Easy24-33 v6™

User manual

Development System

All MikroElektronika´s development systems represent irreplaceable tools for programming and developing microcontroller-based devices. Carefully chosen components and the use of machines of the last generation for mounting and testing thereof are the best guarantee of high reliability of our devices. Due to simple design, a large number of add-on modules and ready to use examples, all our users, regardless of their experience, have the possibility to develop their project in a fast and efficient way.

MikroElektronika

SOFTWARE AND HARDWARE SOLUTIONS FOR EMBEDDED WORLD . . . making it simple

TO OUR VALUED CUSTOMERS

I want to express my thanks to you for being interested in our products and for having confidence in mikroElektronika.
The primary aim of our company is to design and produce high quality electronic products and to constantly improve the performance thereof in order to better suit your needs.

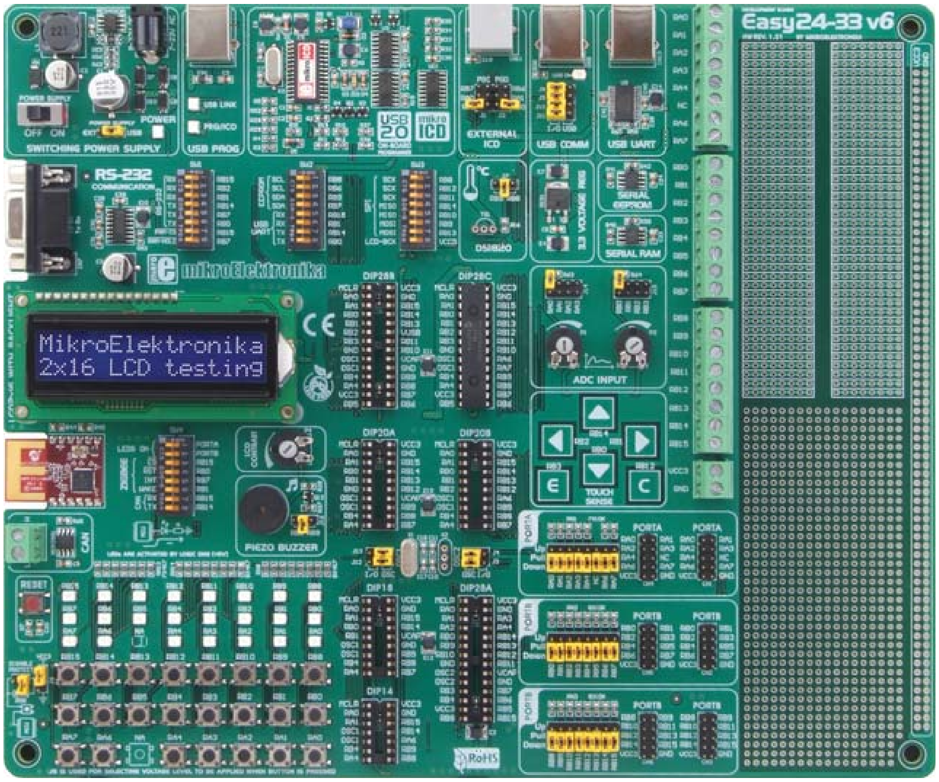
Nebojsa Matic
General Manager

The Microchip name and logo, the Microchip logo, Accuron, dsPIC, KeeLoq, microID, MPLAB, PIC, PICmicro, PICSTART, PRO MATE, PowerSmart, rfPIC and SmartShunt are registered trademarks of Microchip Technology Incorporated in the U.S.A and other countries.

TABLE OF CONTENTS

Introduction to Easy24-33 v6 Development System

The *Easy24-33 v6*™ development system provides a development environment for programming and experimenting with PIC24® microcontrollers from Microchip®. The system includes an on-board programmer with mikroICD support providing an interface between the microcontroller and a PC. You are simply expected to write a program in one of the dsPIC compilers, generate a .hex file and program your microcontroller using the on-board *LvPICFlash*™ programmer. Numerous modules, such as ZigBee®, alphanumeric 2x16 LCD, piezo buzzer, etc, are provided on the board and allow you to easily simulate the operation of the target device.



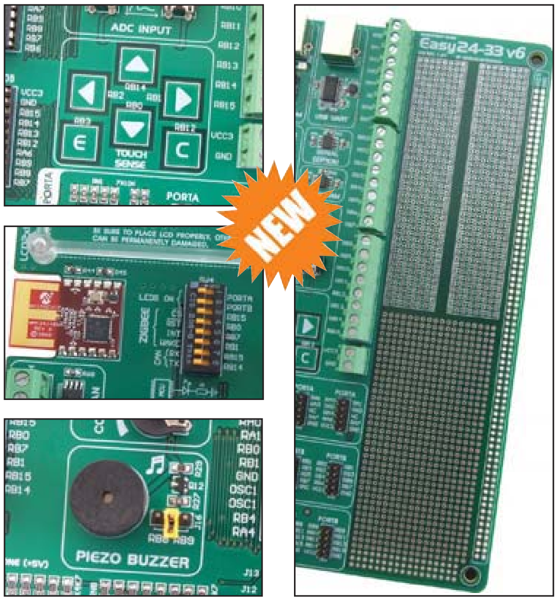
Full-featured development system for PIC24 microcontroller based devices

USB 2.0 on-board programmer

Built-in debugger for testing programs in real time at hardware level

ZigBee communication is based on the IEEE 802.15.4 standard

On-board pads for placing additional boards



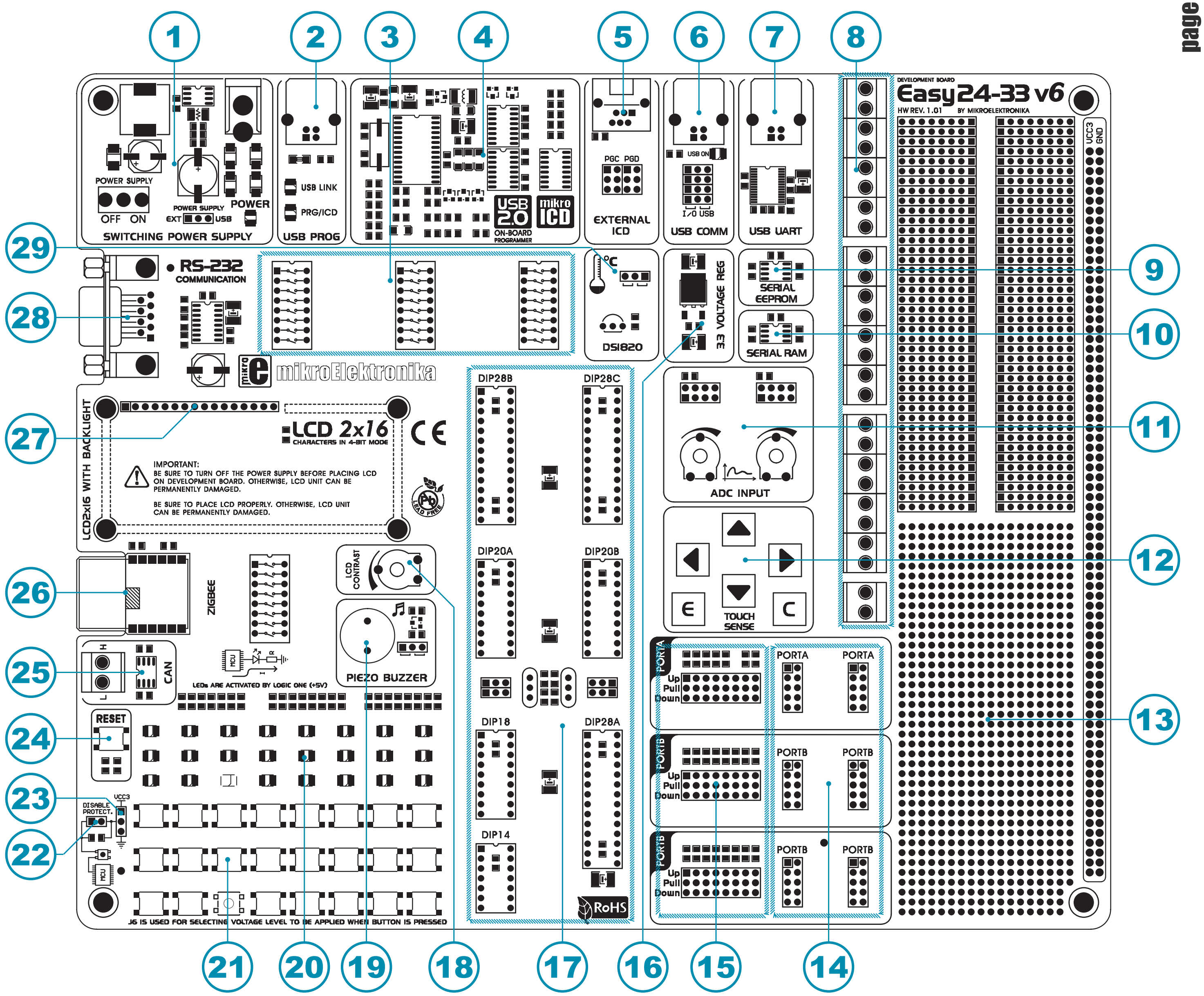
The *mikroProg Suite for PIC* program provides a complete list of all supported microcontrollers. The latest version of this program with updated list of supported microcontrollers can be downloaded from our website at **www.mikroe.com**

Package contains:

Development system:	Easy24-33 v6
CD:	product CD with relevant software
Cables:	USB cable
Documentation:	Manuals for Easy24-33 v6, mikroProg Suite for PIC and mikroICD, Quick gude for installing USB drivers, Electrical schematic of the system

System specification:

Power supply:	over an AC/DC connector (7-23V AC or 9-32V DC); or over a USB programming cable (5V DC)
Power consumption:	~40mA in idle state when all on-board modules are off
Dimension:	26,5 x 22cm (10,4 x 8,6inch)
Weight:	462g (0.89lbs)



Key Features

1. Power supply voltage regulator
2. Built-in programmer's USB connector
3. DIP switches turn on/off on-board modules
4. USB 2.0 programmer with mikroICD support
5. Connector for Microchip's debugger ICD2® or ICD3®
6. USB connector for USB communication
7. USB UART module
8. Connectors for linking microcontroller's I/O pins with pads provided on the development system
9. Serial EEPROM module
10. Serial RAM module
11. A/D converter test inputs
12. Touch sense keypad
13. Pads (proto board)
14. I/O port connectors
15. Jumpers for pull-up/pull-down resistor selection
16. 3.3V voltage regulator
17. Microcontroller sockets
18. Potentiometer for adjusting LCD contrast
19. Piezo buzzer
20. 23 LEDs indicate pins' logic state
21. Push buttons for exciting microcontroller digital inputs
22. Jumper for protective resistor shortening
23. Jumper for selecting push buttons' logic state
24. Reset button
25. CAN communication module
26. ZigBee communication module
27. Alphanumeric LCD connector
28. Connector for RS-232 communication
29. Connector for DS1820 temperature sensor

1.0. Connecting the System to a PC

Step 1:

Follow the instructions provided in the relevant manuals and install the *mikroProg Suite for PIC* program and USB driver from the product CD. USB drivers are essential for the proper operation of the on-board programmer.
In case you already have one of the Mikroelektronika's dsPIC compilers installed on your PC, there is no need to reinstall USB drivers as they are already installed along with the compiler.

Step 2:

Use the USB cable to connect the development system to a PC. One end of the USB cable, with a USB connector of B type, should be connected to the development system, as shown in Figure 1-2, whereas the other end of the cable with a USB connector of A type should be connected to a PC. When establishing a connection, make sure that jumper J17 is placed in the USB position as shown in Figure 1-1.

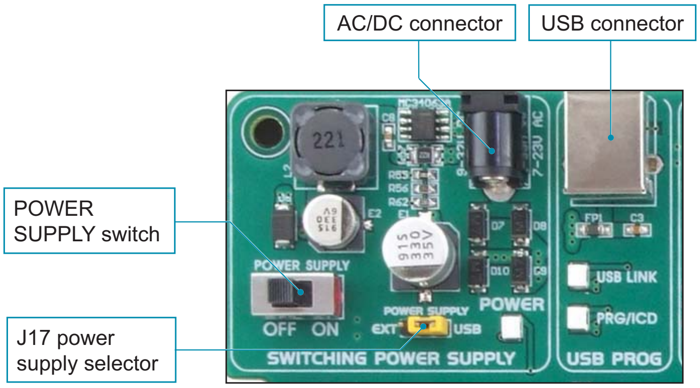


Figure 1-1: Power supply

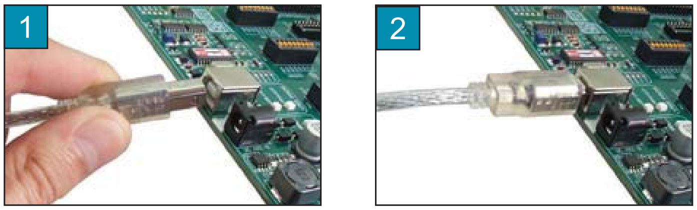


Figure 1-2: Connecting USB cable

Step 3:

Turn on your development system by setting the POWER SUPPLY switch to the ON position. Two LEDs marked as POWER and USB LINK will be automatically turned on indicating that your development system is ready to use. Use the on-board programmer and the *mikroProg Suite for PIC* program to dump a hex code into the microcontroller and employ the system to test and develop your projects.

NOTE: If some additional modules are used, such as LCD, GLCD etc, it is necessary to place them properly on the development system while it is turned off. Otherwise, either can be permanently damaged. Refer to figure below for the proper placing of the additional modules.

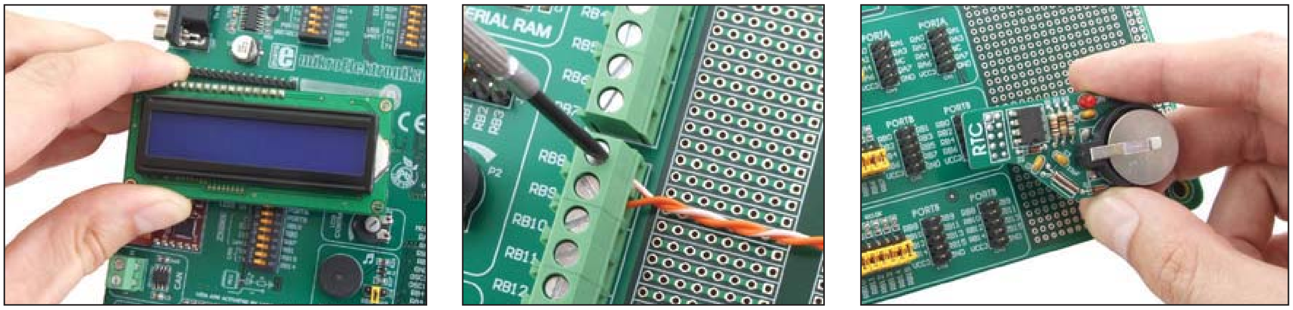


Figure 1-3: Placing additional modules on the board

2.0. Supported Microcontrollers

The *Easy24-33 v6* development system provides seven separate sockets for PIC24 microcontrollers in DIP28, DIP20, DIP18 and DIP14 packages. There are three sockets DIP28A, DIP28B and DIP28C for microcontrollers in DIP28 package, and two sockets DIP20A and DIP20B for microcontrollers in DIP20 package.

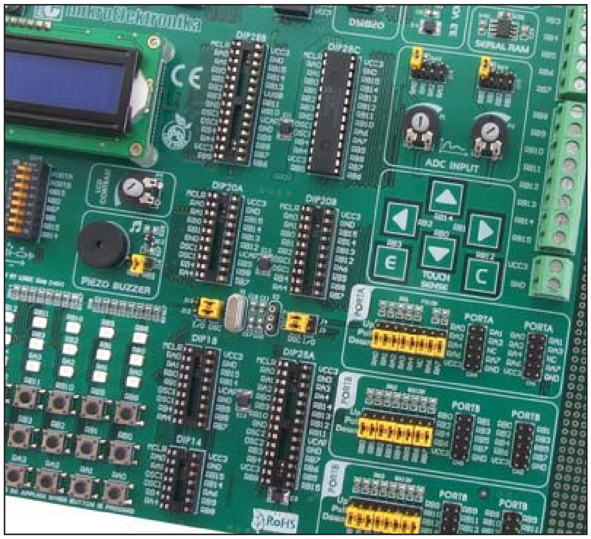


Figure 2-1: Microcontroller sockets

Jumpers J13, J12, J4 and J3 are placed next to the oscillator. They are used to select the function of the microcontroller pins.

Jumper	Position	Function
J13	I/O	RA3 is configured as an I/O pin
	OSC	Pin RA3 is fed with a clock signal from external oscillator X1 (OSC1)
J12	I/O	RA2 is configured as an I/O pin
	OSC	Pin RA2 is fed with a clock signal from external oscillator X1 (OSC1)
J4	I/O	RB2 is configured as an I/O pin
	OSC	Pin RB2 is fed with a clock signal from external oscillator X2 (OSC2)
J3	I/O	RB1 is configured as an I/O pin
	OSC	Pin RB1 is fed with a clock signal from external oscillator X2 (OSC2)

PIC24 microcontrollers can use either built-in or external oscillator as a clock signal source. There is a quartz crystal provided on the board that is used as the external oscillator. Signals generated by this oscillator are used as clock signals for most supported microcontrollers.

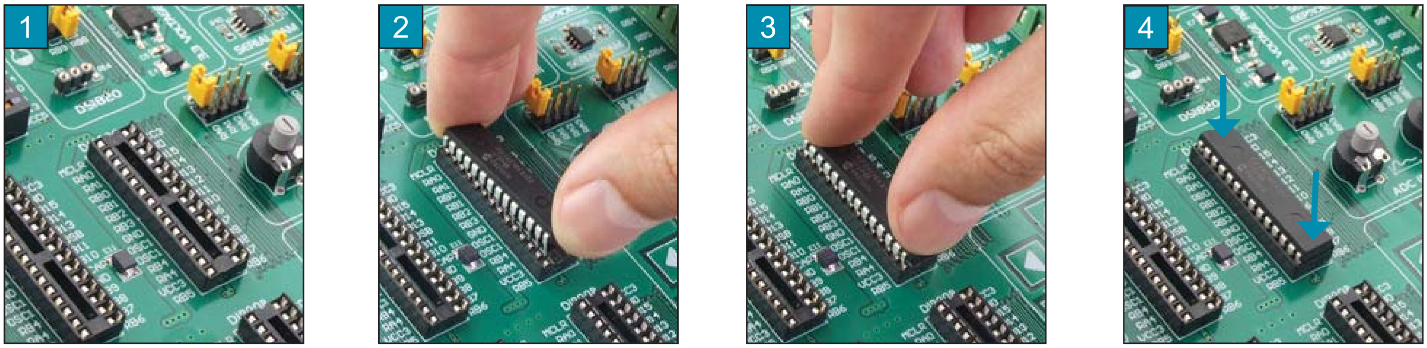


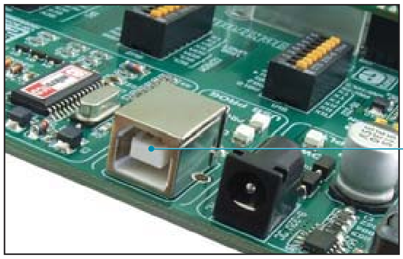
Figure 2-2: Plugging the microcontroller into appropriate socket

Prior to plugging in the microcontroller into the appropriate socket, make sure that the power supply is turned off. Figure 2-2 shows how to plug in a microcontroller correctly. Figure 1 shows an unoccupied DIP28 socket. Place one end of the microcontroller into the socket as shown in Figure 2. Then put the microcontroller slowly down until all the pins match the socket, as shown in Figure 3. Check again that everything is placed correctly and press the microcontroller easily down until it is completely plugged into the socket, as shown in Figure 4.

NOTE: Only one microcontroller may be plugged into the development board at one time.

3.0. On-board LvPICFlash Programmer

A programmer is a necessary tool when working with microcontrollers. It is used to load a hex code into the microcontroller and provides an interface between the microcontroller and a PC. The *Easy24-33 v6* has an on-board *LvPICFlash* programmer. All you need is a .hex file to be loaded into the microcontroller using the *mikroProg Suite for PIC* program. Figure 3-3 shows connection between the compiler, the *mikroProg Suite for PIC* program and the microcontroller.



Programmer's USB connector

Figure 3-1: USB connector's front side

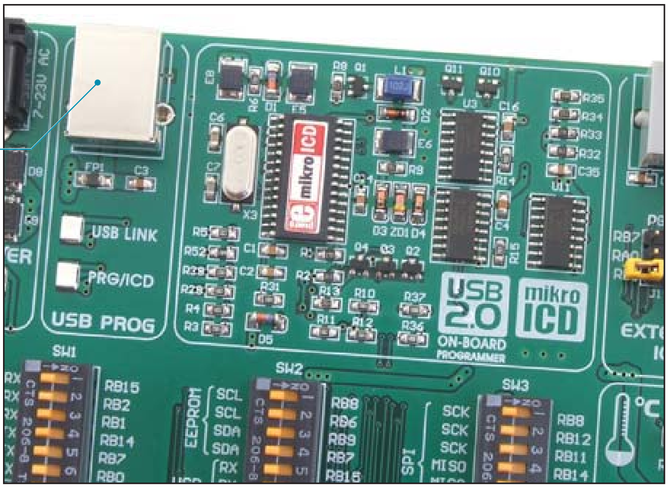
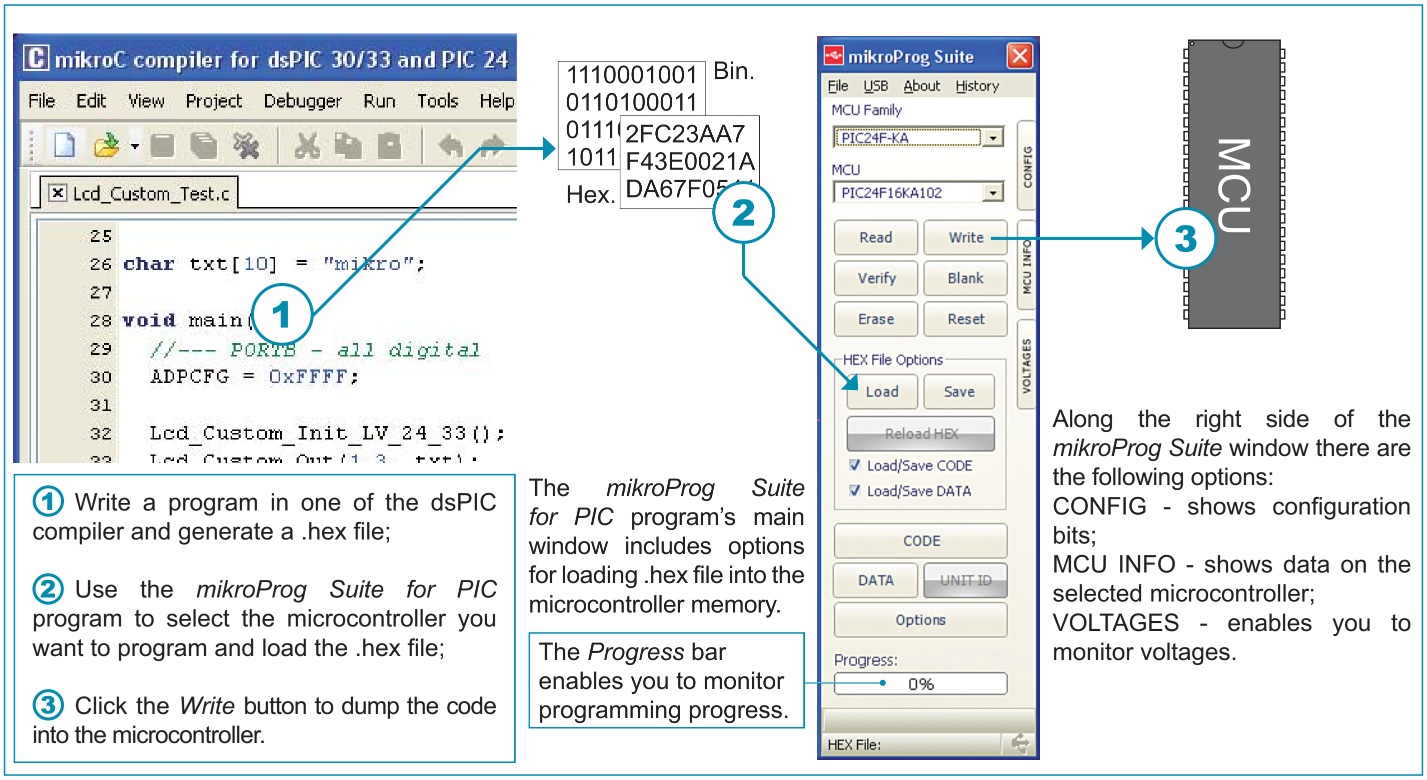


Figure 3-2: On-board programmer



```
char txt[10] = "mikro";

void main(
  //--- PORTB - all digital
  ADPCFG = 0xFFFF;

  Lcd_Custom_Init_LV_24_33();
  Lcd_Custom_Out(1, 3, txt);
```

MCU Family: PIC24F-KA

MCU: PIC24F16KA102

☑ Load/Save CODE

☑ Load/Save DATA

Progress: 0%

1. Write a program in one of the dsPIC compiler and generate a .hex file;

2. Use the *mikroProg Suite for PIC* program to select the microcontroller you want to program and load the .hex file;

3. Click the *Write* button to dump the code into the microcontroller.

The *mikroProg Suite for PIC* program's main window includes options for loading .hex file into the microcontroller memory.

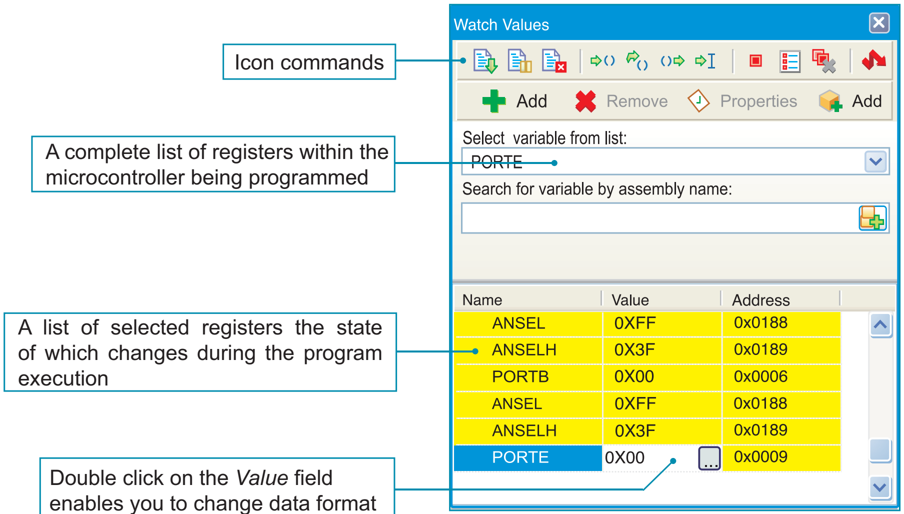
The *Progress* bar enables you to monitor programming progress.

Along the right side of the *mikroProg Suite* window there are the following options:
CONFIG - shows configuration bits;
MCU INFO - shows data on the selected microcontroller;
VOLTAGES - enables you to monitor voltages.

Figure 3-3: Programming process

4.0. mikroICD (In-Circuit Debugger)

The mikroICD (In-Circuit Debugger) is an integral part of the on-board programmer. It is used for testing and debugging programs in real time. The process of testing and debugging is performed by monitoring the state of all registers within the microcontroller while operating in real environment. The mikroICD software is integrated in all PIC compilers designed by Mikroelektronika (mikroBASIC PRO, mikroC PRO, mikroPASCAL PRO, etc.). As soon as the mikroICD debugger starts up, a window called *Watch Values*, appears on the screen, Figure 4-1. The *mikroICD* debugger communicates to the microcontroller through the microcontroller's pins used for programming.



Icon commands

A complete list of registers within the microcontroller being programmed

A list of selected registers the state of which changes during the program execution

Double click on the *Value* field enables you to change data format

Select variable from list: PORTE

Search for variable by assembly name:

Name	Value	Address
ANSEL	0XFF	0x0188
ANSELH	0X3F	0x0189
PORTB	0X00	0x0006
ANSEL	0XFF	0x0188
ANSELH	0X3F	0x0189
PORTE	0X00	0x0009

Figure 4-1: *Watch Values* window

mikroICD debugger options:

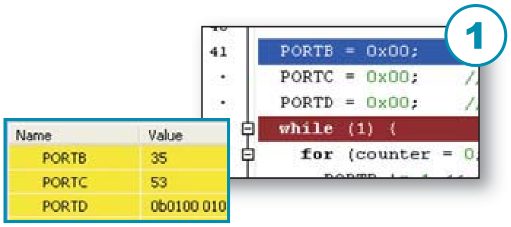
Start Debugger	[F9]
Run/Pause Debugger	[F6]
Stop Debugger	[Ctrl+F2]
Step Into	[F7]
Step Over	[F8]
Step Out	[Ctrl+F8]
Toggle Breakpoint	[F5]
Show/Hide Breakpoints	[Shift+F4]
Clear Breakpoints	[Ctrl+Shift+F5]

Each of these commands is activated via keyboard shortcuts or by clicking the appropriate icon within the *Watch Values* window.

The mikroICD debugger also offers functions such as running a program step by step (single stepping), pausing the program execution to examine the state of currently active registers using breakpoints, tracking the values of some variables etc. The following example illustrates a step-by-step program execution using the *Step Over* command.

Step 1:

In this example the 41st program line is highlighted in blue, which means that it will be executed next. The current state of all registers within the microcontroller can be viewed in the mikroICD *Watch Values* window.



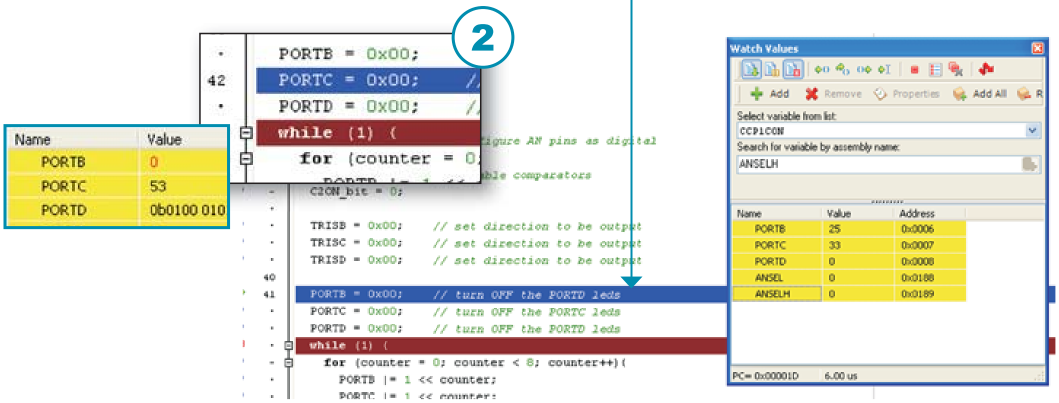
```
41  PORTB = 0x00;
    PORTC = 0x00;      //
    PORTD = 0x00;      //
    while (1) {
      for (counter = 0;
```

Name	Value
PORTB	35
PORTC	53
PORTD	0b0100 010

Step 2:

After the *Step Over* command is executed, the microcontroller will execute the 41st program line. The next line to be executed is highlighted in blue. The state of registers being changed by executing this instruction may be viewed in the *Watch Values* window.

During operation, the program line to be executed next is highlighted in blue, while the breakpoints are highlighted in red. The *Run* command executes the program in real time until it encounters a breakpoint.



```
    PORTB = 0x00;
42  PORTC = 0x00;      //
    PORTD = 0x00;      //
    while (1) {
      for (counter = 0;
```

Name	Value
PORTB	0
PORTC	53
PORTD	0b0100 010

```
                                  figure AN pins as digital
                                  able comparators
    C2ON_bit = 0;

    TRISB = 0x00;      // set direction to be output
    TRISC = 0x00;      // set direction to be output
    TRISD = 0x00;      // set direction to be output
40
41  PORTB = 0x00;      // turn OFF the PORTD leds
    PORTC = 0x00;      // turn OFF the PORTC leds
    PORTD = 0x00;      // turn OFF the PORTD leds
    while (1) {
      for (counter = 0; counter < 8; counter++){
        PORTB |= 1 << counter;
        PORTC |= 1 << counter;
```

Select variable from list: CCP1CON

Search for variable by assembly name: ANSELH

Name	Value	Address
PORTB	25	0x0006
PORTC	33	0x0007
PORTD	0	0x0008
ANSEL	0	0x0188
ANSELH	0	0x0189

PC= 0x00001D 6.00 us

NOTE: For more information on the mikroICD debugger refer to the *mikroICD Debugger* manual.

5.0. ICD Connector

The ICD connector enables communication between the microcontroller and an external ICD debugger/programmer (ICD2 or ICD3) from Microchip. Jumpers J1 and J2 are used for selecting the pin to be fed with programming signals. For the programming signal to be sent to the microcontroller, it is necessary to place both jumpers in the same position, as shown in Figure 5-2. Either RA0 and RA1 or RB0 and RB1 or RB6 and RB7 microcontroller pins can be fed with this signal. The position of jumpers J1 and J2 depends on the microcontroller provided on the board. When the PIC24F16KA102 microcontroller is programmed by means of the external ICD programmer, jumpers J1 and J2 should be placed in the RB0 and RB1 positions, respectively. In case some other microcontroller is used, it is necessary to check which microcontroller pins can be fed with the programming signal.

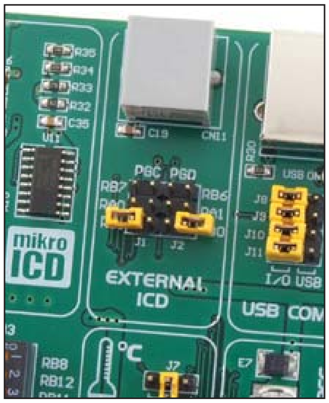


Figure 5-1: ICD connector (default position)

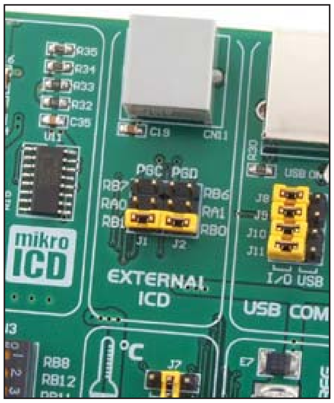


Figure 5-2: Pins RB0 and RB1 are used for programming

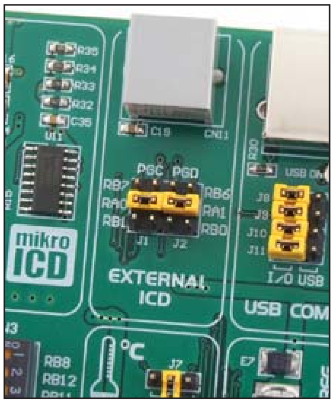


Figure 5-3: Pins RA0 and RA1 are used for programming

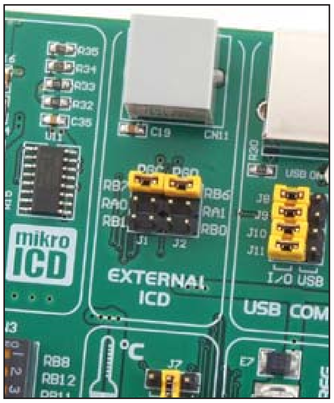


Figure 5-4: Pins RB7 and RB6 are used for programming

External ICD programmer is connected to RB0 and RB1 pins

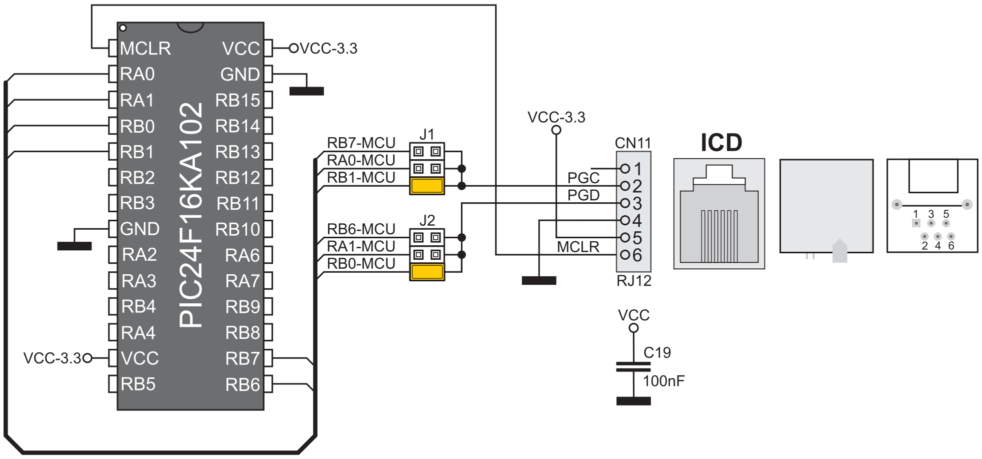


Figure 5-5: ICD connector and microcontroller connection schematic

6.0. Power Supply

The *Easy24-33 v6* development system may use one of two power supply sources:

1. +5V PC power supply through the USB programming cable; and
2. External power supply source connected to an AC/DC connector provided on the development board.

The MC34063A voltage regulator and Gretz rectifier are used to enable external power supply voltage to be either AC (in the range of 7V to 23V) or DC (in the range of 9V to 32V). Jumper J17 is used to select the power supply source. To make advantage of the USB power supply, jumper J17 should be placed in the USB position. When using the external power supply, jumper J17 should be placed in the EXT position. The development system is turned on/off by switching the position of the POWER SUPPLY switch.

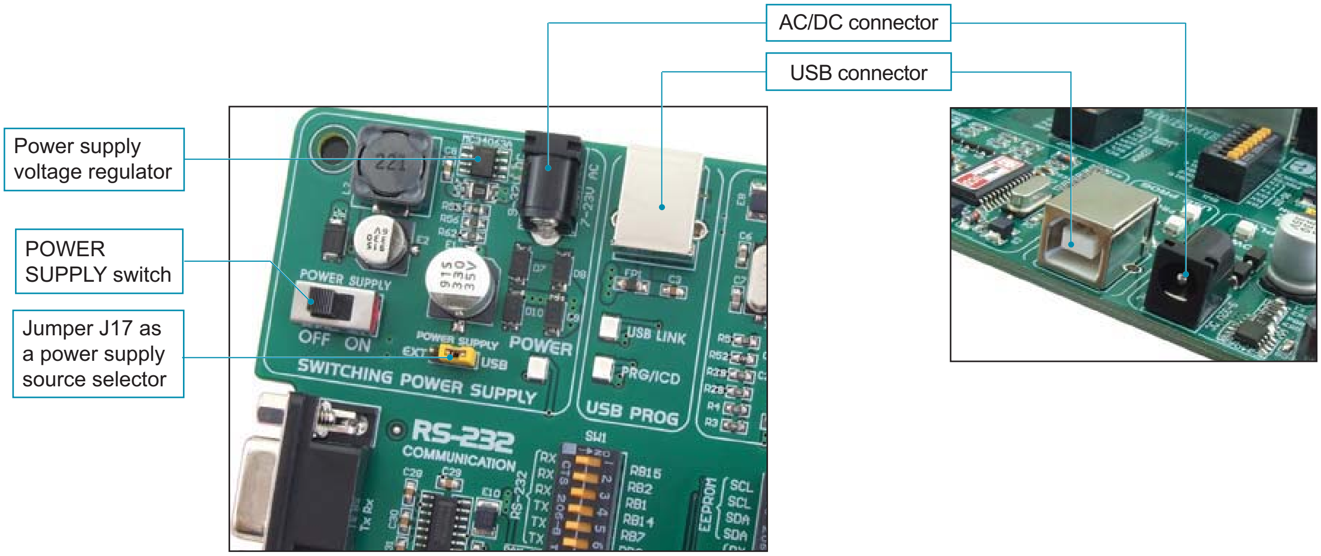

AC/DC connector
USB connector
Power supply voltage regulator
POWER SUPPLY switch
Jumper J17 as a power supply source selector

Figure 6-1: Power supply

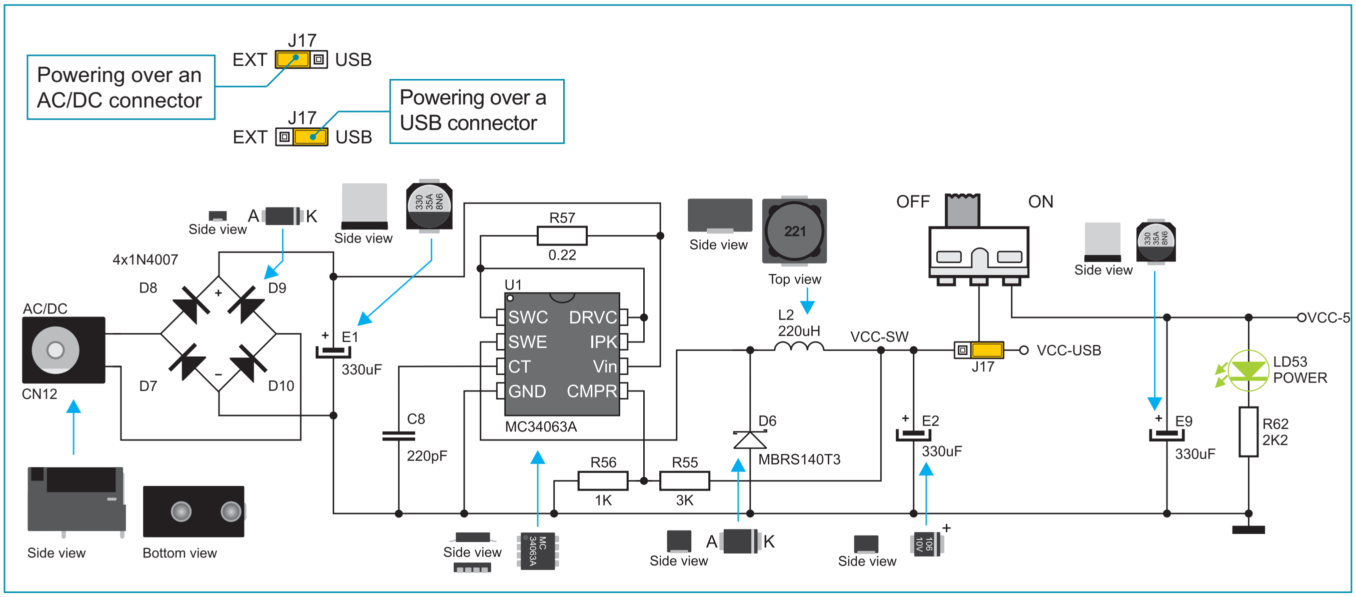


Figure 6-2: Power supply source connection schematic

7.0. A/D Converter Test Inputs

An A/D converter is used for converting an analog voltage into the appropriate digital value. The A/D converter is linear, which means that converted number is linearly dependent on the input voltage value. The A/D converter built into the microcontroller converts an analog voltage into a 10-bit number. Potentiometers P1 and P2 enable voltage to vary between 0 and 3.3V. The microcontroller with a built-in A/D converter is supplied with this voltage via test inputs. Jumpers J14 and J15 are used for selecting either RB0 - RB3 or RA0 - RA3 pins to be supplied with A/D conversion voltage. Resistors R63 and R64 have a protective function and are used to limit current flow through the potentiometer or the microcontroller pin.

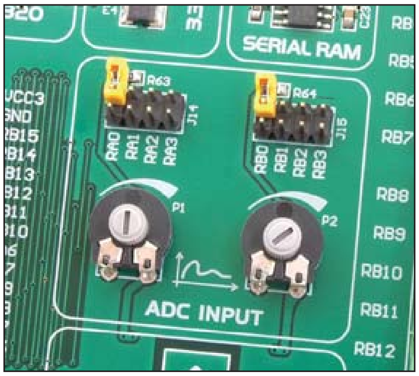


Figure 7-1: A/D converter (jumpers in default position)

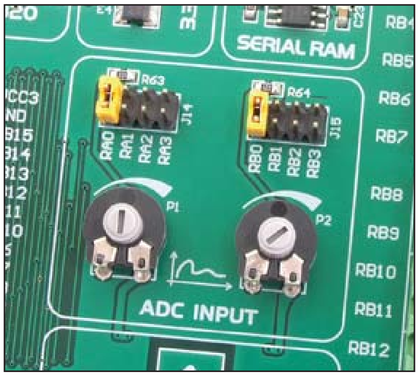


Figure 7-2: Pins RA0 and RB0 used as input pins for A/D conversion

A/D conversion is performed via RB0 and RA0 microcontroller pins

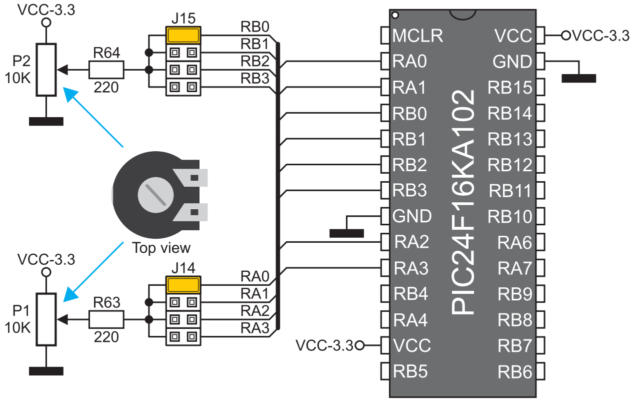


Figure 7-3: A/D converter test inputs and microcontroller connection schematic

NOTE: In order to enable the microcontroller to accurately perform A/D conversion, it is necessary to turn off LEDs and pull-up/ pull-down resistors on the port pins used by the A/D converter.

8.0. Serial EEPROM Module

EEPROM (Electrically Erasable Programmable Read-Only Memory) is a built-in memory module used to store data that should be saved when the power supply goes off. The 24AA01 circuit can store up to 1Kbit data and communicates with the microcontroller through pins RB6 - RB9 using I^{2}C communication. In order to enable connection between the EEPROM module and the microcontroller, it is necessay to set switches 1 (RB8) and 3 (RB9) on the DIP switch SW2 to the ON position. Switches 2 (RB6) and 4 (RB7) on the DIP switch SW2 may be optionally used.

Serial EEPROM is connected to the microcontroller via pins RB8 and RB9

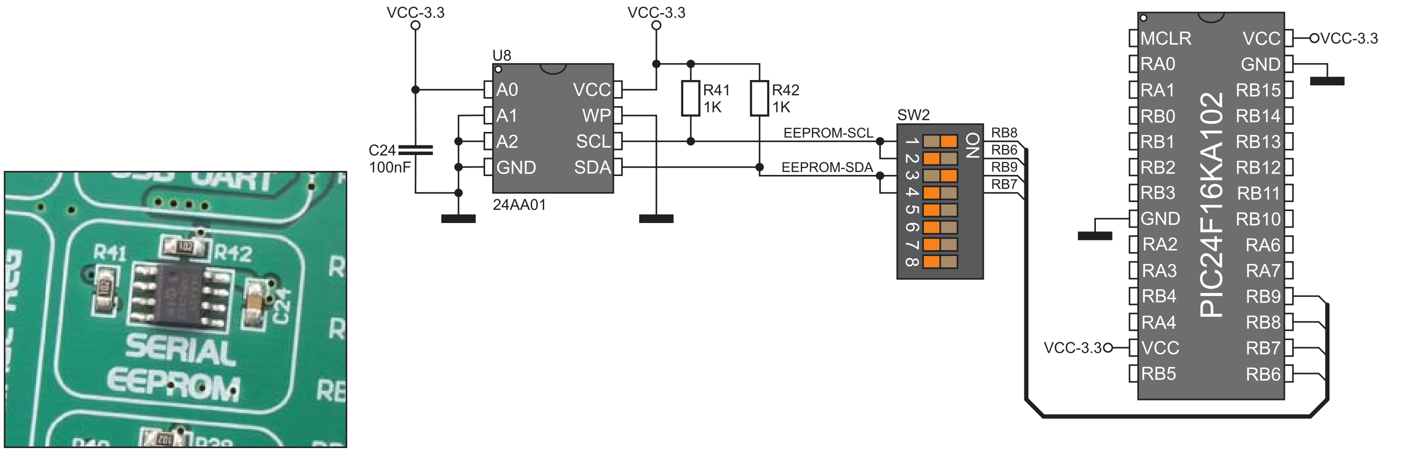


Figure 8-1: Serial EEPROM module and microcontroller connection schematic

9.0. USB Communication Module

The *Easy24-33 v6* development system can communicate with external devices through the USB connector CN9. Only microcontrollers with the built-in USB communication module are capable of using USB communication. Jumpers J8, J9, J10 and J11 are used to determine the fuction of microcontroller pins RB6, RB10, RB11 and RB12. When these jumpers are in the USB position (Figure 9-1), the microcontroller pins are used for USB communication. When they are in the I/O position, the microcontroller pins are used as I/O pins.

Jumpers J8, J9, J10 and J11 are in USB position

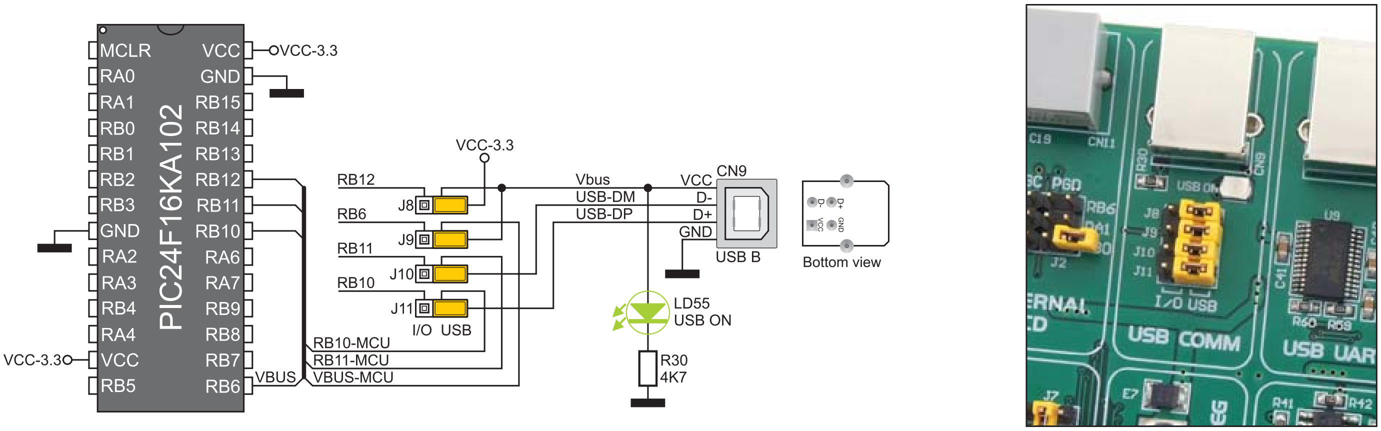
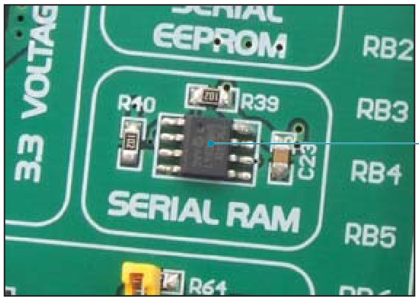


Figure 9-1: USB connector CN9 and microcontroller connection schematic

10.0. Serial RAM Module

Owing to the built-in RAM module, the *Easy24-33 v6* development system is provided with 64Kbit RAM module that can be connected to the microcontroller via SPI interface. The function of the serial RAM module is to expand available RAM of the microcontroller. To establish connection between the microcontroller and RAM module, it is necessary to enable SPI communication by means of the DIP switch SW3. In the example in Figure 10-2, RAM module is connected to the RB8, RB14 and RB9 microcontroller pins. Other microcontroller pins RB12, RB11, RB10 and RB13 may also be optionally used here. The function of the RAM-HOLD# (RB7) switch on the DIP switch SW1 is to hold communication between the microcontroller and RAM module. The RAM-CS (RB15) switch on the DIP switch SW1 is used to control the access to RAM.



64Kbit RAM module can be added to the RAM module built into the microcontroller

Figure 10-1: Serial RAM module

Serial RAM is connected to the microcontroller via RB15, RB7, RB8, RB14 and RB9 pins

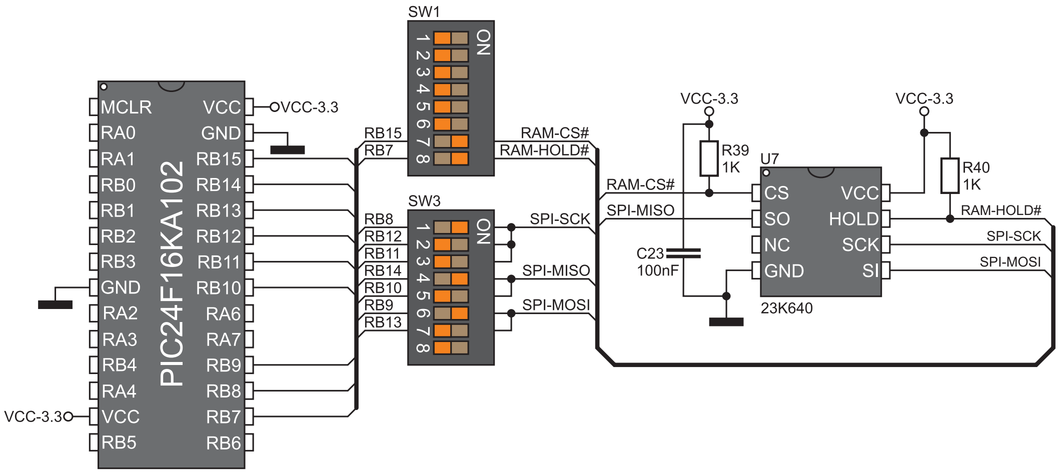


Figure 10-2: Serial RAM module and microcontroller connection schematic

11.0. RS-232 Communication Module

The *Easy24-33 v6* development system features the RS-232 module that serves as an interface between the microcontroller and an external device using RS-232 communication to exchange data. The RS-232 module is connected to that device through a 9-pin SUB-D connector. For communication with the microcontroller, this module uses TX pin for sending and RX pin for receiving data. For the microcontroller to be capable of exchanging data with the RS-2323 module, it is necessary to be provided with a built-in USART module. Connection between the RS-232 module and the microcontroller is established by setting the appropriate switches on the DIP switch SW1 to the ON position. The RS-232 module's TX pin can be connected to the microcontroller via either RB14, RB7 or RB0 microcontroller pin, whereas the RX pin can use either RB14, RB7 or RB0 microcontroller pin for such connection. In the example in Figure 11-2, the microcontroller is connected to the RS-232 module via RB15 and RB14 pins.
In order to enable USART module of the microcontroller to receive input signals which meet the RS-232 standard, it is necessary to adjust voltage levels using an IC circuit such as MAX3232 (MAX232).

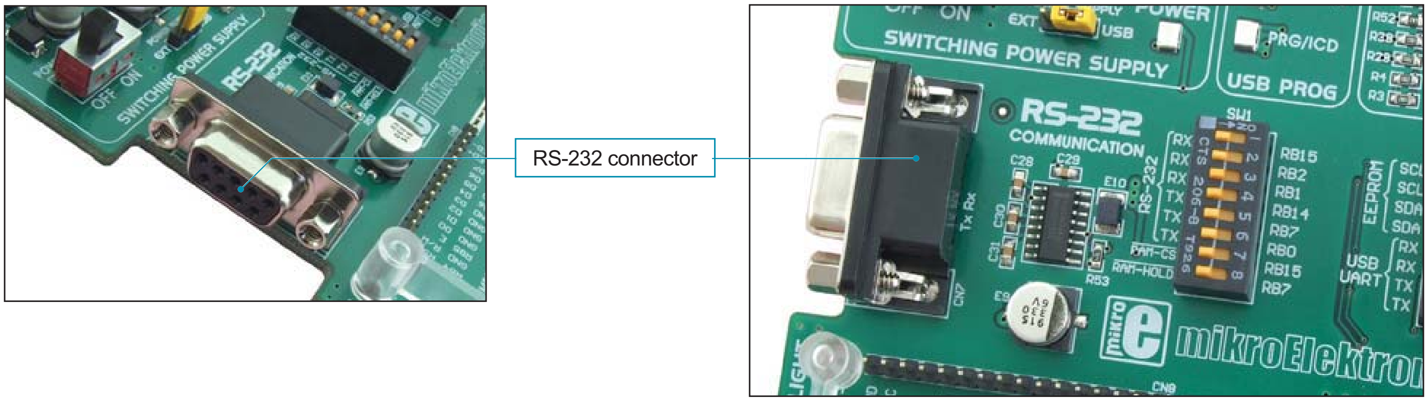


Figure 11-1: RS-232 module

The function of switches 1-6 on the DIP switch SW1 is to determine which microcontroller pins are to be used as RX and TX lines. Refer to figure 11-2.

Port RS-232 is connected to the microcontroller

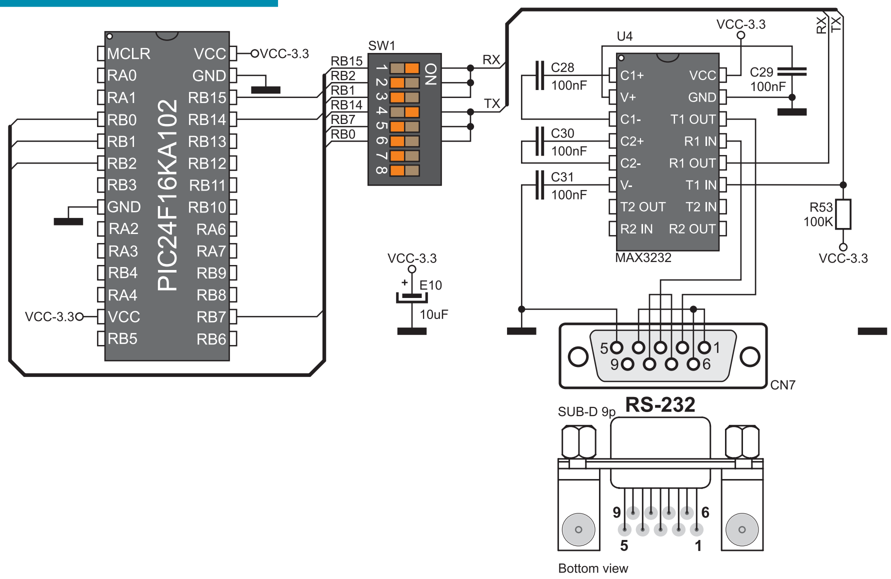


Figure 11-2: RS-232 module and microcontroller connection schematic

NOTE: Make sure that your microcontroller is provided with the USART module as it is not necessarily integrated in all PIC24 microcontrollers.

12.0. CAN Communication Module

CAN (Controller Area Network) is a communication standard primarily intended for use in automotive industry. It enables the microcontroller to communicate to a car device without using a host PC. In addition, such communication is widely used in industrial automation. The *Easy24-33 v6* uses the MCP2551 circuit for CAN communication. This circuit provides an interface between the microcontroller and some peripheral device. To enable connection between the microcontroller and MCP2551, it is necessary to set switches 7(RX) and 8(TX) on the DIP switch SW4 to the ON position. The RX line is connected to the microcontroller via the RB15 pin, whereas the TX line is connected to the microcontroller via the RB14 pin.

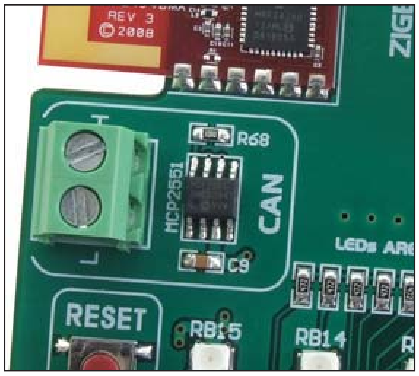


Figure 12-1: CAN module

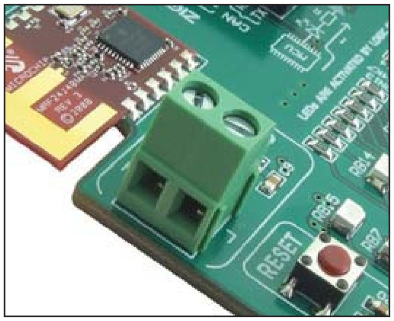


Figure 12-2: CAN module connector

CAN communication is enabled via DIP switch SW4

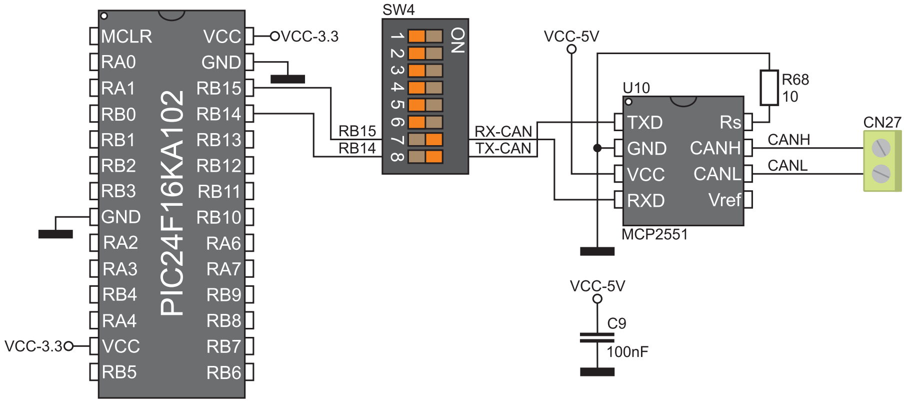
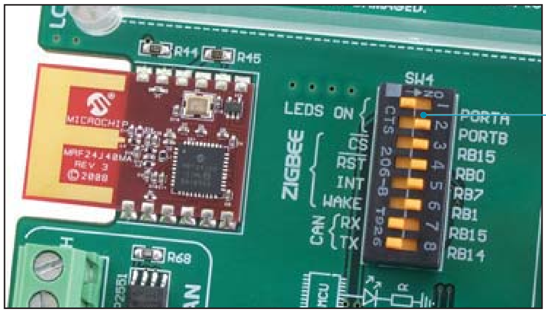


Figure 12-3: Microcontroller and CAN module connection schematic

13.0. ZigBee Communication Module

The operation of the ZigBee communication module is based on the IEEE 802.15.4 standard that defines wireless data transfer on short distances with a low power consumption. The *Easy24-33 v6* development system features the MRF24J40MA ZigBee module. Some of its key features are: 250kbps data rate, 2.4GHz operating frequency, ~20mA power consumption, open area range up to 400m, etc. Communication between the microcontroller and ZigBee module is performed through SPI interface. In order to enable the operation of the ZigBee module, it is necessary to set switches 3, 4, 5 and 6 on the DIP switch SW4 to the ON position. It is also necessary to enable SPI communication through the DIP switch SW3. Which of the switches on the DIP switch SW3 is to be used depends on the arrangement of the following microcontroller pins RB8, RB12, RB11, RB14, RB10, RB9 and RB13. In Figure 13-3, pins RB8, RB14 and RB9 are used for serial communication.



ZigBee module and microcontroller are connected through the DIP switch SW4

Serial SPI communication between ZigBee module and microcontroller is enabled through the DIP switch SW3

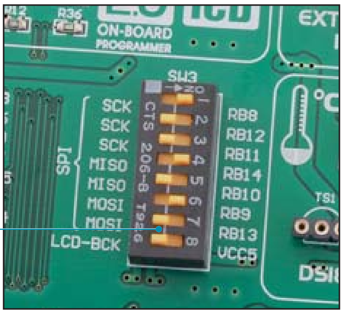


Figure 13-1: MRF24J40MA ZigBee module

Figure 13-2: DIP switch SW3

ZigBee module is connected to the microcontroller through the DIP switches SW3 and SW4

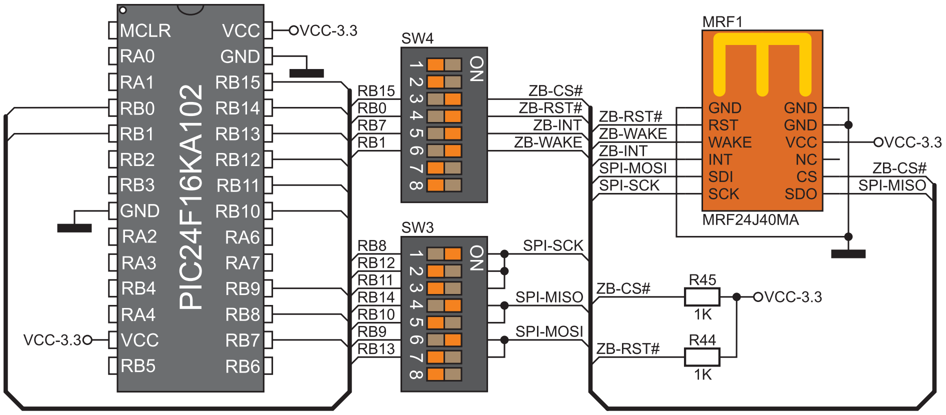


Figure 13-3: ZigBee module and microcontroller connection schematic

14.0. USB UART Module

The USB UART module is an interface between an external USB device and the serial USB module integrated in the microcontroller. In order to establish connection between the microcontroller and USB UART module, it is necessary to set switches 5 (RB15) and 7 (RB14) on the DIP switch SW2 to the ON position. In addition, switches 6 (RB1) and 8 (RB0) on the DIP switch SW2 may be optionally used.

When connecting a UART device and the microcontroller, it is necessary to cross connect data send and data receive lines. In other words, the UART's RX pin should be connected to the microcontroller's TX pin, while the UART's TX pin should be connected to the microcontroller's RX pin. It is not possible to directly connect UART's and microcontroller's TX lines since the microcontroller wouldn't be able to receive nor send data.

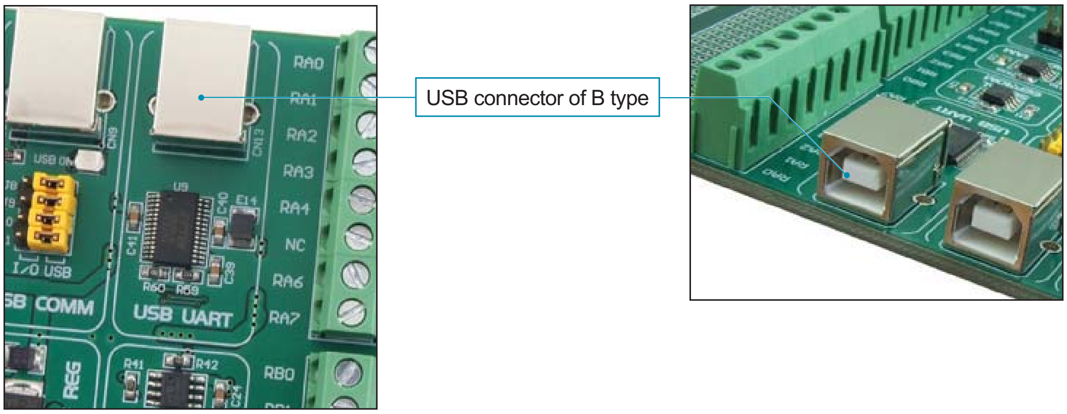

USB connector of B type

Figure 14-1: USB UART module

USB connector of B type is connected to the microcontroller via the FT232RL circuit

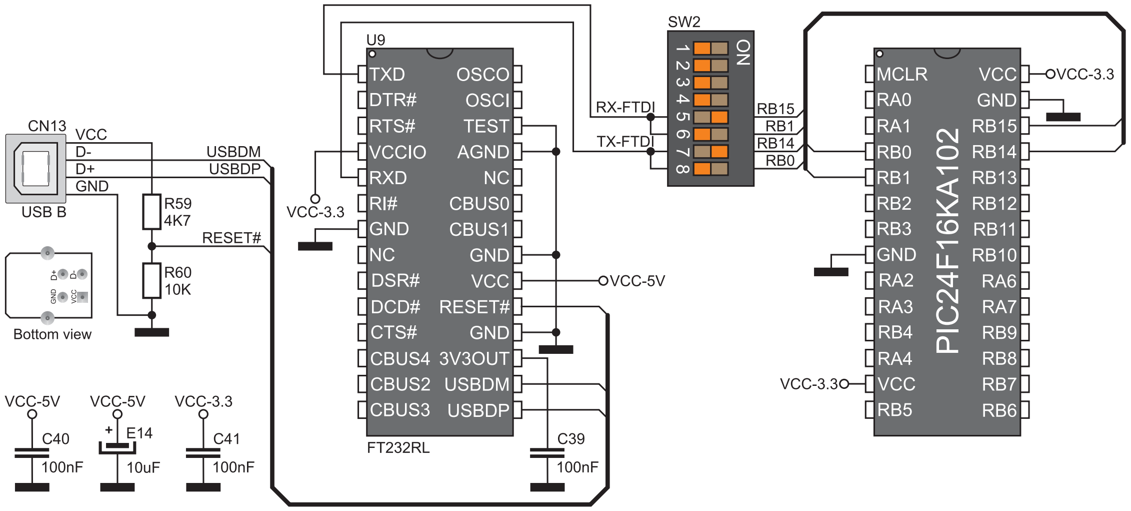
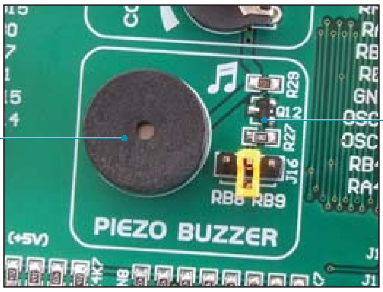


Figure 14-2: USB UART module and microcontroller connection schematic

15.0. Piezo Buzzer

Due to a built-in piezo buzzer, the development system is capable of emitting audio signals. For the piezo buzzer to operate normally it is necessary to generate a voltage signal of specified frequency. The voltage signal is generated in the microcontroller by the appropriate code written to its memory. Remember, when writing the voltage signal generation code, the piezo buzzer's resonant frequency is 3.8kHz. In addition, other frequencies in the range between 20Hz and 20kHz can be used, but the best performance is provided by frequencies ranging between 2kHz and 4kHz. The voltage signal can be generated via the RB8 or RB9 microcontroller pin. Jumper J16 is used to connect the piezo buzzer to the microcontroller pin used for voltage signal generation. In the example shown in Figure 15-2, the voltage signal is generated through the RB8 pin.



Piezo buzzer's resonant frequency 3.8kHz

Transistor Q12 is used to amplify voltage signal generated by the microcontroller

Figure 15-1: Piezo buzzer

Piezo buzzer is connected to the microcontroller via the RB8 pin

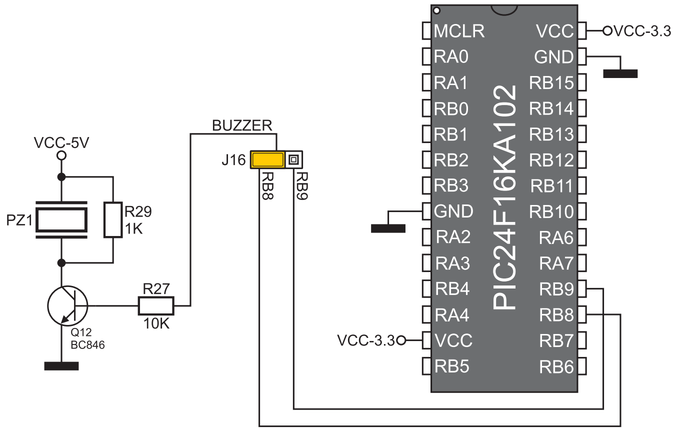


Figure 15-2: Piezo buzzer and microcontroller connection schematic

16.0. DS1820 Temperature Sensor

1-wire® serial communication enables data to be transferred over one single communication line while the process itself is under control of the master device. The advantage of such communication is that only one microcontroller pin is used. All slave devices have by default a unique ID code, which enables the master device to easily identify all devices sharing the same communication interface.

The DS1820 is a temperature sensor that communicates with the microcontroller via the 1-wire communication. It is capable of measuring temperatures within the range of -55 to 125°C and provides ±0.5°C accuracy for temperatures within the range of -10 to 85°C. A power supply voltage of 3.3V is required for its operation. It takes maximum 750ms for the DS1820 to calculate temperature with a 9-bit resolution. The *Easy24-33 v6* development system provides a separate socket for the DS1820. It may use either RB8 or RB9 pin for communication with the microcontroller. Pins to be used for 1-wire communication are selected by jumper J7. In the example in Figure 16-4, the RB9 pin is used for 1-wire communication with the microcontroller.

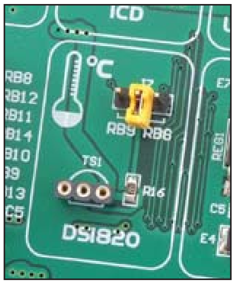


Figure 16-1: DS1820 connector (DS1820 is not placed)

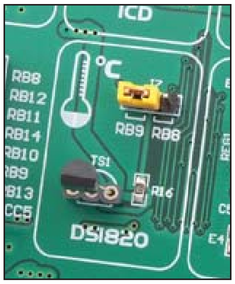


Figure 16-2: J7 in RB9 position (DS1820 is connected to the RB9 pin)

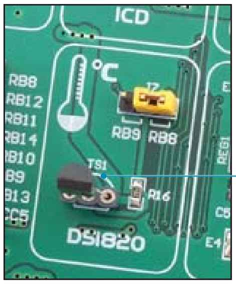


Figure 16-3: J7 in RB8 position (DS1820 is connected to the RB8 pin)

NOTE: Make sure that half-circle on the board matches the round side of the DS1820

Jumper J7 is placed in the RB9 position

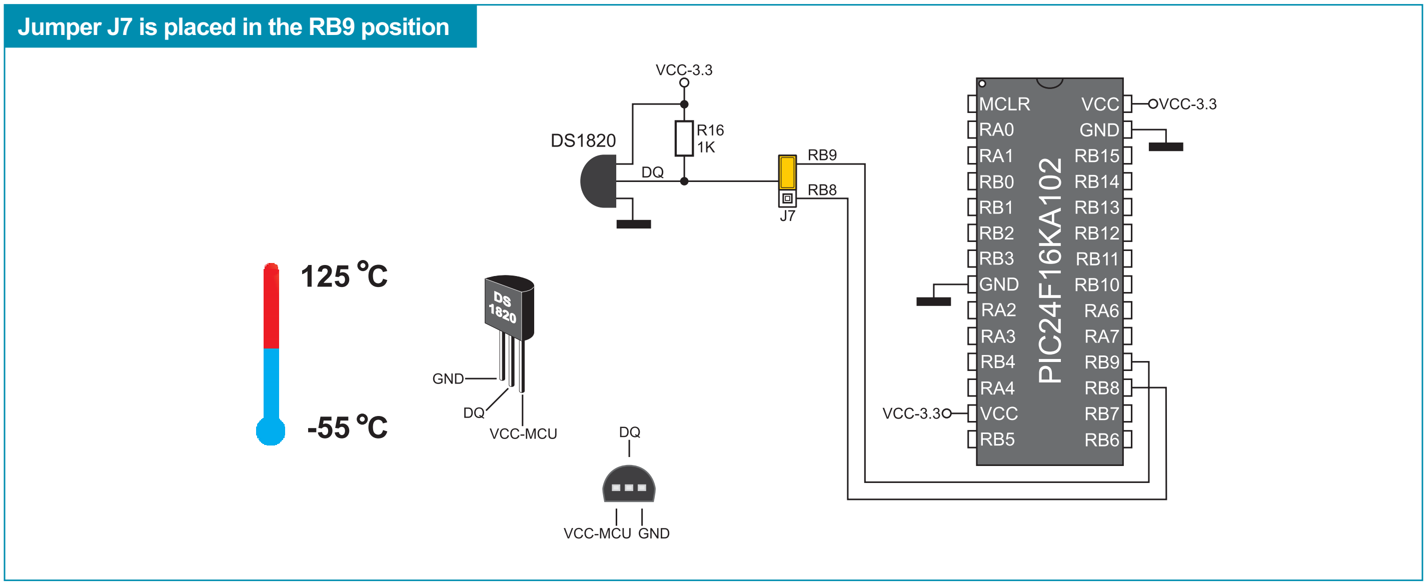


Figure 16-4: DS1820 temperature sensor and microcontroller connection schematic

17.0. LEDs

LED (Light-Emitting Diode) is a highly efficient electronic light source. When connecting LEDs, it is necessary to use a current limiting resistor. A common LED voltage is approximately 2.5V, while the current varies from 1 to 20mA depending on the type of LED. The *Easy24-33 v6* uses LEDs with current I=1mA.

There are 23 LEDs on the *Easy24-33 v6* development system which visually indicate the state of each microcontroller I/O pin. An active LED indicates that a logic one (1) is present on the pin. In order to enable the pin state to be shown, it is necessary to select appropriate port (PORTA or PORTB) using the DIP switch SW4.

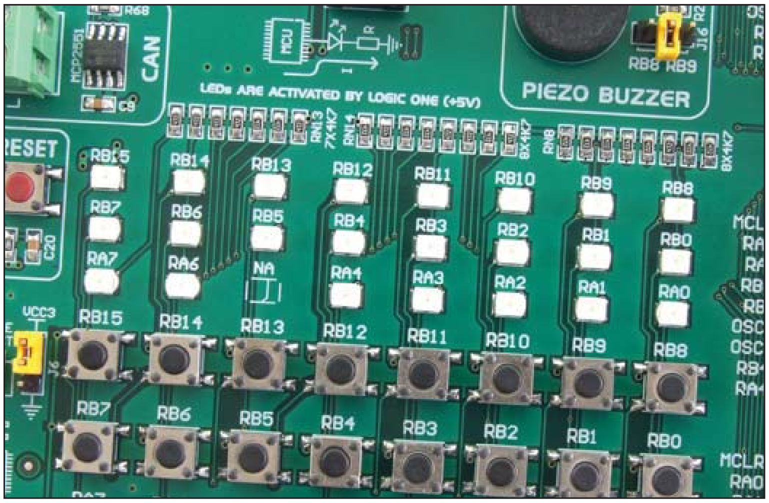
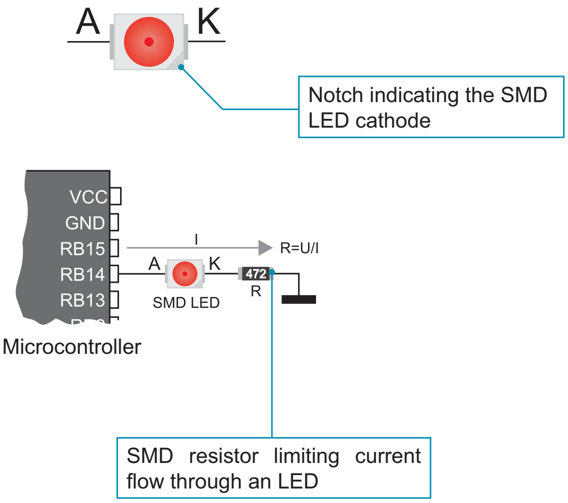


Figure 17-1: LEDs


A
K
Notch indicating the SMD LED cathode
VCC
GND
RB15
RB14
RB13
I
R=U/I
A
K
SMD LED
472
R
Microcontroller
SMD resistor limiting current flow through an LED

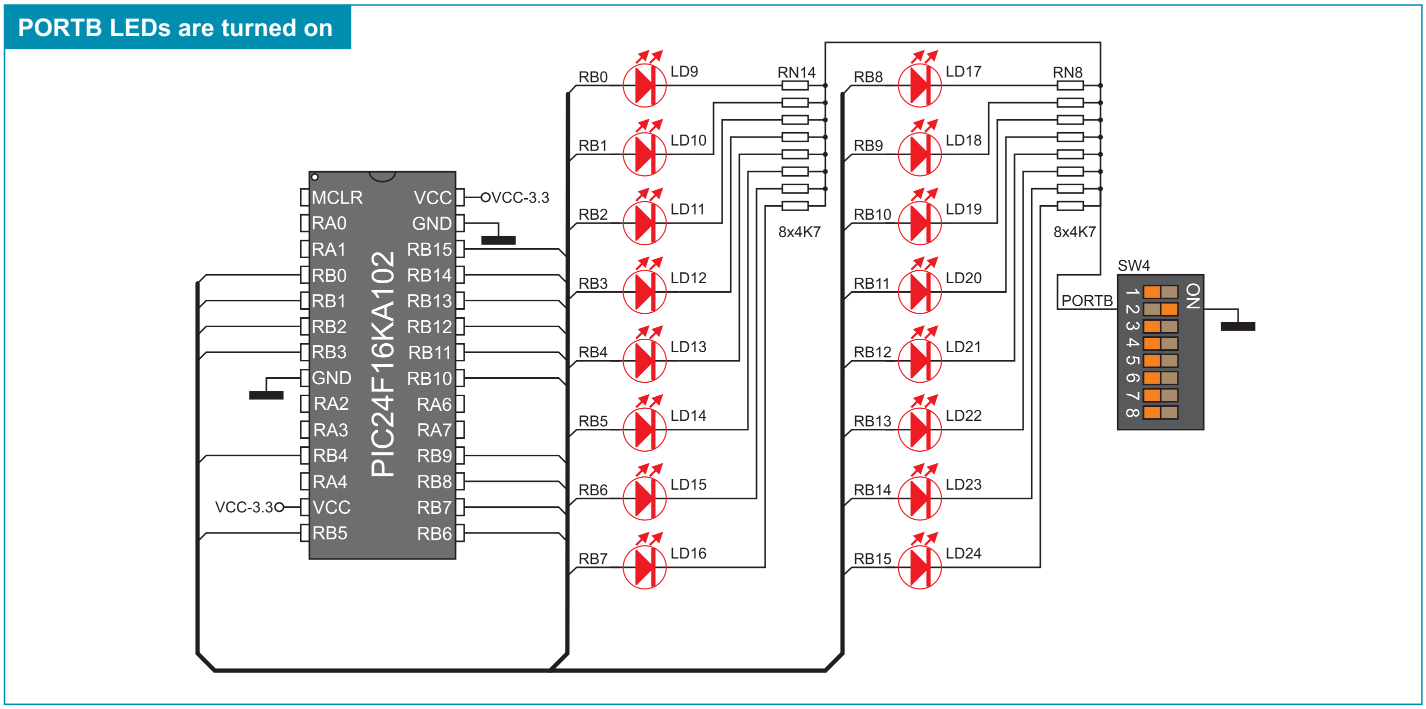


Figure 17-2: Microcontroller's PORTB and LEDs connection schematic

18.0. Push Buttons

The logic state of all microcontroller input pins may be changed by means of push buttons. Jumper J6 is used to determine the logic state to be applied to the desired microcontroller pin by pressing the appropriate push button. The function of the protective resistor is to limit the maximum current, thus preventing the development system and peripheral modules from being damaged in case a short circuit occurs. If needed, advanced users may shorten such resistor using jumper J5. Right next to the push buttons, there is a RESET button which is not directly connected to the MCLR microcontroller pin. The reset signal is generated by the programmer.

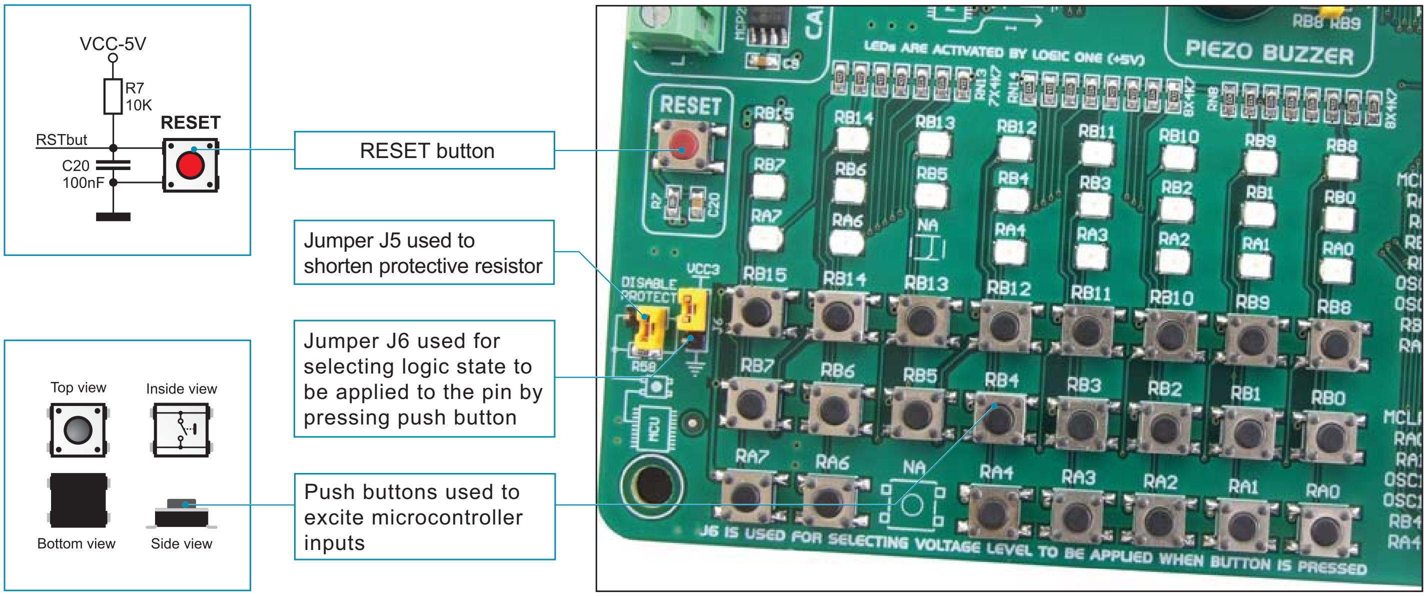

VCC-5V
R7
10K
RESET
RSTbut
C20
100nF
RESET button
Jumper J5 used to shorten protective resistor
Jumper J6 used for selecting logic state to be applied to the pin by pressing push button
Push buttons used to excite microcontroller inputs
Top view
Inside view
Bottom view
Side view
LEDs ARE ACTIVATED BY LOGIC ONE (+5V)
PIEZO BUZZER
DISABLE PROTECT
VCC3
J6 IS USED FOR SELECTING VOLTAGE LEVEL TO BE APPLIED WHEN BUTTON IS PRESSED

Figure 18-1: Push buttons

By pressing any push button when jumper J6 is in the VCC-3.3, a logic one (3.3V) will be applied to the appropriate microcontroller pin, as shown in Figure 18-2.

By pressing a push button, the appropriate pin will be driven high (1)

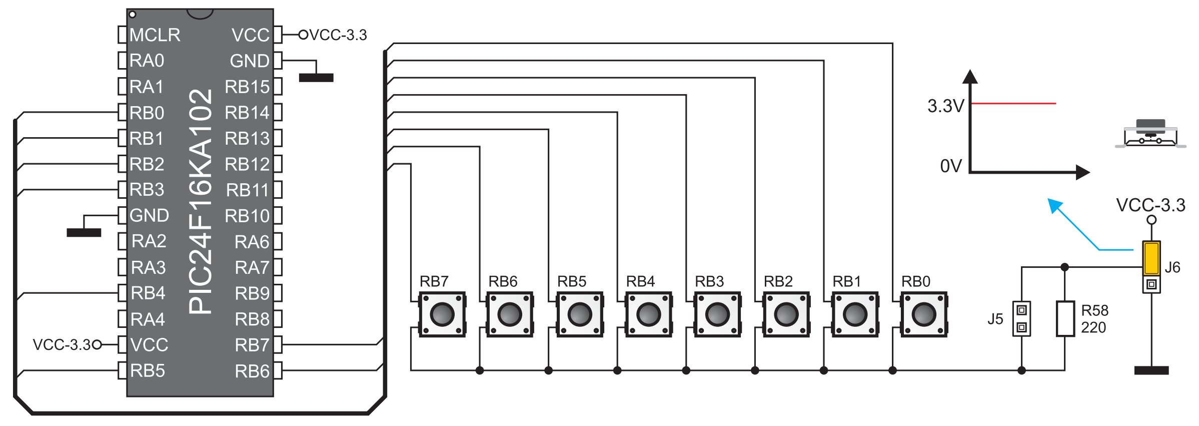
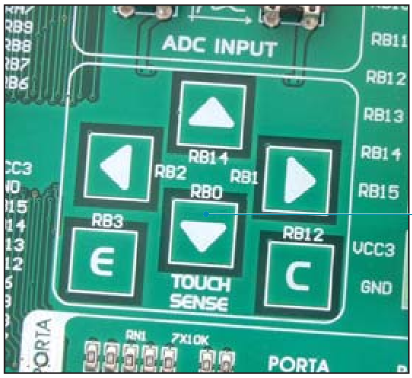


Figure 18-2: Microcontroller’s PORTB and push buttons connection schematic

19.0. Touch Sense Keypad

The *Easy24-33 v6* development system is provided with a set of touch sensitive buttons making the navigation touch sense keypad. The touch sense buttons are arrow-like oriented left, right, up and down. There are also two additional push buttons marked as E and C that can be assigned ENTER and CANCEL functions, respectively. Touch sense buttons are connected to push buttons having the same names that are also connected to the microcontroller's port PORTB. Their function is determined by the programmer while writing program for the microcontroller.

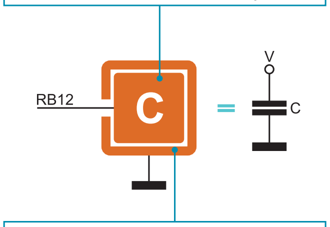
The operation of the touch sense buttons is based on the sensor capacitance change. A touch sense button consists of a small copper surface connected to the microcontroller pin. Around this button, there is a copper surface connected to the ground. These two isolated surfaces form a capacitor. A finger press causes the capacitor capacitance to be changed, which further causes the voltage present on the microcontroller pin to be changed as well. Touch sense buttons are connected to the microcontroller via its pins connected to the built-in A/D converter. Some microcontrollers feature a special CTMU module (Charge Time Measurement Unit) used to measure changes in capacitance. This module can also be used to measure exact time, relative capacitance changes as well as to generate pulses independently from the oscillator.



Have in mind when writing program for the microcontroller that touch sense keypad is connected to port PORTB

Figure 19-1: Touch sense keypad

Copper surface is connected to the RB12 microcontroller pin



It is surrounded by another copper surface connected to the ground

The operation of the touch sense buttons is based on the capacitance change

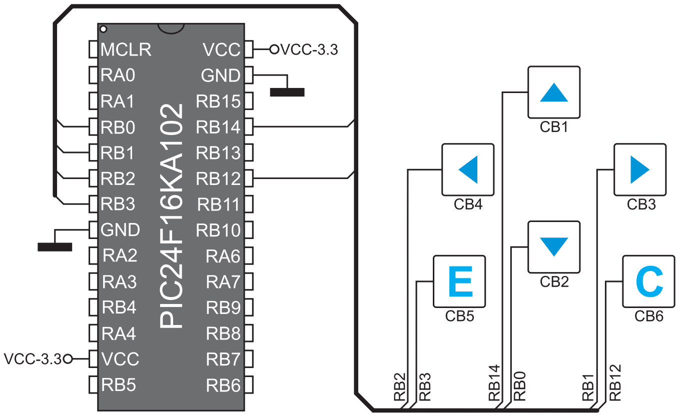


Figure 19-2: Touch sense keypad and microcontroller connection schematic

NOTE: Make sure that pull-up/pull-down resistors on pins RB0, RB1, RB2, RB3, RB12 and RB14 are disabled prior to using the touch sense keypad.

20.0. 2x16 LCD

The *Easy24-33 v6* development system provides an on-board connector for the alphanumeric 2x16 LCD. This connector is linked to the microcontroller via ports PORTA and PORTB. Potentiometer P4 is used to adjust display contrast. Switch LCD-BCK on the DIP switch SW3 is used to turn the display backlight on/off.

Communication between the LCD and the microcontroller is performed in a 4-bit mode. Alphanumeric digits are displayed in two lines each containing up to 16 characters of 7x5 pixels.

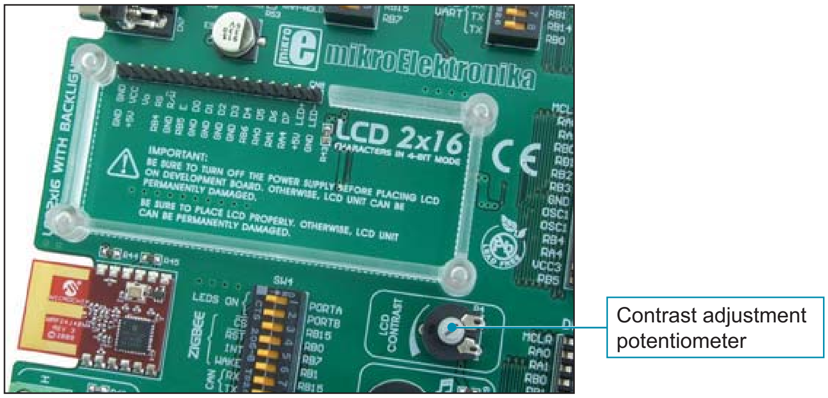


Figure 20-1. Connector for alphanumeric LCD

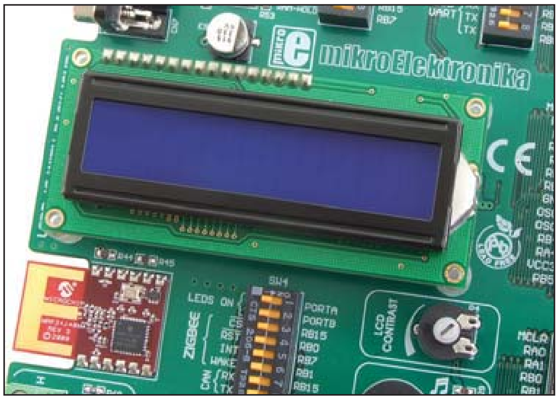


Figure 20-2: Alphanumeric 2x16 LCD

LCD display backlight is on

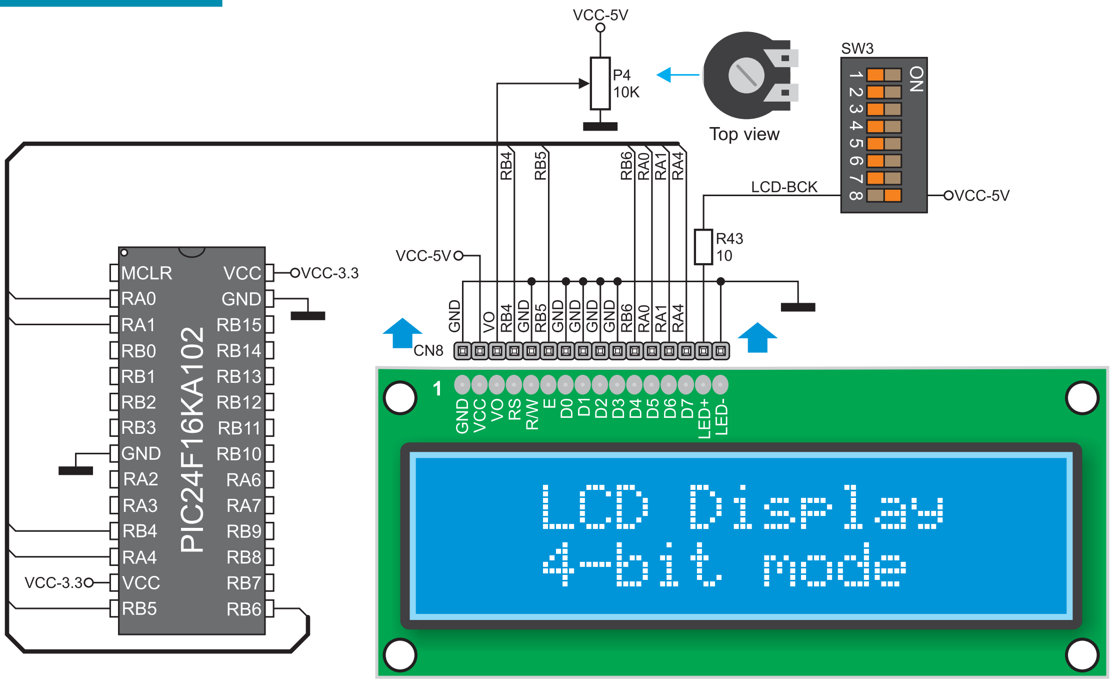


Figure 20-3: Alphanumeric 2x16 LCD and microcontroller connection schematic

21.0. Input/Output Ports

Six 10-pin connectors linked to the microcontroller's I/O ports are provided on the development system. Pins RA0, RA1, RB0, RB1, RB6 and RB7 are used for programming and therefore are not directly connected to the appropriate 10-pin connector, but via the programmer's multiplexer. In addition to these 10-pin connectors, the development system includes screw connectors and pads suitable for placing additional components. Jumpers next to the PORTA and PORTB ports' 10-pin connectors enable connection between pull-up/pull-down resistors and the microcontroller pins.

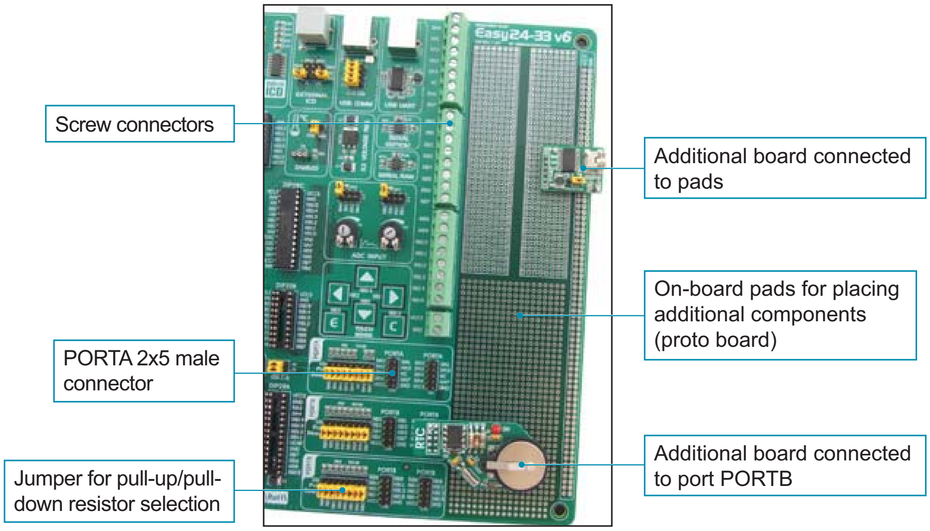

Screw connectors
PORTA 2x5 male connector
Jumper for pull-up/pull-down resistor selection
Additional board connected to pads
On-board pads for placing additional components (proto board)
Additional board connected to port PORTB
Easy24-33 v6

Figure 21-1: I/O ports

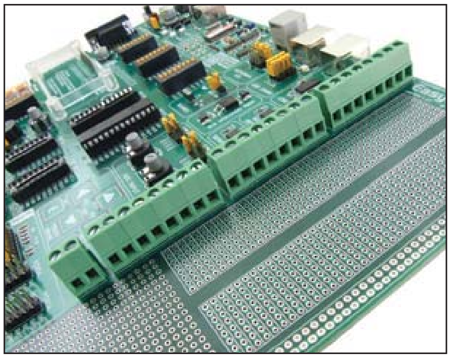


Figure 21-2: Screw connectors are connected to I/O pins

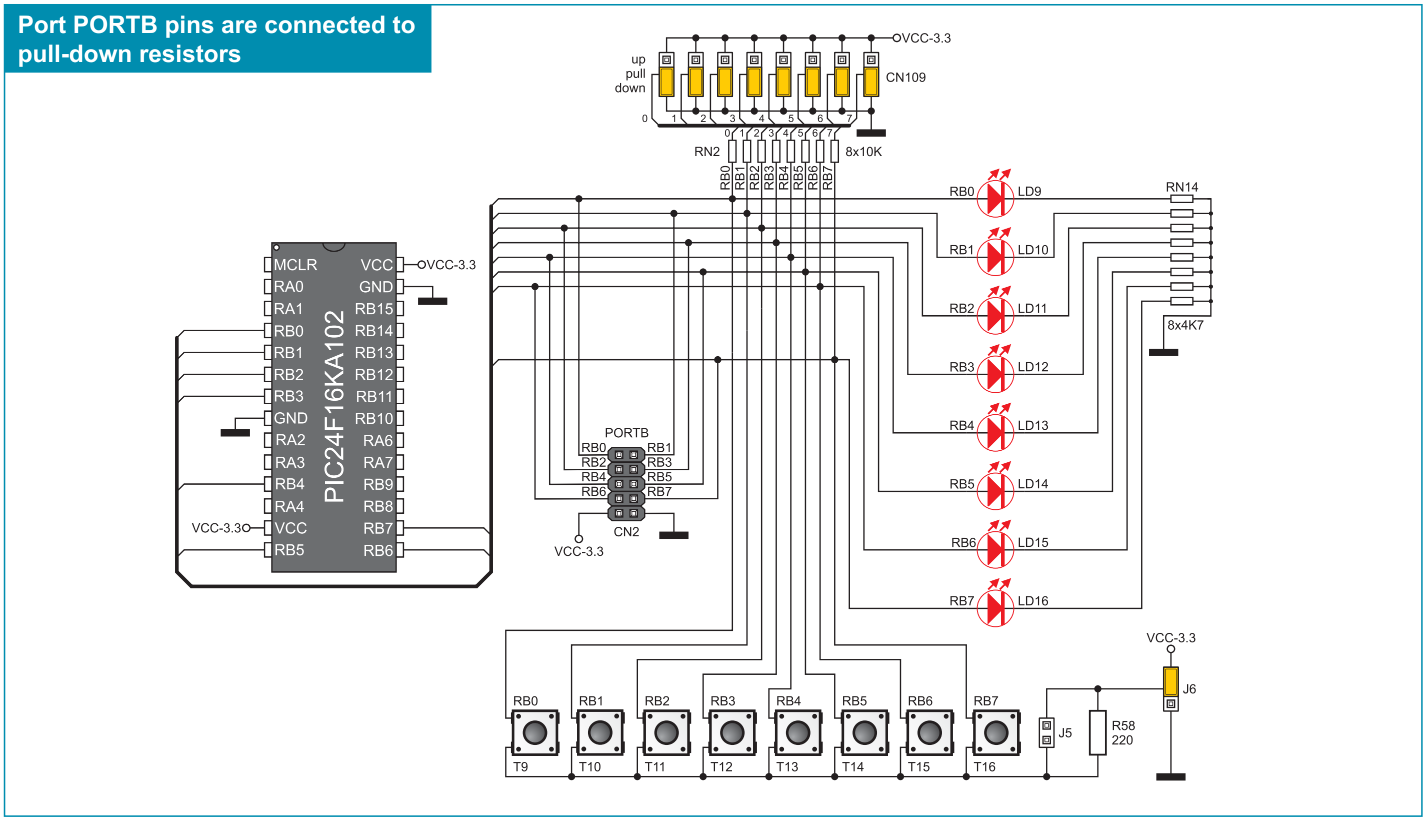
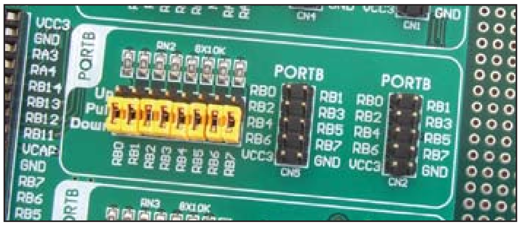


Figure 21-3: Port PORTB connection schematic

Pull-up/pull-down resistors enable you to set the logic level on all microcontroller's input pins when they are in idle state. Such level depends on the position of the pull-up/pull-down jumper. The microcontroller's RB8 pin with the relevant jumpers CN110 and RB8 push button with jumper J6 are used here for the purpose of explaining the performance of pull-up/pull-down resistors. The principle of their operation is the same as for all other microcontroller pins.

In order to enable the PORTB port pins to be connected to pull-down resistors, it is necessary to place jumpers CN110 in the *Down* position first. This enables any PORTB port pin to be supplied with a logic zero (0V) in idle state over jumpers CN110 and 8x10k resistor network. As a result, every time you press the RB8 push button, a logic one (3.3V) will appear on the RB8 pin, provided that jumper J6 is placed in the VCC-3.3 position.



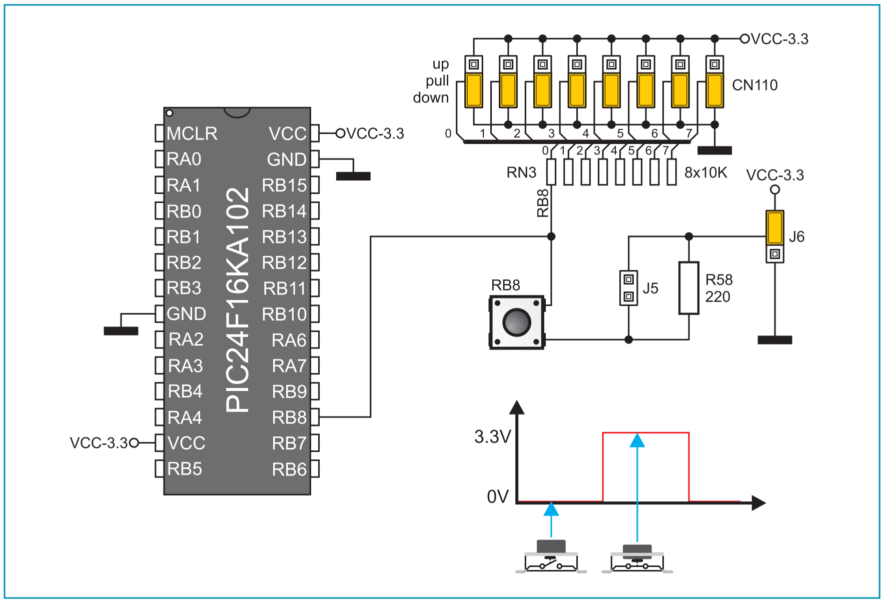
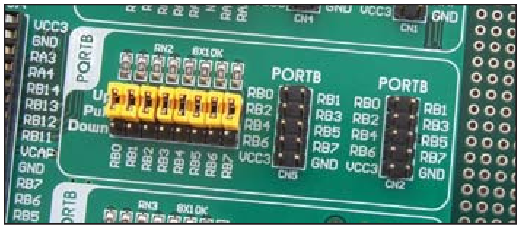

VCC-3.3
up
pull
down
CN110
0 1 2 3 4 5 6 7
RN3
8x10K
RB8
VCC-3.3
J6
J5
R58
220
RB8
MCLR RA0 RA1 RB0 RB1 RB2 RB3 GND RA2 RA3 RB4 RA4 VCC RB5
VCC GND RB15 RB14 RB13 RB12 RB11 RB10 RA6 RA7 RB9 RB8 RB7 RB6
PIC24F16KA102
VCC-3.3
VCC-3.3
3.3V
0V

Figure 20-4: Jumpers CN110 in pull-down position and jumper J6 in VCC-3.3 position

In order to enable port PORTB pins to be connected to pull-up resistors and the port input pins to be supplied with a logic zero (0), it is necessary to place jumpers CN110 in the *Up* position and jumper J6 in the GND position. This enables any port PORTB input pin, when it is in idle state, to be driven high (3.3V) over the 10k resistor. As a result, every time you press the RB8 push button, a logic zero (0V) will appear on the RB8 pin.



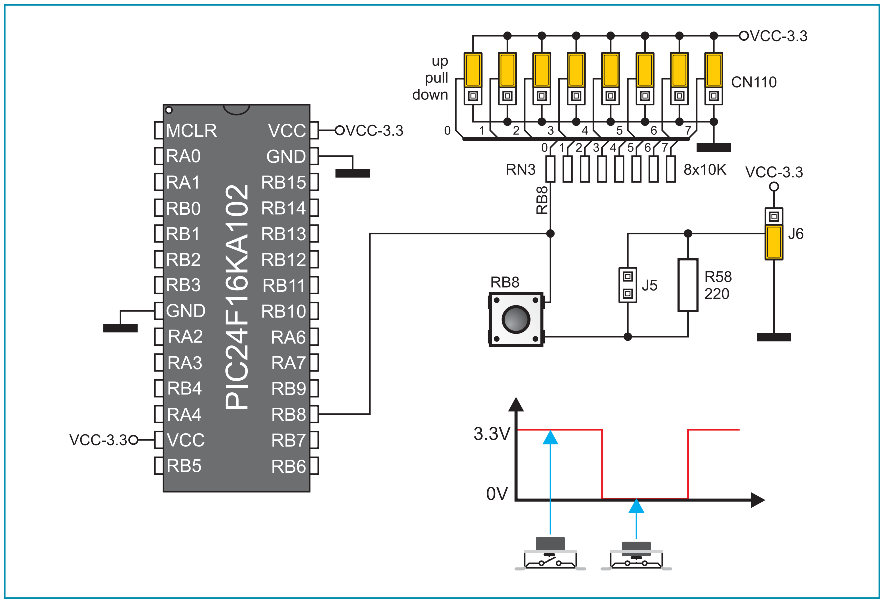

VCC-3.3
up
pull
down
CN110
0 1 2 3 4 5 6 7
RN3
8x10K
RB8
VCC-3.3
J6
J5
R58
220
RB8
MCLR RA0 RA1 RB0 RB1 RB2 RB3 GND RA2 RA3 RB4 RA4 VCC RB5
VCC GND RB15 RB14 RB13 RB12 RB11 RB10 RA6 RA7 RB9 RB8 RB7 RB6
PIC24F16KA102
VCC-3.3
VCC-3.3
3.3V
0V

Figure 20-5: Jumpers CN110 in pull-up and jumper J6 in GND position

In case that jumpers CN110 and J6 have the same logic state, press on any button will not cause input pins to change their logic state.

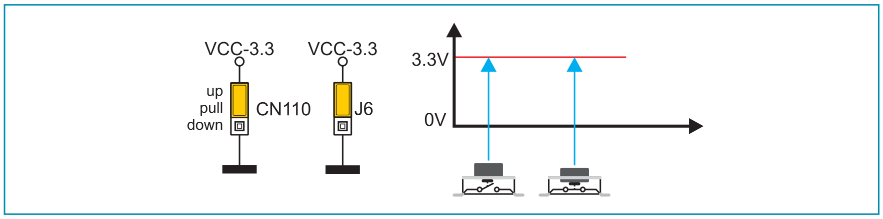


Figure 20-7: Jumpers CN10 and J6 in the same positions

DISCLAIMER

All the products owned by MikroElektronika are protected by copyright law and international copyright treaty. Therefore, this manual is to be treated as any other copyright material. No part of this manual, including product and software described herein, may be reproduced, stored in a retrieval system, translated or transmitted in any form or by any means, without the prior written permission of MikroElektronika. The manual PDF edition can be printed for private or local use, but not for distribution. Any modification of this manual is prohibited.

MikroElektronika provides this manual 'as is' without warranty of any kind, either expressed or implied, including, but not limited to, the implied warranties or conditions of merchantability or fitness for a particular purpose.

MikroElektronika shall assume no responsibility or liability for any errors, omissions and inaccuracies that may appear in this manual. In no event shall MikroElektronika, its directors, officers, employees or distributors be liable for any indirect, specific, incidental or consequential damages (including damages for loss of business profits and business information, business interruption or any other pecuniary loss) arising out of the use of this manual or product, even if MikroElektronika has been advised of the possibility of such damages. MikroElektronika reserves the right to change information contained in this manual at any time without prior notice, if necessary.

HIGH RISK ACTIVITIES

The products of MikroElektronika are not fault – tolerant nor designed, manufactured or intended for use or resale as on – line control equipment in hazardous environments requiring fail – safe performance, such as in the operation of nuclear facilities, aircraft navigation or communication systems, air traffic control, direct life support machines or weapons systems in which the failure of Software could lead directly to death, personal injury or severe physical or environmental damage ('High Risk Activities'). MikroElektronika and its suppliers specifically disclaim any expressed or implied warranty of fitness for High Risk Activities.

TRADEMARKS

The Mikroelektronika name and logo, the Mikroelektronika logo, mikroC, mikroC PRO, mikroBasic, mikroBasic PRO, mikroPascal, mikroPascal PRO, AVRflash, PICflash, dsPICprog, 18FJprog, PSOCprog, AVRprog, 8051prog, ARMflash, EasyPIC5, EasyPIC6, BigPIC5, BigPIC6, dsPIC PRO4, Easy8051B, EasyARM, EasyAVR5, EasyAVR6, BigAVR2, EasydsPIC4A, EasyPSoC4, EasyVR Stamp LV18FJ, LV24-33A, LV32MX, PIC32MX4 MultiMedia Board, PICPLC16, PICPLC8 PICPLC4, SmartGSM/GPRS, UNI-DS are trademarks of Mikroelektronika. All other trademarks mentioned herein are property of their respective companies.

All other product and corporate names appearing in this manual may or may not be registered trademarks or copyrights of their respective companies, and are only used for identification or explanation and to the owners' benefit, with no intent to infringe.

MikroElektronika

SOFTWARE AND HARDWARE SOLUTIONS FOR EMBEDDED WORLD ...making it simple

If you want to learn more about our products, please visit our website at www.mikroe.com

If you are experiencing some problems with any of our products or just need additional information, please place your ticket at www.mikroe.com/en/support

If you have any questions, comments or business proposals, do not hesitate to contact us at office@mikroe.com