CM02 - Radio Communications Module

Technical Specification

The CM02 module, together with its companion RF04 module, form a complete interface between your PC and our range of I2C modules. You can send commands down to your robot and receive telemetry data back up to the PC. Now you have all the modules necessary to control your robot from your PC. The CM02 is powered from your robots battery, which can be anything from 6v to 12v. The CM02 can monitor battery voltage and report this back to the PC (so your robot knows when its hungry!). There are four I2C connectors on the CM02, but your not limited to four I2C devices. In practice, you may only need one connector and route the I2C bus around all the modules you wish. Bit rate on the I2C bus is approximately 100KHz.

License Exempt Frequency Bands

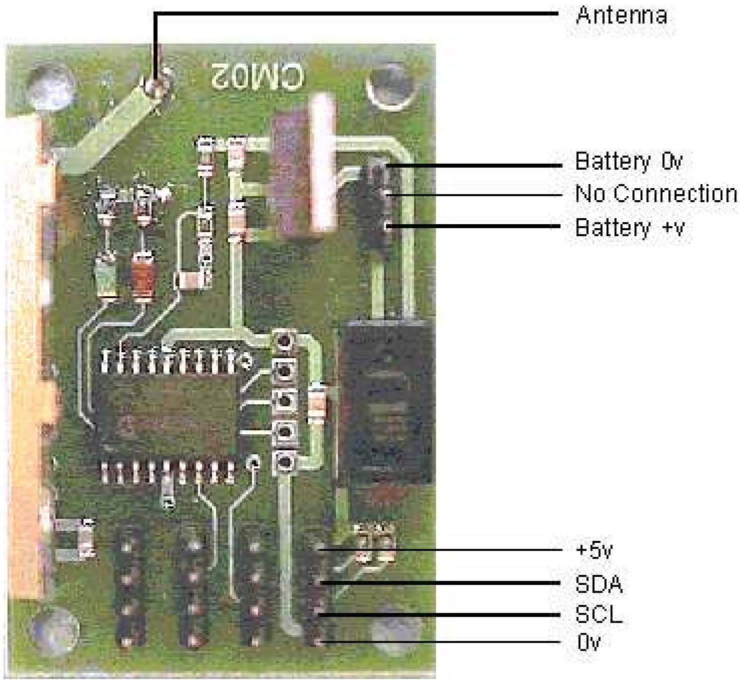
The CM02 is available in two versions. The CM02/400 uses the LPRS EasyRadio ER400TRS module which operates at 433-4MHz. The frequency allocations between 433-4MHz are Pan European and also extend to South Africa, Australia & New Zealand.
The CM02/900 uses the ER900TRS module for EU (868-870MHz) or USA (902-920MHz) markets. Power output up to 10mW. NOTE restrictions in USA to 1mW and 5mW EU.
Be sure to order to check which type may be used in your country, and order matching RF04/CM02 modules, either the RF04/400 and CM02/400 or the RF04/900 and CM02/900.
In 2005, LPRS updated the EasyRadio module, Here are the data sheets for the original ER400TRS and the latest ER400TRS-02.

Power

A 3pin header is used to supply the power for the CM02. A 5v regulator on board supplies the CM02 and other modules. The regulator can supply 1Amp, but in practice this is limited more by the power dissipation, which will be worse, the higher the battery voltage used. Just make sure the heat sink doesn't get too hot! Our test robot, which comprises of CM02, an MD22 motor drive, I2C LCD display, SP03 speech module, CMPS03 compass and eight SRF08's requires 155mA in operation and with a 14v supply the regulator heat sink is just detectably warm. (155mA for the 5v logic - the motors take a lot more, but that doesn't go through the regulator). The slide switch is on the battery side or the regulator. This doesn't switch off the motor power to the MD22. However since the MD22 only takes leakage current when the 5v power is off (we measured 1.5uA) we use it as the robot power switch. If you do that, remember to disconnect the battery before working on your robot. The regulator is a low drop type, so you can use battery voltages of 6v to 12v.

Connections

The I2C connectors are identical and supply 0v, SCL, SDA, and 5v in that order from the outside edge of the PCB. Also the CM02 will require an antenna. At 434MHz, we use a straight piece of solid wire 16.4cm in length as a whip antenna. At 900MHz use an 8cm whip.



Commands
The CM02 responds to commands sent to it over the radio link from the PC. There are just two primary commands:
I2C_CMD (0x55) - This allows you to read or write as much as you wish to anything on the I2C bus - including our own range of modules.
CM02_CMD (0x5A) - A range of commands to the CM02 module, generally to improve selected communications.

Writing directly to any I2C device
Following the I2C_CMD you send the device address, the devices internal register address you want to write to and the number of bytes you're writing. The maximum number of data bytes should not exceed 76 so as not to overflow the CM02's internal buffer.

	Primary CM02 command	Device Address + R/W bit	Device internal register	Number of data bytes	The data bytes
Byte Type	I2C_CMD	Addr+R/W	Reg	Byte Count	Data
Example	0x55	0xE0	0x00	0x01	0x51
Meaning	Primary CM02 command	SRF08 I2C address	SRF08 command Reg	One command byte follows	Start ranging in cm

This 5 byte sequence starts an SRF08 at address 0xE0 ranging. All 5 bytes should be sent to the CM02 in one sequence. A gap will result in the CM02 re-starting its internal command synchronization loop and ignoring the message. After all bytes have been received the CM02 performs the IC2 write operation out to the SRF08 and sends a single byte back to the PC. This returned byte will be 0x00 (zero) if the write command failed and non-zero if the write succeeded. The PC should wait for this byte to be returned (timing out after 500mS) before proceeding with the next transaction.
Here is another write example - this time an 8 byte sequence to initialize the MD22 motor driver:

I2C_CMD	MD22 Addr+R/W	Mode Reg	Data byte count	MD22 mode 1	Left Motor Stopped	Right Motor Stopped	Fast acceleration
0x55	0xB0	0x00	0x04	0x01	0x00	0x00	0x02

Again the CM02 will respond with non-zero if the write succeeded and zero if it failed. A failure means that no acknowledge was received from the I2C device.

Reading directly from any I2C device
This is similar to writing, except that you should add 1 to the device address to make it an odd number. To read from an SRF08 at address 0xE0, you would use 0xE1 as the address. (When the address goes out on the I2C bus, its the 1 in the lowest bit position that indicates a read cycle is happening). The maximum number of data bytes requested should not exceed 76 so as not to overflow the CM02's internal buffer. Here is an example of reading the two byte bearing from the CMPS03 compass module:

I2C_CMD	CPMS03 I2C address + Read bit	CMPS03 bearing register	Number of bytes to read
0x55	0xC1	0x02	0x02

The CM02 will perform the read operation on the I2C bus and send two bytes back to the PC - high byte first. The PC should wait for both bytes to be returned (timing out after 500mS) before proceeding with the next transaction.

CM02 Commands
The CM02 command format is shown below:

CM02_CMD	CM02 Command	Data 1	Data2
0x5A	See below	Command Specific	Command Specific

The CM02 commands are always a four byte sequence. They start with the CM02_CMD primary command which is followed by the CM02 command itself. Two data bytes follow which can be any junk if not used, but they must be included to make up the 4 byte command sequence. These commands are:

Hex	Command	Bytes returned	Purpose
0x01	REVISION	1	Returns the CM02 firmware revision number
0x02	NEW_ADDRESS	1	Changes SRF08 I2C address
0x03	BATTERY	2	High/Low bytes of battery voltage
0x04	SCAN1	6	Send motor data - return battery, compass & sonar data
0x05	SCAN2	9	Same but for 2 SRF08's
0x06	SCAN3	12	3 SRF08's
0x07	SCAN4	15	4
0x08	SCAN6	21	6
0x09	SCAN8	27	8
0x0A	SCAN12	39	12
0x0B	SCAN16	51	All 16 possible SRF08's

REVISION is used to read the CM02 firmware revision. It returns a single byte indicating the revision number. The two data bytes are unused and can be anything, but they must be sent.
NEW_ADDRESS command is used to change an SRF08's I2C address to a different address. The new address should be in the first of the two data bytes. The second data byte is unused and can be anything, but it must be sent. Changing the address on the SRF08 requires 4 separate transactions on the I2C bus. The CM02 know how to change an SRF08's I2C address and just needs you to send it the new address using this command. When using it, make sure you only have one SRF08 connected, otherwise you will set every SRF08 on the bus to the same address. The single return byte is the new address sent back when the task is complete.
BATTERY returns a two byte word (high byte first) indicating the battery voltage. This is the voltage you are supplying the CM02 with. It's a 10-bit number, but it is left justified so should be treated as a 16-bit number. It is the actually output of the A/D converter in the PIC16F88 chip used on the CM02 module. The high byte can be used on its own and the battery voltage treated as a byte reading. Divide the high byte by 16.347 or the word by 4198 to get a direct reading of the battery voltage. (Remember you're on the PC now, and floating point divides are easy).
SCAN This command is very useful when using our modules in your robot. It assumes you have an MD22 motor controller, a CMPS03 compass module and a number of SRF08 rangefinders. SCAN1 assumes 1 SRF08, SCAN8 assumes 8 SRF08's. The two data bytes contain the Left and Right motor speed values for the MD22 motor controller. After sending the new motor speeds to the MD22, the CM02 will send a return frame comprising the battery voltage as a single byte. This is the same as the high byte of the BATTERY command above. This is followed by two bytes of compass bearing - high byte first, and then three bytes for each SRF08. The first of the three bytes is the SRF08's light sensor reading. The next two bytes is the range - high byte first.
For example, if the SCAN2 command is used, you would receive a 9 byte return:

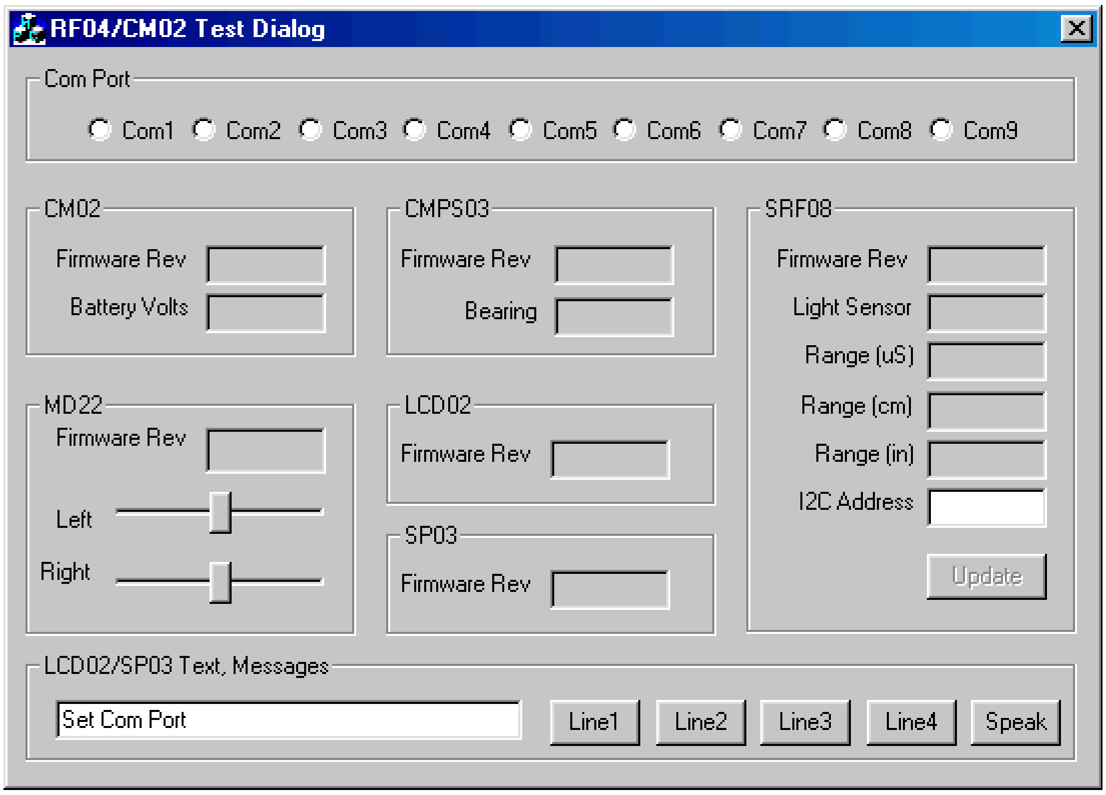
Battery Volts	Compass bearing high byte	Compass bearing low byte	SRF08 at 0xE0 Light sensor	SRF08 at 0xE0 Range high byte	SRF08 at 0xE0 Range low byte	SRF08 at 0xE2 Light sensor	SRF08 at 0xE2 Range high byte	SRF08 at 0xE2 Range low byte

SRF08 data is always returned starting with address 0xE0, 0xE2, 0xE4 - going up one address at a time until all requested SRF08's data has been sent.
After sending the data back up to the PC, the CM02 automatically issues a new ranging command to all SRF08s. The ranging command used is 82 (0x52) which returns the results in uS. To convert to cm divide by 58 and to convert to inches divide by 148.
SRF08 addresses should have been set up before running this command and the MD22 should be initialized to the mode and acceleration required. One more important feature. The SCAN command also sets up a 500mS timer on the CM02. If another SCAN command is not received within this time, a command is automatically sent to the MD22 to stop the motors. This is to prevent your robot wandering out of control if it ventures outside of the range of the radio link.
After initialization, our test robot is controlled entirely with a SCAN8 command.

LEDs
There are two status Leds on the CM02. A red Led indicates power is on and the green Led flashes briefly when a

command is received over the radio link.

RF04/CM02 Test Software
To help with testing the RF04/CM02 radio link, I have put together the following test dialog. Download it here. It's a small Windows/PC program, and looks like this when you first kick it off.


RF04/CM02 Test Dialog

Com Port
Com1 Com2 Com3 Com4 Com5 Com6 Com7 Com8 Com9

CM02
Firmware Rev
Battery Volts

CMPS03
Firmware Rev
Bearing

SRF08
Firmware Rev
Light Sensor
Range (uS)
Range (cm)
Range (in)
I2C Address
Update

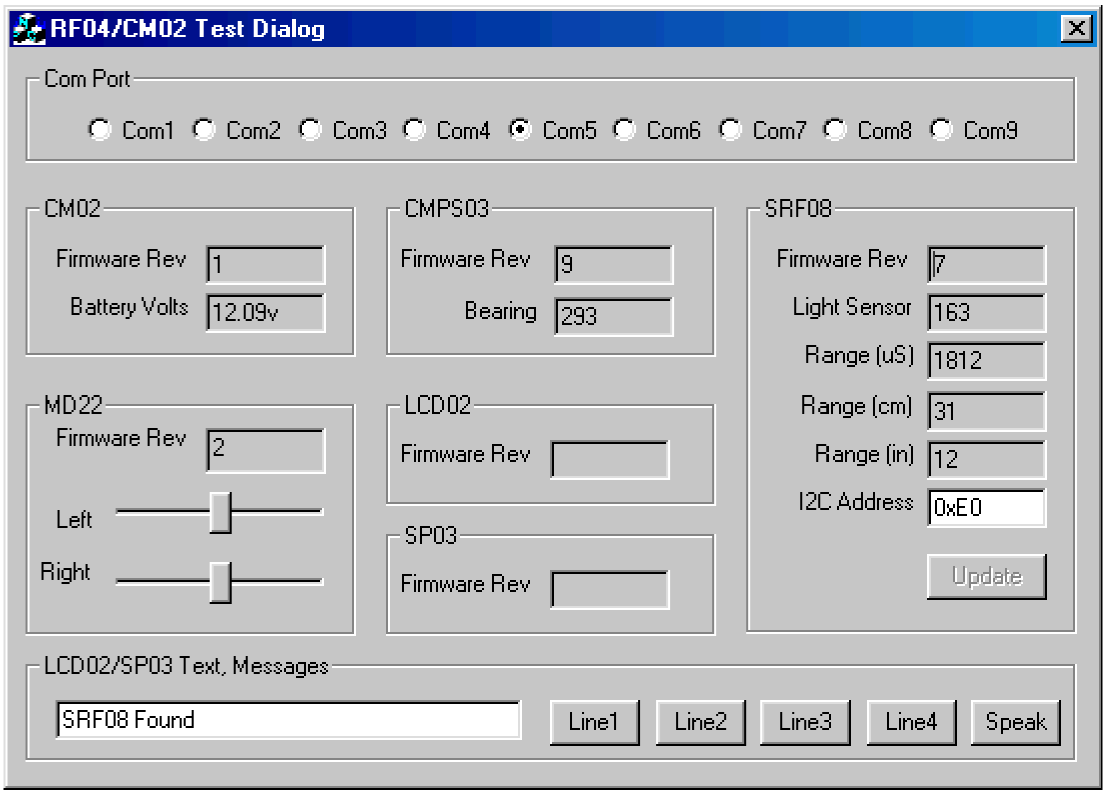
MD22
Firmware Rev
Left
Right

LCD02
Firmware Rev

SP03
Firmware Rev

LCD02/SP03 Text, Messages
Set Com Port
Line1 Line2 Line3 Line4 Speak

You should first select the com port that the RF04 has been instantiated to. There are instructions on the RF04 technical page about how to find which com port its located at. You only have to set the com port once. If you have a CM02 powered up and in range, you will see the Firmware Revision and Battery voltage appear on screen.


RF04/CM02 Test Dialog

Com Port
Com1 Com2 Com3 Com4 [✓ Com5] Com6 Com7 Com8 Com9

CM02
Firmware Rev 1
Battery Volts 12.09v

CMPS03
Firmware Rev 9
Bearing 293

SRF08
Firmware Rev 7
Light Sensor 163
Range (uS) 1812
Range (cm) 31
Range (in) 12
I2C Address 0xE0
Update

MD22
Firmware Rev 2
Left
Right

LCD02
Firmware Rev

SP03
Firmware Rev

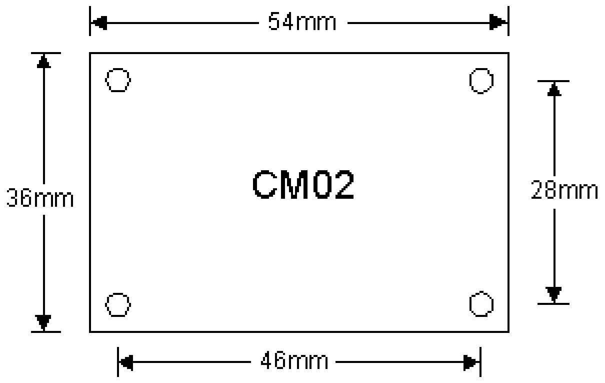
LCD02/SP03 Text, Messages
SRF08 Found
Line1 Line2 Line3 Line4 Speak

In this case Rev1 and 12.09v coming from the battery. In this example the CM02 is fitted to our test robot - Chucky. It has an MD22 fitted to drive the motors, and you will see this is firmware Rev2. The sliders controls the speed of the motors. There is a CMPS03 compass module fitted - Rev9 and showing a bearing of 293 degrees. An SRF08 has been found at address 0xE0. Chucky actually has 12 SRF08's fitted, the one shown is simply the first that was found. Chucky does not have an SP03 module fitted - yet, or an LCD module. If they are connected they will appear in the LCD02 and SP03 panels. Text can be sent to the SP03 or LCD02 modules by typing it into the edit box and pressing one of the Line/Speak buttons.

The Edit box in the SRF08 panel shows the currently found SRF08. If you change this address, the "Update" button, which is normally grayed out, becomes active. If you press it, a command will be sent to the CM02 to change the SRF08's address. This is very useful if you're putting multiple SRF08's on your robot. Make sure you only have one SRF08 connected when you do this, otherwise you will reprogram every SRF08 to the new address. The display is live - with battery volts, compass and sonar being continuously updated.

I also have and example of using the RF04 and CM02, along with the CMPS03, MD22 and 12 SRF08's, here.

PCB Footprint


54mm
36mm
CM02
28mm
46mm