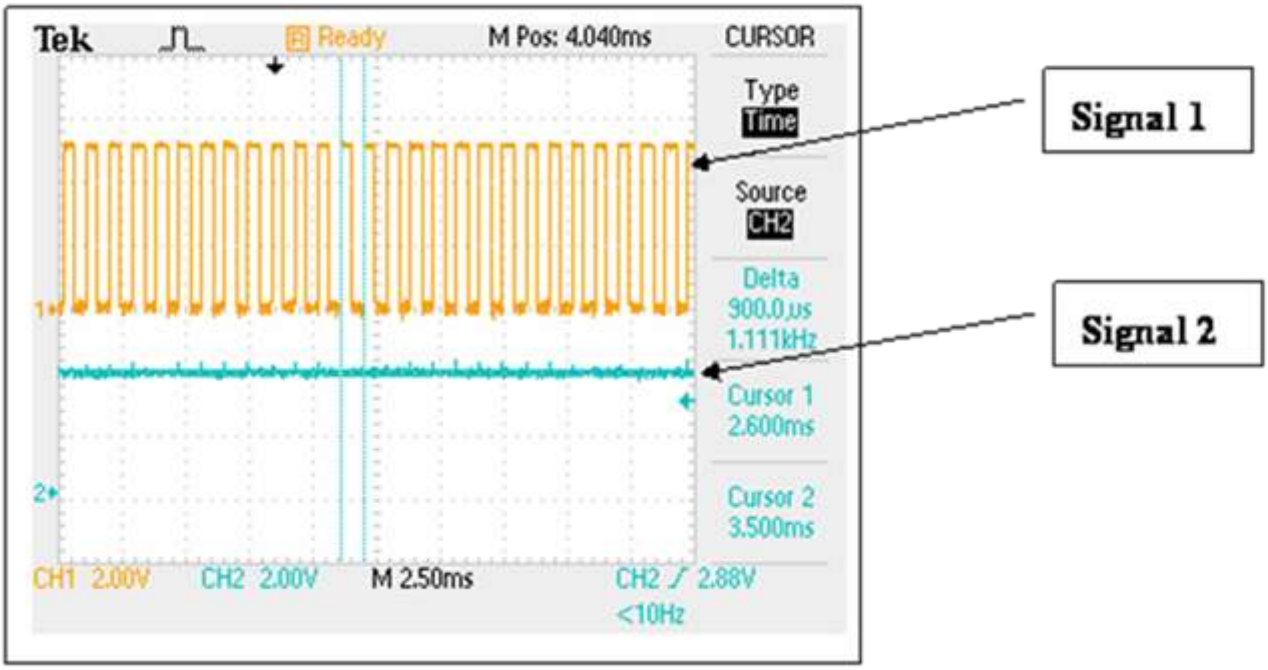
input signal=1000Hz, 50/50 duty cycle à pulse width=500 μs

Signal 2 =

input signal after Xor and Low Pass Filter, now Xor=0 and the Low Pass Filter is enabled. Signal Pulse=500 μs=Low Pass Filter/2.

Nearly all pulses are filtered, but sometimes certain noises will not be filtered.

If the input signal is increased to 1100Hz, then all pulses will be filtered as shown in Figure 1-12:



Input signal=1100Hz & Low Pass Filter Enabled=1μs

In summary, apply the minimum 1μs on Low Pass Filters.

The result of the signal being processed by the Low Pass Filter as follows:

Input signal frequency(Hz)	After Low Pass Filter processing	Reference
Input signal <500Hz (Low Pass Filter=1μs)	All signals will be passed	Figure 1
Input signal =500Hz (Low Pass Filter=1μs)	All signals should be passed	Figure 2
Input signal =600Hz (Low Pass Filter=1μs)	Some signals will be filtered and some will be passed	Figure 3
Input signal =900Hz (Low Pass Filter=1μs)	Many signals will be filtered and few will be passed	Figure 4
Input signal =1000Hz (Low Pass Filter=1μs)	Nearly all signals are filtered	Figure 5
Input signal =1100Hz (>1k Hz) (Low Pass Filter=1μs)	All signals will be filtered	Figure 6

For the same reason, if the signal pulse=Low Pass Filter, certain pulses may be filtered. Therefore, it is recommended to set the cycle time of Low Pass Filter about 5% less than the cycle time of input signal pulse as shown below:

Input pulse =1 ms = 1000 μs à set Low Pass Filter <=950 μs

if Input pulse = 100 μs , set Low Pass Filter <= 95 μs

The minimum Low Pass Filter = 1 μs , input signal < 475K, 50/50 duty cycle

As a result, the maximum speed of the 8084 is recommended to 450K, 50/50 duty cycle.

2.3. Operation Mode

Operation Mode	Description	Number of counter and frequency sets
00	Dir/Pulse counting mode	4 sets
01	Up/Down counting mode	4 sets
02	Frequency mode	8 sets
03	Up counting mode	8 sets
04	Quadrant Counting mode	4 sets

The input channels mapping table and working modes are indicated below:

	Mode 00	Mode 01	Mode 02	Mode 03	Mode 04
A0	Pulse 0	Up 0	Frequency 0	Up 0	A0
B0	Dir 0	Down 0	Frequency 1	Up 1	B0
A1	Pulse 2	Up 2	Frequency 2	Up 2	A1
B1	Dir 2	Down 2	Frequency 3	Up 3	B1
A2	Pulse 4	Up 4	Frequency 4	Up 4	A2
B2	Dir 4	Down 4	Frequency 5	Up 5	B2
A3	Pulse 6	Up 6	Frequency 6	Up 6	A3
B3	Dir 6	Down 6	Frequency 7	Up 7	B3

CountN =

the counter value for channel N, 32bit wide, from -2147483648 to 2147483647

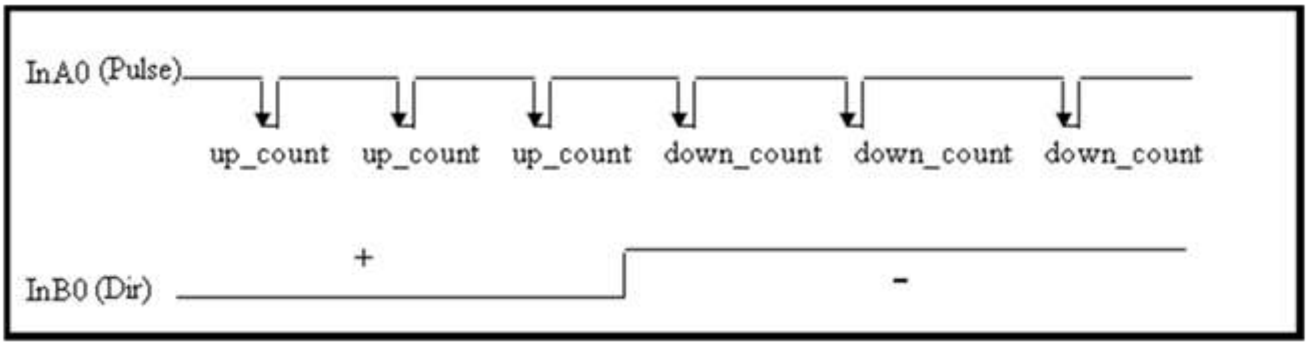
OverflowN =

the counting overflow number for channel N, 16bit wide, from -32768 to 32767

Total Counting Value bit = 32bit + 16bit = 48bit

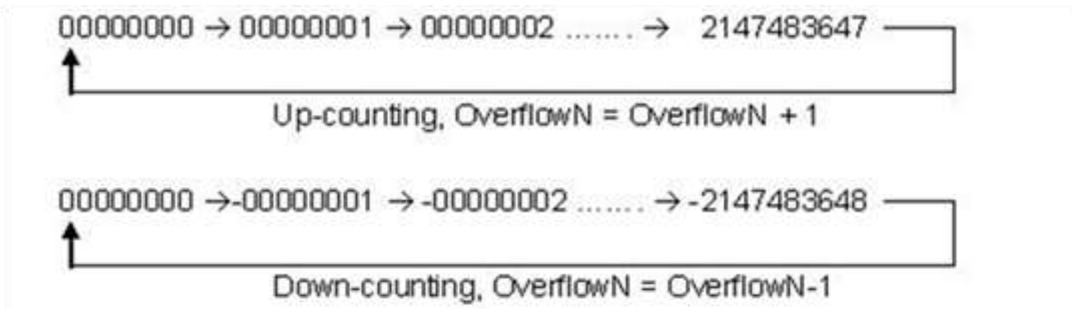
2.3.1. Mode 00: Pulse /Dir Counting

The counter operation for mode 00 (Dir/Pulse mode) is as follows:



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

The counter operation is given as follows:



Pulse/Dir Counter	Counting Variable	Total Counting Value
A0, B0	Count0, Overflow0	Count0 + Overflow0 * 2147483648
A1, B1	Count2, Overflow2	Count2 + Overflow2 * 2147483648
A2, B2	Count4, Overflow4	Count4 + Overflow4 * 2147483648
A3, B3	Count6, Overflow6	Count6 + Overflow6 * 2147483648

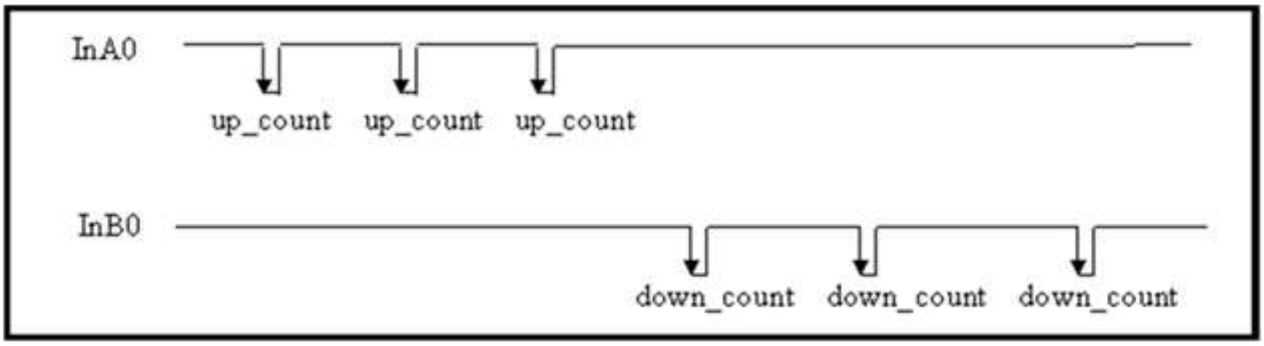
CountN = the counter value for channel N, 32bit wide, from -2147483648 to 2147483647

OverflowN = the counting overflow number for channel N, 16bit wide, from -32768 to 32767

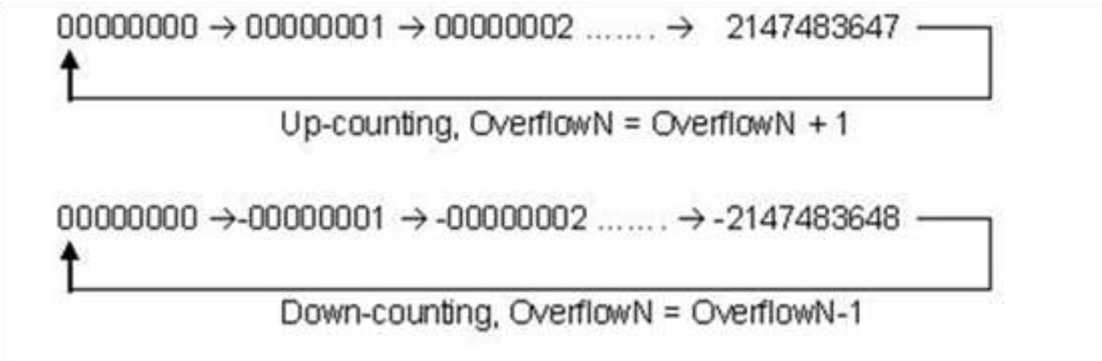
Total Counting Value bit = 32bit + 16bit = 48bit

2.3.2. Mode 01: Up/Down Counting

The counter operation for mode 01 (Up/Down mode) is as follows:



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



Up/Down Counter	Counting Variable	Total Counting Value
A0, B0	Count0, Overflow0	Count0 + Overflow0 * 2147483648
A1, B1	Count2, Overflow2	Count2 + Overflow2 * 2147483648
A2, B2	Count4, Overflow4	Count4 + Overflow4 * 2147483648
A3, B3	Count6, Overflow6	Count6 + Overflow6 * 2147483648

CountN =

the counter value for channel N, 32bit wide, from -2147483648 to 2147483647

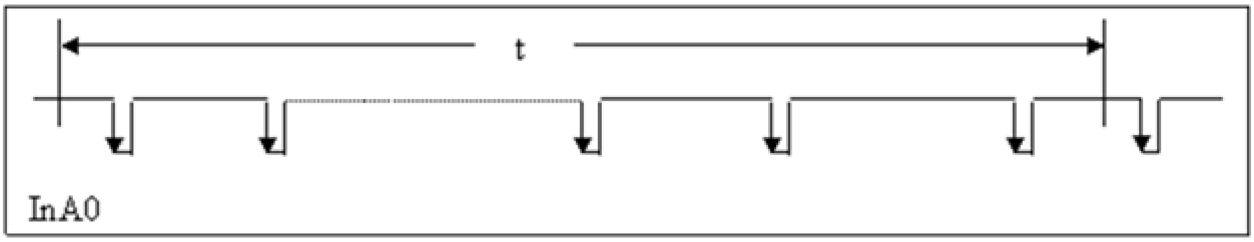
OverflowN =

the counting overflow number for channel N, 16bit wide, from -32768 to 32767

Total Counting Value bit = 32bit + 16bit = 48bit

2.3.3. Mode 02: Frequency Mode

The frequency operation for mode 02 is as follows:



Frequency	Frequency Variable
A0	Frequency0
B0	Frequency1
A1	Frequency2
B1	Frequency3
A2	Frequency4
B2	Frequency5
A3	Frequency6
B3	Frequency7

Period of update time t =0.33 second is the default setting. A user defined command can be used to change the value of t for special applications.

Frequency = Counter value / Period of scan time

Assume t = 0.1 seconds,

If count = 1 à frequency = 1/(0.1/1) = 10 Hz

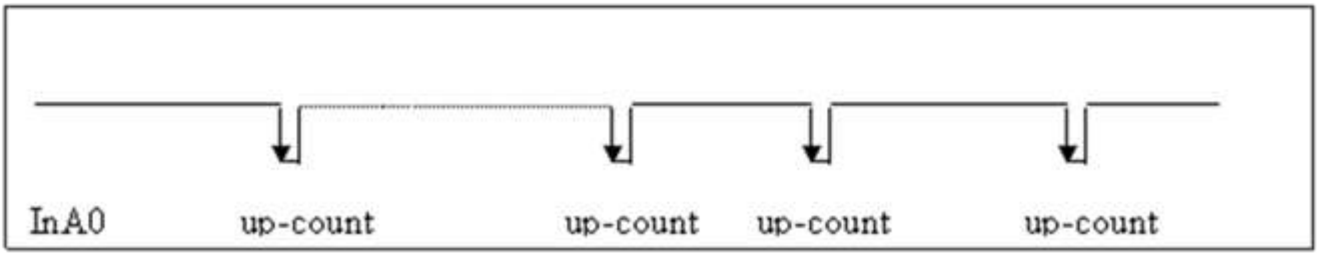
If count = 10 à frequency = 1/(0.1/10) = 100 Hz

All frequency channels will be updated every 0.1 seconds for t= 0.1 seconds.
The software driver provides three ways to adjust.
They are Auto select, Low and High Frequency. (The default is Auto select)
The default configuration data is as follows:

Auto Frequency = the frequency channel will be updated every 330 million seconds;
Low Frequency = the frequency channel will be updated every 1000 million seconds;
High Frequency = the frequency channel will be updated every 100 million seconds;

2.3.4. Mode 03: Up Counting

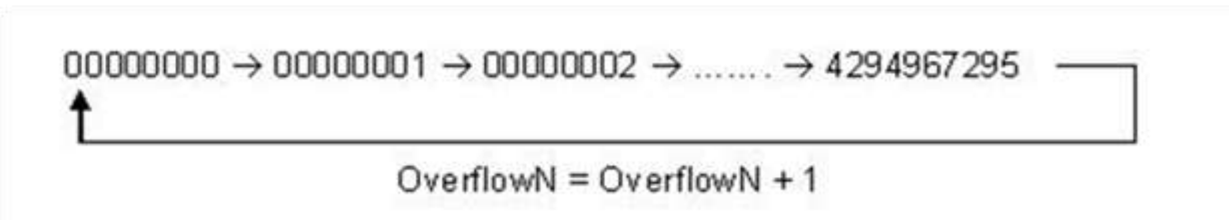
The counter operation for mode 03 is as follows:



Counter_0 will increment by one for every falling edge of InA0

Up Counter	Counting Variable	Total Counting Value
A0	Count0, Overflow0	Count0 + Overflow0 * 4294967296
B0	Count1, Overflow1	Count1 + Overflow1 * 4294967296
A1	Count2, Overflow2	Count2 + Overflow2 * 4294967296
B1	Count3, Overflow3	Count3 + Overflow3 * 4294967296
A2	Count4, Overflow4	Count4 + Overflow4 * 4294967296
B2	Count5, Overflow5	Count5 + Overflow5 * 4294967296
A3	Count6, Overflow6	Count6 + Overflow6 * 4294967296
B3	Count7, Overflow7	Count7 + Overflow7 * 4294967296

The counter operation is as follows:



CountN =

current counter value for channel N, 32bit wide, from 0 to 4294967295

OverflowN =

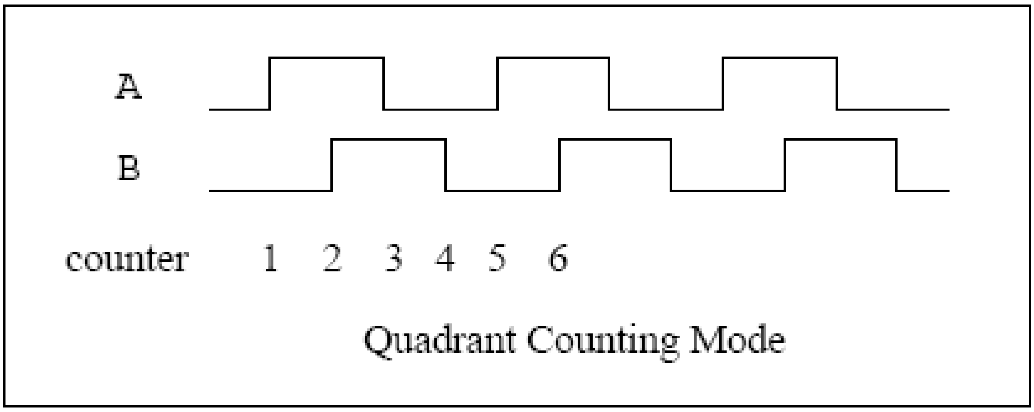
The counting overflow number for channel N, 16bit wide, from 0 to 65535

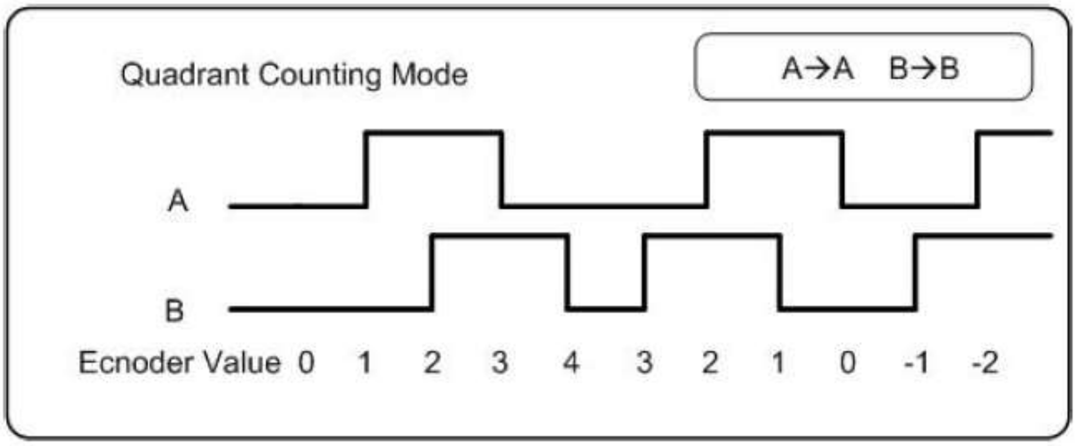
Total Counting Value = CountN + OverflowN * 4294967296

Total Counting Bit = 32bit + 16bit = 48bit

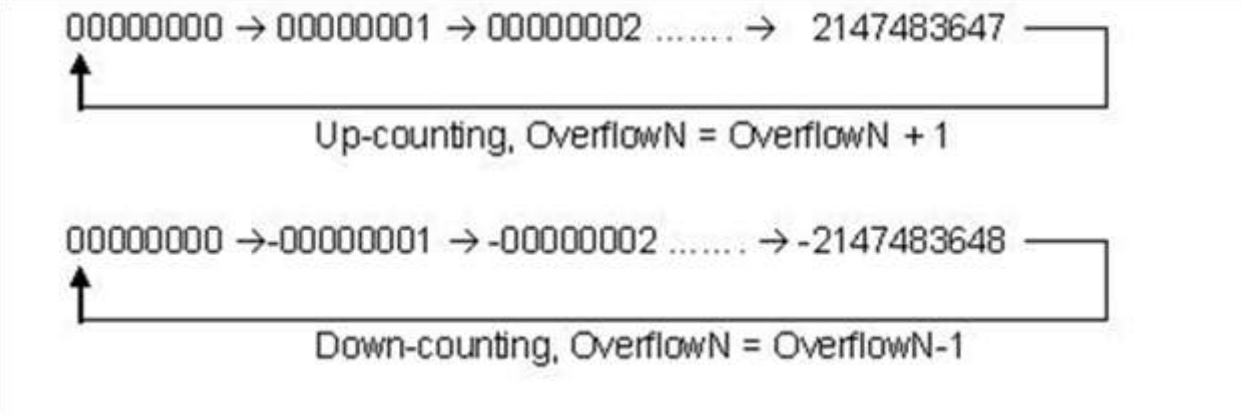
2.3.5. Mode 04: Quadrant Counting

The counter operation for mode 04 is as follows:





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


00000000 → 00000001 → 00000002 .... → 2147483647
Up-counting, OverflowN = OverflowN + 1
00000000 →-00000001 → -00000002 .... → -2147483648
Down-counting, OverflowN = OverflowN-1

3. API for Linux PAC

ICP DAS provides a range of demo programs for different platforms that can be used to verify the functions of the I-8084W. The source code contained in these programs can also be reused in your own custom programs if needed.

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

1. First, user need to download LinPAC SDK, which is includes GNU toolchain, Libraries, header, examples files, etc.
2. Check the power cable, Ethernet cable, VGA monitor, the communication cable between controller and PC has been connected well, and then check the I-8084W has been plugged in the controller.
3. Next, check the communication between controller and PC is fine, and download the demo program files to the controller.
4. The following is a list of the locations where both the demo programs and associated libraries can be found on either the ICP DAS web site or the enclosed CD, and I-8084W use the same library and demo.
   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

The following is a list of the functions provided in the LinPAC SDK library - libi8k.a for Linux PAC.

Function	Description
i8084W_GetLibVersion	This function is used to read the version information for the currently installed library
i8084W_InitDriver	This function is used to initialize the 8084W Library. Note that this function MUST be used before using any other function in the library,
i8084W_SetChannelMode	This function is used to set the operation mode for a specified channel.
i8084W_ReadChannelMode	This function is used to read the operation mode for a specified channel.
i8084W_ReadCntABPhase	This function is used to read the counter value for a specified channel in Quadrant Counting mode
i8084W_ReadCntPulseDir	This function is used to read the counter value for a specified channel in Dir/Pulse counting mode.
i8084W_ReadCntUpDown	This function is used to read the counter value for a specified channel in Up/Down counting mode.
i8084W_ReadCntUp	This function is used to read the counter value for a specified channel in Up counting mode
i8084W_ReadFreqInFloat	This function is used to read the frequency of the specified channel in Frequency mode.
i8084W_ClrCnt	This function is used to clear the counter value for a specified channel regardless of the operation mode.
i8084W_ClrAllCnt	This function is used to clear the counter value for all channels regardless of the operation mode.
i8084W_RecoverDefaultSetting	This function is used to restore the default settings of the I-8084W module installed in a specified slot.
i8084W_ReadXorRegister	This function is used to read the status of the XOR register for a specified channel.
i8084W_SetXorRegister	This function is used to set the status of the XOR register for a specified channel.
i8084W_ReadLowPassFilterUs	This function is used to read the width of the low pass filter in microseconds.
i8084W_SetLowPassFilterUs	This function is used to set the width of the low pass filter in microseconds.
i8084W_ReadLowPassFilterStatus	This function is used to read the status of the low pass filter for a specified channel.
i8084W_SetLowPassFilterStatus	This function is used to set the status of the low pass filter for a specified channel.

Function	Description
i8084W_ReadFreqMode	This function is used to read the frequency mode for a specified channel.
i8084W_SetFreqMode	This function is used to set the frequency mode for a specified channel.
i8084W_ReadFreqTimeoutValue	This function is used to read the timeout value for reading the frequency.
i8084W_SetFreqTimeoutValue	This function is used to set the timeout value for reading the frequency.
i8084W_ReadDIXor	This function is used to read the Digital Input value after the XOR operation has been performed.
i8084W_ReadDIXorLPF	This function is used to read the Digital Input value after the both the XOR and the low pass filter operations have been performed.

3.1. i8084W_GetLibVersion

Get the version number of I-8084 library.

Syntax

int i8084W_GetLibVersion()

Return Values

The version information for the Library file

3.2. i8084W_InitDriver

Configure the I-8084 with the setting stored in the EEPROM.

If there is no settings stored in the EEPROM, the function will call i8084W_RecoverDefaultSetting.

Syntax

```
int i8084W_InitDriver(int Slot)
```

Parameter and Return Values

Slot:

1~8

Return:

0 → OK

-1 → Module not found

>0 → Some Pulse/Dir counters have one count offset (+1)

Bit0=1 A0 has one count offset (+1)

Bit2=1 A1 has one count offset (+1)

Bit4=1 A2 has one count offset (+1)

Bit6=1 A3 has one count offset (+1)

(due to the input channel is high)

3.3. i8084W_SetChannelMode

There are four operation modes in I-8048W. The function is used to set the operation mode of channel of I-8084W.

Syntax

int i8084W_SetChannelMode(int Slot, int Channel, int Mode)

Parameter and Return Values

Slot:

1~8

Channel:

0 ~ 7

Mode:

0 → Dir/Pulse Counter

1 → Up/Down Counter

2 → Frequency

3 → Up Counter

4 --> AB Phase

Return:

0 → No error

1 → The Pulse/Dir counter has one count offset (+1)

(due to the input channel is high)

3.4. i8084W_AutoScan

Auto scan the I-8084W to updates 8 channels.

Syntax

int i8084W_AutoScan(void)

Parameter and Return Values

Remark

This function is used to update the hardware counter values.

The hardware counter is 16-bit.

User's code must call the function or i8084W_ReadCntPulseDir, i8084W_ReadCntUpDown, i8084W_ReadFreq, i8084W_ReadCntUp before the hardware counter is overflow.

Under very high speed signal input, for example: 450K Hz, the 16-bit counter is overflow round 145 ms.

To avoid the overflow situation, user's code is recommended to call i8084W_AutoScan every 70 ms.

3.5. i8084W_ReadCntABPhase

The function is used to read ABPhase counter.

Syntax

int i8084W_ReadCntABPhase(int Slot, int Channel,long *Cnt32U,int *Overflow)

3.6. i8084W_ReadCntPulseDir

The function is used to Read Pulse/Dir Counter

Syntax

int i8084W_ReadCntPulseDir(int Slot, int Channel,long *Cnt32U,int *Overflow)

Parameter and Return Values

Slot:

1~8

Channel:

0 ~ 7

Cnt32L =

32-bit → UpDown Counter

Bit31= 0 → Up Count (count > 0)

Bit31 = 1 →Down Count (count < 0)

Overflow =

number of overflow

Total count =

over * 0 x 80000000 + count

Example A:

Over = 1 , count = 16384,

total count = (1) * 0 x 80000000 + 16384 = 2147500032

Example B:

Over = -1 , count = -8192,

total count = (-1)*0x80000000 -8192 = -2147491840

3.7. i8084W_ReadCntUpDown

The function is used to Read UpDown Counter

Syntax

int i8084W_ReadCntUpDown(int Slot, int Channel,long *Cnt32U,int *Overflow)

Parameter and Return Values

Slot:

1~8

Channel:

0 ~ 7

Cnt32L =

32-bit → UpDown Counter

Bit31= 0 → Up Count (count > 0)

Bit31 = 1 →Down Count (count < 0)

Overflow =

number of overflow

Total count

= over * 0 x 80000000 + count

Example A:

Over = 1 , count = 16384,

total count = (1) * 0 x 80000000 + 16384 = 2147500032

Example B:

Over = -1 , count = -8192,

total count = (-1)*0x80000000 -8192 = -2147491840

3.8. i8084W_ReadFreq

The function is used to read frequency.

Syntax

int i8084W_ReadFreq(int Slot, int Channel, unsigned long *Freq);

int i8084W_ReadFreq(int Slot, int Channel, float *Freq);

Parameter and Return Values

Slot:

1~8

Channel:

0 ~ 7

Freq:

Unit = Hz

3.9. i8084W_ReadCntUp

The function is used to Read Up Counter

Syntax

int i8084W_ReadCntUp(int Slot, int Channel, unsigned long *Cnt32U, unsigned int

*OverFlow)

Parameter and Return Values

Slot:

1~8

Channel:

0 ~ 7

Cnt32U =

32-bit Up Counter

Overflow =

number of Overflow

Total count =

over * 0x100000000 + count

ExampleA:

over=1 , count=16384,

total count = (1)*0x100000000 +16384 = 4294983680

3.10. i8084W_ClrCnt

Clear Counter

Syntax

int i8084W_ClrCnt(int Slot, int Channel)

Parameter and Return Values

Slot:

1~8

Channel:

0 ~ 7

Return:

0 → No error

1 → The Pulse/Dir counter has one count offset (+1).

It is due to the pulse channel is high.

The correct initial situation is:

Pulse channel is low or open

dir signal is high or low.

3.11. i8084W_RecoverDefaultSetting

The function is used to recover default setting of I-8048W.

Syntax

void i8084W_RecoverDefaultSetting(int Slot)

Parameter and Return Values

Slot:

1~8

Remark

Default settings:

XOR register=0

Channel mode= 3 (Up counter mode)

Frequency operate mode = 0 (Auto mode)

Frequency update time: Auto mode = 330 ms

Low freq mode = 1000 ms

High freq mode = 100 ms

Low Pass Filter status = disable

Low Pass Filter signal width = 1 ms

3.12. i8084W_ReadXorRegister

Syntax

int i8084W_ReadXorRegister(int Slot, int Channel, int *XorReg)

Parameter and Return Values

Slot:

1~8

Channel:

0~7

XorReg:

0 → Low active (signal from High to Low, count changed)

1 → High acitve (signal from Low to High, count changed)

Return:

0 → OK

Others → Error codes

3.13. i8084W_SetXorRegister

Syntax

int i8084W_SetXorRegister(int Slot, int Channel, int XorReg)

Parameter and Return Values

Slot:

1~8

Channel:

0~7

XorReg:

0 → Low active (signal from High to Low, count count changed)

1 → High acitve (signal from Low to High, count changed)

Return:

0 → No error

1 → The Pulse/Dir counter has one count offset (+1)

(due to the input channel is high)

3.14. i8084W_ReadChannelMode

Syntax

int i8084W_ReadChannelMode(int Slot, int Channel, int *Mode)

Parameter and Return Values

Slot:

1~8

Channel:

0~7

Mode:

0 → Dir/Pulse Counter

1 → Up/Down Counter

2 → Frequency

3 → Up Counter

4 --> AB Phase

3.15. i8084W_ReadLowPassFilter_Us

Read Low Pass Filter

Syntax

```
int i8084W_ReadLowPassFilter_Us(int Slot, int Channel, unsigned int *Us);
```

Parameter and Return Values

Slot:

1~8

Channel:

0~7

Us:

1~32767, pulse width, unit = 0.001 ms

3.16. i8084W_SetLowPassFilter_Us

Set Low Pass Filter

Syntax

```
int i8084W_SetLowPassFilter_Us(int Slot, int Channel, unsigned int Us)
```

Parameter and Return Values

Slot:

1~8

Channel:

0~7

Us:

1~32767, pulse width, unit = micro second

3.17. i8084W_ReadLowPassFilter_Status

Syntax

```
void i8084W_ReadLowPassFilter_Status(int Slot, int Channel, int *Status);
```

Parameter and Return Values

Slot:

1~8

Channel:

0~7

Status:

0 = disable

1 = enable

3.18. i8084W_SetLowPassFilter_Status

Syntax

void i8084W_SetLowPassFilter_Status(int Slot,int Channel,int Status);

Parameter and Return Values

Slot:

1~8

Channel:

0~7

Status:

0 = disable

1 = enable

3.19. i8084W_ReadFreqMode

Syntax

void i8084W_ReadFreqMode(int Slot, int Channel, int *Mode);

Parameter and Return Values

Slot:

1~8

Channel:

0~7

*Mode:

0 = Auto

1 = Low Frequency

2 = High Frequency

3.20. i8084W_SetFreqMode

Syntax

void i8084W_SetFreqMode(int Slot, int Channel, int Mode);

Parameter and Return Values

Slot:

1~8

Channel:

0~7

*Mode:

0 = Auto

1 = Low Frequency

2 = High Frequency

3.21. i8084W_ReadFreqUpdateTime

Reads the update time used by frequency measurement algorithm

Syntax

void i8084W_ReadFreqUpdateTime(int Slot, int *AutoMode_UpdateTime ,

int *LowMode_UpdateTime, int *HighMode_UpdateTime);

Parameter and Return Values

Slot:

1~8

AutoMode_UpdateTime =

time period for Auto mode, unit: ms

LowMode_UpdateTime =

time period for Low Frequency mode, unit: ms

LowMode_UpdateTime =

time period for High Frequency mode, unit: ms

3.22. i8084W_SetFreqUpdateTime

Sets the update time used by frequency measurement algorithm

Syntax

int i8084W_SetFreqUpdateTime(int Slot, int AutoMode_UpdateTime, int

LowMode_UpdateTime, int HighMode_UpdateTime);

Parameter and Return Values

Slot:

1~8

AutoMode_UpdateTime =

time period for Auto mode, unit: ms

LowMode_UpdateTime =

time period for Low Frequency mode, unit: ms

LowMode_UpdateTime =

time period for High Frequency mode, unit: ms

3.23. i8084W_ReadDI_Xor

Syntax

int i8084W_ReadDI_Xor(int Slot, int *DI);

Parameter and Return Values

Slot:

1~8

*DI: Bit0 = DI of A0 after XorControl

*DI: Bit1 = DI of B0 after XorControl

…

*DI: Bit7 = DI of B3 after XorControl

Return:

0 → OK

<> 0 → Error codes

3.24. i8084W_ReadDI_XorLPF

Syntax

```
int i8084W_ReadDI_XorLPF(int Slot, int *DI);
```

Parameter and Return Values

Slot:

1~8

*DI: Bit0 = DI of A0 after XorControl & Low Pass Filter

*DI: Bit1 = DI of B0 after XorControl & Low Pass Filter

…

*DI: Bit7 = DI of B3 after XorControl & Low Pass Filter

Return:

0 → OK

<> 0 → Error codes

3.25. i8084W_EepWriteEnable

Write_Enable EEPROM

Syntax

int i8084W_EepWriteEnable(int Slot);

Parameter and Return Values

Slot:

1~8

Return:

0 → OK

Others → Error codes

3.26. i8084W_EepWriteDisable

Write_Disable EEPROM

Syntax

int i8084W_EepWriteDisable(int Slot);

Parameter and Return Values

Slot:

1~8

Return:

0 → OK

Others → Error codes

3.27. i8084W_EepWriteWord

Write 16-bit data to EEP

Syntax

int i8084W_EepWriteWord(int Slot, int Addr, int Value);

Parameter and Return Values

Slot:

1~8

Addr:

0~39 for users

40~63 for 8084 configuration

Value = two bytes integer

Return:

0 → OK

-1 → Address error

3.28. i8084W_EepReadWord

Read 16-bit data to EEP

Syntax

int i8084W_EepReadWord(int Slot, int Addr, int *Value);

Parameter and Return Values

Slot:

1~8

Addr:

0~39 for users

40~63 for 8084 configuration

Value = two bytes integer

Return:

0 → OK

-1 → Address error