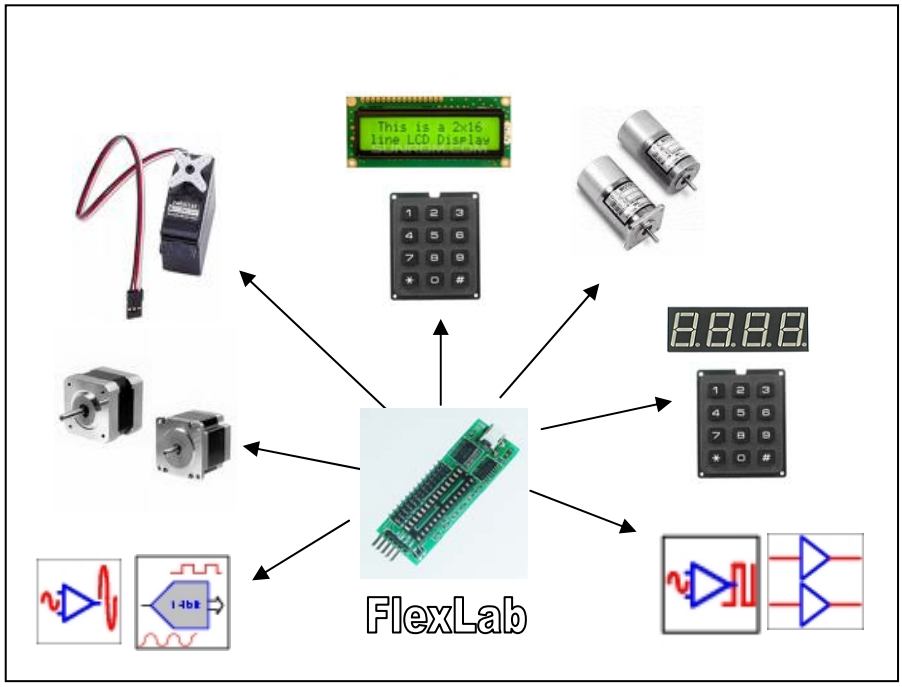
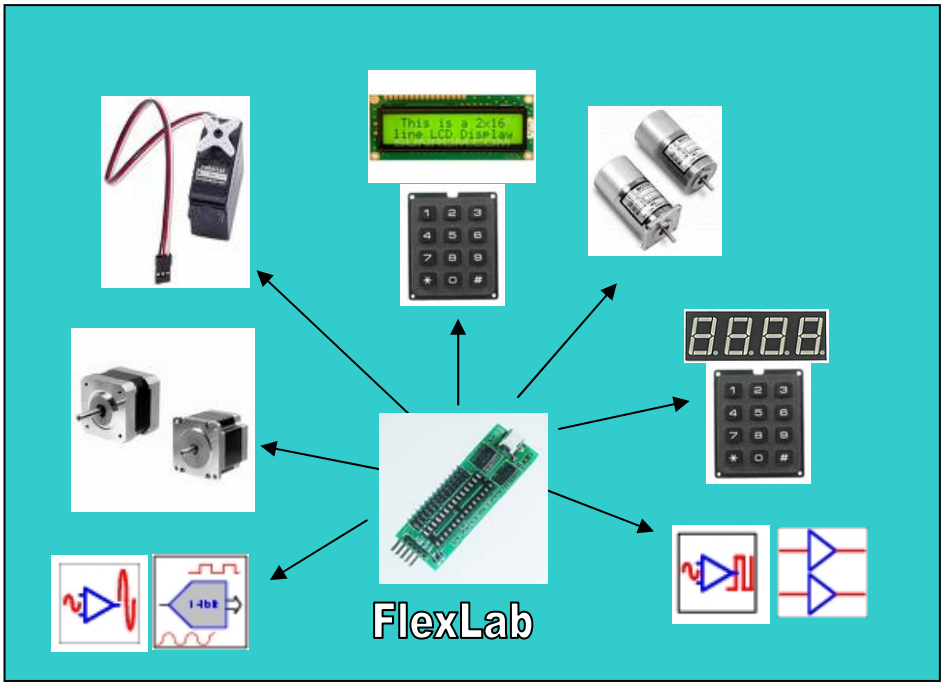


USB Data Acquisition Lab
«FlexLab»

User manual

CONTENTS

1 Introduction.



FlexLab provides 3 basic functions:

1. **USB Data Acquisition System with programmable structure**
2. **USB to I2C Interface Converter**
3. **USB to Serial TTL Interface Converter**

Key Features

- Data acquisition system with 6 preprogrammed configurations.
- On-board 26 pin connector for data acquisition system.
- Mini USB connector and Full speed 12 Mbps USB-to-Serial converter.
- Support the dual interface: standard 100Kbps I2C interface and Serial TTL interface.
- The module is a master for I2C bus.
- 5 pin connector for I2C/Serial bus.
- On-board the pull up resisters for I2C bus.
- 2 LEDs to indicate USB traffic.
- The module is self powered from the USB and can supply up to 50mA at 5V for external circuitry.
- Small form factor (2.1” x 0.9”, 54 x 23 mm).

USB Data Acquisition System with programmable structure

FlexLab can be directly connected to computer USB port as the intelligent peripherals.

FlexLab module includes configurable blocks of analog circuits and digital logic, as well as programmable interconnect. This architecture allows the user to create customized peripheral configurations, to match the requirements of each individual application.
Multiple different hardware function sets can be implemented in one single module under software control. It enables a designer to dynamically change the configurations repeatedly “on-the-fly” while the device is running.

The FlexLab module includes 6 preprogrammed configurations. The configurations can be switched in run time. The dynamic reconfiguration time is less than 200 µs.

FlexLab module includes the configurations as shown in Table 1.

Table1.

Configuration Number	Description
1	16 channels RC servo control
2	Data Acquisition System: 4 analog channels with programmable gain amplifier, 14 bit ADC, two 9 bit analog outputs, 8 digital inputs/outputs
3	Character LCD and 4x4 matrix Keypad control
4	LED 7-Segment Display and Keypad control (supports Single-Digit, Dual-Digit, Triple-Digit, Quad-Digit Displays and 4x4 Keypad)
5	Up to 4 DC motor control (includes four 8 bit PWM blocks), 16 digital inputs/outputs
6	Stepper motor control, 16 digital inputs/outputs

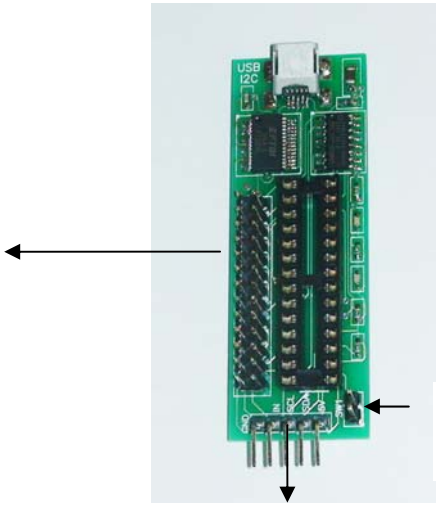
2 FlexLab DESCRIPTION.

2.1 FlexLab Connection.

FlexLab board includes the three connectors: mini USB connector, data acquisition system connector and I2C/Serial interface connector.
The diagram below shows the module connections.

Data acquisition system connector.

Pin Name	Pin No.	Pin No.	Pin Name
P0.0	1	2	P2.0
P0.1	3	4	P2.1
P0.2	5	6	P2.2
P0.3	7	8	P2.3
P0.4	9	10	P2.4
P0.5	11	12	P2.5
P0.6	13	14	P2.6
P0.7	15	16	P2.7
SDA/RX	17	18	P1.4
SCL/TX	19	20	P1.5
SEL1	21	22	P1.6
SEL2	23	24	P1.7
GND	25	26	VDD



I2C/Serial interface connector

Pin No.	5	4	3	2	1
Pin Name	GND	IN	SCL	SDA	VDD

Table 1 shows the pin assignment of I2C/Serial connector.

Table 1

Pin No.	Pin Name	Description
1	+5V	The +5V supply from USB. The module can supply up to 50mA to external devices. Leave this pin unconnected if your external device requires more than 50mA, or has its own supply. **Do not apply your own 5V supply to this pin.**
2	SDA / RX	I2C SDA signal (RX – for serial interface). This pin should be connected directly to the SDA pin on your I2C device.
3	SCL / TX	I2C SCL signal (TX – for serial interface). This pin should be connected directly to the SCL pin on your I2C device.
4	INT	Interface select pin. Leave this pin unconnected if you use I2C interface and connect to ground for Serial interface. **Note:** You can also use jumper SW1 on board to select the communication interface.
5	GND	Ground pin must be connected to the 0V (Ground) on your external device.

Table 2 shows the pin assignments of data acquisition system connector.

Table 2

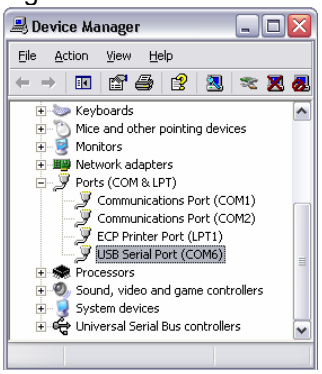
Pin No.	Pin Name	Description
1	P0.0	Port 0, Bit 0
2	P2.0	Port 2, Bit 0
3	P0.1	Port 0, Bit 1
4	P2.1	Port 2, Bit 1
5	P0.2	Port 0, Bit 2
6	P2.2	Port 2, Bit 2
7	P0.3	Port 0, Bit 3
8	P2.3	Port 2, Bit 3
9	P0.4	Port 0, Bit 4
10	P2.4	Port 2, Bit 4
11	P0.5	Port 0, Bit 5
12	P2.5	Port 2, Bit 5
13	P0.6	Port 0, Bit 6
14	P2.6	Port 2, Bit 6
15	P0.7	Port 0, Bit 7
16	P2.7	Port 2, Bit 7
17	SDA	I2C SDA signal (RX – for serial interface)
18	P1.4	Port 1, Bit 4
19	SCL	I2C SCL signal (TX – for serial interface)
20	P1.5	Port 1, Bit 5
21	SEL1	Select the communication protocol , Bit 1
22	P1.6	Port 1, Bit 6
23	SEL2	Select the communication protocol , Bit 2
24	P1.7	Port 1, Bit 7
25	GND	Ground connection
26	VDD	The +5V supply from USB. The module can supply up to 50mA to external devices. Leave this pin

		unconnected if your external device requires more than 50mA, or has its own supply. **Do not apply your own 5V supply to this pin.**

Software and Hardware Installation:

1. Install the Virtual Com Port Drivers. The module uses the FTDI FT232R USB chip to communicate with USB. The drivers are available directly from FTDI website (www.ftdichip.com). The drivers appear to the PC as an extra Com Port. Application software accesses the USB device in the same way as it would access a standard Windows Com Port. The serial data format is: one start bit, 8 data bits, and one stop bit. The baud rate is 9600 bit/s.
2. Connect the module to computer USB port. To see which COM port has been assigned to module, right click on "My Computer" desktop icon and select the "Device Manager" tab. Now scroll down and open the "Ports (COM & LPT)" tab. You should see the USB serial port string with port number (Fig.1).

Fig.1



2.2 FlexLab communication interfaces.

The module supports 2 interfaces:

- I2C interface
- Serial TTL interface

I2C Interface

Industry standard Philips I^2C bus compatible interface.
Data rate 100 kbps.

Serial Interface

Baud Rates 4800, 9600 and 19200 bits per second (default 9600 after power up or reset).
8 Bits per character
None Parity
1 Stop Bit
None Flow Control

The communication interface is determined by the states of the single jumper. When the jumper is present (factory default) the module is in serial mode. If the jumper is removed the module is in I2C mode. The mode jumper is only checked as part of the power-up sequence.
Once the interface has been changed, the module parameters will restored in DEFAULT values.

To restore the module default parameters (module address):

1. Change the jumper state.
2. Power up.
3. Power down.
4. Change back the jumper state.
5. Power up.

2.3 FlexLab technical parameters.

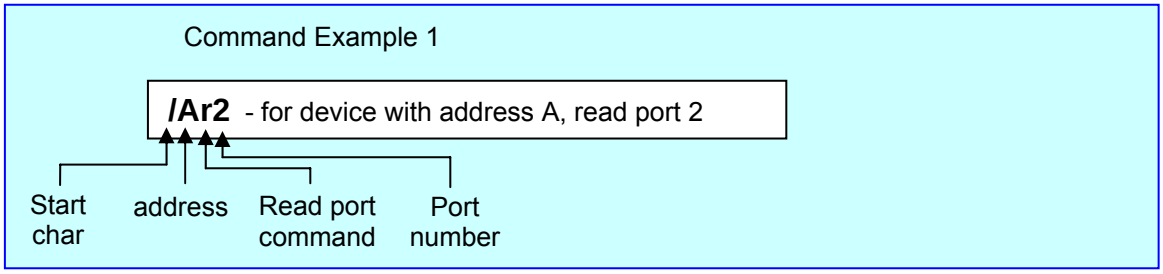
- 5V Operating Voltage
- 0 – 5 V voltage range for analog inputs/outputs
- 25 mA Sink on all digital Inputs/Outputs
- 30 mA on analog outputs
- Industrial Temperature Range -40C to +85C
- Board size - 2.1” x 0.9” (54 x 23 mm)

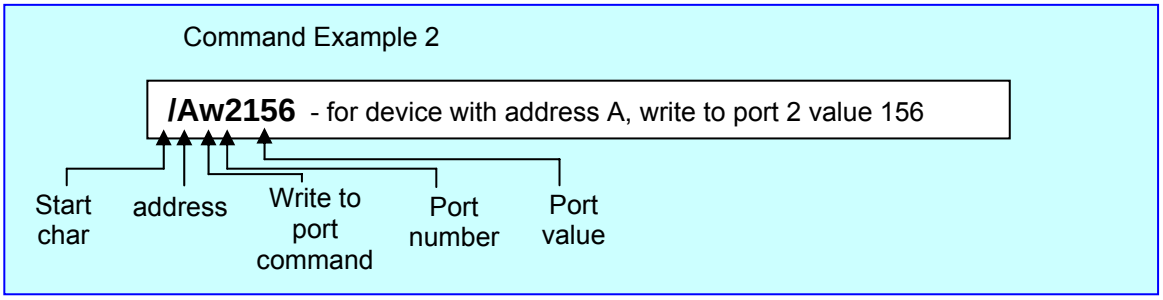
2.4 Command Format.

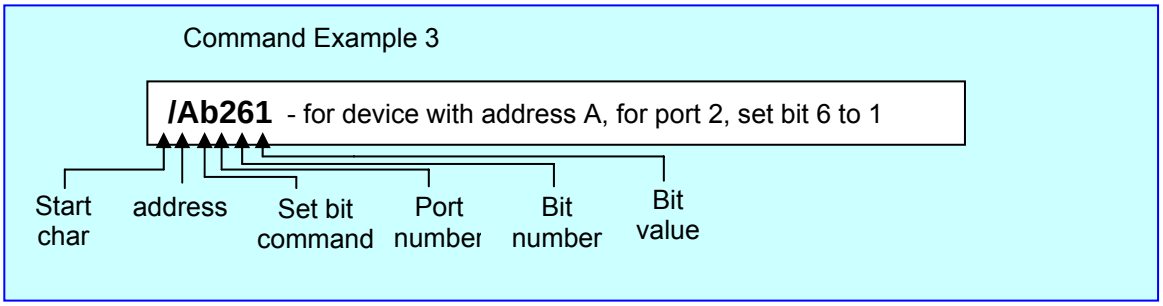
Commands to the FlexLab module are ASCII Character Strings.
All commands start with the ’/’ character and a single alpha device address. Then a one letter command code followed by the parameters. The command is terminated by the <CR> (carriage return) character.
Command Syntax:

1. **/<address><command><parameter1><CR>**
2. **/<address><command><parameter1><parameter2><CR>**
3. **/<address><command><parameter1><parameter2><parameter3><CR>**

Spaces or other punctuation characters are not allowed in the command character sequence.
ALL commands are CASE SENSITIVE.







3 PREPROGRAMMED CONFIGURATION DESCRIPTION.

FlexLab module includes 6 preprogrammed configurations. The configurations can be switched in run time. The dynamic reconfiguration time is less than 200 µs.
Every configuration has a command set. FlexLab includes also four common commands:

- Read the device information
- Set the new device address (ID)
- Set the configuration number
- Set the baud rate for serial port

No.	Command Description	Command Format	Parameters	Example
1	**I** – Read the device information	**/<addr>I<CR>**	/ - start char, <addr> - device address (ID), I – command char, <CR> - carriage return	**/AI** – read the information from device with address A. The device return message (8 chars): F1,A,2,1 Where F01 – device type, A – device address, 2 – configuration number, 1 – revision number
2	**B** – Set the new device address	**/<addr>B<newaddr><CR>**	/ - start char, <addr> - device address, B – command char, <newaddr> - new address, <CR> - carriage return	**/ABD** – set for device with address A the new address D
3	**C** – Set the configuration number	**/<addr>C<newconf><CR>**	/ - start char, <addr> - device address, C – command char, <newconf> - configuration number, <CR> - carriage return	**/AC2** – set for device with address A the configuration number 2
3	**Y** – Set the baud rate for serial port	**/<addr>Y<rate><CR>**	/ - start char, <addr> - device address (if address = '0' 0x30 all devices accept this command), Y – command char, <rate> - baud rate for serial port (0 – 4800, 1 – 9600, 2 – 19200; default 9600 after power up or reset), <CR> - carriage return	**/0Y2** – set for all devices the baud rate 19200

Note: Wait 50ms after "Set new device address" and "Set the configuration number" commands to write the data in internal flash memory, before send the next command.

3.1 Configuration 1. RC Servo Control.

In configuration 1 FlexLab module works as 16 channel servo controller. The device generates 16 continuous streams of pulses that are 500 to 2500 microseconds long, repeated fifty times per second. The pulse resolution is one microsecond. It provides very accurate control of the servo position.

Every servo has independent control. The device can set the position and speed for each servo, disable or enable servo in run time.

Fig.3.1 shows the device pin assignments in configuration 1.

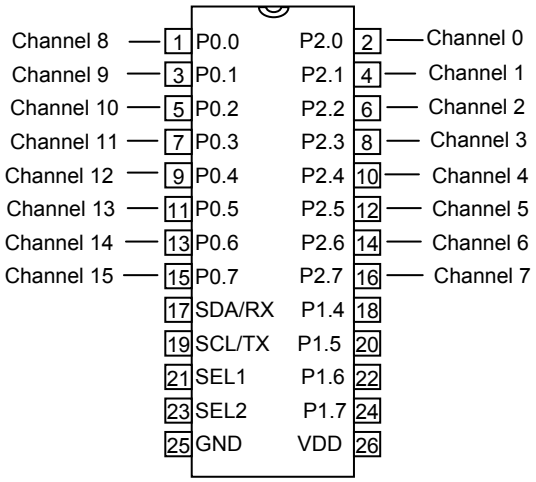


Fig.3.1

Command Set for Configuration 1.

No.	Command Description	Command Format	Parameters	Example
1	**A** – Set the servo position	**/<addr>A<ser><pos><CR>**	/ - start char, <addr> - device address (ID), A – command char, <ser> - servo number (0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F), <pos> - position(500 to 2500), <CR> - carriage return	**/AA30850** – for device A set servo 3 position to 850.
2	**V** – Set the servo speed	**/<addr>V<ser><spd><CR>**	/ - start char, <addr> - device address, V – command char, <ser> - servo number (0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F), <spd> - speed (0 to 255), <CR> - carriage return	**/AV2150** – for device A set servo 2 speed to 150
3	**J** – Disable servo	**/<addr>J<ser><CR>**	/ - start char, <addr> - device address, J – command char, <ser> - servo number (0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F), <CR> - carriage return	**/AJ4** – for device with address A disable servo 4

Note: Set the servo position command enables the current RC servo.

3.2 Configuration 2. Data Acquisition System.

The configuration 2 includes the Data Acquisition System:

- 4 analog channels with programmable gain amplifier
- 14 bit Analog to Digital Converter (ADC)
- two 9 bit analog outputs (DAC)
- 8 digital inputs/outputs

Fig. 3.2 illustrates the system implementation insight in FLEXEL device.

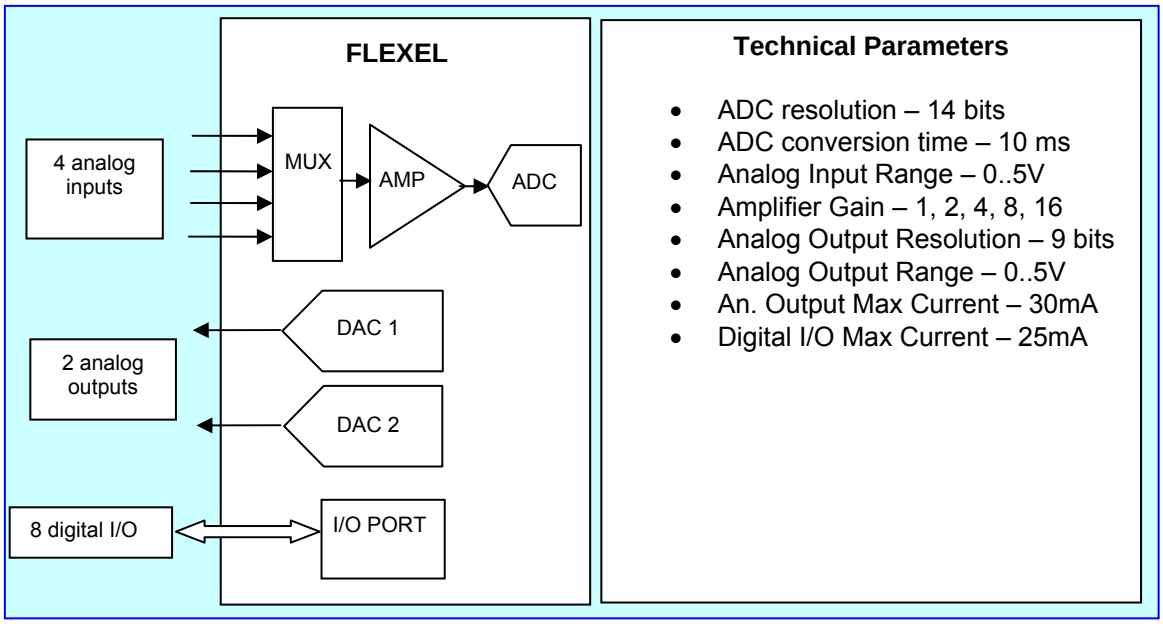
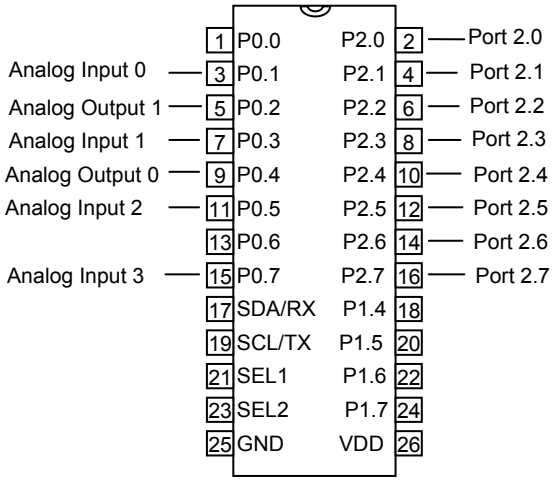


Fig.3.3 shows the device pin assignments in configuration 2.



Command Set for Configuration 2.

No.	Command Description	Command Format	Parameters	Example
1	**i** – Read analog input	**/<addr>i<ch><gain><CR>**	/ - start char, <addr> - device address (ID), i – command char, <ch> - channel (0 to 3), <gain> - amplifier gain (1 to	**/Ai13** – for device A read analog channel 1 with amplifier gain number 3. The device will return

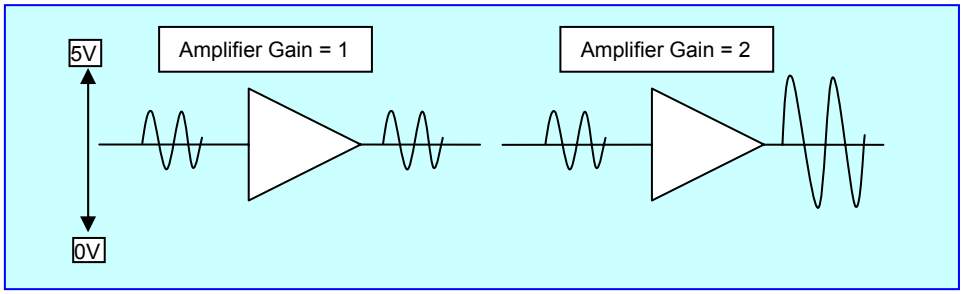
			5), <CR> - carriage return	5 chars of ADC value.
2	**t** – Write to analog output	**/<addr>V<ch><val><CR>**	/ - start char, <addr> - device address, t – command char, <ch> - channel (0,1), <val> - code value (000 to 511), <CR> - carriage return	**/At0215** – for device A write to analog output 0 value 215
3	**d** – Set port direction	**/<addr>d<prt><dir><CR>**	/ - start char, <addr> - device address, d – command char, <prt> - port number (2), <dir> - direction (0 – input, 1 – output), <CR> - carriage return	**/Ad21** – for device with address A set port 2 direction to output
4	**r** – Read port	**/<addr>r<prt><CR>**	/ - start char, <addr> - device address, r – command char, <prt> - port number (2), <CR> - carriage return	**/Ar2** – for device with address A read port 2. The device will return 3 chars of port value.
5	**w** – Write to port	**/<addr>w<prt><val><CR>**	/ - start char, <addr> - device address, w – command char, <prt> - port number (2), <val> - port value, <CR> - carriage return	**/Aw2123** – for device with address A write to port 2 value 123
6	**u** – Set bit direction	**/<addr>u<p><b><d><CR>**	/ - start char, <addr> - device address, u – command char, <p> - port number (2), <b> - bit number (0 to 7), <d> - direction (0 – input, 1 – output), <CR> - carriage return	**/Au261** – for device with address A set port 2 bit 6 direction to output
7	**q** – Read bit	**/<addr>q<prt><bit><CR>**	/ - start char, <addr> - device address, q – command char, <prt> - port number (2), <bit> - bit number (0 to 7), <CR> - carriage return	**/Aq25** – for device with address A read port 2 bit 5. The device will return 1 char of bit value (0,1).
8	**b** – Write bit	**/<addr>b<p><b><v><CR>**	/ - start char, <addr> - device address, b – command char, <p> - port number (2), <b> - bit number (0 to 7), <v> - bit value (0,1), <CR> - carriage return	**/Ab251** – for device with address A write port 2 bit 5 to 1.

Amplifier Gain Setup

Amplifier Gain Number	Amplifier Gain Value
1	1
2	2
3	4
4	8
5	16

Note: Analog signal measurement.

The FlexLab device operates on a single power VDD =5 volts. Analog signals in most systems are typically of both positive and negative polarity around some reference or ground. The FlexLab only handles signals of positive polarity with respect to VDD. An artificial ground is constructed on the chip to provide a reference point for signals of both polarities; this reference is called Analog Ground = VDD/2.



The expected ADC code is 0 for 0 volts, 8192 for Analog Ground and 16383 for 5 volts.
The programmed gain amplifier has not rail to rail input. The calibration procedure is recommended for precision measurement:

1. Connect the reference voltage Vref1 to the analog input and measure the ADC code ADC1.
2. Connect the reference voltage Vref2 to the analog input and measure the ADC code ADC2.
3. Use the formula to calculate input voltage Vx :

   Vx = Vref1 + (ADCx – ADC1)*K, where K = (Vref2 – Vref1) / (ADC2 – ADC1).

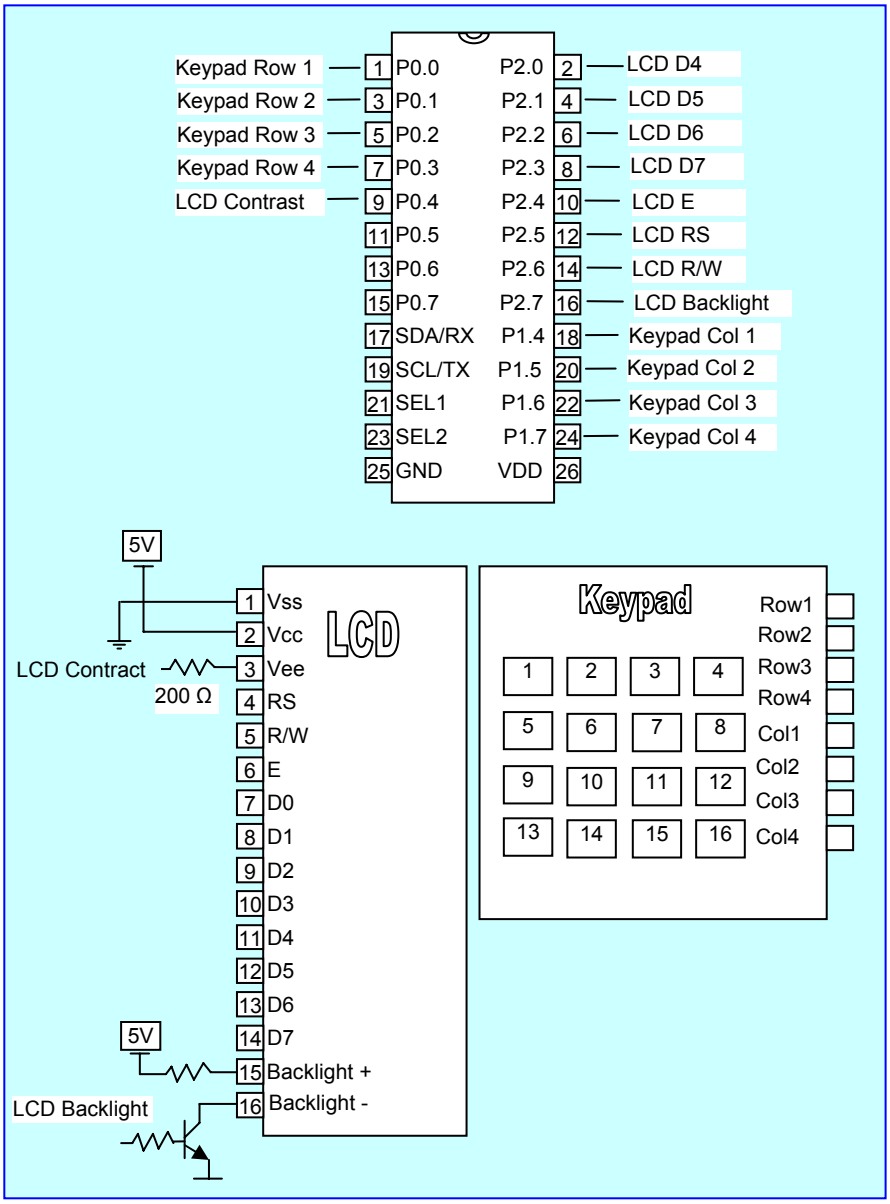
   Example:
   Vref1 = 1V; ADC1 = 3200;
   Vref2 = 4V; ADC2 = 12700;

   For ADCx = 5340 Vx = 1 + (5340-3200) * (4 – 1) / (12700 – 3200) = 1.6758 V.

3.3 Configuration 3. Character LCD and Keypad control.

The configuration 3 supports the character LCD and 4x4 matrix keypad control.

Fig.3.4 shows the device pin assignments in configuration 3 and LCD and Keypad connections.



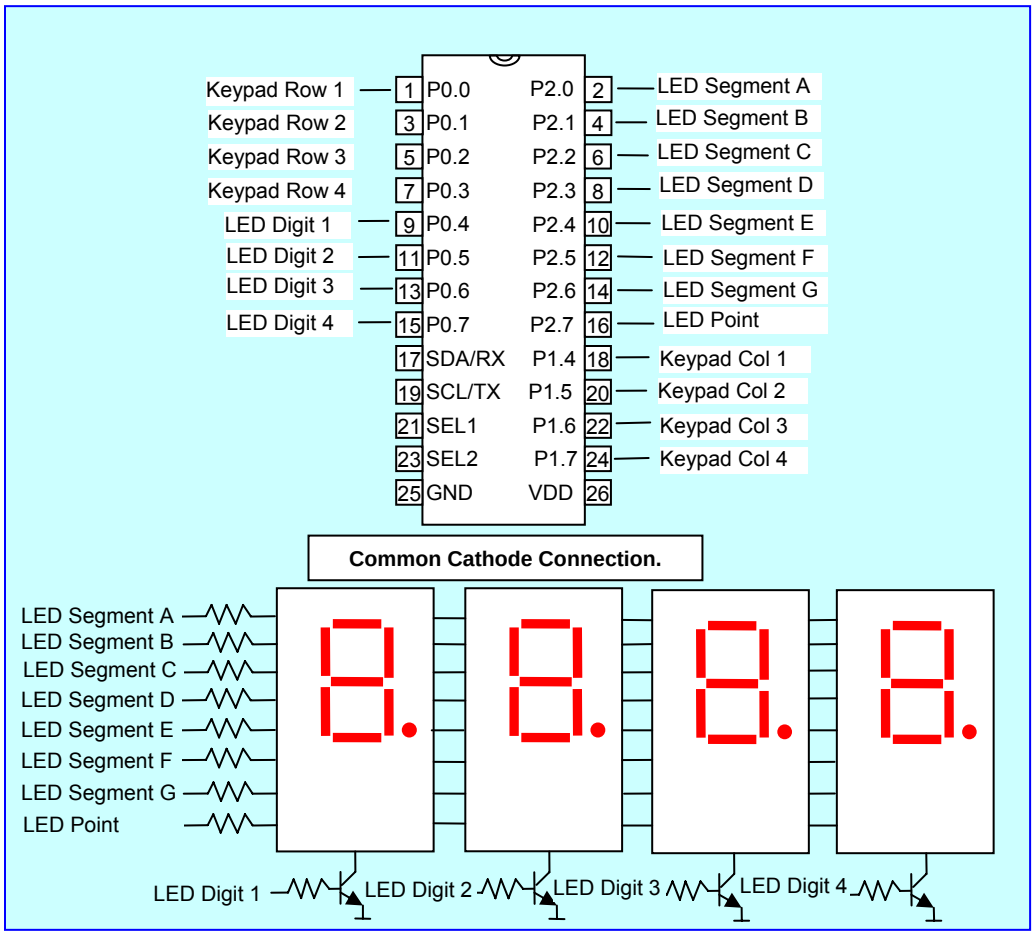
Command Set for Configuration 3.

No.	Command Description	Command Format	Parameters	Example
1	**c** – Clear LCD	**/<addr>c<CR>**	/ - start char, <addr> - device address (ID), c – command char, <CR> - carriage return	**/Ac** – for device A clear LCD.
2	**p** – Set the cursor position LCD	**/<addr>p<str><pos><CR>**	/ - start char, <addr> - device address, p – command char, <str> - string number(0,1,2,3), <pos> - cursor position (00 to 19), <CR> - carriage return	**/Ap215** – for device A set LCD cursor: - string 2 - position 15
3	**t** – LCD print string	**/<addr>t<str><CR>**	/ - start char, <addr> - device address, t – command char, <str> - string (up to 20 chars), <CR> - carriage return	**/AtHello** – for device with address A print string “Hello” at current cursor position
4	**x** – Set LCD contrast	**/<addr>x<val><CR>**	/ - start char, <addr> - device address, x – command char, <val> - contrast value (1…255), <CR> - carriage return	**/Ax35** – for device with address A set LCD contrast value 35
5	**b** – LCD set the backlight	**/<addr>b<val><CR>**	/ - start char, <addr> - device address, b – command char, <val> - backlight Value (0 – backlight off, 1- backlight on), <CR> - carriage return	**/Ab1** – for device with address A turn on the backlight
6	**k** – Read the keypad	**/<addr>k<CR>**	/ - start char, <addr> - device address, k – command char, <CR> - carriage return	**/Ak** – for device with address A read the keypad. The device will return the two chars of keypad number (00 – if not button pressed or 01 to 16 button number)

3.4 Configuration 4. LED 7-Segment Display and Keypad control.

The configuration 4 supports Single-Digit, Dual-Digit, Triple-Digit, Quad-Digit Displays and 4x4 matrix Keypad control.

Fig.3.5 shows the device pin assignments in configuration 4 and LED and Keypad connections.



Command Set for Configuration 4.

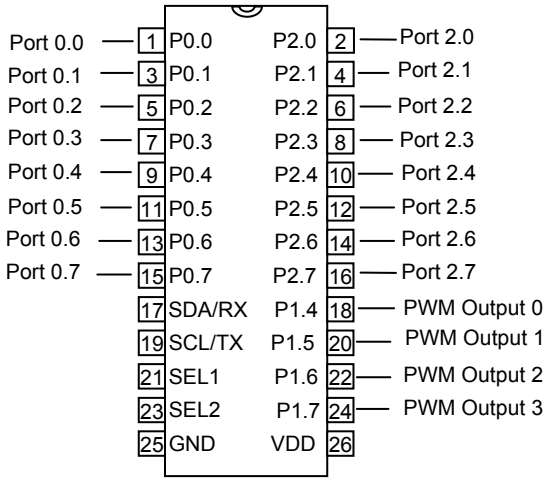
No.	Command Description	Command Format	Parameters	Example
1	**p** – Set LED poit position	**/<addr>p<s><d><CR>**	/ - start char, <addr> - device address (ID), p – command char, <s> - enable poit display (0 – off, 1 – on), <d> - digit location (1…4), <CR> - carriage return	**/Ap12** – for device A set LED point for digit 2.
2	**t** – LED print string	**/<addr>t<p><l><v><CR>**	/ - start char, <addr> - device address, t – command char, <p> - start digit position (1…4), <l> - length of digit to display (1…4), <v> - value to display (0…9999), <CR> - carriage return	**/At12342** – for device with address A print string “12.34”.
3	**k** – Read the keypad	**/<addr>k<CR>**	/ - start char, <addr> - device address, k – command char,	**/Ak** – for device with address A read the keypad.

			<CR> - carriage return	The device will return the two chars of keypad number (00 – if not button pressed or 01 to 16 button number)

3.5 Configuration 5. DC Motor Control.

The configuration 5 provides four 8 bit PWM blocks to control up to 4 DC motor and 16 digital inputs/outputs for additional tasks.

Fig.3.6 shows the device pin assignments in configuration 5.



Command Set for Configuration 5.

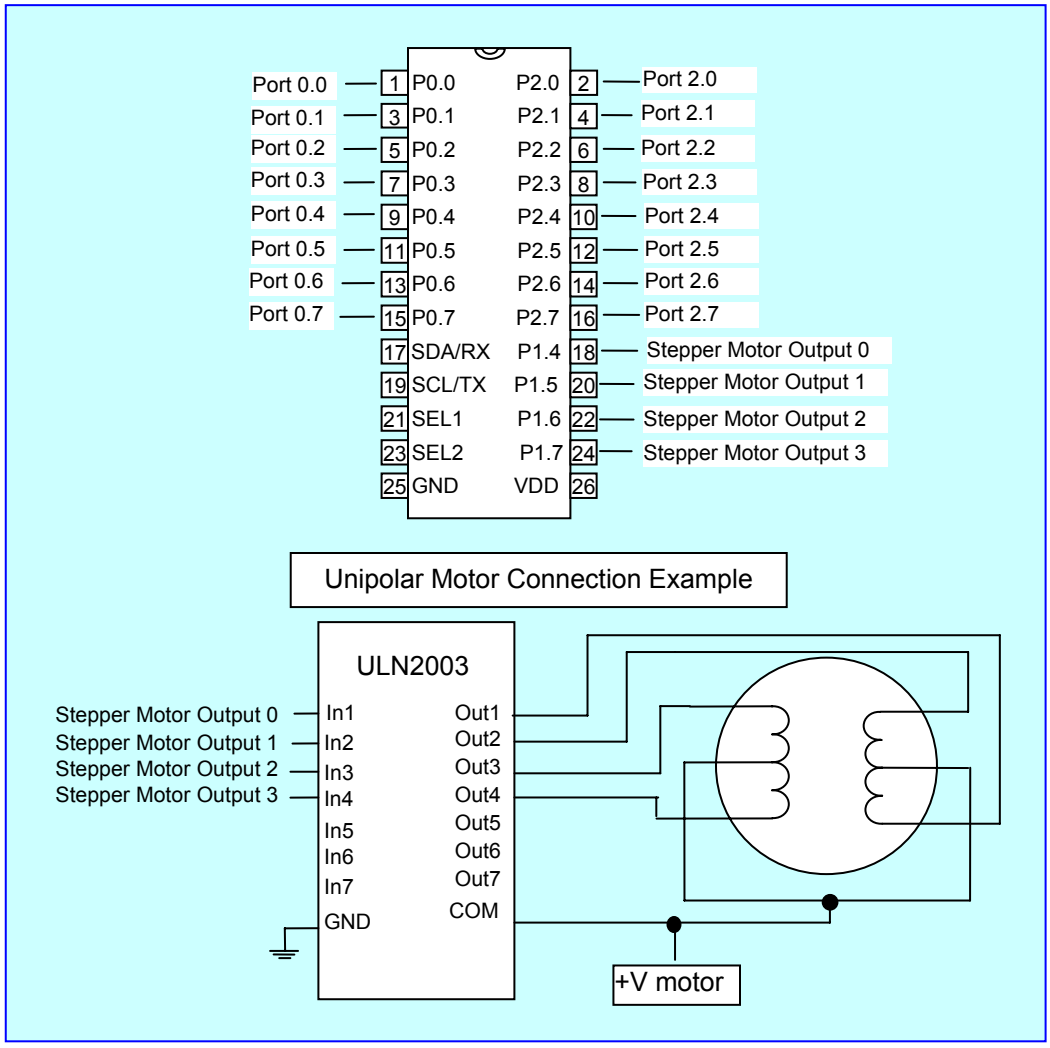
No.	Command Description	Command Format	Parameters	Example
1	**p** – Set PWM pulse width	**/<addr>p<ch><val><CR>**	/ - start char, <addr> - device address (ID), p – command char, <ch> - PWM number (0 to 3), <val> - pulse width (0 to 255, 0 – disable PWM), <CR> - carriage return	**/Ap2150** – for device A set PWM 2 pulse width to 150.
2	**d** – Set port direction	**/<addr>d<prt><dir><CR>**	/ - start char, <addr> - device address, d – command char, <prt> - port number (2), <dir> - direction (0 – input, 1 – output), <CR> - carriage return	**/Ad21** – for device with address A set port 2 direction to output
3	**r** – Read port	**/<addr>r<prt><CR>**	/ - start char, <addr> - device address, r – command char, <prt> - port number (2), <CR> - carriage return	**/Ar2** – for device with address A read port 2. The device will return 3 chars of port value.
4	**w** – Write to port	**/<addr>w<prt><val><CR>**	/ - start char, <addr> - device address, w – command char, <prt> - port number (2), <val> - port value, <CR> - carriage return	**/Aw2123** – for device with address A write to port 2 value 123

5	**u** – Set bit direction	**/<addr>u<p><b><d><CR>**	/ - start char, <addr> - device address, u – command char, <p> - port number (2), <b> - bit number (0 to 7), <d> - direction (0 – input, 1 – output), <CR> - carriage return	**/Au261** – for device with address A set port 2 bit 6 direction to output
6	**q** – Read bit	**/<addr>q<prt><bit><CR>**	/ - start char, <addr> - device address, q – command char, <prt> - port number (2), <bit> - bit number (0 to 7), <CR> - carriage return	**/Aq25** – for device with address A read port 2 bit 5. The device will return 1 char of bit value (0,1).
7	**b** – Write bit	**/<addr>b<p><b><v><CR>**	/ - start char, <addr> - device address, b – command char, <p> - port number (2), <b> - bit number (0 to 7), <v> - bit value (0,1), <CR> - carriage return	**/Ab251** – for device with address A write port 2 bit 5 to 1.

3.6 Configuration 6. Stepper Motor Control.

The configuration 6 provides the stepper motor control and 16 digital inputs/outputs for additional tasks.

Fig.3.7 shows the device pin assignments in configuration 6.

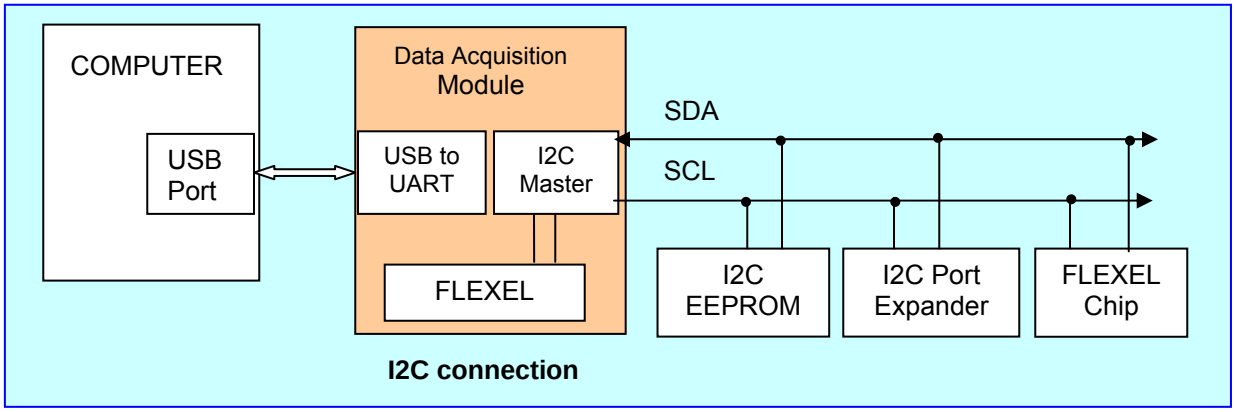


Command Set for Configuration 6.

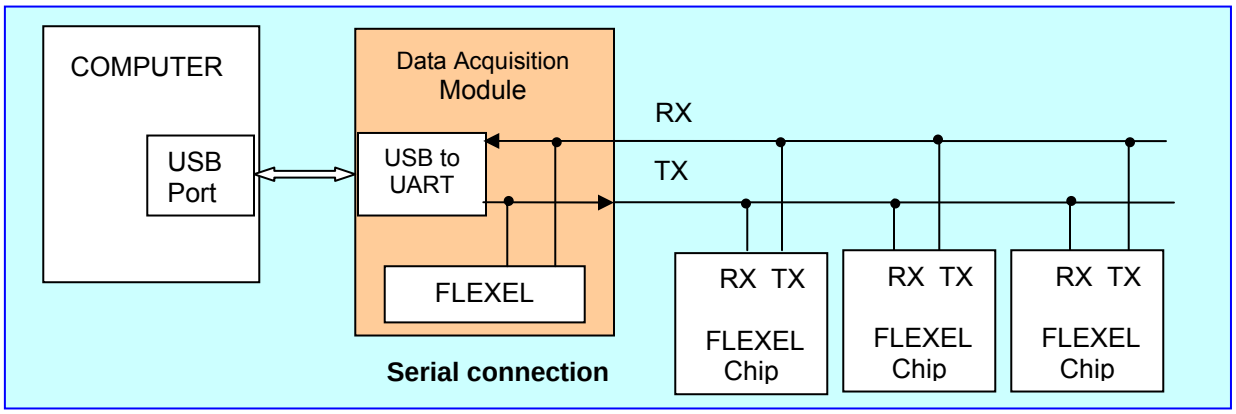
No.	Command Description	Command Format	Parameters	Example
1	**j** – Stepper Motor Setup	**/<addr>j<mt><mode><CR>**	/ - start char, <addr> - device address (ID), j – command char, <mt> - motor type (0 – unipolar, 1 - bipolar), <mode> - motor mode (0 – full step, 1 - half step), <CR> - carriage return	**/Aj01** – for device A setup the unipolar motor in half step mode.
2	**n** – Set the number of steps	**/<addr>n<stps><CR>**	/ - start char, <addr> - device address, n – command char, <stps> - number of steps (-32767 to 32767), <CR> - carriage return	**/An1500** – for device with address A make 1500 steps.
3	**t** – Set the step delay	**/<addr>t<dly><CR>**	/ - start char, <addr> - device address, t – command char, <dly> - step delay (1 to 1000 ms), <CR> - carriage return	**/At132** – for device with address A set the step delay 132 ms.
4	**r** – Read port	**/<addr>r<prt><CR>**	/ - start char, <addr> - device address, r – command char, <prt> - port number (2), <CR> - carriage return	**/Ar2** – for device with address A read port 2. The device will return 3 chars of port value.
5	**w** – Write to port	**/<addr>w<prt><val><CR>**	/ - start char, <addr> - device address, w – command char, <prt> - port number (2), <val> - port value, <CR> - carriage return	**/Aw2123** – for device with address A write to port 2 value 123
6	**u** – Set bit direction	**/<addr>u<p><b><d><CR>**	/ - start char, <addr> - device address, u – command char, <p> - port number (2), <b> - bit number (0 to 7), <d> - direction (0 – input, 1 – output), <CR> - carriage return	**/Au261** – for device with address A set port 2 bit 6 direction to output
7	**q** – Read bit	**/<addr>q<prt><bit><CR>**	/ - start char, <addr> - device address, q – command char, <prt> - port number (2), <bit> - bit number (0 to 7), <CR> - carriage return	**/Aq25** – for device with address A read port 2 bit 5. The device will return 1 char of bit value (0,1).
8	**b** – Write bit	**/<addr>b<p><b><v><CR>**	/ - start char, <addr> - device address, b – command char, <p> - port number (2), <b> - bit number (0 to 7), <v> - bit value (0,1), <CR> - carriage return	**/Ab251** – for device with address A write port 2 bit 5 to 1.

4 ADDITIONAL DEVICE CONNECTION.

Additional devices can be connected to the Lab board.



I2C connection



Serial connection

Command Description.

The computer initiates an I2C-bus data transfer through a series of ASCII commands. The commands start from S character (0x53) and terminated with a P character (0x50). Unrecognized commands are ignored by the USB/I2C module.

To prevent the computer from handing the module due to an unfinished command sequence, the Data Acquisition module has a time-out feature. The delay between any two bytes of
data coming from the computer 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.

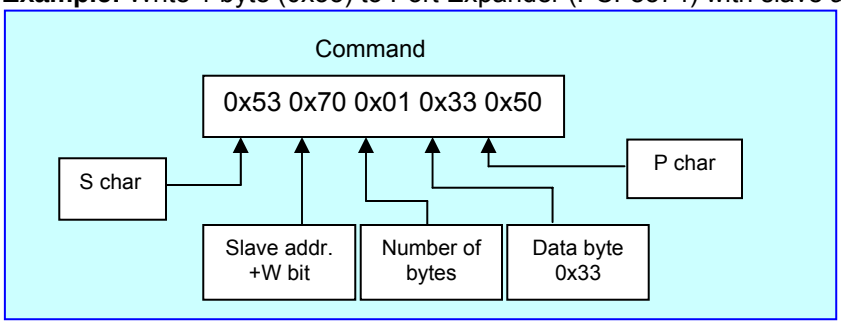
The module supports:

1. Read/Write single byte for non-registered devices (as I/O Expanders).
2. Read multiple bytes without setting new address (as the pressure sensors, ADC, etc).
3. Read/Write single or multiple bytes for 1 byte addressed devices (as EEPROM with 1 byte address, etc).
4. Read/Write single or multiple bytes for 2 byte addressed devices (as EEPROM with 2 byte address).
5. The full range of commands for FLEXEL chips.

Write N bytes to I2C device.
The computer issues the write command by sending an S character followed by an I2C-bus slave device address, the total number of bytes to be sent, and I2C-bus data which begins with the first byte (DATA 0) and ends with the last byte (DATA N). The frame is then terminated with a P character. Once the computer issues this command, the module will access the I2C-bus slave device and start sending the I2C-bus data bytes.
Note that the second byte sent is the I2C-bus device slave address. The least significant bit (W) of this byte must be set to 0 to indicate this is an I2C-bus write command.

The computer sends

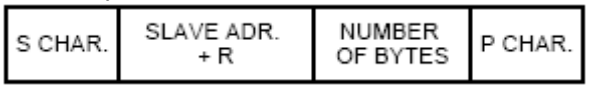
Example: Write 1 byte (0x33) to Port Expander (PCF8574) with slave address 0x38



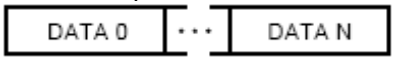
Read N bytes from I2C device.

The computer issues the read command by sending an S character followed by an I2C-bus slave device address, and the total number of bytes to be read from the addressed I2C-bus slave. The frame is then terminated with a P character. Once the computer issues this command, the USB/I2C module will access the I2C-bus slave device, get the correct number of bytes from the addressed I2C-bus slave, and then return the data to the computer.
Note that the second byte sent is the I2C-bus device slave address. The least significant bit (R) of this byte must be set to 1 to indicate this is an I2C-bus read command.

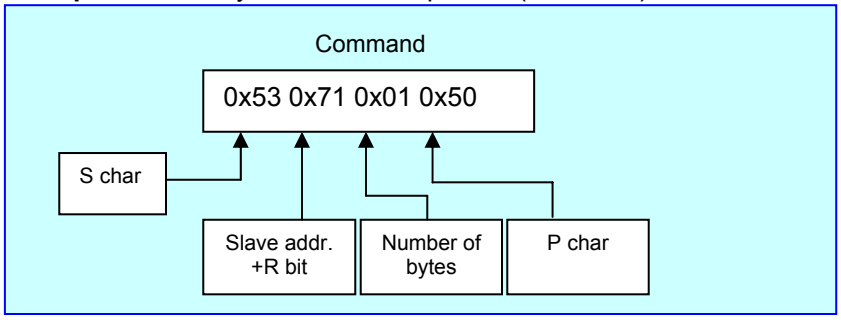
The computer sends



Module responds



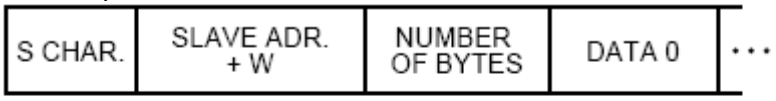
Example: Read 1 byte from Port Expander (PCF8574) with slave address 0x38.



Repeated START: read after write.

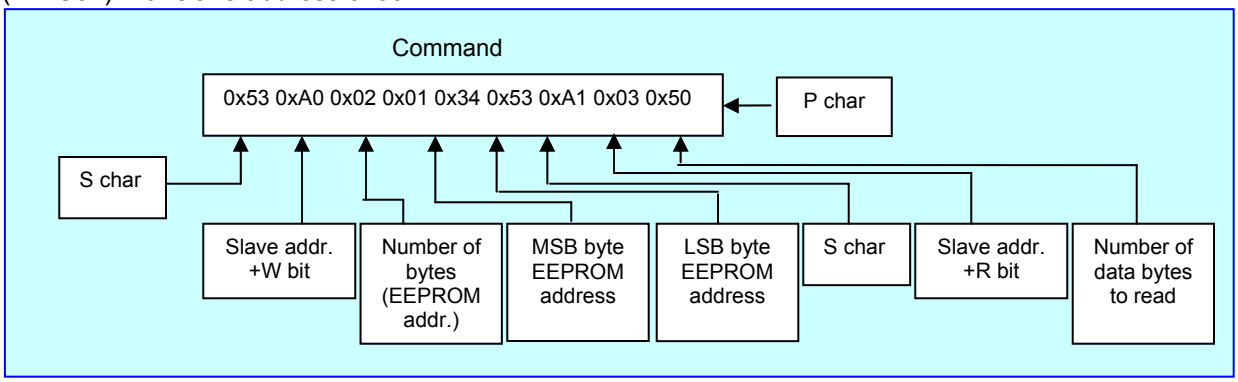
The USB/I2C module also supports ‘read after write’ command as specified in the I2C-bus specification. This allows a read command to be sent after a write command without having to issue a STOP condition between the two commands.
The computer issues a write command as normal, then immediately issues a read command without sending a STOP (P) character after the write command.

The computer sends



Module responds

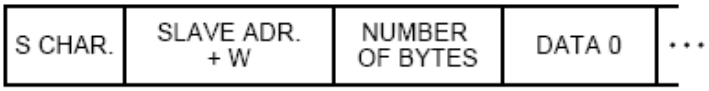
Example: Read 3 bytes (start from address 0x0134) from EEPROM with 2 byte address (24LC32) with slave address 0x50.

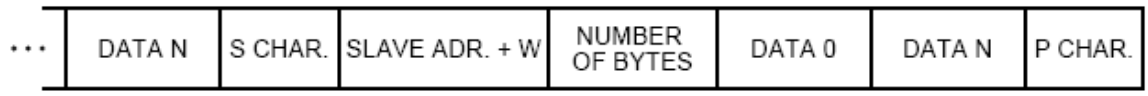


Repeated START: write after write.

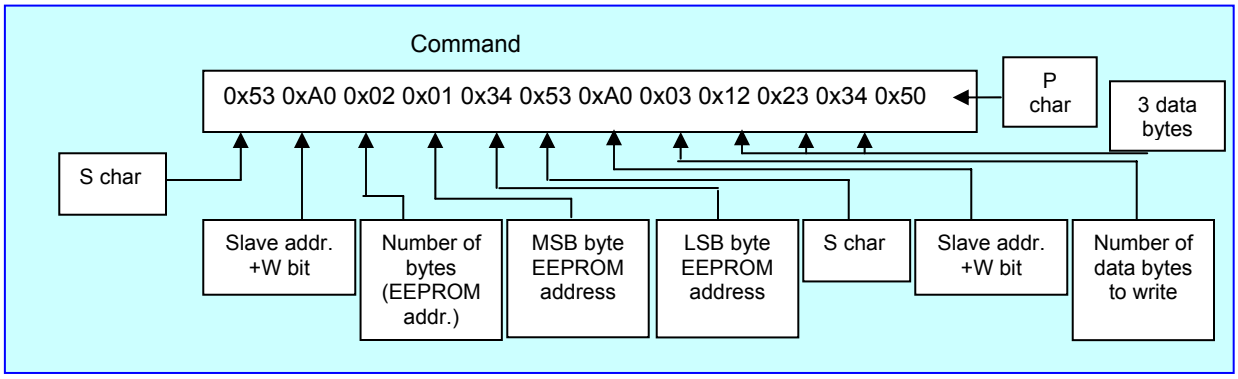
The USB/I2C module also supports ‘write after write’ command as specified in the I2C-bus specification. This allows a write command to be sent after a write command without having to issue a STOP condition between the two commands.
The computer issues a write command as normal, then immediately issues a second write command without sending a STOP (P) character after the first write command.

The computer sends





Example: Write 3 bytes (start from address 0x0134) to EEPROM with 2 byte address (24LC32) with slave address 0x50.



Communication with FLEXEL Chips.

The computer initiates an I2C-bus data transfer through a series of ASCII commands. The commands start from '/' character (0x2F) and terminated with a CR character (0x0D - carriage return).

See the FLEXEL CHIP USER MANUAL for FLEXEL command description.