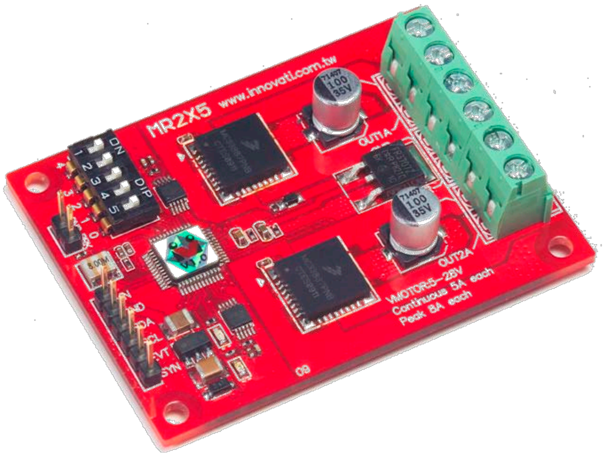
Innovati's MR 2x5

Dual DC Motor Controller Module

Version: V1.0



Product Overview: Innovati's MR 2x5 Module can reach the goal of freely controlling two DC motors simultaneously through simple commands. It allows the user to change the rotation speed of the motor any time and acquire the current status of the motor, including the rotation and the direction. The user can also obtain the value of the electric current through the commands.

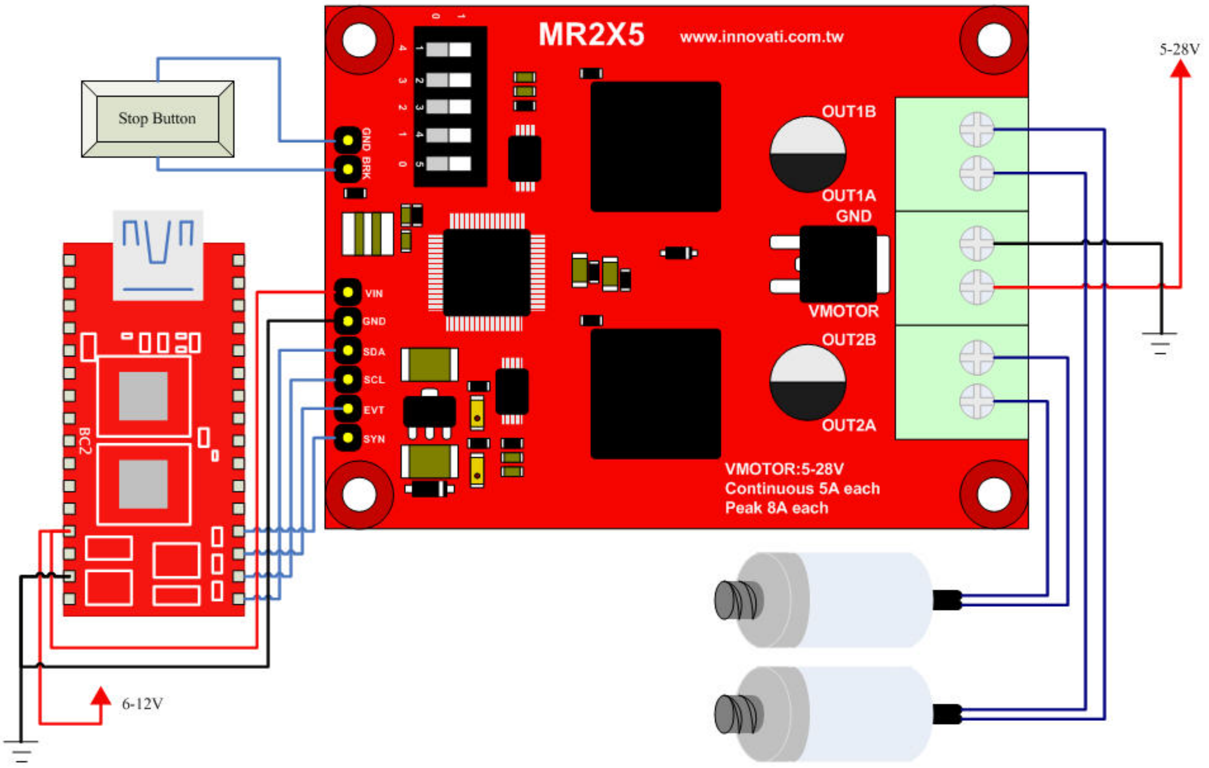
Applications:

- Control the driving of the motor so as to configure the forward or backward movement of the model vehicle. With the speed difference setting, the turning of the vehicle's moving direction can be performed.
- It allows the user to control the motors for different rotation directions so as to control the forward/backward and left/right movements, such as those in the control of robotic arms.
- It can be connected with a small fan for controlling the wind strength.
- It can monitor the electric current of the motor and allow the user to configure over-current protection to stop motor operation.

Product Features:

- It allows the user to configure the execution of commands on two motors at the same time.
- It can provide a continuous current within the range of ±5A.(Peak current up to ±7.8A)
- The maximum allowable input voltage for normal operation can be up to 28 V.(Up to 40V for the efficiency reduction condition)
- The maximum internal PWM current control frequency can be up to 8k Hz.
- Provides automatic shut down protection against overheat (~175°C).
- Provides protection against current overload.
- With the brake command, it can rapidly stop the motion of the motor.
- Provides up to 256 levels of rotation speed settings.
- Through the commands, it allows two motors to operate at different speeds in different directions at the same time.
- By using the commands, it is easy to obtain the current status of the rotation speed or direction of the motors.
- Provides commands for reading the value of the electric current and allows the user to configure the mechanism for stopping the motor under over-current conditions.

Connection: Place the ID switch directly on the required number, and then connect the cmdBUS to the corresponding pins on the BASIC Commander so that the user can perform the required operations through the BASIC Commander. According to the pin connection shown in the following figure, connect the motors to be controlled to the corresponding input pins OUT1AB and OUT2AB. And then connect the pins VMOTOR and GND on the right hand side of the module shown in the following figure to the power supply required for the motors respectively. During the connection, please notice the polarity of the voltage. The figure shows a possible connection of the stop button on the left hand side. If an external stop button is required, the user can connect a button according to the figure with one terminal connected to the input pin for the stop button and the other terminal connected to the ground.



Product Specifications:

These pins are dedicated for connecting the stop button. Connect the pin BRK to an external button and then connect the other terminal of the button to the ground. When the button is pressed, the motor operation is stopped.

Module ID Setting Switch: The module ID of the LCD module can be configured with the binary digits from the right to the left. This ID number allows the BASIC Commander to determine the module to be controlled during the operation.

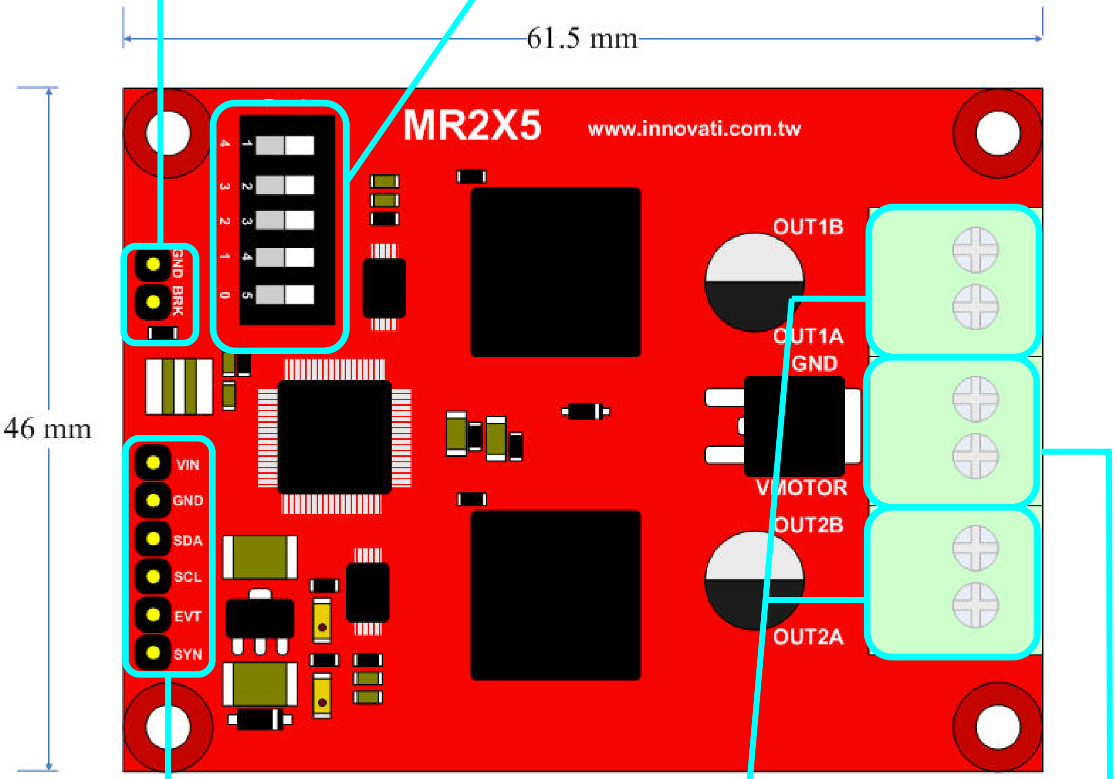


Figure 1 Description of pins and switches on the module

Pins for cmdBUS: Connect these pins to the corresponding pins on the BASIC Commander for controlling the Motor module through the BASIC Commander. While connecting, please notice the pin assignment. Connect Vin to the Vin on the BASIC Commander. Incorrect pin connection may cause damage to the module.)

Motor connection pins. Please connect OUT1A and OUT1B to the first motor to be controlled; connect OUT2A and OUT2B to the second motor to be controlled.

Power supply connection pins for the motor controller module. Please connect the power supply wires for the motor to these pins.

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
I_{DD}	Operating Current	7.5	No Load	—	14.7	—	mA
f_{pwm}	PWM Output frequency	—	—	—	8K	—	Hz

Table 1 Characteristics of the operating current (room temperature at 25 °C)

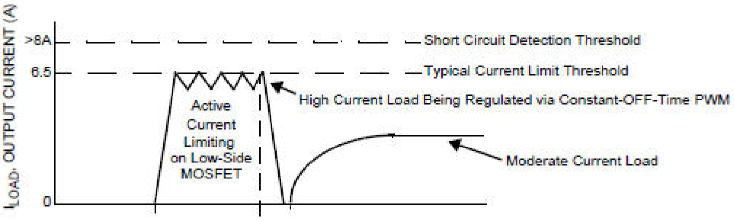
Test Condition: T_A=25℃, V_M=30V

Characteristic	Symbol	Limits			
		Min.	Typ.	Max.	Units
Operating Voltage Range	V_+	5	-	28	V
Output ON-Resistance 5.0 V ≦ V_+ ≦ 28 V, T_J = 25℃	$R_{DS(ON)}$	-	120	-	mΩ
8.0 V ≦ V_+ ≦ 28 V, T_J = 150℃		-	-	225	
5.0 V ≦ V_+ ≦ 8 V, T_J = 150℃		-	-	300	
Active Current Limiting Threshold (via Internal Constant OFF-Time PWM) on Low-Side MOSFETs	I_{LIM}	5.2	6.5	7.8	A
High-Side Short Circuit Detection Threshold	I_{SCH}	11	-	-	A
Low-Side Short Circuit Detection Threshold	I_{SCL}	8.0	-	-	A
Leaking Current V_{OUT} = V_+	$I_{OUT(LEAK)}$	-	100	200	μA
V_{OUT} = Ground		-	30	60	
Output MOSFET Body Diode Forward Voltage Drop I_{OUT} = 3.0 A	V_F	-	-	2.0	V
Thermal Shutdown Temp.	T_{LIM}	175	-	225	℃
Thermal Shutdown Hysteresis.	T_{HYS}	10	-	30	℃

Table 2 Electrical Characteristics of a Motor

Overheat Protection: The overheat protection circuit is used for automatically breaking the circuit when the internal temperature up to 175°C inside the drive IC is detected. At this time, the motor operation will be stopped. When the temperature is decreased by about 15°C, the protection circuits will automatically recover the circuit connection and the motor will continue its previous operation.

Current Limit Protection: Please refer to the following figure. When the H-bridge starts output, the current will increase as the motor rotates faster. When the current value exceeds 6.5A (as the Current Limit Threshold indicated in the following figure), the H-bridge output will be stopped. After the next clock of the internal oscillator is sent out, the current transmission will be continued. In such a way, the operation is repeatedly limited within the range as shown in the figure.



Precautions for Operations:

The Motor Module provides the connection pins for two motors separately. Please ensure that

the connected motors are DC motors.
Operating Temperature of the Module: 0℃ ~ 70℃ (Please confirm the operating temperature for the motor separately)
Storage Temperature of the Module: -50℃~125℃

List of Commands:

The following list shows the commands dedicated to controlling the MR 2x5 module. The command name and parameters which should be input are shown in bold or bold-italic typefaces. The words in bold typeface should not be changed while being input. The words in bold-italic typefaces can be filled with parameters in properly defined format by the user. Please note that the words in uppercase or lowercase are regarded as the same word while entering the program in the innoBASIC Workshop.

Before executing the program for MR 2x5, please define the corresponding parameters and the module ID at the beginning of the program, for example:

Peripheral *ModuleName* As MR2x5 @ *ModuleID*

Command Format	Command Function
Commands for the acceleration of the motor	
BackwardA(*DutyCycle*)	Command Motor A, Motor B, or Motors A and B to perform the backward rotation operation and set the rotation speed of the motor according to the value specified by ***DutyCycle***. The input value should be an integer in the range of 0~255 (a larger value of ***DutyCycle*** will lead to a faster rotation speed).
BackwardAB(*DutyCycleA, DutyCycleB*)	
BackwardB(*DutyCycle*)	
BackwardDual(*DutyCycleAll*)	
ForwardA(*DutyCycle*)	Command Motor A, Motor B, or Motors A and B to perform the forward rotation operation and set the rotation speed of the motor according to the value specified by ***DutyCycle***. The input value should be an integer in the range of 0~255 (a larger value of ***DutyCycle*** will lead to a faster rotation speed).
ForwardAB(*DutyCycleA, DutyCycleB*)	
ForwardB(*DutyCycle*)	
ForwardDual(*DutyCycleAll*)	
Commands for stopping the motor	
BrakeA()	Stop the motions of Motor A, Motor B, or Motors A and B immediately.
BrakeB()	
BrakeDual()	
StopA()	Stop the motions of Motor A, Motor B, or Motors A and B.
StopB()	
StopDual()	
Commands for parameter setting and status	
GetDCA(*DutyCycle*)	Get the rotation speed of Motor A or Motor B and store it in ***DutyCycle***. (A larger value of ***DutyCycle*** leads to a faster rotation speed.) The retrieved value of ***DutyCycle*** will be an integer in the range of 0~255.
GetDCB(*DutyCycle*)	
GetDCAB(*DutyCycleA, DutyCycleB*)	
GetDirA(*Dir*)	Get the rotation direction of Motor A or Motor Band store it in ***Dir***. (For the value of ***Dir***, 0 represents forward rotation and 1 represents backward rotation.)
GetDirB(*Dir*)	
GetDirAB(*DirA, DirB*)	
SetDCA(*DutyCycle*)	Set the rotation speed of Motor A, Motor B, or Motors A and

SetDCAB(*DutyCycleA, DutyCycleB*)	B with the value of ***DutyCycle***. The input value should be an integer in the range of 0~255. (A larger value of ***DutyCycle*** leads to a faster rotation speed.)
SetDCB(*DutyCycle*)	
SetDCDual(*DutyCycleAll*)	
SetDirA(*Dir*)	Set the rotation direction of Motor A, Motor B, or Motors A and B with the value of ***Dir***. (For the value of ***Dir***, 0 represents forward rotation and 1 represents backward rotation.)
SetDirAB(*DirA, DirB*)	
SetDirB(*Dir*)	
SetDirDual(*DirAll*)	
GetVelA(*DutyCycle*)	Get the rotation speed of Motor A or Motor B with its sign indicating the rotation direction and store it in ***DutyCycle***. A larger absolute value of ***DutyCycle*** leads to a faster rotation. A positive value represents forward rotation and a negative value represents backward rotation. The retrieved value of ***DutyCycle*** will be an integer in the range of -255~255.
GetVelB(*DutyCycle*)	
GetVelAB(*DutyCycleA, DutyCycleB*)	
SetVelA(*DutyCycleA*)	Set the rotation speed of Motor A, Motor B, or Motors A and B with the value of ***DutyCycle***. The input value should be an integer in the range of -255~255. A larger absolute value of ***DutyCycle*** leads to a faster rotation. A positive value represents forward rotation and a negative value represents backward rotation.
SetVelB(*DutyCycleB*)	
SetVelAB(*DutyCycleA, DutyCycleB*)	
SetVelDual(*DutyCycle*)	
GetCurr(*CurrA, CurrB*)	Get the values of the currents of Motors A and B and store them in ***CurrA*** and ***CurrB***, respectively. The retrieved values of ***CurrA*** and ***CurrB*** will be integers in the range of 0~1023. **＊3**
SetCurrMode(*Mode, Time, Curr*)	Set the interrupt mode for the current overload with ***Mode***; set the time required for holding the over current to activate the corresponding mode with ***Time***; and set the limit of the current with ***Curr***. There are four modes available for configuration as follows 0: Disable the monitoring of current 1: Activate the monitoring of current and then activate the notification event when the current exceeds the preset current limit. 2: Activate monitoring of current, activate the notification event when the current exceeds the preset current limit and then perform the Stop operations on both motors. 3: Activate the monitoring of current, activate the notification event when the current exceeds the preset current limit and then perform the Brake operations on both motors. The input value of ***Time*** can be an integer in the range of 0~255 in unit of 100 ms. The input value of ***Curr*** can be an integer in the range of 0~1023.
GetCurrMode(*Mode, Time,*	Get the current monitoring mode and store it in ***Mode***; get

***Curr*)**	the preset time and store it in ***Time***; and get the preset current and store it in ***Curr***. The retrieved value of ***Mode*** will be an integer in the range of 0~3. The retrieved value of ***Time*** will be an integer in the range of 0~255. The retrieved value of ***Curr*** will be an integer in the range of 0~1023.
Command for the stop button ＊1	
ClrBrakeButStatus()	Clear the status of the stop button as 0. After the stop button is pressed, if the status is not cleared, the motor operation cannot be re-started.
***bStatus*=GetBrakeButStatus()**	Get the current status of the stop button. ***bStatus*** 0: The stop button is not pressed 1: The stop button has been pressed but the clear operation has not been performed yet.
Commands for fault status detection ＊2	
EnFaultStop()	Enable the fault-stop mechanism. When a fault occurs, the module will stop the rotation of the motor. In addition, the commands for re-starting the motors cannot be performed before the clear fault status command is performed. The default value is “disable.”
DisFaultStop()	Disable the fault-stop mechanism. When a fault state occurs, the system will automatically clear the fault state and continue the rotation of the motor. But if the fault notification event is activated, it will send the fault notification event. The default value for the fault-stop mechanism is “disable.”
***bStatus*=GetFaultStatus()**	Get the fault status and store it in ***bStatus***. If the fault-stop mechanism is disabled, the fault status will be automatically cleared to 0 and the previously configured motor rotation will be continued. However, if the fault-stop mechanism is enabled, once a fault occurs, the fault status will remains as 1 and the motor rotation operation cannot be performed until the clear fault status command is performed.
ClrFaultStatus()	Clear the fault status for the fault-stop mechanism. If the fault-stop mechanism is enabled, once a fault occurs, it is necessary to perform this command before the motor rotation can be re-started.
EnFaultEvent()	Enable the fault notification event. The default is “enable.”
DisFaultEvent()	Disable the fault notification event. The default value is “disable.”
RestoreStatus()	Restore all parameter settings at the moment when the fault occurs.

＊1 The commands for the stop button are effective only when an external stop button is installed.

＊2 The fault status will generate a related fault notification event only when the rotation speed of the motor is greater than 2.

＊3 The retrieved value is the digitally converted voltage which is built from the trip current of the motor current. The trip current is 1/375 of the total current. The trip current passes through a 220Ω resistor to build the output voltage.

Application Events Provided by the Module:

Event	Activation Condition
FaultEvent	When the motor protection mechanism is activated due to the detection of the fault status such as overheat, and the fault notification event is enabled, this event will be generated.
OverCurrAEvent	When one of the current monitoring modes 1~3 is configured, if the current of Motor A exceeds the preset current limit and maintains the level for a time longer than the preset time limit, the corresponding event is activated.
OverCurrBEvent	When one of the current monitoring modes 1~3 is configured, if the current of Motor B exceeds the preset current limit and maintains the level for a time longer than the preset time limit, the corresponding event is activated.

Demonstration Program:

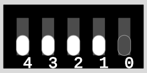
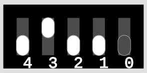
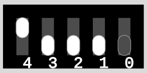
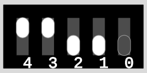
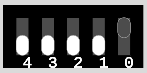
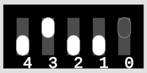
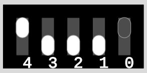
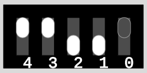
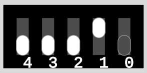
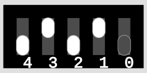
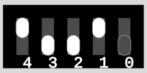
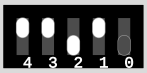
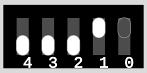
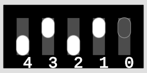
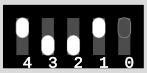
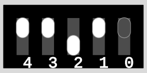
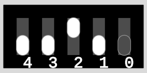
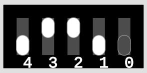
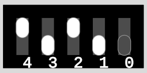
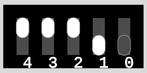
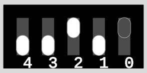
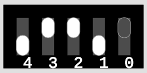
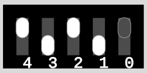
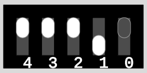
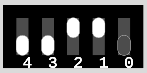
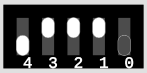
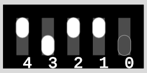
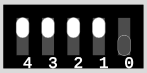
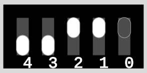
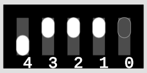
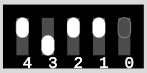
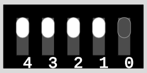
```
Peripheral myMotor As MR2x5 @ 0            '   Set the module ID as 0

Sub Main()
    Debug CLS
    myMotor.ForwardDual(200)                '   Let the two motors rotate forwards at
                                            '   the same time
    Pause 3000
    myMotor.StopDual()                      '   Stop the two motors
    Pause 3000
    myMotor.BackwardDual(200)               '   Let the two motors rotate backwards at
                                            '   the same time
    Pause 3000
    myMotor.SetDirDual(0)                   '   Reverse the rotation direction of the
                                            '   two motors to the forward direction
    Pause 3000
    myMotor.SetDCDual(150)                  '   Change the rotation speed of the motors
                                            '   to 150
    Pause 3000
    myMotor.BrakeDual()                     '   Stop the two motors rapidly
    Pause 3000
End Sub
```

Appendix

1. Known problems:

2. List of the Configuration of the Module ID Switch:

	0		8		16		24
	1		9		17		25
	2		10		18		26
	3		11		19		27
	4		12		20		28
	5		13		21		29
	6		14		22		30
	7		15		23		31