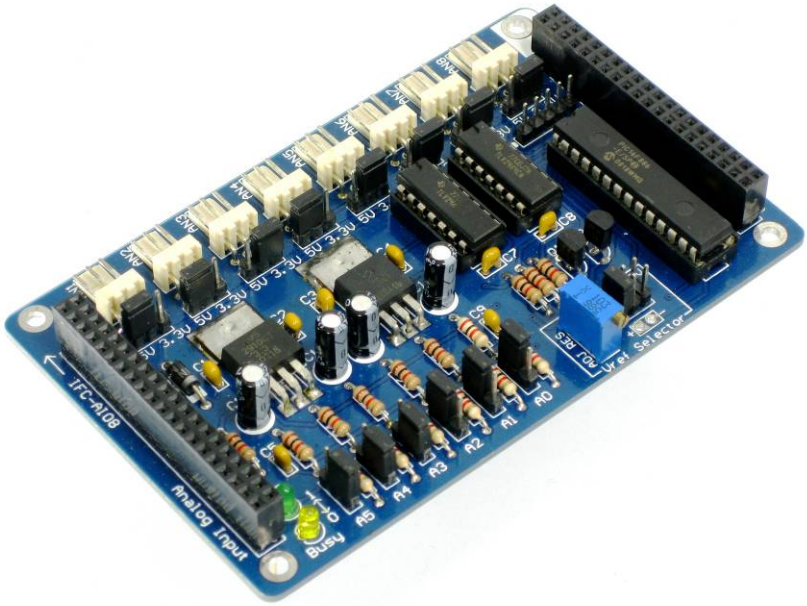
IFC-AI08
Interface Free Controller
Analog Input Card



Card Library Functions

V1.0

Sept 2008

Function Prototype for Analog Input (AI08)

This document explains the functions offered to control the analog input card (AI08). The function will be called in the main program which will be loaded into the main board card (MB00). Beside this document, the explanation of each function will also be found in the AI08 slave card header file (iic_ai.h). Table 1 shows the functions for AI08.

Function Prototype	Remarks	Parameter Description
void ai_sampling_conf(unsigned char **add**, unsigned char **an_sel**, unsigned short **average**);	ai_sampling_conf(**add_ai1**,**5**,**500**)	**add** = card address **an_sel** = select which port needed to configure an_sel => **1** --- AN1 an_sel => **2** --- AN2 an_sel => **3** --- AN3 an_sel => **4** --- AN4 an_sel => **5** --- AN5 an_sel => **6** --- AN6 an_sel => **7** -- AN7 an_sel => **8** -- AN8 **Average** = **1-65535** Decide the period for the analog read result to refresh. Refresh rate = average*10ms 1(default) = fastest analog read 65535 = slowest analog read.
void ai_bit_conf(unsigned char **add**, unsigned char **bit_num**);	ai_bit_conf(**add_ai1**, **0b00001111**);	**add** = card address **bit_num** = 1 byte = 8 bits (each bit represent an analog input port). To configure each port either to convert the analog input to 8 bits or 10 bits digital value **0** = analog value is convert into 8 bit digital value. **1** = analog value is convert into 10 bit digital value.
unsigned short ai_read(unsigned char **add**, unsigned char **an_sel**);	ai_read(**add_ai1**,**5**)	**add** = card address **an_sel** = select which ADC port to read. an_sel => **1** --- AN1 an_sel => **2** --- AN2 an_sel => **3** --- AN3 an_sel => **4** --- AN4 an_sel => **5** --- AN5 an_sel => **6** --- AN6 an_sel => **7** --- AN7 an_sel => **8** --- AN8

unsigned char ai_cmphl(unsigned char **add**, unsigned char **selection**, unsigned char **highest_lowest**);	ai_cmphl(**add_ai1**,**0b11111111**,**1**)	**add** = card address **selection** = 1 byte = 8 bits (Each bit represents an analog input port). To select which port ready to be compared. **0** = not activated for comparison **1** = activated for comparison **highest_lowest** = to compare and select which port is highest or lowest. highest_lowest => **1** (compare for highest reading) highest_lowest => **0** (compare for lowest reading)
unsigned short ai_vref_read(unsigned char **add**)	ai_vref_read(**add_ai1**)	**add** = card address
unsigned char ai_cmp2(unsigned char **add**, unsigned char **selection1**, unsigned char **selection2**);	ai_cmp2(**add_ai1**,**4**,**8**)	**add** = card address **selection1** and **selection2** are for user to select which port to compare. Selection1 => **1** --- AN1 Selection1 => **2** --- AN2 Selection1 => **3** --- AN3 Selection1 => **4** --- AN4 Selection1 => **5** --- AN5 Selection1 => **6** --- AN6 Selection1 => **7** --- AN7 Selection1 => **8** --- AN8 Selection2 => **1** --- AN1 Selection2 => **2** --- AN2 Selection2 => **3** --- AN3 Selection2 => **4** --- AN4 Selection2 => **5** --- AN5 Selection2 => **6** --- AN6 Selection2 => **7** --- AN7 Selection2 => **8** --- AN8 if selection1 > selection2 = 1 if selection1 < selection2 = 0 if selection1 = selection2 = 2

Table 1 Function Prototype for Analog Input (AI08)

IFC-AI08 is fully compatible in IFC series which it can be stacked on IFC system as one of the input card. Functions are provided for user to conveniently control the AI08 card through MB00 main board card. This section will give examples on how to use this function in main program.

void ai_sampling_conf(unsigned char **add**, unsigned char **an_sel**, unsigned short **average**);

This function is used to configure sampling rate for each port. Every 10ms, the AI08 card will refresh the conversion stalling register for every analog port with new conversion result not matter whether the port has connected to an input device or not. This function is used to determine after how many conversions (once per 10ms) for particular port, the conversion stalling register will refresh to the average result from the **average** times of conversion. **an_sel** is used to select which port is going to configure (1-8). **average** is a value needed to decide the period for the conversion stalling register to refresh with the average value. **average** is range from 1 - 65,535. Refresh rate = **average** x 10ms. Example below shows that for analog port 1, the conversion stalling register will only refresh a new value (average value of 2000 conversion) after 2000*10 milliseconds (20 second). The default value for **average** is 1 for every port if this function do not called.

```
ai_sampling_conf(ai_add,1,2000);
```

void ai_bit_conf(unsigned char **add**, unsigned char **bit_num**)

This function is used for user to configure the conversion of each analog input port either to 8 bit digital value or 10 bit digital value. **bit_num** is one byte and can be divided into 8 bits variable which each bit represent every analog input port. User can configure each bit of bit_num either 0 or 1. If 0 is set mean the analog value is converted into 8 bit digital value and if 1 is set mean the analog value is converted into 10 bit digital value. Figure below show example of bit_num represent each analog input port.

Port 8	**Port 7**	**Port 6**	**Port 5**	**Port 4**	**Port 3**	**Port 2**	**Port 1**
x	x	x	x	x	x	x	x
bit 7 (MSB)							bit0(LSB)

bit 7-0 **x** : convert each analog input port either to 8 bit digital value or 10 bit digital value.

1 = analog value is converted into 10bit digital value.
0 = analog value is converted into 8bit digital value

Below is an example showing the method to configure analog input port. From the example below**, bit_num** is set to a number in binary form 0b00001111.

```
ai_bit_conf(ai_add, 0b00001111);
```

From the example above Port1, Port2, Port3 and Port4 are set to 10 bits analog to digital conversion meanwhile Port8, Port7 Port6 and Port5 are set to 8 bits analog to digital conversion.

unsigned short ai_read(unsigned char **add**, unsigned char **an_sel**)

This function is used to read the conversion stalling register which stall the final conversion result. They are 8 analog input ports can be read. **an_sel** is for user to select which port to read(1-8). User can refer table 1 above to select **an_sel.** The example below shows the sample code to read input 8. A 8 bits data (conversion result) will be returned by calling the function if the port is set to 8bits digital conversion meanwhile a 16 bits data (conversion result) will be return with the 6 MSB don't care if the port is set to 10bits digital conversion.

```
ai_read(ai_add,8);
```

unsigned char ai_cmphl(unsigned char **add**, unsigned char **selection**, unsigned char **highest_lowest**)

This function is used to compare the selected analog port to figure out which port has the highest or lowest value conversion. **selection** is a one byte and can be divided into 8 bits variable and each bit represents every analog port. If a particular bit is set to 1 means the representative port is ready to be compared and vice versa. **highest_lowest** is for user to set either to figure out the highest conversion value port (set to 1) or the lowest conversion value port (clear to 0). Below is example to call this function in main program.

```
ai_cmphl(add_ai1,0b00001111,1);
```

From the example above, **selection** is set to 0b00001111, means user want to compare port 1-4 only. User can use decimal number instead of binary for **selection**. For example, user can replace 0b00001111 to 15 for **selection**. **highest_lowest** = 1 means user wants to compare and figure out the highest value among selected analog input port. The function will return 1 byte to represent the comparison result. Figure below show CP04 control panel display return value.

Return value is 00000100. Means port 3(AN3) is the highest value among all 4 ports.

Make sure the selected ports ready to be compared are in the same ADC resolution (8 bits or 10 bits).

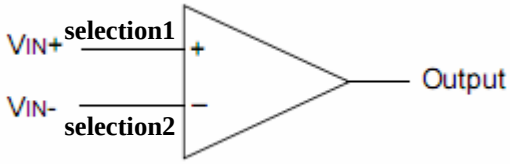
unsigned short ai_vref_read(unsigned char **add**)

This function offers user another method to know the adjusted Vref value through programming besides using multimeter. This function will return a 10bits (0-1023) value and need a CP04 card to display the value. The voltage of V_{ref} can be calculated through following equation:

$$V_{ref} = \frac{\text{Value display on LCD}}{1023} \text{ X } 5V$$

unsigned char ai_cmp2(unsigned char **add**, unsigned char **selection1**, unsigned char **selection2**)

This function offers a more simple way to compare 2 analog input ports. A single comparator is shown in figure below along with the relationship between the analog input levels and the digital output.



When the analog voltage at V_{IN+} is less than the analog voltage at V_{IN-}, the output of the comparator is a digital low level (0). When the analog voltage at V_{IN+} is greater than the analog voltage at V_{IN-}, the output of the comparator is a digital high level (1). For this function selection1 is defined as V_{IN+} and selection2 is defined as V_{IN-}. So, If selection1 > selection2, value 1 will be returned from this function, if selection1 < selection2, value 0 will be return from this function. But if selection1 = selection2, value 2 will be return from this function. **selection1** and **selection2** are for user to select which port to compare. Below is a

sample of source code on how to call this function. From example below, user call this function to compare the digital value from port 4 (selection1) and port 8 (selection2).

```
ai_cmp2(ai_add,4,8);
```

Make sure the selected ports ready to be compared are in the same ADC resolution (8 bits or 10 bits).

Note: User is reminded to add header file (iic.h and iic_ai.h) and object file (iic.o and iic_ai.o) for IFC-MB00 and IFC-AI08 each time open a new project for IFC. User also need to include card h file at the beginning of the program. Please refer sample source cod e for the example to include card h file.

Prepared by
Cytron Technologies Sdn. Bhd.
19, Jalan Kebudayaan 1A,
Taman Universiti,
81300 Skudai,
Johor, Malaysia.

Tel: *+607-521 3178*
Fax: *+607-521 1861*

URL: *www.cytron.com.my*
Email: *support@cytron.com.my*
sales@cytron.com.my