LSI3188

Application Specific
Quadrature Encoder / Linear Scale
Counter Card

Software Manual (V1.0)

健昇科技股份有限公司

JS AUTOMATION CORP.

新北市汐止區中興路 100 號 6 樓
6F., No.100, Zhongxing Rd.,
Xizhi Dist., New Taipei City, Taiwan
TEL：+886-2-2647-6936
FAX：+886-2-2647-6940
http://www.automation.com.tw
http://www.automation-js.com/
E-mail：control.cards@automation.com.tw

Correction record

Version	Record
1.0	firmware v1.0 up

Contents

1. How to install the software of LSI3188 4

1.1 Install the PCI driver 4

2. Where to find the file you need 5

3. About the LSI3188 software 6

3.1 What you need to get started 6

3.2 Software programming choices 6

4. LSI3188 Language support 7

4.1 Building applications with the LSI3188 software library 7

4.2 LSI3188 Windows Libraries 7

5. Basic concepts of digital I/O control 8

5.1 Types of I/O classified by isolation 8

5.2 Types of Output classified by driver device 8

5.3 Input debounce 9

5.4 Input interrupt 10

5.5 Read back of Output status 10

6. Basic concepts of quadrature encoder counter 11

6.1 Signal input type 11

6.2 Input debounce time 12

6.3 Input polarity 12

6.4 Homing (counter clear mode) 13

7. Basic concepts of counter compare function 15

8. Function format and language difference 16

8.1 Variable data types 17

8.2 Programming language considerations 18

9. Flow chart of application implementation 20

9.1 LSI3188 Flow chart of application implementation 20

10. Software overview and dll function 23

10.1 Initialization and close 23

LSI3188_initial 23

LSI3188_close 23

LSI3188_info 23

10.2 Input/Output function 24

LSI3188_port_polarity_set 24

LSI3188_port_polarity_read 25

LSI3188_debounce_time_set 26

LSI3188_debounce_time_read 26

LSI3188_port_set 27

LSI3188_port_read 27

LSI3188_point_set 28

LSI3188_point_read........28
10.3 Timer function........29
LSI3188_timer_set........29
LSI3188_timer_start........29
LSI3188_timer_stop........30
LSI3188_TC_set........30
LSI3188_TC_read........31
10.4 Interrupt function........32
LSI3188_IRQ_mask_set........32
LSI3188_IRQ_mask_read........33
LSI3188_IRQ_process_link........33
LSI3188_IRQ_enable........34
LSI3188_IRQ_disable........34
LSI3188_IRQ_status_read........35
11. Application specific counter function........36
11.1 Position signal and position counter........38
LSI3188_CI_mode_set........39
LSI3188_CI_mode_read........40
LSI3188_CIO_polarity_set........41
LSI3188_CIO_polarity_read........42
LSI3188_CIO_read........43
LSI3188_counter_control_set........43
LSI3188_counter_control_read........44
LSI3188_counter_set........44
LSI3188_counter_read........44
11.2 Homing (to clear counter)........45
LSI3188_HOMING_mode_set........45
LSI3188_HOMING_mode_read........48
11.3 Compare function........49
LSI3188_trigger_mask_set........52
LSI3188_trigger_mask_read........52
LSI3188_trigger_preload_set........53
LSI3188_trigger_preload_read........53
LSI3188_trigger_out_width_set........54
LSI3188_trigger_out_width_read........54
LSI3188_trigger_direction_set........55
LSI3188_trigger_direction_read........55
12. Dll list........56
13. LSI3188 Error codes summary........58
13.1 LSI3188 Error codes table........58

1. How to install the software of LSI3188

1.1 Install the PCI driver

The PCI card is a plug and play card, once you add a new card on the window system will detect while it is booting. Please follow the following steps to install your new card.

In WinXP/7 and up system you should: (take Win XP as example)

--Make sure the power is off

--Plug in the interface card

--Power on

--A hardware install wizard will appear and tell you it finds a new PCI card
Do not response to the wizard, just Install the file
(..\LSI3188\Software\WinXP_7\ or if you download from website please execute the file
LSI3188_Install.exe to get the file)

--After installation, power off

--Power on, it's ready to use

For more detail of step by step installation guide, please refer the file "installation.pdf " on the CD come with the product or register as a member of our user's club at:
http://automation.com.tw/
to download the complementary documents.

2. Where to find the file you need

WinXP/7 and up

The directory will be located at

.. \ JS Automation \LSI3188\API\   (header files and lib files for VB,VC,BCB,C#)

.. \ JS Automation \LSI3188\Driver\   (backup copy of LSI3188 drivers)

.. \ JS Automation \LSI3188\exe\   (demo program and source code)

The system driver is located at **..\system32\Drivers** and the DLL is located at **..\system**.

For your easy startup, the demo program with source code demonstrates the card functions and help file.

3. About the LSI3188 software

LSI3188 software includes a set of dynamic link library (DLL) and system driver that you can utilize to control the interface card's functions.

Your LSI3188 software package includes setup driver, tutorial example and test program that help you how to setup and run appropriately, as well as an executable file which you can use to test each of the LSI3188 functions within Windows' operation system environment.

3.1 What you need to get started

To set up and use your LSI3188 software, you need the following:

- LSI3188 software
- LSI3188 hardware

  Main board

  Wiring board (Option)

3.2 Software programming choices

You have several options to choose from when you are programming LSI3188 software. You can use Borland C/C++, Microsoft Visual C/C++, Microsoft Visual Basic, or any other Windows-based compiler that can call into Windows dynamic link libraries (DLLs) for use with the LSI3188 software.

4. LSI3188 Language support

The LSI3188 software library is a DLL used with WinXP/7 and up. You can use these DLL with any Windows integrating development environment that can call Windows' DLLs.

4.1 Building applications with the LSI3188 software library

The LSI3188 function reference topic contains general information about building LSI3188 applications, describes the nature of the LSI3188 files used in building LSI3188 applications, and explains the basics of making applications using the following tools:

Applications tools

- Microsoft Visual C/C++
- Borland C/C++
- Microsoft Visual C#
- Microsoft Visual Basic
- Microsoft VB.net

If you are not using one of the tools listed, consult your development tool reference manual for details on creating applications that call DLLs.

4.2 LSI3188 Windows Libraries

The LSI3188 for Windows function library is a DLL called **LSI3188.dll**. Since a DLL is used, LSI3188 functions are not linked into the executable files of applications. Only the information about the LSI3188 functions in the LSI3188 import libraries is stored in the executable files.

Import libraries contain information about their DLL-exported functions. They indicate the presence and location of the DLL routines. Depending on the development tools you are using, you can make your compiler and linker aware of the DLL functions through import libraries or through function declarations.

Refer to **Table 1** to determine to which files you need to link and which to include in your development to use the LSI3188 functions in LSI3188.dll.

Header Files and Import Libraries for Different Development Environments		
Language	**Header File**	**Import Library**
Microsoft Visual C/C++	LSI3188.h	LSI3188VC.lib
Borland C/C++	LSI3188.h	LSI3188BC.lib
Microsoft Visual C#	LSI3188.cs	
Microsoft Visual Basic	LSI3188.bas	
Microsoft VB.net	LSI3188.vb	

Table 1

5. Basic concepts of digital I/O control

The digital I/O control is the most common type of PC based application. For example, on the main board, printer port is the TTL level digital I/O.

5.1 Types of I/O classified by isolation

If the system and I/O are not electrically connected, we call it is isolated. There are many kinds of isolation: by transformer, by photo-coupler, by magnetic coupler,… Any kind of device, they can break the electrical connection without breaking the signal is suitable for the purpose.

Currently, photo-coupler isolation is the most popular selection, isolation voltage up to 2000V or over is common. But the photo-coupler is limited by the response time, the high frequency type cost a lot. The new selection is magnetic coupler, it is design to focus on high speed application.

The merit of isolation is to avoid the noise from outside world to enter the PC system, if the noise comes into PC system without elimination, the system maybe get “crazy” by the noise disturbance. Of course the isolation also limits the versatile of programming as input or output at the same pin as the TTL does. The inter-connection of add-on card and wiring board maybe extend to several meters without any problem.

The non-isolated type is generally the TTL level input/output. The ground and power source of the input/output port come from the system. Generally you can program as input or output at the same pin as you wish. **The connection of wiring board and the add-on board is limited to 50cm or shorter** (depends on the environmental noise condition).

5.2 Types of Output classified by driver device

There are several devices used as output driver, the relay, transistor or MOS FET, SCR and SSR.

Relay is electric- mechanical device, it life time is about 1,000,000 times of switching. But on the other hand it has many selections such as high voltage or high current. It can also be used to switch DC load or AC load.

Transistor and MOS FET are basically semi-permanent devices. If you have selected the right ratings, it can work without switching life limit. But the transistor or MOS FET can only work in DC load condition.

The transistor or MOS FET also have another option is source or sink. For PMOS or PNP transistor is source type device, the load is one terminal connects to output and another connects to common ground, but NPN or NMOS is one terminal connects to output and the other connects to VCC+. **If you are concerned about hazard from high DC voltage while the load is floating, please choose the source type driver device.**

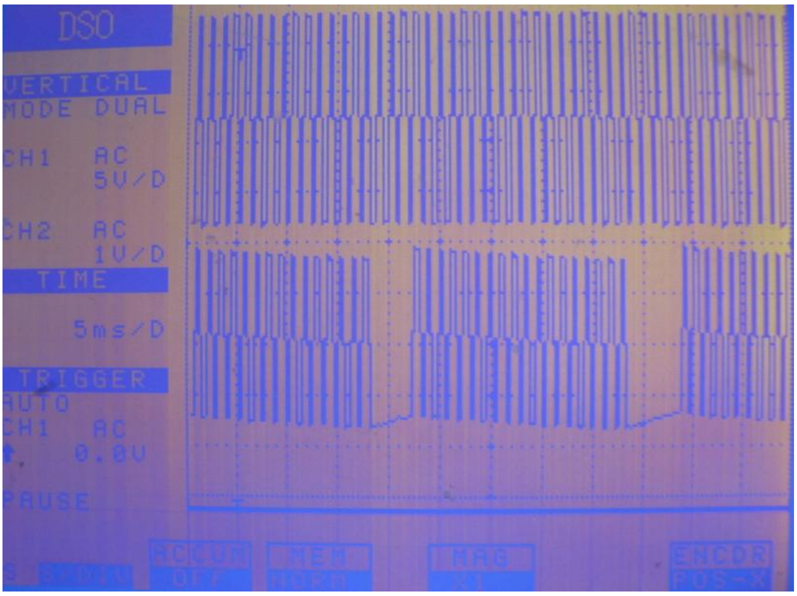
SCR (or triac) is seldom direct connect to digital output, but his relative SSR is the most often selection. In fact, SSR is a compact package of trigger circuit and triac. You can choose zero cross trigger (output command only turn on the output at power phase near zero to eliminate surge) or direct turn on type. SSR is working in AC load condition.

5.3 Input debounce

Debounce is the function to filter the input jitters. From the microscope view of a switch input, you will see the contact does not come to close or release to open clearly. In most cases, it will contact-release-contact-release… for many times then go to steady state (ON or OFF). If you do not have the debounce function, you will read the input at high state and then next read will get low state, this maybe an error data for your decision of contact input.

Debounce can be implemented by hardware or software. Analog hardware debounce circuit will have fixed time constant to filter out the significant input signal, if you want to change the response time, the only way is to change the circuit device.

If digital debounce is implemented, maybe several filter frequency you can choose. To choose the filter frequency, please keep the Nyquist–Shannon sampling theorem in mind: filter sample frequency must at least twice of the input frequency. The following sample is a bad selection of debounce filter, the input frequency is not as low as les than half of the sample frequency, the output will generate a beat frequency.



<- Input frequency at 835Hz

<- Output of digital filter,
Please note the beat frequency.

Digital debounce circuit work at 1KHZ sample rate and observe the output of filter from 835Hz input

Software debounce will consume the CPU time a lot, we do not recommend to use except for you really know you want.

5.4 Input interrupt

You can scan the input by polling, but the CPU will spend a lot of time to do null task. Another way is use a timer to sample the input at adequate time (remind the Nyquist–Shannon sampling theorem, at least double of the input frequency). The third one is directly allows the input to generate interrupt to CPU. To use direct interrupt from input, the noise coupled from input must take special care not to mal-trigger the interrupt. LSI3188 card has 8 bit isolated digital input and 8 bit isolated digital output. Each input can be configured as external interrupt source.

5.5 Read back of Output status

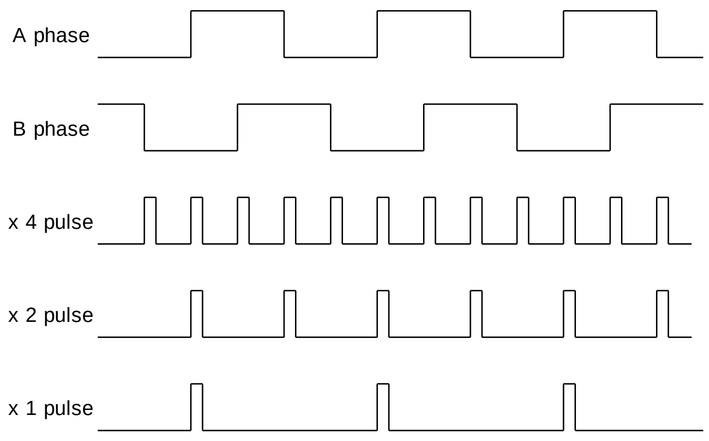
Some applications need to read back the output status, if the card do not provide output status read back, you can use a variable to store the status of output before you really command it output. Some cards provide the read back function but please note that **the read back status is come from the output register, not from the real physical output.**

6. Basic concepts of quadrature encoder counter

6.1 Signal input type

In LSI3188 card, there are 3 major signal types can be count.

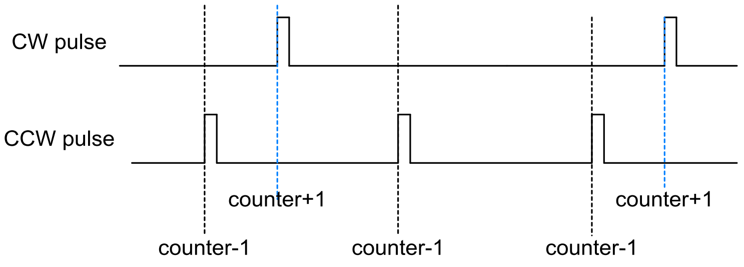
Quadrature input type



The left diagram shown that A phase leads B, if we take A leads B as up count and the counting pulse of up count will depends on the multiple rate.

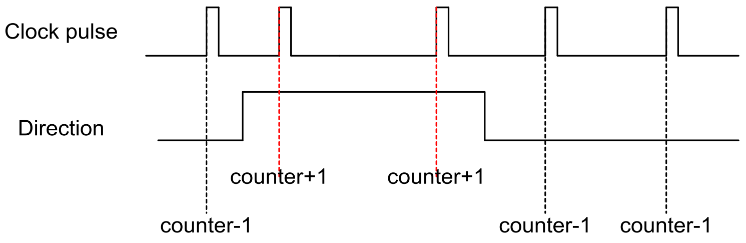
On the other hand, if B phase leads A phase, the counter will be down count.

CW and CCW input type (Dual pulse mode)



The left diagram shown that CW and CCW pulses. Any CW pulse input will increase counter by 1 and any CCW pulse input will decrease counter by 1.

Clock and direction input type (Single pulse mode)



The left diagram shown that Clock and Direction pulses. Any Clock pulse input will increase counter by 1 while the Direction signal is make and any Clock pulse input will decrease counter by 1 while Direction signal is break.

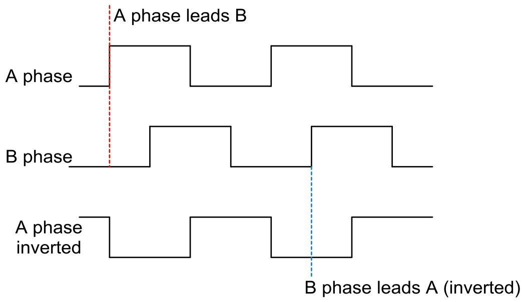
6.2 Input debounce time

If the counter input signal comes from the noisy environment, the input needs to filter out the unwanted signal and keep the meaningful signals to go through to counter. A programmable debounce digital filter put in the way of input signal to drop out the unwanted signal is a good choice.

Users can use the default debounce time constant or change depending on the signal speed and environment noise. A noisy environment normally needs large time constant to drop out the unwanted signal and high pulse rate limits the time constant you can choose. At default, the debounce function will drop the pulse duration less than 1us (debounce frequency 1M). You can choose one from 512K, 1M, 2M, 4M, 8M, 16M to meet your requirement.

6.3 Input polarity

For the maximum flexibility, the polarity function will change the input signal to meet the requirements of the following function blocks. Say A phase leads B in your external signal input, you can invert the A phase to change to B phase leads A phase without actually change the wiring.

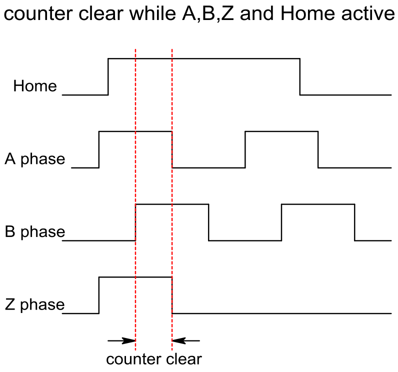


From left diagram, you can see A phase change polarity is equivalent to change A phase from lead state to lag state.

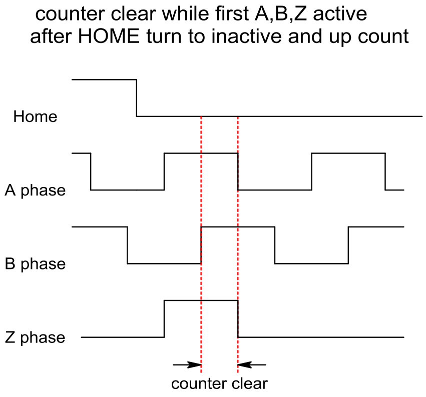
6.4 Homing (counter clear mode)

Normal counters use external asynchronous reset to clear counter but the quadrature counter generally provides more versatile functions to fit the need of different applications. In most quadrature counter applications the counter clear function also called as counter “HOMING”.

There are several modes to do homing:

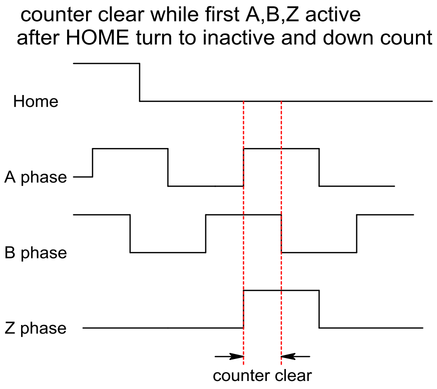


The counter will be cleared while A,B Z and Home are all “make”.


counter clear while first A,B,Z active
after HOME turn to inactive and up count
Home
A phase
B phase
Z phase
counter clear

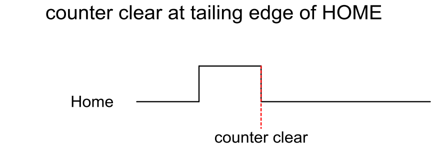
The counter will be cleared while the following conditions meet:

1. HOME switch turns into “BREAK” state.
2. first A,B Z are all “MAKE” and counter up counts (suppose A phase leads B phase).

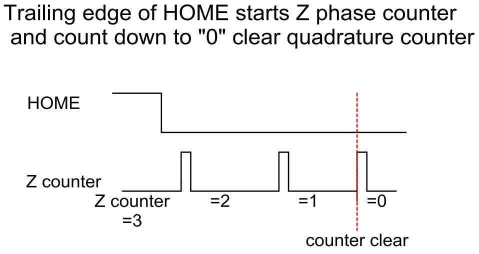

counter clear while first A,B,Z active
after HOME turn to inactive and down count
Home
A phase
B phase
Z phase
counter clear

The counter will be cleared while the following conditions meet:

1. the HOME switch turns into “BREAK” state.
2.first A,B Z are all “MAKE” and counter down counts (suppose B phase leads A phase).



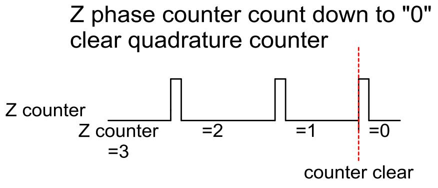
The counter will be cleared at the tailing edge of the HOME switch.


Trailing edge of HOME starts Z phase counter
and count down to "0" clear quadrature counter
HOME
Z counter
Z counter
=3
=2
=1
=0
counter clear

The counter will be cleared the following conditions meet:

1. the HOME switch turns into “BREAK” state.

2. Since the HOME tailing edge, Z phase counter counts down to “0”



The counter will be cleared while Z phase counter counts down to “0”.

7. Basic concepts of counter compare function

The most powerful function of LSI3188 card is the high speed comparison function. You can use this function to trigger external devices such as CCD camera to catch vision data.

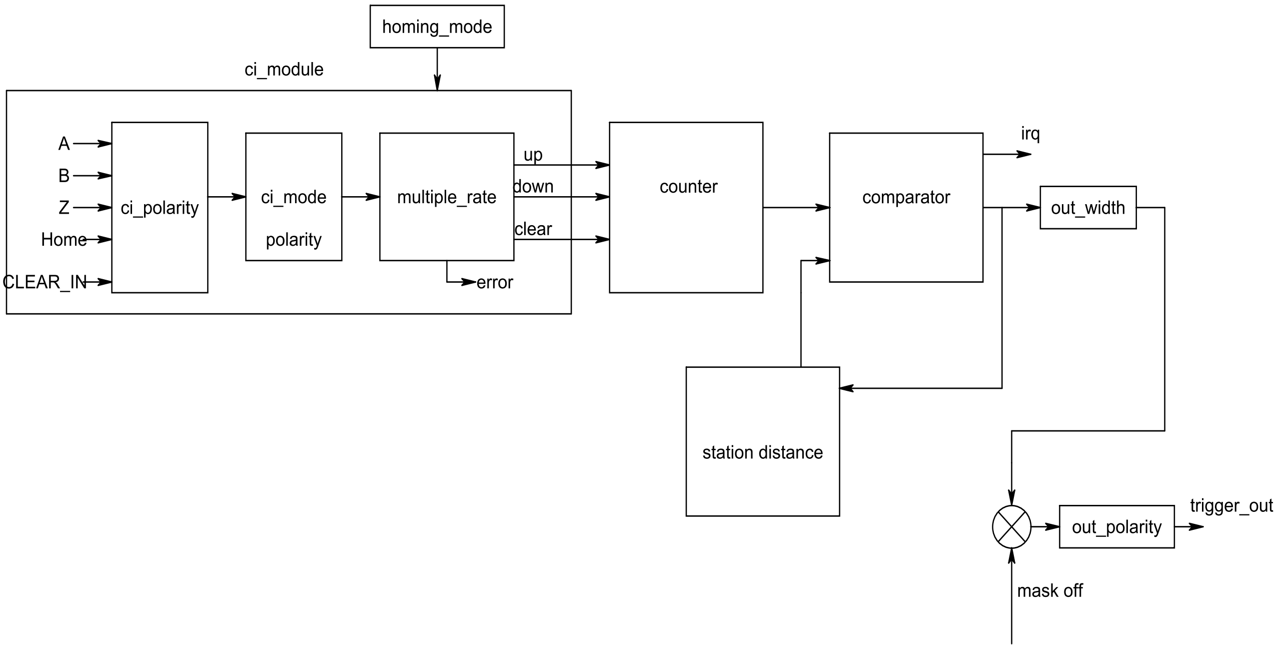


fig. 7.1 Function block of encoder counter card

From the above diagram, while the comparator compares equal, it will generate a trigger output and maintain the pulse at out_width duration. For more detailed application specific compare function, please refer Chapt. 11 Application specific counter function

8. Function format and language difference

Every LSI3188 function is consist of the following format:

Status = function_name (parameter 1, parameter 2, … parameter n);

Each function returns a value in the **Status** global variable that indicates the success or failure of the function. A returned **Status** equal to zero that indicates the function executed successfully. A non-zero status indicates failure that the function did not execute successfully because of an error, or executed with an error.
Note : **Status** is a 32-bit unsigned integer.

The first parameter to almost every LSI3188 function is the parameter **CardID** which is located the driver of LSI3188 board you want to use those given operation. The **CardID** is assigned by DIP/ROTARY SW. You can utilize multiple devices with different card CardID within one application; to do so, simply set the hardware and pass the appropriate **CardID** to each function.
Note: **CardID** is set by DIP/ROTARY SW (**0x0-0xF**)

8.1 Variable data types

Every function description has a parameter table that lists the data types for each parameter. The following sections describe the notation used in those parameter tables and throughout the manual for variable data types.

Primary Type Names					
Name	**Description**	**Range**	**C/C++**	**Visual BASIC**	**Pascal (Borland Delphi)**
u8	8-bit ASCII character	0 to 255	char	Not supported by BASIC. For functions that require character arrays, use string types instead.	Byte
i16	16-bit signed integer	-32,768 to 32,767	short	Integer (for example: deviceNum%)	SmallInt
u16	16-bit unsigned integer	0 to 65,535	unsigned short for 32-bit compilers	Not supported by BASIC. For functions that require unsigned integers, use the signed integer type instead. See the i16 description.	Word
i32	32-bit signed integer	-2,147,483,648 to 2,147,483,647	long	Long (for example: count&)	LongInt
u32	32-bit unsigned integer	0 to 4,294,967,295	unsigned long	Not supported by BASIC. For functions that require unsigned long integers, use the signed long integer type instead. See the i32 description.	Cardinal (in 32-bit operating systems). Refer to the i32 description.
f32	32-bit single-precision floating-point value	-3.402823E+38 to 3.402823E+38	float	Single (for example: num!)	Single
f64	64-bit double-precision floating-point value	-1.797683134862315E+308 to 1.797683134862315E+308	double	Double (for example: voltage Number)	Double

Table 1

8.2 Programming language considerations

Apart from the data type differences, there are a few language-dependent considerations you need to be aware of when you use the LSI3188 API. Read the following sections that apply to your programming language.

Note: Be sure to include the declaration functions of LSI3188 prototypes by including the appropriate LSI3188 header file in your source code. Refer to 4.2 LSI3188 Windows Libraries for the header file appropriate to your compiler.

8.2.1 C/C++

For C or C++ programmers, parameters listed as Input/Output parameters or Output parameters are pass-by-reference parameters, which means a pointer points to the destination variable should be passed into the function. For example, the Read Port function has the following format:

Status = LSI3188_port_read(u8 CardID, u8 port, u8*data);

where **CardID** and **port** are input parameters, and **data** is an output parameter. Consider the following example:

```
u8 CardID, port;
u8 data,
u32 Status;
Status = LSI3188_port_read (CardID, port, &data);
```

8.2.2 Visual basic

The file LSI3188.bas contains definitions for constants required for obtaining LSI3188 Card information and declared functions and variable as global variables. You should use these constants symbols in the LSI3188.bas, do not use the numerical values.

In Visual Basic, you can add the entire LSI3188.bas file into your project. Then you can use any of the constants defined in this file and call these constants in any module of your program. To add the LSI3188.bas file for your project in Visual Basic 4.0, go to the **File** menu and select the **Add File... option**. Select LSI3188.bas, which is browsed in the LSI3188 \ API directory. Then, select **Open** to add the file to the project.

To add the LSI3188.bas file to your project in Visual Basic 5.0 and 6.0, go to the **Project** menu and select **Add Module**. Click on the Existing tab page. **Select** LSI3188.bas, which is in the LSI3188 \ API directory. Then, select **Open** to add the file to the project.

8.2.3 Borland C++ builder

To use Borland C++ builder as development tool, you should generate a .lib file from the .dll file by implib.exe.

```
 implib LSI3188BC.lib LSI3188.dll
```

Then add the **LSI3188BC.lib** to your project and add

#include “LSI3188.h” to main program.

Now you may use the dll functions in your program. For example, the Read Port function has the following format:

Status = LSI3188_port_read(u8 CardID, u8 port, u8*data);

where **CardID** and **port** are input parameters, and **data** is an output parameter. Consider the following example:

u8 CardID, port;

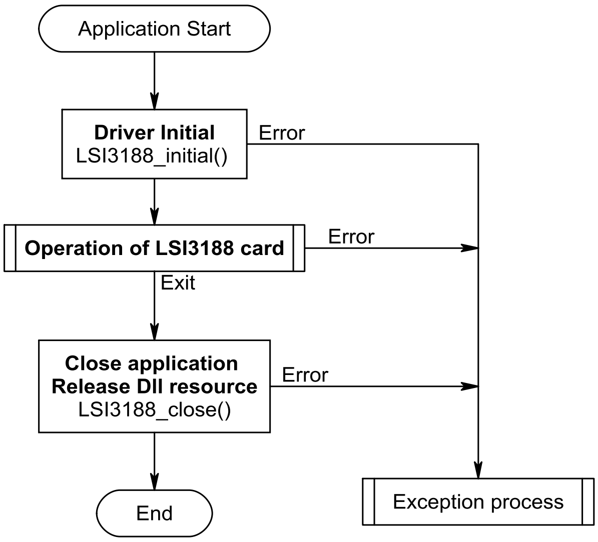
u8 data;

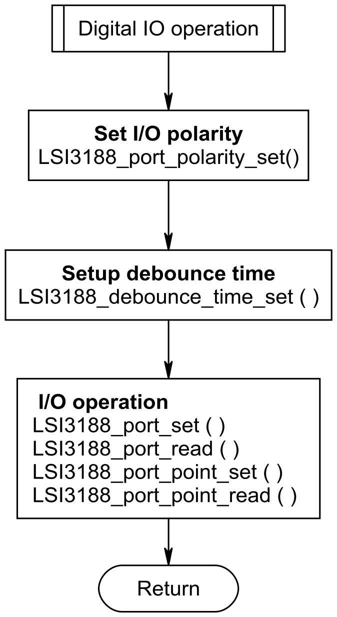
u32 Status;

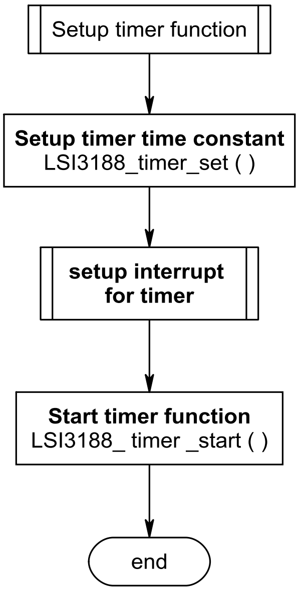
Status = LSI3188_port_read(CardID, port, &data);

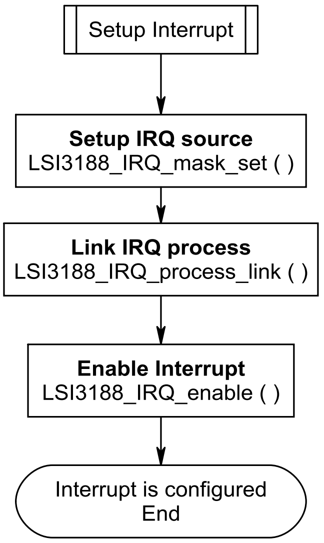
9. Flow chart of application implementation

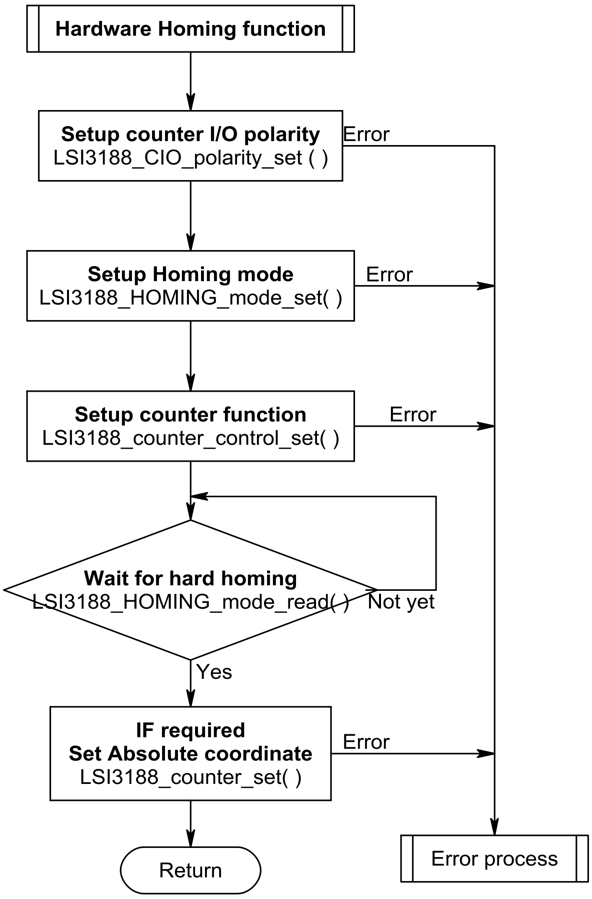
9.1 LSI3188 Flow chart of application implementation

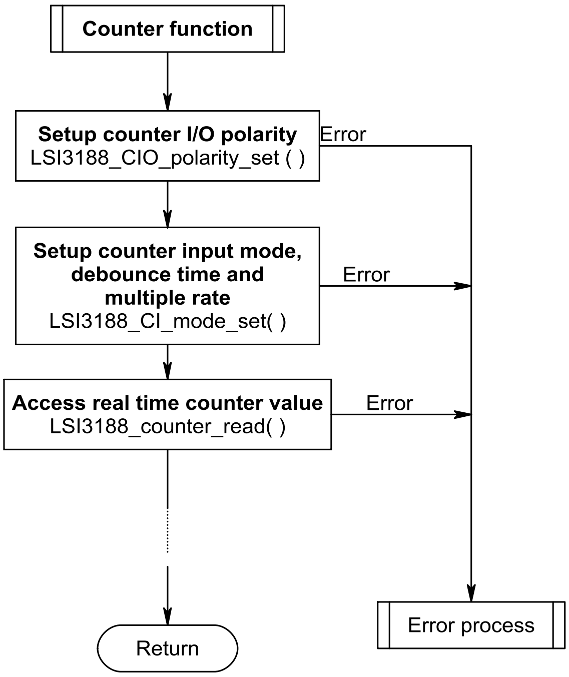


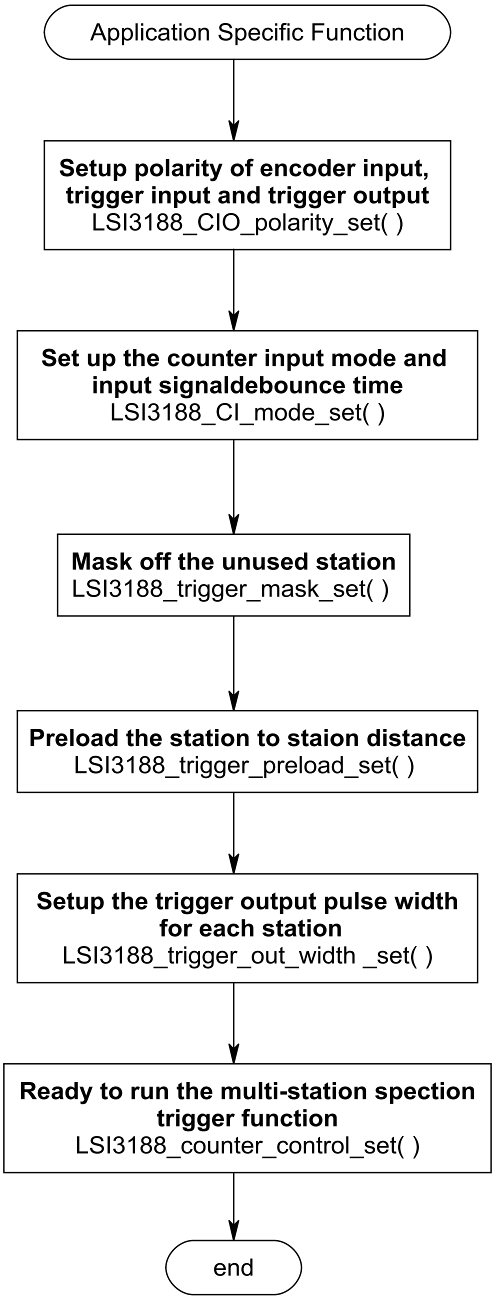







Hardware Homing function
Setup counter I/O polarity
LSI3188_CIO_polarity_set ( )
Error
Setup Homing mode
LSI3188_HOMING_mode_set( )
Error
Setup counter function
LSI3188_counter_control_set( )
Error
Wait for hard homing
LSI3188_HOMING_mode_read( )
Not yet
Yes
IF required
Set Absolute coordinate
LSI3188_counter_set( )
Error
Return
Error process


Counter function
Setup counter I/O polarity
LSI3188_CIO_polarity_set ( )
Error
Setup counter input mode,
debounce time and
multiple rate
LSI3188_CI_mode_set( )
Error
Access real time counter value
LSI3188_counter_read( )
Error
Return
Error process


Application Specific Function
Setup polarity of encoder input, trigger input and trigger output
LSI3188_CIO_polarity_set( )
Set up the counter input mode and input signaldebounce time
LSI3188_CI_mode_set( )
Mask off the unused station
LSI3188_trigger_mask_set( )
Preload the station to staion distance
LSI3188_trigger_preload_set( )
Setup the trigger output pulse width for each station
LSI3188_trigger_out_width _set( )
Ready to run the multi-station spection trigger function
LSI3188_counter_control_set( )
end

10. Software overview and dll function

10.1 Initialization and close

You need to initialize system resource each time you run your application,

LSI3188_initial( ) will do.

Once you want to close your application, call

LSI3188_close( ) to release all the resource.

If you want to know the physical address assigned by OS, use

LSI3188_info( ) to get the address.

- **LSI3188_initial**

Format : **u32 status =LSI3188_initial (void)**

Purpose: Initial the LSI3188 resource when start the Windows applications.

- **LSI3188_close**

Format : **u32 status = LSI3188_close (void);**

Purpose: The LSI3188_close () function is corresponded with LSI3188_initial ( ) function to make LSI3188 card windows application program completely ended and memory fully be released.

- **LSI3188_info**

Format : **u32 status =LSI3188_info(u8 CardID, u16 *IO_address, u16 *TC_address)**

Purpose: Read the physical I/O address assigned by O.S..

Parameters:

Input:

Name	Type	Description
CardID	u8	assigned by DIP/ROTARY SW

Output:

Name	Type	Description
IO_address	u16	physical I/O address assigned by OS
TC_address	u16	physical timer/counter I/O address assigned by OS

10.2 Input/Output function

For the easy use of digital input / output or the signal and control input / output, the logic polarity configure as you need will release the complexity of your application. Use

LSI3188_port_polarity_set ( ) to set digital input/output port logic polarity.

LSI3188_port_polarity_read ( ) to read back the digital input/output port logic polarity.

To eliminate the input noise, debounce filter is a good solution. LSI3188 card provides software input debounce circuit, before using the digital input, selecting an adequate filter frequency by:

LSI3188_debounce_time_set( ) and read back setting by

LSI3188_debounce_time_read( ).

To output data

LSI3188_port_set( ) will do.

To read digital input /output status

LSI3188_port_read ( ) will do.

To set a dedicate digital output, use

LSI3188_point_set( ) and to read back the digital input/output status by

LSI3188_point_read( )

- **LSI3188_port_polarity_set**

Format : **u32 status = LSI3188_port_polarity_set (u8 CardID, u8 port, u8 polarity)**

Purpose: To set LSI3188 card's digital I/O port polarity.

Parameters:

Input:

Name	Type	Description
CardID	u8	assigned by DIP/ROTARY SW
port	u8	0: input port 1: output port
polarity	u8	b7: IN07 for input port,OUT07 for output port 0: normal polarity (default) 1: inverse polarity …. b0: IN00for input port,OUT00 for output port 0: normal polarity (default) 1: inverse polarity

- **LSI3188_port_polarity_read**

Format : **u32 status = LSI3188_port_polarity_read (u8 CardID, u8 port, u8 *polarity)**

Purpose: To read back polarity of digital I/O port point.

Parameters:

Input:

Name	Type	Description
CardID	u8	assigned by DIP/ROTARY SW
port	u8	0: input port 1: output port

Output:

Name	Type	Description
polarity	u8	b7: IN07 for input port,OUT07 for output port 0: normal polarity (default) 1: inverse polarity …. b0: IN00for input port,OUT00 for output port 0: normal polarity (default) 1: inverse polarity

- **LSI3188_debounce_time_set**

Format : **u32 status = LSI3188_debounce_time_set (u8 CardID, u8 debounce_time)**

Purpose: Set the input port debounce time

Parameters:

Input:

Name	Type	Description
CardID	u8	assigned by Rotary SW
debounce_time	u8	Debounce time selection: 0: no debounce 1: debounce frequency 100 Hz, filter out duration less than 10ms (default) 2: debounce frequency 200 Hz, filter out duration less than 5ms 3: debounce frequency 1K Hz, filter out duration less than 1ms

- **LSI3188_debounce_time_read**

Format : **u32 status = LSI3188_debounce_time_read (u8 CardID, u8 * debounce_time)**

Purpose: Read back the input port debounce time configuration

Parameters:

Input:

Name	Type	Description
CardID	u8	assigned by Rotary SW

Output:

Name	Type	Description
debounce_time	u8	Debounce time selection: 0: no debounce 1: debounce frequency 100 Hz, filter out duration less than 10ms (default) 2: debounce frequency 200 Hz, filter out duration less than 5ms 3: debounce frequency 1K Hz, filter out duration less than 1ms

- **LSI3188_port_set**

Format : **u32 status = LSI3188_port_set (u8 CardID, u8 port, u8 data)**

Purpose: To set LSI3188 card's DIO output.

Parameters:

Input:

Name	Type	Description
CardID	u8	assigned by DIP/ROTARY SW
port	u8	0: invalid 1: output port
data	u8	b7: OUT07 for output port …. b0: OUT00 for output port

- **LSI3188_port_read**

Format : **u32 status = LSI3188_port_read (u8 CardID, u8 port, u8 *data)**

Purpose: To read LSI3188 card's DIO port status.

Parameters:

Input:

Name	Type	Description
CardID	u8	assigned by DIP/ROTARY SW
port	u8	0: input port 1: output port

Output:

Name	Type	Description
data	u8	b7: state of IN07 for input port or OUT07 for output port …. b0: state of IN00 for input port or OUT00 for output port

- **LSI3188_point_set**

Format : **u32 status = LSI3188_point_set (u8 CardID, u8 port, u8 point, u8 state)**

Purpose: To set LSI3188 card’s digital input/output point.

Parameters:

Input:

Name	Type	Description
CardID	u8	assigned by DIP/ROTARY SW
port	u8	0: invalid 1: output port
point	u8	Point designated 7~0 for OUT07~OUT00
state	u8	Data (0 or 1) will set the designated pint

- **LSI3188_point_read**

Format : **u32 status = LSI3188_point_read (u8 CardID, u8 port, u8 point, u8 *state)**

Purpose: To read LSI3188 card’s digital input/output point status.

Parameters:

Input:

Name	Type	Description
CardID	u8	assigned by DIP/ROTARY SW
port	u8	0: input port 1: output port
point	u8	Point designated 7~0 for b7~b0

Output:

Name	Type	Description
state	u8	Returned status (0 or 1) of the designed bit

10.3 Timer function

The build in 32 bit timer based on 1 us time base can be used as system clock to generate interrupt for periodical task.

To setup timer or change time constant

LSI3188_timer_set( ) and start by

LSI3188_timer_start( ) and stop by

LSI3188_timer_stop( )

If you want to dedicated control the timer associated registers, use

LSI3188_TC_set( ) to set registers and use

LSI3188_TC_read( ) to read back settings.

- **LSI3188_timer_set**

Format : **u32 status = LSI3188_timer_set (u8 CardID, u32 time_constant)**

Purpose: To setup timer operation mode or update timer

Parameters:

Input:

Name	Type	Description
CardID	u8	assigned by DIP/ROTARY switch
time_constant	u32	Timer constant based on 1us clock

Note:

1. Time constant is based on 1us clock, period T= (time_constant +1) * 1us
2. If you also enable the timer interrupt, the period T must at least longer than the system interrupt response time else the system will be hanged by excess interrupts.

- **LSI3188_timer_start**

Format : **u32 status = LSI3188_timer_start (u8 CardID)**

Purpose: To start timer operation mode

Parameters:

Input:

Name	Type	Description
CardID	u8	assigned by DIP/ROTARY switch

- **LSI3188_timer_stop**

Format : **u32 status = LSI3188_timer_stop (u8 CardID)**

Purpose: To stop timer operation mode

Parameters:

Input:

Name	Type	Description
CardID	u8	assigned by DIP/ROTARY switch

- **LSI3188_TC_set**

Format : **u32 status=LSI3188_TC_set (u8 CardID, u8 index, u32 data)**

Purpose: To load data to timer related registers

Parameters:

Input:

Name	Type	Description
CardID	u8	assigned by DIP/ROTARY SW
index	u8	0: TC_CONTROL 1: PRELOAD 2: TIMER
data	u32	For TC_CONTROL 0: stop timer operation 1: timer run For PRELOAD or TIMER Data is the constant to be load

Note: PRELOAD is the register for timer to re-load, the value will be valid while timer count to zero and reload the data.

- **LSI3188_TC_read**

Format : **u32 status=LSI3188_TC_read (u8 CardID, u8 index, u32 *data)**

Purpose: To read data from timer related registers

Parameters:

Input:

Name	Type	Description
CardID	u8	assigned by DIP/ROTARY SW
index	u8	0: TC_CONTROL 1: PRELOAD 2: TIMER

Output:

Name	Type	Description
data	u32	Data read back

Note: Meaning of setting or return value of different index

index	register	value	meaning
0	TC_CONTROL	0~1	0:timer stops operation 1: timer runs
1	PRELOAD	1~0xffffffff (1~ 4294967295)	timer preload value
2	TIMER	1~0xffffffff (1~ 4294967295)	Timer value on the fly

Note:

For example, you want to watch the timer counting on the fly, use
LSI3188_TC_read (CardID,index, *data)   //CardID as you assign, index=2
to read back the timer value.

10.4 Interrupt function

There are 3 interrupt sources for your quick response application,

Digital input: IN07~IN00 generate interrupt
Timer: time up interrupt
Counter: compare equal
can generate interrupt.

To use the interrupt service, the first step
LSI3188_IRQ_mask_set ( ) to mask off the undesired interrupt source.
LSI3188_IRQ_mask_read( ) to read back the mask.
After the mask set, you can link your service routine to interrupt by:
LSI3188_IRQ_process_link( ), then enable or disable by:
LSI3188_IRQ_enable( ) to enable, or
LSI3188_IRQ_disable( ) to disable the function.
If you want to check the interrupt status to identify which is the interrupt source,
LSI3188_IRQ_status_read( ) will do and it also clears the interrupt status.

- **LSI3188_IRQ_mask_set**

Format : **u32 status = LSI3188_IRQ_mask_set (u8 CardID,u8 source, u8 mask)**

Purpose: Mask off interrupt source of port0 IN07~IN00 or timer,counter

Parameters:

Input:

Name	Type	Description
CardID	u8	assigned by Rotary SW
source	u8	0: digital io block 1: timer /counter block 2: trigger output block
mask	u8	**Digital IO block:** Any bit set to 1 of b7~b0 means IN07~IN00 can generate interrupt **Timer /Counter block:** b0=1, enable timer cross zero to generate interrupt, else disable. **Trigger output block:** Any bit set to 1 of b7~b0 means Trigger07~ Trigger00 can generate interrupt

- **LSI3188_IRQ_mask_read**

Format : **u32 status = LSI3188_IRQ_mask_read (u8 CardID,u8 source,u8 *mask)**

Purpose: read back interrupt mask of port0 b7~b0 or timer/counter

Parameters:

Input:

Name	Type	Description
CardID	u8	assigned by Rotary SW
source	u8	0: digital io block 1: timer/counter block 2: trigger output block

Output:

Name	Type	Description
mask	u8	Digital block: Any bit set to 1 of b7~b0 means IN07~IN00 can generate interrupt **Timer /Counter block:** b0=1, enable timer cross zero to generate interrupt, else disable. **Trigger output block:** Any bit set to 1 of b7~b0 means Trigger07~Trigger00 can generate interrupt

- **LSI3188_IRQ_process_link**

Format : **u32 status = LSI3188_IRQ_process_link (u8 CardID,**
void (__stdcall *callbackAddr)(u8 CardID))

Purpose: Link irq service routine to driver

Parameters:

Input:

Name	Type	Description
CardID	u8	assigned by Rotary SW
callbackAddr	void	callback address of service routine

- **LSI3188_IRQ_enable**

Format : **u32 status = LSI3188_IRQ_enable (u8 CardID, HANDLE *phEvent)**

Purpose: Enable interrupt from selected source

Parameters:

Input:

Name	Type	Description
CardID	u8	assigned by Rotary SW

Output:

Name	Type	Description
phEvent	HANDLE	event handle

- **LSI3188_IRQ_disable**

Format : **u32 status = LSI3188_IRQ_disable (u8 CardID)**

Purpose: Disable interrupt from selected source

Parameters:

Input:

Name	Type	Description
CardID	u8	assigned by Rotary SW

- **LSI3188_IRQ_status_read**

Format : **u32 status = LSI3188_IRQ_status_read (u8 CardID,u8 source,**
u8 *Event_Status)

Purpose: To read back the interrupt status to identify the source

Parameters:

Input:

Name	Type	Description
CardID	u8	assigned by Rotary SW
source	u8	0: digital io block 1: timer block 2: trigger output block

Output:

Name	Type	Description
Event_Status	u8	Digital block: Any bit set to 1 of b7~b0 means port0 IN07~IN00 has generated interrupt **Timer /counter block:** b0:S_TIMER Timer cross 0 will set S_TIMER flag **Trigger output block:** Any bit set to 1 of b7~b0 means Trigger07~ Trigger 00 has generated interrupt

Note:

1. Status read back will also clear the on board status register.
2. The status will reflect the on board digital input or timer count up status are irrelevant to the IRQ_MASK