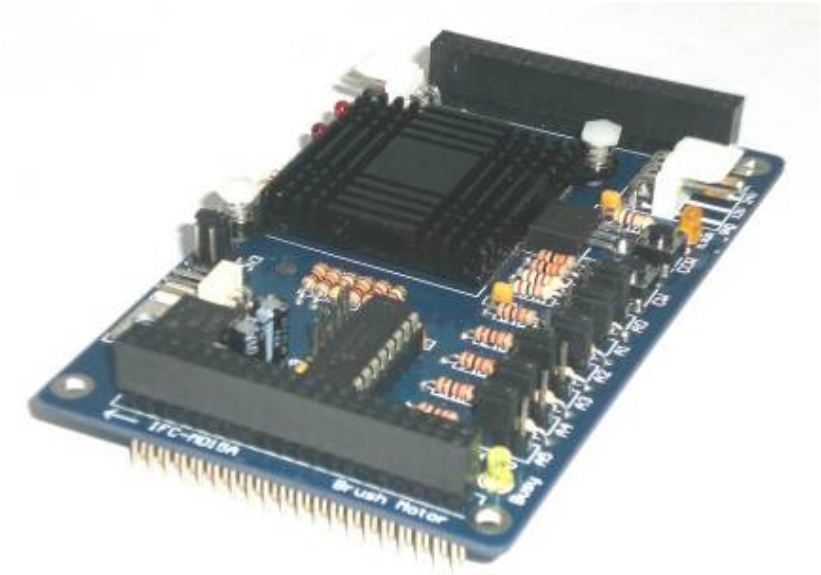
IFC-MD15A
Interface Free Controller
Brush Motor Card



Card Library Functions for Visual C# Express and Visual Basic Express

V1.0

Apr 2009

Function Prototype for Brush motor (MD15A)

This document explains the function prototype for controlling IFC-MD15A using PC through IFC-CI00. User may also use 'object browser' under Microsoft Visual C# to view the summary, parameter and return value description of IFC-MD15A function prototype. User need to add reference 'ifc_ci.dll' and 'ifc_md.dll' for IFC-CI00 and IFC-MD15A card in order to control/communicate IFC-MD15A using PC. Please note that before user start the programming, user need to initialize the 'ifc.ifc_ci' and 'ifc.ifc_md' in order to use the functions to control IFC-MD15A. Example of creating a 'ifc.ifc_ci' class called 'ifc1' and 'ifc.ifc_md' class called md1:

```
static ifc.ifc_ci ifc1 = new ifc.ifc_ci(74);
ifc.ifc_md md1 = new ifc.ifc_md(ifc1, 4);
```

For 'ifc.ifc_ci' class, user need to specified the COM Port that is connected to IFC-CI00 and for 'ifc.ifc_md' class, user need to specified the IFC-CI00 in use and also the address for IFC-MD15A. Please make sure that the address must be unique and different with other IFC card in the IFC system.

Function Prototype	Example	Summary	Parameter Description	Return Value
ifc_md(ifc.ifc_ci *ifc_ci*, int *address*)	**ifc.ifc_md**(ifc1, 5)	Initializes a new instance of the ifc.ifc_md class using the specified ifc.ifc_ci and address for IFC-MD15A.	*ifc_ci*: ifc.ifc_ci in use. *address*: Address for IFC-MD15A, in range of 0 to 63. (int)	
ifc_md(ifc.ifc_ci *ifc_ci*, byte *address*)	**ifc.ifc_md**(ifc1, 5)	Initializes a new instance of the ifc.ifc_md class using the specified ifc.ifc_ci and address for IFC-MD15A.	*ifc_ci*: ifc.ifc_ci in use. *address*: Address for IFC-MD15A, in range of 0 to 63. (byte)	
void md_alcon(bool *autoreset*, int *response*)	**md1.md_alcon**(true, 820)	To configure internal alarm of IFC-MD15A.	*autoreset*: True to enable autoreset feature and false to disable autoreset feature of IFC-MD15A. Autoreset feature is enabled by default. (bool) *response*: Response value proportional to the acceleration after autoreset occur. Default value of response is 820. (int)	

void md_alcon(int *autoreset*, int *response*)	**md1.md_alcon**(1, 820)	To configure internal alarm of IFC-MD15A.	*autoreset*: 1 to enable autoreset feature and 0 to disable autoreset feature of IFC-MD15A. Autoreset feature is enabled by default. (int) *response*: Response value proportional to the acceleration after autoreset occur. Default value of response is 820. (int)	
void md_alrst()	**md1.md_alrst**()	To manually reset the alarm.		
md_alstat()	**md1.md_alstat**()	To get the alarm status of IFC-MD15A.		True if alarm occurred and false otherwise. (bool)
void md_brake()	**md1.md_brake**()	To brake Motor.		
void md_ccw()	**md1.md_ccw**()	To change the Motor direction to counter-clockwise.		
void md_cw()	**md1.md_cw**()	To change the Motor direction to clockwise.		
void md_enclr()	**md1.md_enclr**()	To clear Encoder value.		
md_enstat()	**md1.md_enstat**()	To read Encoder stand alone encoding status.		Return true if Encoder is in progress. (bool)
md_enval()	**md1.md_enval**()	To read Encoder value.		Encoder value in 16-bit. (int)

md_runstat()	**md1.md_runstat**()	To read Motor run status.		Return false if Motor is stopped or braked. (bool)
void md_speed(int *pwm*)	**md1.md_speed**(180)	To change the speed of Motor.	*pwm*: Speed value of Motor in range of 0 to 255. (int)	
void md_speed(byte *pwm*)	**md1.md_speed**(255)	To change the speed of Motor.	*pwm*: Speed value of Motor in range of 0 to 255. (byte)	
md_spval()	**md1.md_spval**()	To read Motor speed value.		To read Motor speed value.
void md_stop()	**md1.md_stop**()	To stop Motor.		
md_encon	**md1.md_encon**	To enable stand alone encoding process for Encoder.		

Table 1 Function Prototype for MD15A

Table 2 is function prototype for sub function of **md_encon**'. These functions are to enable stand alone encoding process for Encoder.

Function Prototype	Example	Summary	Parameter Description
void brake(int *enc_data*)	**md1.md_encon.brake**(1500)	Motor brake after targeted encoder value is reached.	Motor brake after targeted encoder value is reached.
void ccw(int *enc_data*, int *act_val*)	**md1.md_encon.ccw**(2000, 100)	Motor change direction to counter-clockwise after targeted encoder value is reached.	*enc_data*: Targeted encoder value in range of 0 to 65535. (int) *act_val*: Motor speed after targeted encoder value is reached. (int)
void cw(int *enc_data*, int *act_val*)	**md1.md_encon.cw**(1500, 150)	Motor change direction to clockwise after targeted encoder value is reached.	*enc_data*: Targeted encoder value in range of 0 to 65535. (int) *act_val*: Motor speed after targeted encoder value is reached. (int)
void none(int *enc_data*)	**md1.md_encon.none**(2000)	No action after targeted encoder value is reached.	*enc_data*: Targeted encoder value in range of 0 to 65535. (int)
void speed(int *enc_data*, int *act_val*)	**md1.md_encon.speed**(2000, 150)	Motor change speed after targeted encoder value is reached.	*enc_data*: Targeted encoder value in range of 0 to 65535. (int) *act_val*: Motor speed after targeted encoder value is reached. (int)
void stop(int *enc_data*)	**md1.md_encon.stop**(2000)	Motor stop after targeted encoder value is reached.	*enc_data*: Targeted encoder value in range of 0 to 65535. (int)

Table 2

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