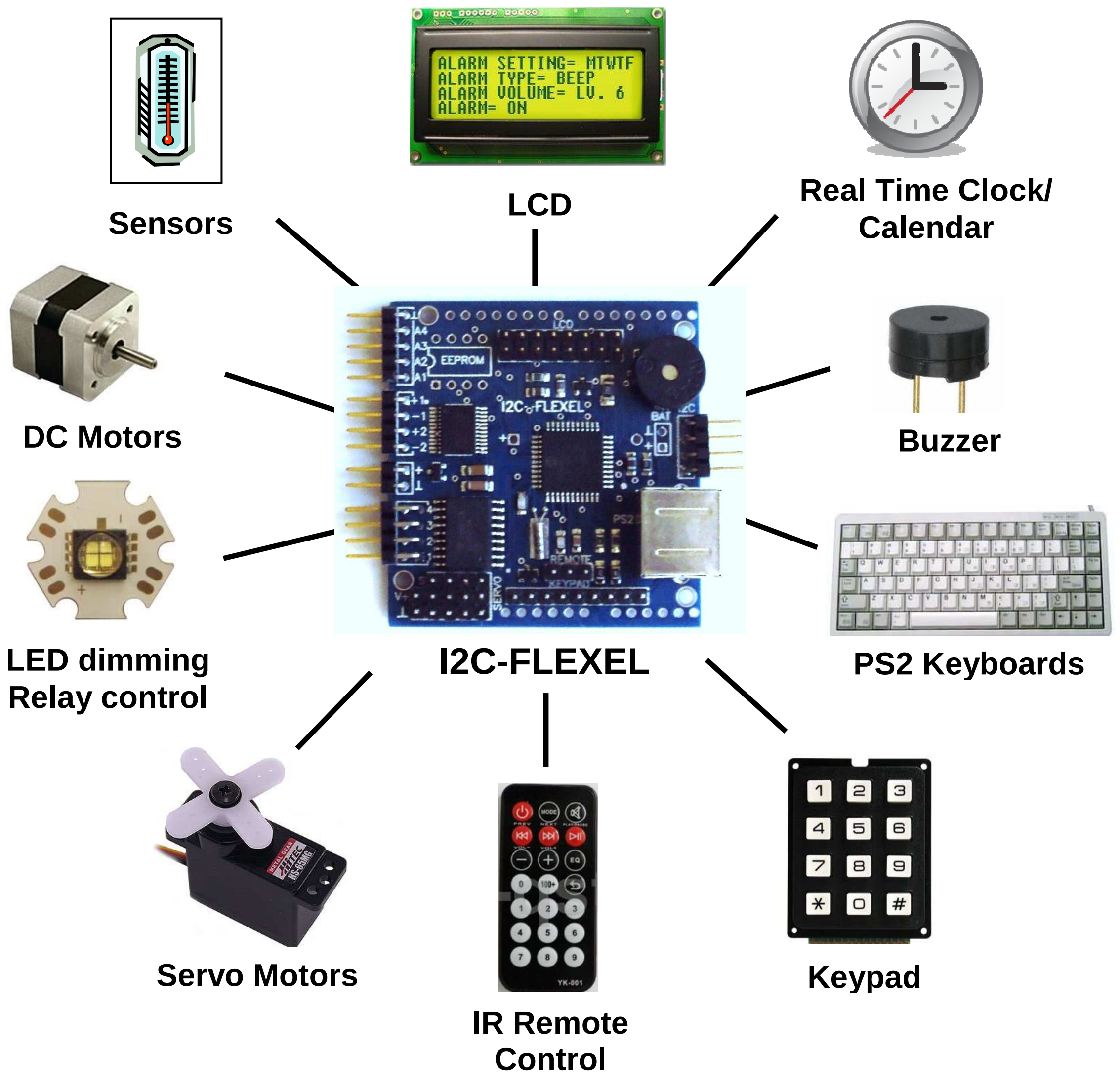
Data Acquisition Module with I2C interface «I2C-FLEXEL»

User's Guide



CONTENTS

1 Introduction.

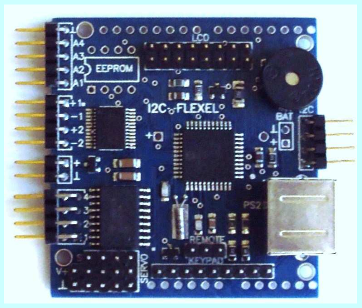
"I2C-FLEXEL" is a versatile, general-purpose data acquisition module with I2C interface. The module is designed as Arduino shield and also can be connected to other microcontrollers with I2C interface.
The module includes a 16-bit microcontroller and works as co-processor for Arduino or your microcontroller. The embedded pre-programmed microcontroller provides control for external peripherals and takes care of communication with I2C bus.

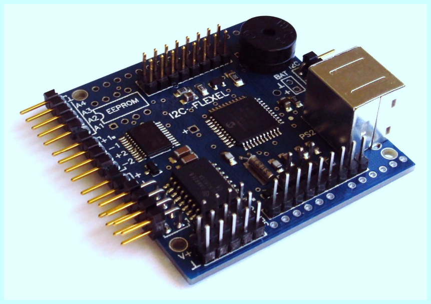
The module is connected to I2C bus as slave device and supports a simple command structure to communicate with Arduino or your microcontroller. Only two lines are used to connect the module to Arduino. All other Arduino inputs / outputs can be used for other tasks.

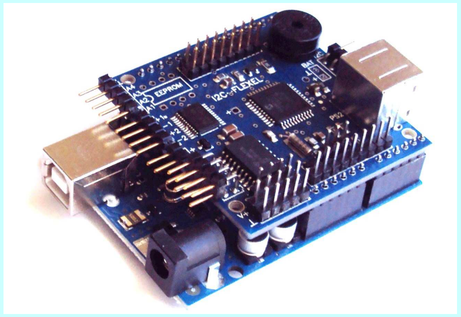
The module features:

- Communicate over standard I2C bus (100 Kbit speed) as slave device
- Control two bidirection DC motors with max current 1.2A per motor
- Four high power outputs (control relays, high current LEDs …)
- RGB LEDs dimming with up to 750mA current per channel
- Up to 4 servo motor control
- Four channels 12 Bit Analog to Digital Converter
- Support the character LCD16x2 and 20x4 with LCD backlight and contrast setup via software
- Infrared (IR) remote control
- Support a matrix keypad up to 16 keys (4 rows by 4 columns) or up to 8 buttons
- Support a standard keyboard with PS2 interface
- Real Time Clock and Calendar (RTCC)
- Buzzer control
- Socket for EEPROM chip with I2C interface
- Small form factor - size 2.0" x 2.1" (51mm x 53mm)

I2C-FLEXEL Board



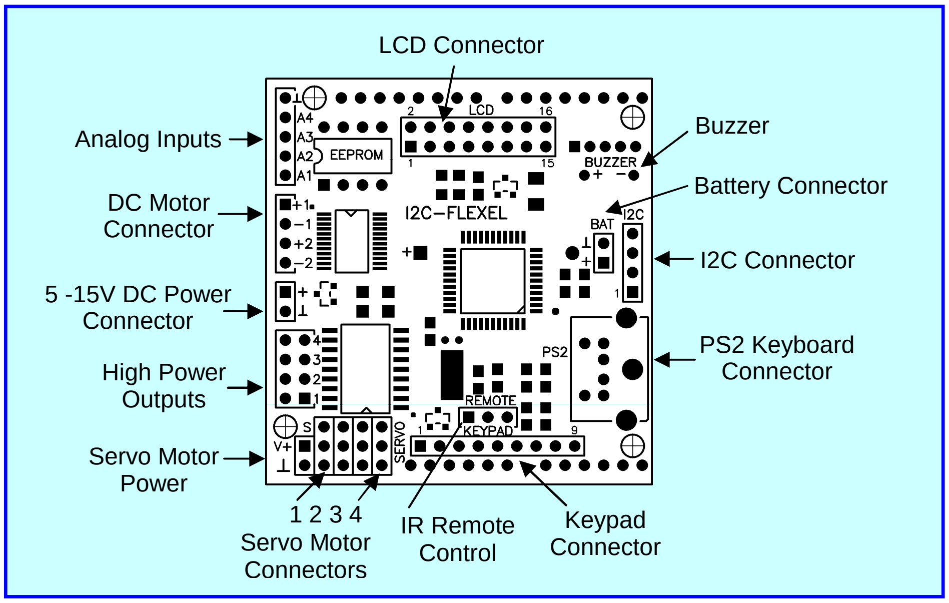




I2C-FLEXEL as Arduino Shield

2 Module connection.

The Fig.1 shows the module connection to the external peripheral devices.



2.1 I2C interface connector.

Table 2.1 shows the connector pin assignments.

Table 2.1

Pin No.	Pin Name	Description
1	SDA	I2C SDA signal. This pin should be connected directly to the SDA pin on your I2C device. The pin is connected to Arduino A4 pin.
2	SCL	I2C SCL signal. This pin should be connected directly to the SCL pin on your I2C device. The pin is connected to Arduino A5 pin.
3	GND	Ground connection
4	VDD	The +5V supply. Connected to Arduino 5V pin. **Do not apply your own 5V power supply to this pin, if the module is connected to Arduino.**

Note: I2C-FLEXEL module includes pull-up resistors for SDA and SCL lines.

2.2 Analog Input connector.

The module includes 12 Bit Analog to Digital Converter with 4 analog inputs.
Table 2.2 shows the connector pin assignments.

Table 2.2

Pin No.	Pin Name	Description
1	A1	Analog Input 1
2	A2	Analog Input 2
3	A3	Analog Input 3
4	A4	Analog Input 4
5	GND	Ground connection

2.3 DC Power connector.

The DC Power connector is used to connect DC power supply for DC motors and high power outputs. The power supply voltage should be in range from 5V to 13.5V (max 15V).
I2C-FLEXEL module provides a power supply reverse polarity protection.

Table 2.3 shows the connector pin assignments.

Table 2.3

Pin No.	Pin Name	Description
1	V+	V+ power supply
2	GND	Ground connection

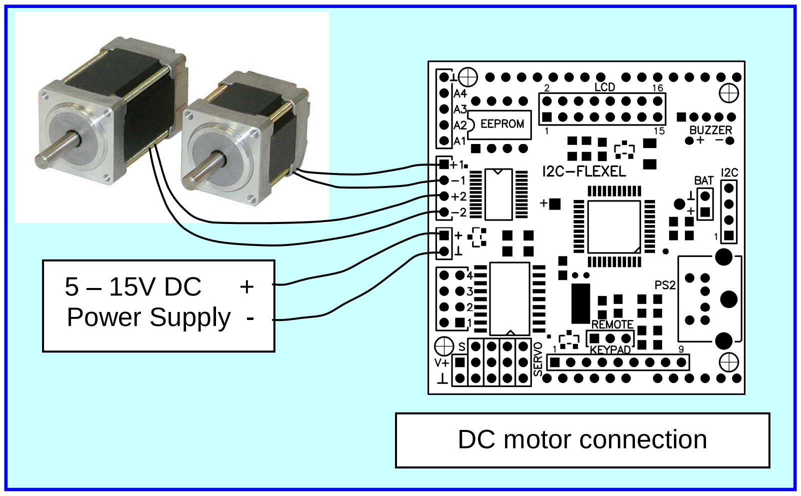
2.4 DC Motor connector.

The module includes a dual DC motor driver with bidirectional PWM motor speed control.
Max motor current is 1.2A (peak 3A), a motor voltage range from 5V to 13.5V (max 15V), PWM frequency – 4 KHz.

Table 2.4 shows the connector pin assignments.

Table 2.4

Pin No.	Pin Name	Description
1	1+	Motor 1 connection
2	1-	Motor 1 connection
3	2+	Motor 2 connection
4	2-	Motor 2 connection



2.5 High Power Output connector.

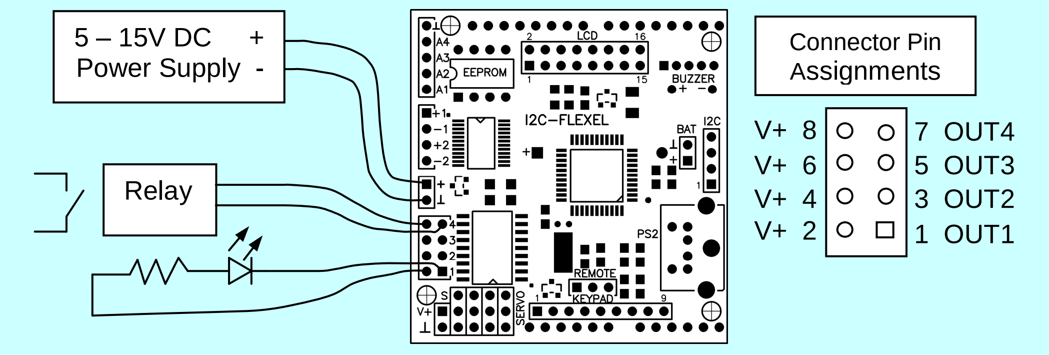
The module includes a chip with high-current Darlington transistor array and provides 4 high power outputs. The chip includes also the common-cathode clamp diodes for swithing inductive loads.Max current is 750mA per output (max 2.5A for all four outputs).

Note: The module embedded microcontroller pins are shared between the high power outputs and servo motor control.
For example, if you use a high power pin 2 to control relay, you can not use servo motor with number 2.

Table 2.5 shows the connector pin assignments.

Table 2.5

Pin No.	Pin Name	Description
1	1	Output 1
2	V+	V+ power supply
3	2	Output 2
4	V+	V+ power supply
5	3	Output 3
6	V+	V+ power supply
7	4	Output 4
8	V+	V+ power supply



Relay and high power LED connection

2.6 Servo Motor Power Supply connector.

The Servo Motor Power connector is used to connect 5V power supply for servo motors.
Take attention to power supply connection. I2C-FLEXEL module does NOT provide a reverse polarity protection for this power supply.
Table 2.6 shows the connector pin assignments.

Table 2.6

Pin No.	Pin Name	Description
1	V+	+5V power supply
2	GND	Ground connection

2.7 Servo Motor connector.

I2C-FLEXEL module provides control up to four servo motors with 1 microsecond pulse resolution.

Note: The module embedded microcontroller pins are shared between the high power outputs and servo motor control.
For example, if you use a high power pin 2 to control relay, you can not use servo motor with number 2.

Table 2.7 shows the connector pin assignments.

Table 2.7

Pin No.	Pin Name	Description
1	S	Servo control signal
2	V+	+5V power supply
3	GND	Ground connection

2.8 LCD connector.

The module provides control for 16x2 and 20x4 Character LCD with HD44780 chip.
The LDC backlight and contrast may be adjusted under program control to compensate for differing lighting conditions and viewing angles.

Table 2.8 shows the connector pin assignments.

Table 2.8

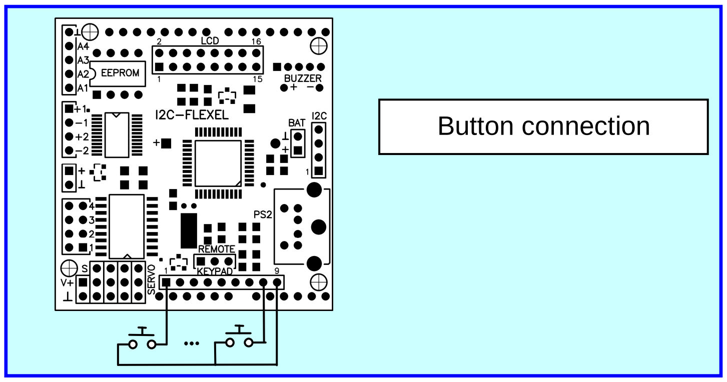
Pin No.	Pin Name	Description
1	VSS	Ground connection
2	VDD	+5V. Connected to Arduino 5V pin.
3	V0	Contrast control voltage
4	RS	RS signal
5	R/W	R/W signal
6	E	E signal
7	DB0	DB0 data line. Not connected
8	DB1	DB1 data line. Not connected
9	DB2	DB2 data line. Not connected
10	DB3	DB2 data line. Not connected
11	DB4	DB4 data line
12	DB5	DB5 data line
13	DB6	DB6 data line
14	DB7	DB7 data line
15	LED-	Backlight Cathode
16	LED+	Backlight Anode

2.9 Keypad connector.

The module supports a matrix keypad up to 16 keys (4 row x 4 coumn) or up to 8 buttons.
Table 2.9 shows the connector pin assignments.

Table 2.9

Pin No.	Pin Name	Description
1	Row1	Keypad Row 1 / Button 1
2	Row2	Keypad Row 2 / Button 2
3	Row3	Keypad Row 3 / Button 3
4	Row4	Keypad Row 4 / Button 4
5	Col1	Keypad Col 1 / Button 5
6	Col2	Keypad Col 2 / Button 6
7	Col3	Keypad Col 3 / Button 7
8	Col4	Keypad Col 4 / Button 8
9	GND	Ground connection



2.10 PS2 Keyboard connector.

The module supports a standard computer keyboard with PS2 interface. The keyboard can be connected directly to PS2 connector.

2.11 Infrared (IR) Remote Control connector.

The module supports infrared remote control with NEC protocol. The infrared detector should be connected to remote control connector.
Table 2.11 shows the connector pin assignments.

Table 2.11

Pin No.	Pin Name	Description
1	OUT	Infrared detector output
2	GND	Ground connection
3	V+	Detector power supply. Connected to Arduino 5V.

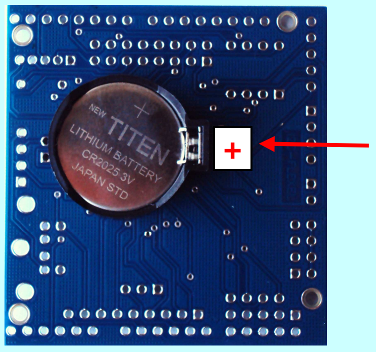
2.12 Battery connector.

The external battery provides the power for embedded microcontroller and supports the Real Time Clock and Calendar if the basic power is disconnected. The module includes a socket for coin battery type CR2032. Also the external battery can be connected to module if you do not use the CR2032 battery. **Do NOT use the external battery and CR2032 battery together.**
The battery voltage should be in range from 3V to 4.5V.

Table 2.12 shows the connector pin assignments.

Table 2.12

Pin No.	Pin Name	Description
1	BAT+	Battery positive output
2	GND	Ground connection



CR2025 battery socket is located at the bottom of the board.

Take attention to the socket and battery polarity orientation.

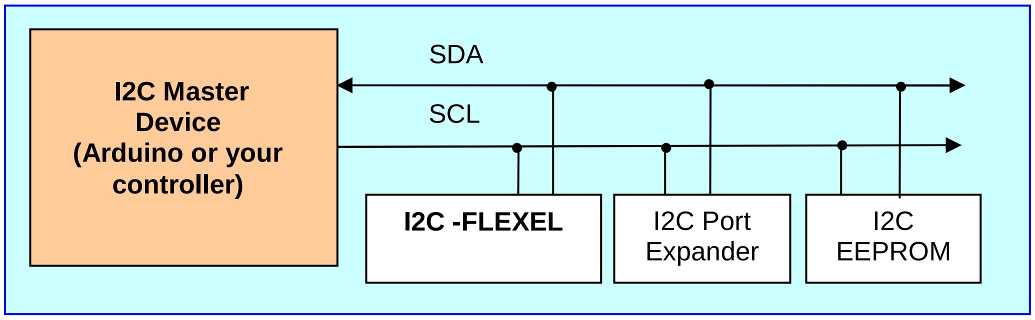
2.13 EEPROM socket.

The module provides 8 pin socket to connect the EEPROM chip with I2C interface.
EEPROM chip I2C address is 96 (0x60 HEX format).
It gives you an opportunity to add an additional EEPROM for your project and use the standard Arduino EEPROM library or your software to communicate with EEPROM.

2.14 Communication interface.

The I2C-bus is for 2-way, 2-line communication between different ICs or modules. The two lines are a serial data line (SDA) and a serial clock line (SCL). Both lines must be connected to a positive supply via a pull-up resistor when connected to the output stages of a device.
Note: I2C-FLEXEL module includes the pull-up resistors for SDA and SCL lines.

The diagram below shows the module connections.



I2C-FLEXEL is a Slave Divice on I2C bus and supports the I2C communication 100 kbps.

Each I2C device must have its own unique address (ID).
I2C-FLEXEL address is 72 (0x48 HEX format).

The I2C Master Device initiates an I2C-bus data transfer through a series of commands.
To prevent the Master from handing the module due to an unfinished command sequence, the I2C-FLEXEL module has a time-out feature. The delay between any two bytes of data coming from Master should be less than 255 ms. If this condition is not met, the module will time-out and clear the receive buffer. The module then starts to wait for the next command from the computer.

3 Commands.

The module is controlled using ASCII characters. The character decimal 254 (0xFE) is a command prefix. Any data sent to the I2C-FLEXEL that is not prefixed by the command prefix (0xFE) will be displays on the LCD.

NOTE: To display the char on LCD, just send its ASCII number, a number from 0x00 to 0x07 displays the user defined custom character, 0x20 to 0x7F displays the standard set of characters. Numbers from 0xA0 to 0xFD display characters and symbols those are factory-masked on the LCD controller and 0xFE is reserved for function command.

After power up I2C-FLEXEL module set up the LCD and other peripherals. Initializing time is approximately 90 milliseconds. Arduino or your microcontroller software must have an appropriate delay before to send the command to I2C-FLEXEL module.

Command Summary

Prefix	Command	Parameter	Description
			Module Control Commands
0xFE	0x24	None	Read Firmware version
0xFE	0x25	1 byte	Set High Power pin
0xFE	0x26	1 byte	Clear High Power pin
0xFE	0x27	2 bytes	Set PWM for High Power pin
0xFE	0x28	3 bytes	Set Servo Motor Position
0xFE	0x29	2 bytes	Set PWM for DC Motor
0xFE	0x2A	None	Get Time
0xFE	0x2B	None	Get Date
0xFE	0x2C	None	Get Time and Date
0xFE	0x2D	3 bytes	Set Time
0xFE	0x2E	4 bytes	Set Date
0xFE	0x2F	7 bytes	Set Time and Date
0xFE	0x30	1 byte	Get Analog Input Value
0xFE	0x31	1 byte	Set Keypad Mode
0xFE	0x32	None	Read Keypad
0xFE	0x33	None	Read Buttons
0xFE	0x34	None	Read PS2 Keyboard
0xFE	0x35	None	Read Remote Control
0xFE	0x36	1 byte	Set Buzzer Time

			LCD Control Commands
0xFE	0x03	1 byte	Set LCD Backlight Brightness
0xFE	0x04	1 byte	Set LCD Contrast
0xFE	0x0A	None	Turn On Display
0xFE	0x0B	None	Turn Off LCD Display
0xFE	0x0C	2 bytes	Set LCD Cursor Position
0xFE	0x0D	None	Home Cursor
0xFE	0x0E	None	Turn On Underline Cursor
0xFE	0x0F	None	Turn Off Underline Cursor
0xFE	0x10	None	Move Cursor Left One Space
0xFE	0x11	None	Move Cursor Right One Space
0xFE	0x12	None	Turn On Blinking Cursor
0xFE	0x13	None	Turn Off Blinking Cursor
0xFE	0x14	None	Clear LCD Screen
0xFE	0x15	Variable	Print String on LCD
0xFE	0x1A	9 bytes	Load LCD Custom Characters

3.1 Module Control Commands.

Read Firmware Version
Syntax hexadecimal 0xFE 0x24

Parameter	Length	Description
None	None	Read the firmware version number

Description: I2C Master (Arduino) issues the read firmware version command by sending two bytes 0xFE 0x24. Module returns data byte with firmware version number.

Set High Power Pin
Syntax hexadecimal 0xFE 0x25 [Pin Number]

Parameter	Length	Description
[Pin Number]	1 byte	Set High Power pin

Description: This command sets the high power pin, the single byte parameter select the pin number in range from 1 to 4.

Clear High Power Pin
Syntax hexadecimal 0xFE 0x26 [Pin Number]

Parameter	Length	Description
[Pin Number]	1 byte	Reset High Power pin

Description: This command reset the high power pin, the single byte parameter select the pin number in range from 1 to 4.

Set PWM for High Power Pin
Syntax hexadecimal 0xFE 0x27 [pin] [pwm]

Parameter	Length	Description
[pin] [pwm]	2 bytes	Put PWM for high power pin

Description: This command sets the high power pin as PWM output with PWM duty cycle according [pwm]. The first parameter [pin] selects the pin number.
The pin number – from 1 to 4, PWM value – from 0 to 255.
Note: PWM frequency is 100 Hz. Use this pin mode for high power LED dimming or the heater control.

Set Servo Motor Position
Syntax hexadecimal 0xFE 0x28 [servo] [MSB][LSB]

Parameter	Length	Description
[servo] [MSB][LSB]	3 bytes	Set servo motor position

Description: This command sets the servo motor position. The first parameter [servo] selects the servo motor in range from 1 to 4. [MSB] and [LSB] bytes define the servo motor position in range from 600 to 2400 with 1 microsecond resolution.

Note: The module embedded microcontroller pins are shared between the high power outputs and servo motor control.
For example, if you use a high power pin 2 to control relay, you can not use servo motor with number 2.

Set PWM for DC Motor
Syntax hexadecimal 0xFE 0x29 [mot/dir] [pwm]

Parameter	Length	Description
[mot/dir] [pwm]	2 bytes	Put PWM for high power pin

Description: This command sets the direction and PWM duty cycle for DC motor. The first parameter [mot/dir] selects the motor and direction, the second byte [pwm] defines the PWM duty cycle in range from 0 to 255.
First parameter [mot/dir] format: Bit 2 defines the motor: 0 – motor 1, 1 – motor 2; Bit 1 – sets the motor direction: 0 – forward, 1 – reverse.

Get Time
Syntax hexadecimal 0xFE 0x2A

Parameter	Length	Description
None	None	Read the time

Description: I2C Master (Arduino) issues read the time from RTCC by sending two bytes 0xFE 0x2A. Module returns 3 data bytes.
Byte 1 – seconds (**BCD** codification, 00-59)
Byte 2 – minutes (**BCD** codification, 00-59)
Byte 3 – hours (**BCD** codification, 00-24)

Example: 24 second BCD codification
Bit7 Bit 0

0	0	1	0	0	1	0	0

Get Date
Syntax hexadecimal 0xFE 0x2B

Parameter	Length	Description
None	None	Read the date

Description: I2C Master (Arduino) issues read the date from RTCC by sending two bytes 0xFE 0x2B. Module returns 4 data bytes.
Byte 1 – day of the week (**BCD** codification, 00-06)
Byte 2 – day of the month (**BCD** codification, 01-31)
Byte 3 – month (**BCD** codification, 01-12)
Byte 4 – year (**BCD** codification, 00-99)

Get Time and Date
Syntax hexadecimal 0xFE 0x2C

Parameter	Length	Description
None	None	Read the time and date

Description: I2C Master (Arduino) issues read the time and date from RTCC by sending two bytes 0xFE 0x2C. Module returns 7 data bytes.
Byte 1 – seconds (**BCD** codification, 00-59)
Byte 2 – minutes (**BCD** codification, 00-59)
Byte 3 – hours (**BCD** codification, 00-24)
Byte 4 – day of the week (**BCD** codification, 00-06)
Byte 5 – day of the month (**BCD** codification, 01-31)
Byte 6 – month (**BCD** codification, 01-12)
Byte 7 – year (**BCD** codification, 00-99)

Set Time
Syntax hexadecimal 0xFE 0x2D [sec][min][hour]

Parameter	Length	Description
[sec][min] [hour]	3	Set the time

Description: I2C Master (Arduino) sets the time by sending two bytes 0xFE 0x2D and 3 data bytes.
Data byte 1 – seconds (**BCD** codification, 00-59)
Data byte 2 – minutes (**BCD** codification, 00-59)
Data byte 3 – hours (**BCD** codification, 00-24)

Set Date
Syntax hexadecimal 0xFE 0x2E [wday][day][month][year]

Parameter	Length	Description
[wday][day] [month][year]	4	Set the date

Description: I2C Master (Arduino) sets the date by sending two bytes 0xFE 0x2E and 4 data bytes.
Data byte 1 – day of the week (**BCD** codification, 00-06)

Data byte 2 – day of the month (**BCD** codification, 01-31)
Data byte 3 – month (**BCD** codification, 01-12)
Data byte 4 – year (**BCD** codification, 00-99)

Set Time and Date
Syntax hexadecimal 0xFE 0x2F [sec][min][hour]

Parameter	Length	Description
[sec][min] [hour][wday] [day][month] [year]	7	Set the time and date

Description: I2C Master (Arduino) sets the time and date by sending two bytes 0xFE 0x2F and 7 data bytes.
Data byte 1 – seconds (**BCD** codification, 00-59)
Data byte 2 – minutes (**BCD** codification, 00-59)
Data byte 3 – hours (**BCD** codification, 00-24)
Data byte 4 – day of the week (**BCD** codification, 00-06)
Data byte 5 – day of the month (**BCD** codification, 01-31)
Data byte 6 – month (**BCD** codification, 01-12)
Data byte 7 – year (**BCD** codification, 00-99)

Get Analog Input Value
Syntax hexadecimal 0xFE 0x30 [input]

Parameter	Length	Description
[input]	1 byte	Read the analog input

Description: I2C Master (Arduino) issues read the analog input by sending two bytes 0xFE 0x30 and one data byte. Data byte defines the input number in range from 1 to 4. Module returns 2 data bytes with 12 bit ADC value.

			Data byte 1								Data byte 2				
Bit7							Bit0	Bit7							Bit 0
0	0	0	0	MSB	x	x	x	x	x	x	x	x	x	x	LSB
				ADC bit12											ADC bit 1

Set Keypad Mode
Syntax hexadecimal 0xFE 0x31 [mode]

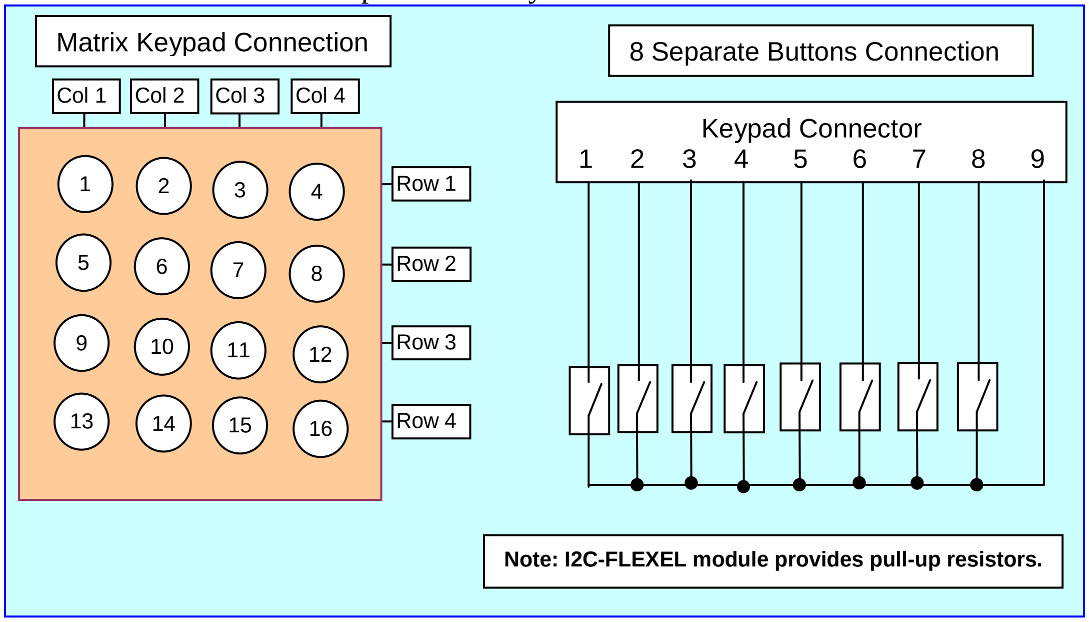
Parameter	Length	Description
[mode]	1 byte	Set keypad mode

Description: I2C Master (Arduino) sets the keypad mode by sending two bytes 0xFE 0x31 and 1 data byte. Data byte defines the keypad mode.
Data byte = 0x00 - matrix keypad 4 row x 4 column (default at power up).
Data byte = 0x01 - 8 button port.
Data byte = 0x02 - keyboard with PS2 interface.

Read Keypad
Syntax hexadecimal 0xFE 0x32

Parameter	Length	Description
None	None	Read the keypad port buffer

Description: I2C Master (Arduino) issues read the keypad by sending two bytes 0xFE 0x32. Module returns 1 data bytes. If there is no key data in the keypad buffer, 0 is returned. If there is key data, a key code from 0x01 to 0x10 is returned, depending on what key was pressed.
Note: I2C-FLEXEL module provides 16 bytes FIFO buffer.



Read Buttons
Syntax hexadecimal 0xFE 0x33

Parameter	Length	Description
None	None	Read the keypad port buffer

Description: I2C Master (Arduino) issues read the keypad port by sending two bytes 0xFE 0x33. Module returns 1 data bytes. If there is no data in the keypad buffer, 0 is returned. If there is data, a button code from 0x01 to 0x08 is returned, depending on what button was pressed.
Note: I2C-FLEXEL module provides 16 bytes FIFO buffer.

Read PS2 Keyboard
Syntax hexadecimal 0xFE 0x34

Parameter	Length	Description
None	None	Read the keypad port buffer

Description: I2C Master (Arduino) issues read the keypad by sending two bytes 0xFE 0x34. Module returns 1 data bytes. If there is no key data in the keypad buffer, 0 is returned. If

there is key data, a key code from 0x01 to 0x10 is returned, depending on what key was pressed.
Note: I2C-FLEXEL module provides 16 bytes FIFO buffer.

Read Infrared (IR) Remote Control
Syntax hexadecimal 0xFE 0x35

Parameter	Length	Description
None	None	Read the remote control buffer

Description: I2C Master (Arduino) issues read the remote control buffer by sending two bytes 0xFE 0x35. Module returns 1 data bytes. If there is no data in the buffer, 0 is returned. If there is remote control data, a received code is returned.
Note: I2C-FLEXEL module provides 32 bytes FIFO buffer.

Set Buzzer Time
Syntax hexadecimal 0xFE 0x36 [count]

Parameter	Length	Description
[count]	1 byte	Create the buzzer pulse

Description: I2C Master (Arduino) initiates the buzzer pulse by sending two bytes 0xFE 0x36 and 1 data byte. Data byte defines the buzzer pulse time. Data byte value is in range from 1 to 255 counts. 1 count = 5 milliseconds. The buzzer pulse frequency is 2 KHz.

3.2 LCD Control Commands.

Set Backlight Brightness
Syntax hexadecimal 0xFE 0x03 [brightness]

Parameter	Length	Description
[brightness]	1 byte	Set backlight brightness (0 – 250)

Description: This command sets the backlight value, the single byte parameter select the desired brightness. The module modulates the backlight via a transistor. This allows to set different brightness settings.
The command requires 1 ms to take effect; therefore, the subsequent input must have an appropriate delay.
Default 80

Set Contrast
Syntax hexadecimal 0xFE 0x04 [contrast]

Parameter	Length	Description
[contrast]	1 byte	Set LCD contrast (0 – 100)

Description: This command sets the LCD contrast value, the single byte parameter select the desired contrast. The module set the contrast voltage.
The command requires 1 ms to take effect; therefore, the subsequent input must have an appropriate delay.
Default 20

Turn On Display
Syntax hexadecimal 0xFE 0x0A

Parameter	Length	Description
None	None	Turn on LCD screen

Description: This command turns on the LCD display screen.
Default LCD screen is on

Turn Off Display
Syntax hexadecimal 0xFE 0x0B

Parameter	Length	Description
None	None	Turn off LCD screen

Description: This command turns off the LCD display screen.
Default LCD screen is on

Set Cursor Position
Syntax hexadecimal 0xFE 0x0C [cool] [row]

Parameter	Length	Description
[row] [cool]	2 bytes	Put cursor at location specified row and cool

Description: This command moves the cursor to a specified location where the next character will be displayed. The row value - from 0 to 3, the column value - from 0 to 19.
The command requires 2 ms to take effect; therefore, the subsequent input must have an appropriate delay.

Home Cursor
Syntax hexadecimal 0xFE 0x0D

Parameter	Length	Description
None	None	Position cursor at line 0 column 0

Description: This command moves the cursor move the cursor to line 0, column 0 of the LCD screen.
The command requires 2 ms to take effect; therefore, the subsequent input must have an appropriate delay.

Turn On Underline Cursor
Syntax hexadecimal 0xFE 0x0E

Parameter	Length	Description
None	None	Turn on underline cursor

Description: This command turn on the underline cursor, the cursor position is where the next character will appear.
Default: The underline cursor is off.

Turn Off Underline Cursor
Syntax hexadecimal 0xFE 0x0F

Parameter	Length	Description
None	None	Turn off underline cursor

Description: This command turns off the underline cursor.
Default: The underline cursor is off.

Move Cursor Left One Space
Syntax hexadecimal 0xFE 0x10

Parameter	Length	Description
None	None	Move cursor left one space

Description: This command moves the cursor position left 1 space.
Default: None.

Move Cursor Right One Space
Syntax hexadecimal 0xFE 0x11

Parameter	Length	Description
None	None	Move cursor right one space

Description: This command moves the cursor position right 1 space.
Default: None.

Turn On Blinking Cursor
Syntax hexadecimal 0xFE 0x12

Parameter	Length	Description
None	None	Turn on the blinking cursor

Description: This command turns on the blinking cursor; both the cursor and the character on the cursor will blink.
Default: The blinking cursor is off.

Turn Off Blinking Cursor
Syntax hexadecimal 0xFE 0x13

Parameter	Length	Description
None	None	Turn off the blinking cursor

Description: This command turns off the blinking cursor.
Default: The blinking cursor is off.

Clear Screen
Syntax hexadecimal 0xFE 0x14

Parameter	Length	Description
None	None	Clear LCD and move cursor to line 1 column 1

Description: This command clears the display and place the cursor at line 1 column 1.
Default: None.
The command requires 2 ms to take effect; therefore, the subsequent input must have an appropriate delay.

Print String
Syntax hexadecimal 0xFE 0x15 [count][string]

Parameter	Length	Description
[count][string]	variable	Print string

Description: This command prints the string with length [count] to the LCD at the present cursor position.

Load Custom Characters

Syntax hexadecimal 0xFE 0x1A [addr][d0 … d7]

Parameter	Length	Description
[addr][d0 … d7]	9 bytes	Load custom characters, [addr] 1 byte – custom character address from 0 to 7, [d0 … d7] 8 bytes – custom character pattern bit map

Description: LCD module has space for 8 custom characters. Each custom character is 5 pixels wide by 8 pixels high.
The [addr] parameter indicates which custom character is defining, and must have a value from 0 to 7.
Following the [addr] parameter are 8 bytes that define the custom character. Bits 0 to 4 each byte byte will each define a pixel character.
The command requires 2 ms to take effect; therefore, the subsequent input must have an appropriate delay.

Example: The bit map for character 'X'.

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**	**Hex**
Byte 1	0	0	0	**1**	0	0	0	**1**	0x11
Byte 2	0	0	0	0	**1**	0	**1**	0	0x0A
Byte 3	0	0	0	0	0	**1**	0	0	0x04
Byte 4	0	0	0	0	**1**	0	**1**	0	0x0A
Byte 5	0	0	0	**1**	0	0	0	**1**	0x11
Byte 6	0	0	0	0	0	0	0	0	0x00
Byte 7	0	0	0	0	0	0	0	0	0x00
Byte 8	0	0	0	0	0	0	0	0	0x00