EasyPIC® 6

User manual

A large number of useful peripherals, ready-to-use practical code examples and a broad set of add-on boards make MikroElektronika development systems fast and reliable tools that can satisfy the needs of experienced engineers and beginners alike.

Development system

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 having confidence in MikroElektronika.
It is our intention to provide you with the best quality products. Furthermore, we will continue to improve our performance to better suit your needs.

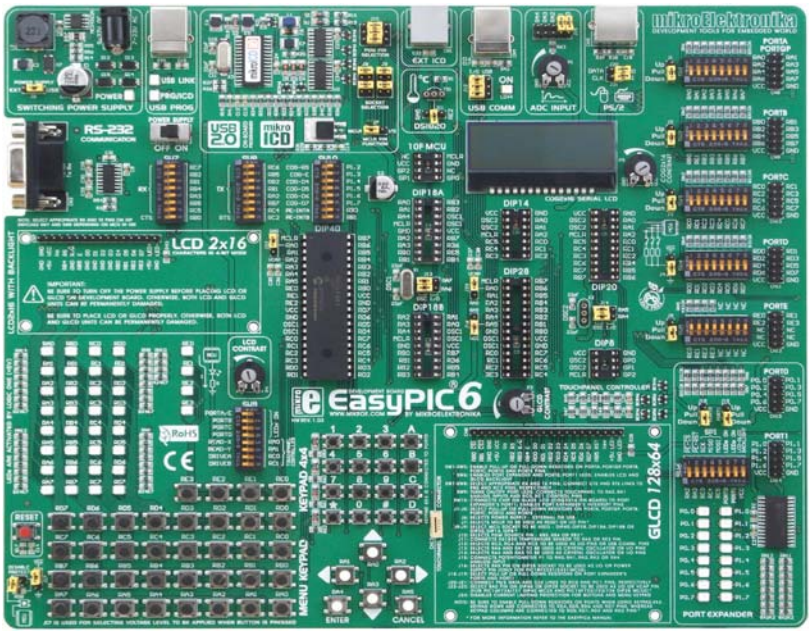
Nebojsa Matic
General Manager

TABLE OF CONTENTS

page

Introduction to EasyPIC® 6 Development Board

The EasyPIC6 development system is an extraordinary development tool suitable for programming and experimenting with PIC® microcontrollers from MICROCHIP®. The board includes an on-board programmer with mikroICD® support (In-Circuit Debugger) providing an interface between the microcontroller and the PC. You are simply expected to write a code in some of our compilers, generate a .hex file and program your microcontroller using the *PICflash*® programmer. Numerous on-board modules, such as 128x64 graphic LCD display, 2x16 LCD display, on-board 2x16 LCD display, keypad 4x4, port expander etc., allow you to easily simulate the operation of the target device.



Full-featured and user-friendly development board for PIC microcontrollers

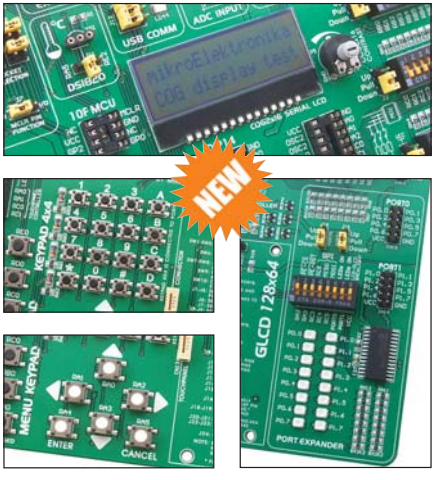
High-Performance USB 2.0 On-Board Programmer

Hardware In-Circuit Debugger for step by step debugging at hardware level

Port Expander provides easy I/O expansion (2 additional ports) using serial interface

On-Board 2x16 serial LCD Display

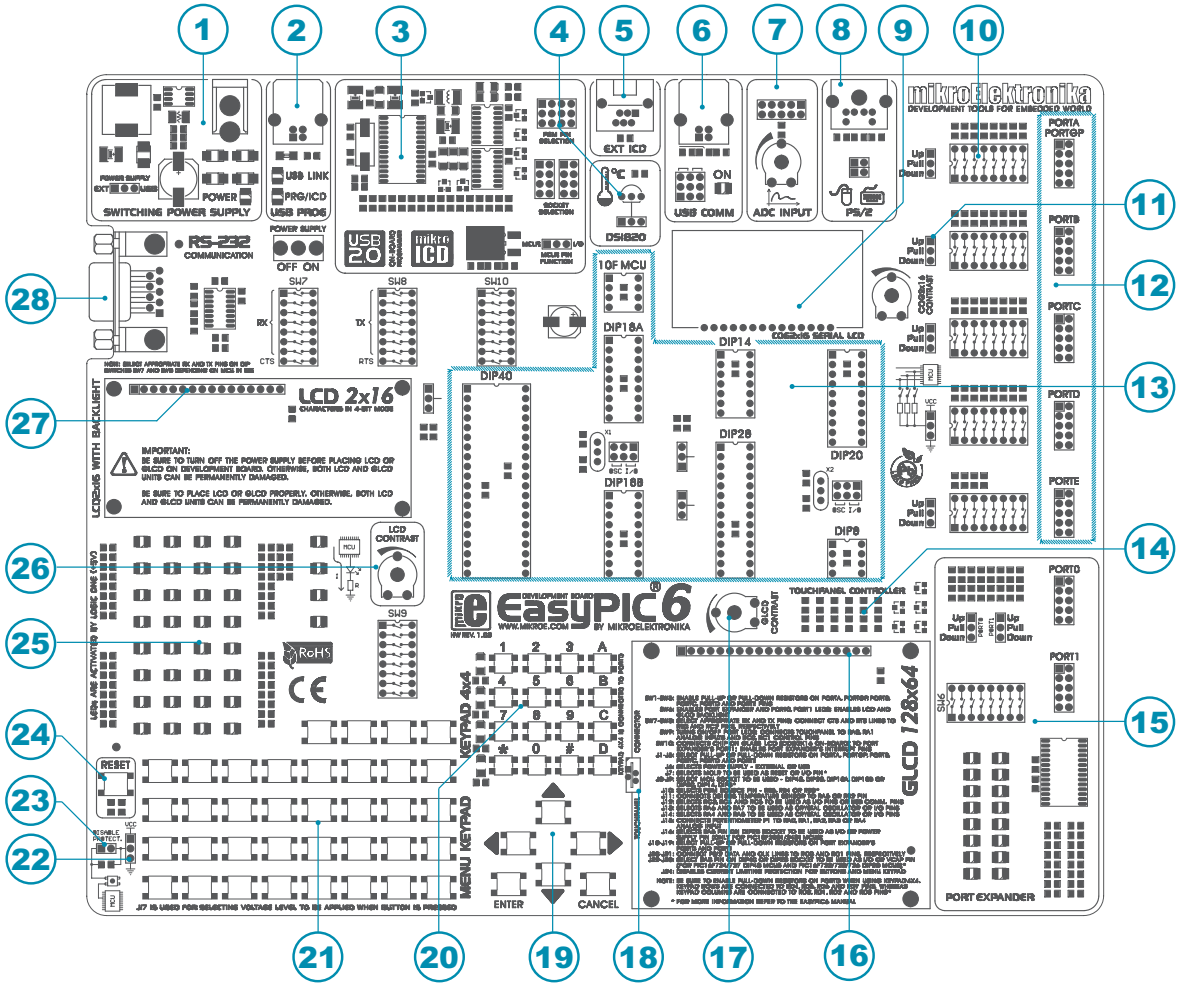
Graphic LCD display with backlights



The *PICflash* 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 **www.mikroe.com**

Package contains:	
Development board:	*EasyPIC6*
CD:	*product CD* with appropriate software
Cables:	*USB cable*
Documentation:	*EasyPIC6* manual, *mikroICD* manual, *PICflash* manual, *Installing USB drivers* manual and *Electrical Schematic* of the *EasyPIC6* development system

System specification:	
Power supply:	over a DC connector (7V to 23V AC or 9V to 32V DC); or over a USB cable (5V DC)
Power consumption:	up to 40mA (depending on how many on-board modules are currently active)
Size:	26,5 x 22cm (10,43 x 8,66inch)
Weight:	~417g (0.919lbs)



Key Features

1. Power supply voltage regulator
2. On-board programmer USB connector
3. USB 2.0 programmer with mikroICD support
4. DS1820 temperature sensor socket
5. External MICROCHIP debugger (ICD2 or ICD3) connector
6. USB communication connector
7. A/D converter test inputs
8. PS/2 connector
9. On-board 2x16 LCD display
10. DIP switches to enable pull-up/pull-down resistors
11. Port pins' pull-up/pull-down mode selection
12. I/O port connectors
13. PIC microcontroller sockets
14. Touch panel controller
15. Port expander
16. 128x64 graphic LCD display connector
17. 128x64 graphic LCD display contrast potentiometer
18. Touch panel connector
19. Menu keypad
20. Keypad *4x4*
21. Push buttons to simulate digital inputs
22. Logic state selector
23. Protective resistor ON/OFF jumper
24. Reset button
25. 36 LEDs to indicate pins' logic state
26. Alphanumeric LCD display contrast adjustment
27. Alphanumeric LCD display connector
28. RS-232 communication connector

1.0. Connecting the System to your PC

Step 1:

Use the USB cable to connect the EasyPIC6 development system to your PC. One end of the USB cable provided with a connector of the USB **B** type should be connected to the development system as shown in Figure 1-2, whereas the other end of the cable (USB **A** type) should be connected to your PC. When establishing a connection, make sure that jumper J6 is placed in the USB position as shown in Figure 1-1.

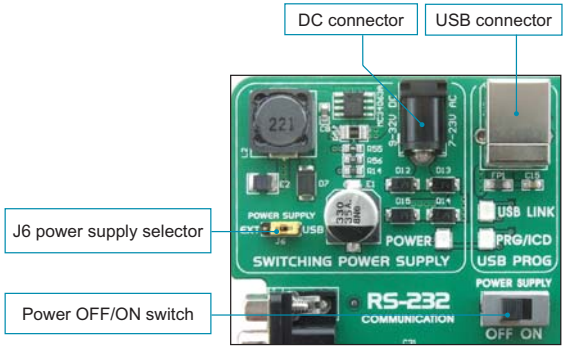


Figure 1-1: Power supply

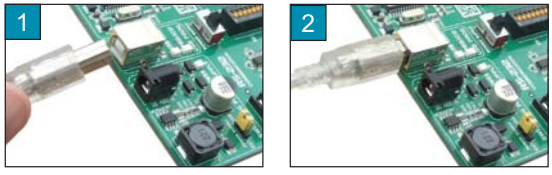


Figure 1-2: Connecting USB cable (jumper J6 in the USB position)

Step 2:

Follow the instructions for installing USB drivers and the *PICflash* programmer provided in the relevant manuals. It is not possible to program PIC microcontrollers without having these devices installed. In case that you already have some of the MikroElektronika's compilers installed on your PC, there is no need to reinstall the *PICflash* programmer as it will be automatically installed along with compiler installation.

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 to indicate that your development system is ready for use. Use the *PICflash* programmer to dump a code into the microcontroller and employ the board to test and develop your projects.

NOTE: If you use some additional modules, such as LCD, GLCD, extra boards etc., it is necessary to place them properly on the development system before it is turned on. Otherwise, they can be permanently damaged.

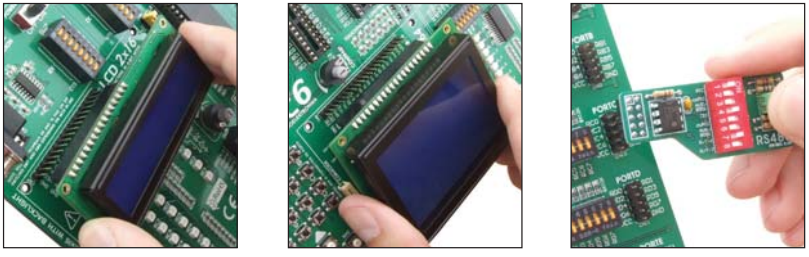


Figure 1-3: Placing additional modules on the board

2.0. Supported Microcontrollers

The *EasyPIC6* development system provides eight separate sockets for PIC microcontrollers in DIP40, DIP28, DIP20, DIP18, DIP14 and DIP8 packages. These sockets allow supported devices in DIP packages to be plugged directly into the development board.
There are two sockets for PIC microcontrollers in DIP18 package provided on the board. Which of these sockets you will use depends solely on the pinout of the microcontroller in use. The EasyPIC6 development system comes with the microcontroller in a DIP40 package.

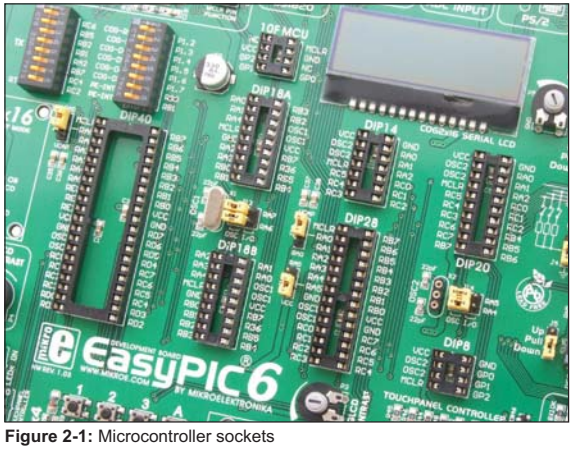


Figure 2-1: Microcontroller sockets

Jumpers next to the sockets are used for selecting functions of the microcontroller pins:

Jumper	Position
J22	RA0 - I/O pin VCAP - filter capacitor (for 16F724/727)
J23	VCAP - filter capacitor (for 16F722/723) RA0 - I/O pin
J16	RA5 - I/O pin VCC - 18F2331/2431 power supply
J13	OSC - RA6, RA7 are OSC. pins I/O - RA6, RA7 are I/O pins
J14	OSC - RA4, RA5 are OSC. pins I/O - RA4, RA5 are I/O pins

PIC microcontrollers normally use a quartz crystal for the purpose of stabilizing clock frequency. The EasyPIC6 provides two sockets for quartz-crystal. Microcontrollers in DIP18A, DIP18B, DIP28 and DIP40 packages use socket X1 (OSC1) for quartz-crystal. If microcontrollers in DIP8, DIP14 and DIP20 packages are used, it is necessary to move quartz crystal from socket X1 to socket X2 (OSC2). Besides, it is also possible to replace the existing quartz-crystal with another one. The value of the quartz-crystal depends on the maximum clock frequency allowed. Microcontrollers being plugged into socket 10F use their own internal oscillator and are not connected to any of the aforementioned quartz-crystal sockets.

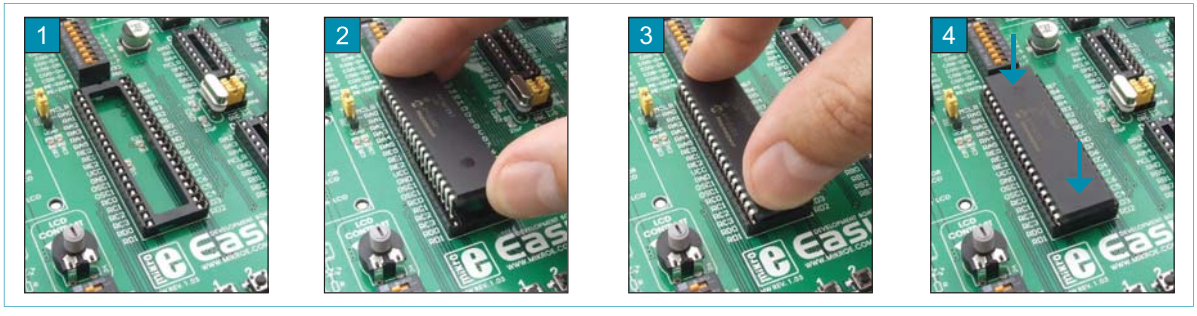


Figure 2-2: Plugging microcontroller into appropriate socket

Prior to plugging the microcontroller into the appropriate socket, make sure that the power supply is turned off. Figure 2-2 shows how to correctly plug a microcontroller. Figure 1 shows an unoccupied 40-pin DIP 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 thereof 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 the same time.

3.0. On-Board USB 2.0 PICflash Programmer

The *PICflash* programmer is an obligatory tool when working with microcontrollers. The EasyPIC6 has an on-board *PICflash* programmer with mikroICD support which allows you to establish a connection between the microcontroller and your PC. Use the *PICflash* programmer to load a HEX file into the microcontroller. Figure 3-2 shows the connection between a compiler, *PICflash* programmer and microcontroller.

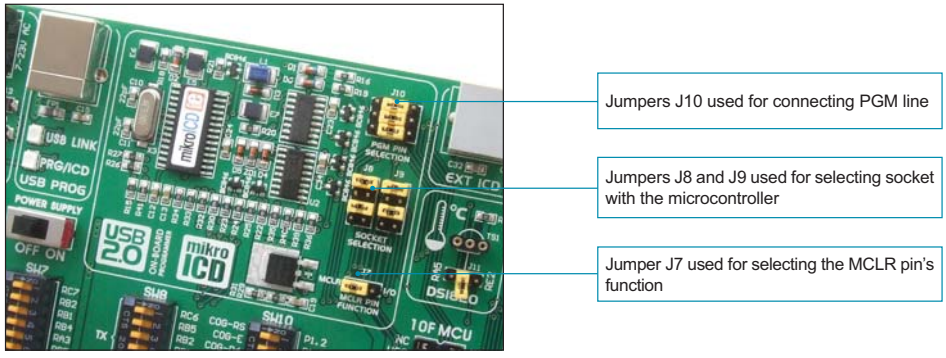


Figure 3-1: *PICflash with mikroICD* programmer

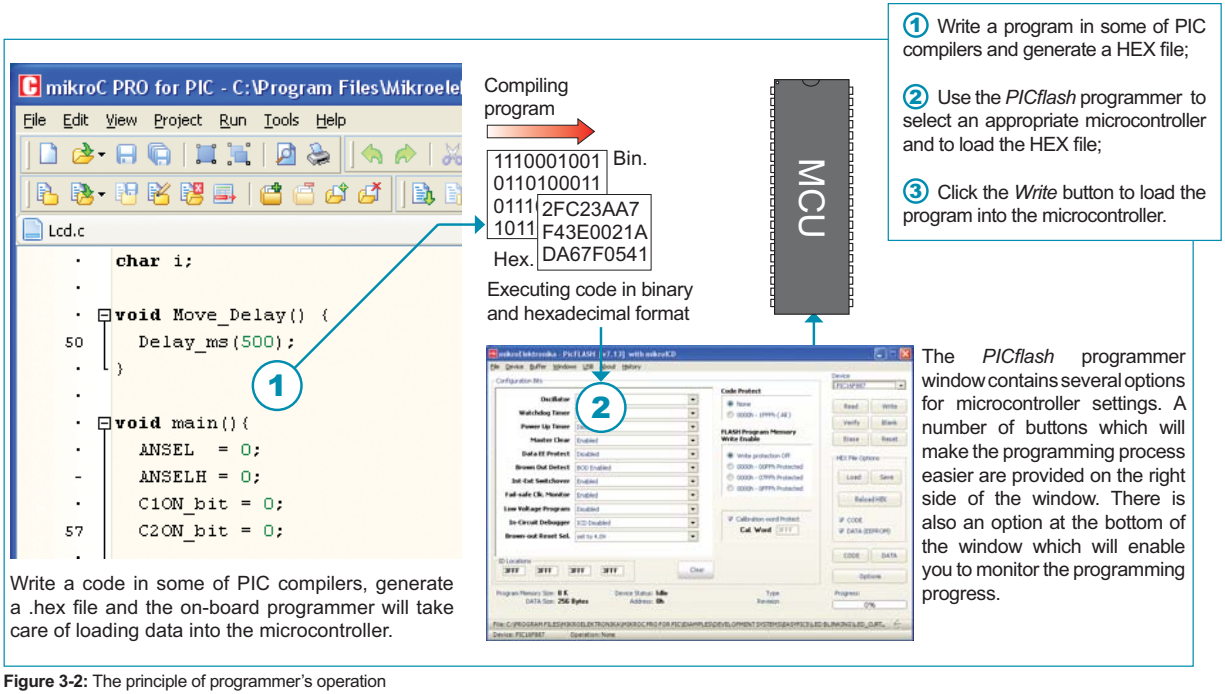


Figure 3-2: The principle of programmer's operation

NOTE: For more information on the *PICflash* programmer refer to the relevant manual provided in the EasyPIC6 development system package.

There are two ways of programming PIC microcontrollers: Low Voltage and High Voltage programming modes. The *PICflash* programmer uses solely High Voltage programming mode during its operation. This mode requires voltage higher than the microcontroller's power supply voltage (the range between 8V to 14V, depending on the type of the microcontroller in use) to be brought to the MCLR/Vpp pin in order so that the process of programming/debugging may be performed.

The Low Voltage programming mode can be enabled/disabled using configuration bits of the microcontroller. If the Low Voltage programming mode is enabled, the programming process is initiated by applying a logic one (1) to the PGM pin. Unlike this mode, the High Voltage programming mode is always enabled and the programming process starts by applying a high voltage to the MCLR/Vpp pin.

All PIC microcontrollers have the Low Voltage programming mode enabled by default. In some rare cases, in order to enable the microcontroller to be programmed in the High Voltage programming mode, it is necessary to apply a logic zero (0) to the PGM pin, which prevents the microcontroller from entering the Low Voltage programming mode. Depending on the microcontroller in use, it is possible to select one of the following pins RB3, RB4 and RB5 to be used as the PGM pin. Jumper J10 is used as the PGM pin selector as shown in Figure 3-3.

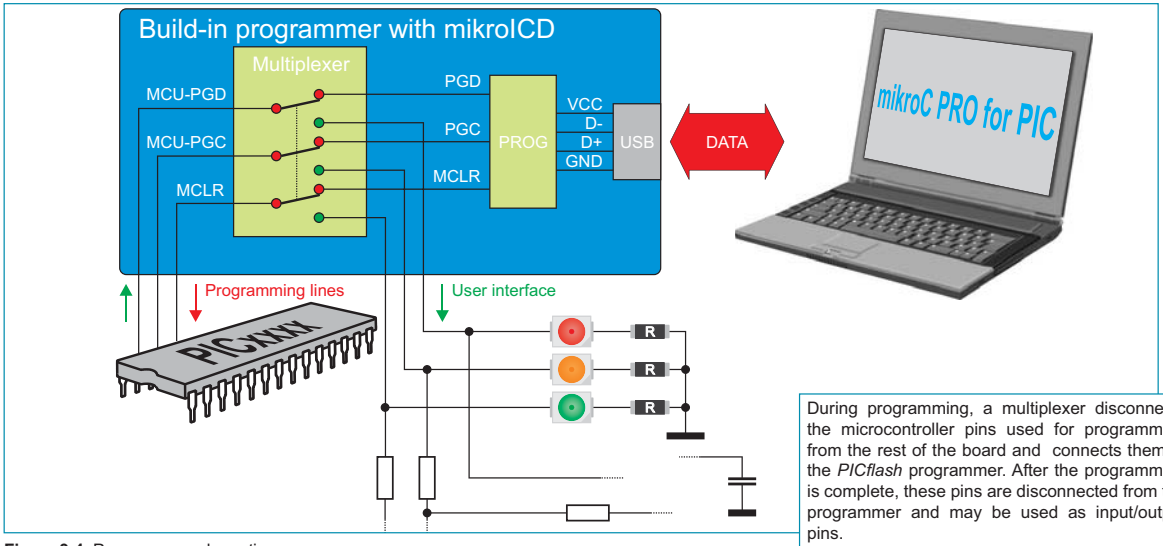
Jumper J10 default position when RB3, RB4 and RB5 pins are not connected to the PGM line.

Jumper J10 position when the PGM line is connected to the RB5 pin.

Jumper J10 position when the PGM line is connected to the RB4 pin.

Jumper J10 position when the PGM line is connected to the RB3 pin.

Figure 3-3: Various positions of jumper J10



During programming, a multiplexer disconnects the microcontroller pins used for programming from the rest of the board and connects them to the *PICflash* programmer. After the programming is complete, these pins are disconnected from the programmer and may be used as input/output pins.

Figure 3-4: Programmer schematic

Microcontroler is plugged into one of the following sockets: DIP40, DIP28 DIP18A or DIP18B. (Default position)

Microcontroller is plugged into one of the following sockets: DIP20, DIP14 or DIP8.

Jumpers J8 and J9 are used for selecting the socket to receive the programming signal. Figure 3-5 shows the position of jumpers J8 and J9 depending on DIP sockets in use.

Figure 3-5: The position of jumpers J8 and J9

MCLR used as the MCLR/Vpp pin.

MCLR used as an I/O pin.

Figure 3-6: The position of jumper J7

The function of the MCLR (Master Clear) pin depends on the position of jumper J7. When placed in the left-hand position, the MCLR pin has default function, i.e. is used as MCLR/Vpp. Otherwise, when the jumper is placed in the right-hand position, the MCLR pin is available as an I/O pin.

4.0. mikroICD (In-Circuit Debugger)

The mikroICD (In-Circuit Debugger) is an integral part of the on-board programmer. It is used for the purpose of 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 compilers designed by mikroElektronika (mikroBASIC®, mikroC® and mikroPASCAL®). As soon as the mikroICD debugger starts up, a window, as shown in figure below, appears.
The *mikroICD* debugger communicates with the PC through the programming pins which cannot be used as I/O pins while the process of the program debugging is in progress.

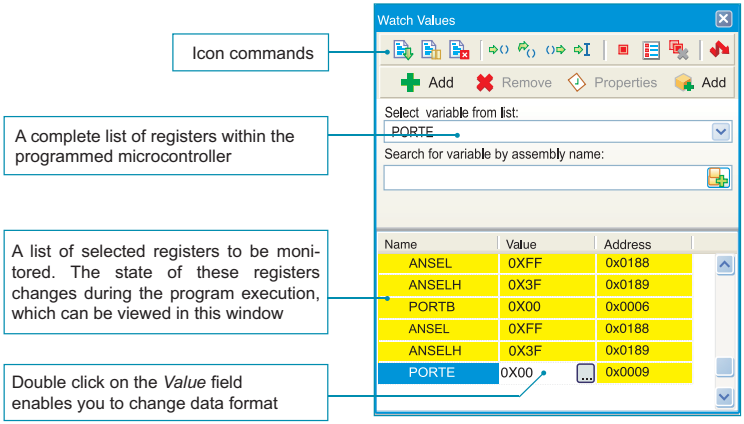


Figure 4-1: mikroICD Watch Values window

mikroICD debugger options:

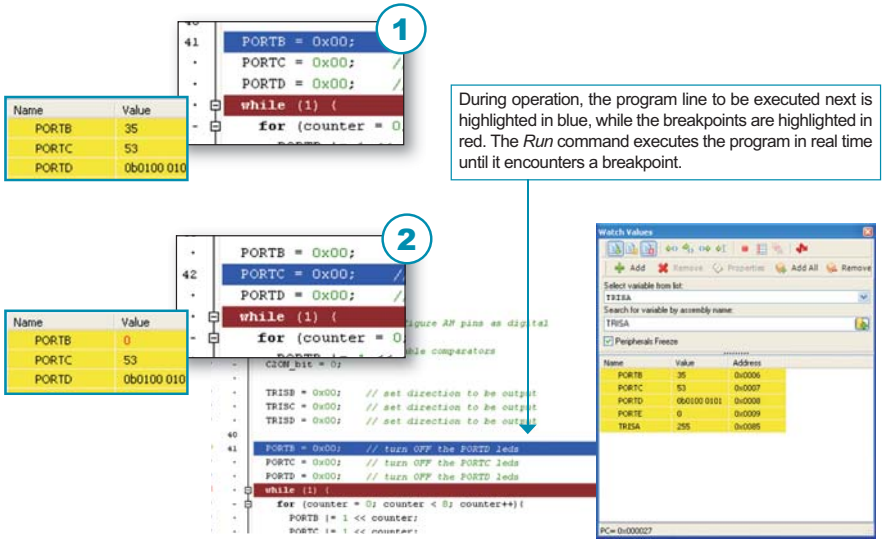
Start Debugger	[F9]
Run/Pause Debugger	[F6]
Stop Debugger	[Ctrl+F2]
Step Into	[F7]
Step Over	[F8]
Step Out	[Ctrl+F8]
Toggle Breakpoint	[F5]
Show/Hide Breakpoints	[Shift+F4]
Clear Breakpoints	[Ctrl+Shift+F4]

Each of these commands is activated via keyboard shortcuts or by clicking 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 microontroller can be viewed in the mikroICD *Watch Values* window.

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.



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

5.0. Power Supply

The EasyPIC6 development system may use one of two power supply sources:

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

The MC34063A voltage regulator is used for enabling 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 J6 is used as power supply selector. When using USB power supply, jumper J6 should be placed in the USB position. When using external power supply, jumper J6 should be placed in the EXT position. The development system is turned OFF/ON by changing the setting on the OFF/ON switch respectively.

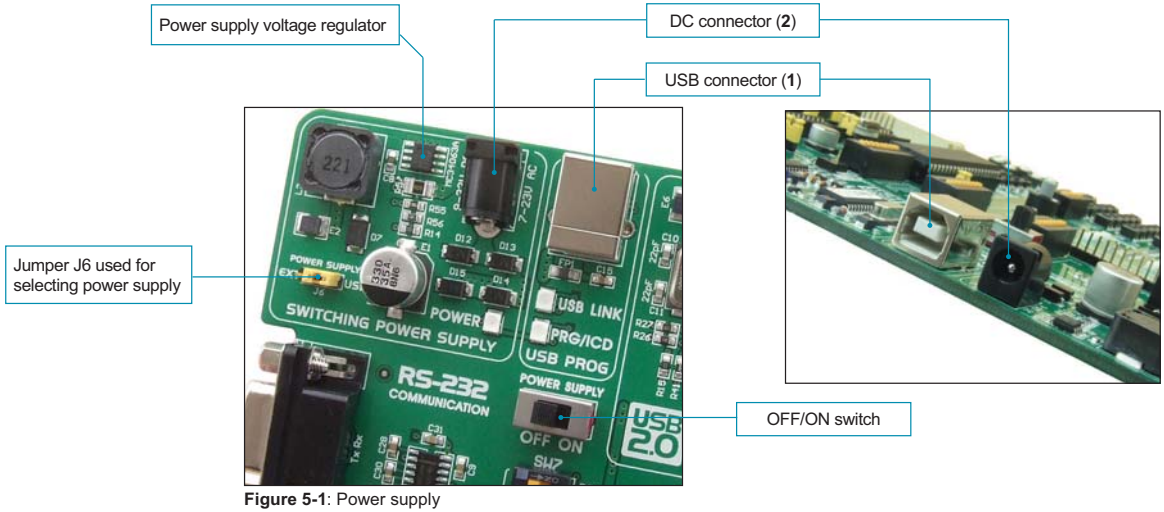


Figure 5-1: Power supply

The programmer uses the MOSFET switch for suspending power supply on the development system during programming. When the process of programming is complete, the programmer enables the development system to be supplied with power.

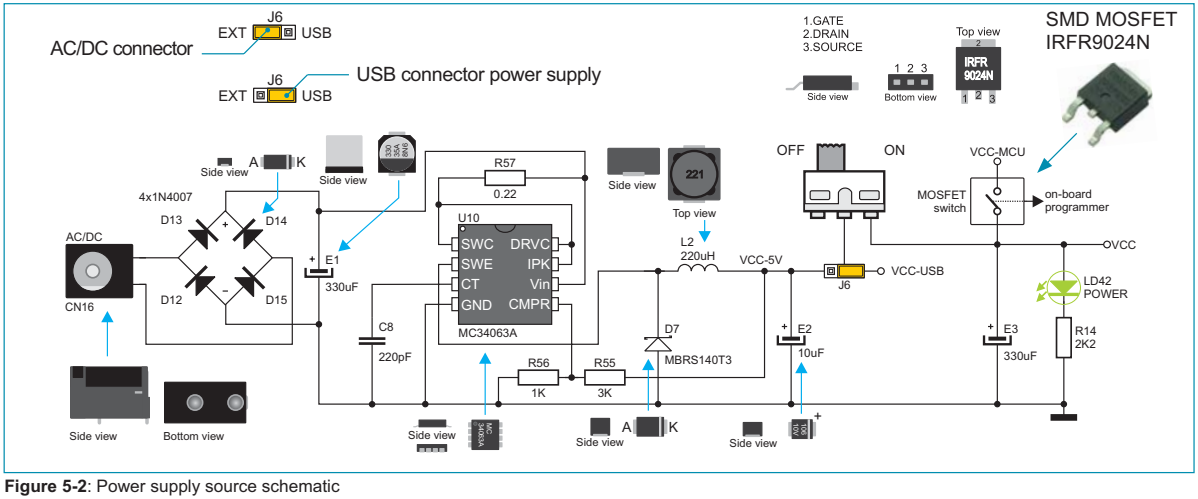


Figure 5-2: Power supply source schematic

6.0. RS-232 Communication Interface

RS-232 serial communication is performed through a 9-pin SUB-D connector and the microcontroller USART module. In order to enable such communication, it is necessary to establish a connection between RX and TX communication lines (*handshaking* lines CTS and RTS are optionally used) and microcontroller pins provided with USART module using a DIP switch. The microcontroller pins used in such communication are marked as follows: RX - *receive data*, TX - *transmit data*, CTS - *clear to send* and RTS - *request to send*. Baud rate goes up to 115kbps.
The USART (universal synchronous/asynchronous receiver/transmitter) is one of the most common ways of exchanging data between the PC and peripheral components. In order to enable the USART module of the microcontroller to receive input signals with different voltage levels, it is necessary to provide a voltage level converter such as MAX-202C.

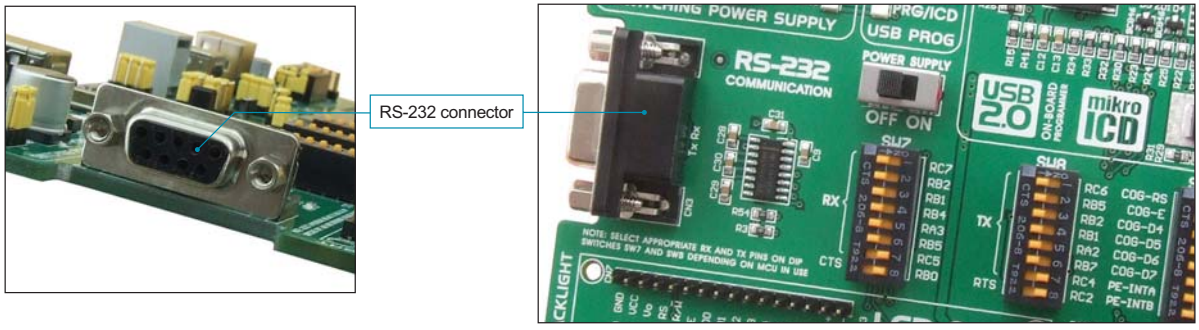


Figure 6-1: RS-232 module

The function of DIP switches SW7 and SW8 is to determine which of the microcontroller pins are to be used as RX and TX lines. The microcontroller pinout varies depending on the type of the microcontroller. Figure 6-2 shows the microcontroller in DIP40 package (PIC16F887).

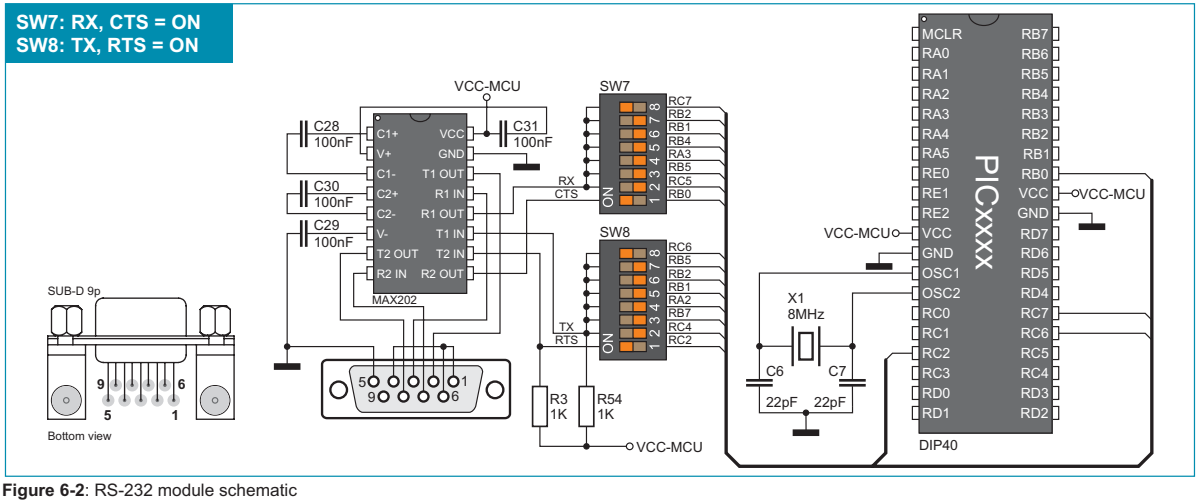


Figure 6-2: RS-232 module schematic

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

7.0. PS/2 Communication Interface

The **PS/2** connector enables input units, such as keyboard and mouse, to be connected to the development system. In order to enable PS/2 communication, it is necessary to correctly place jumpers J20 and J21, thus connecting DATA and CLK lines to the microcontroller pins RC0 and RC1. Do not connect/disconnect input units to the PS/2 connector while the development system is turned on as it may permanently damage the microcontroller.

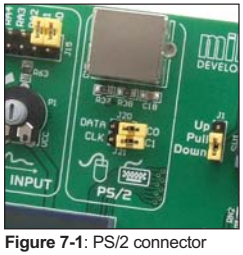


Figure 7-1: PS/2 connector (J20 and J21 are not connected)

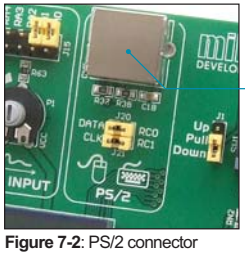
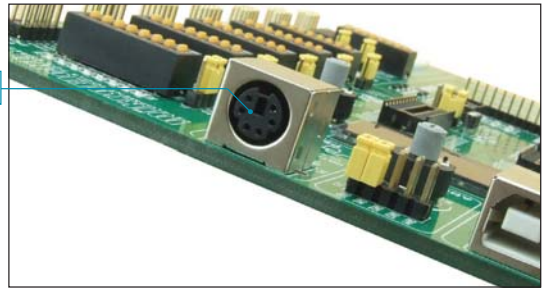


Figure 7-2: PS/2 connector (J20 and J21 are connected)



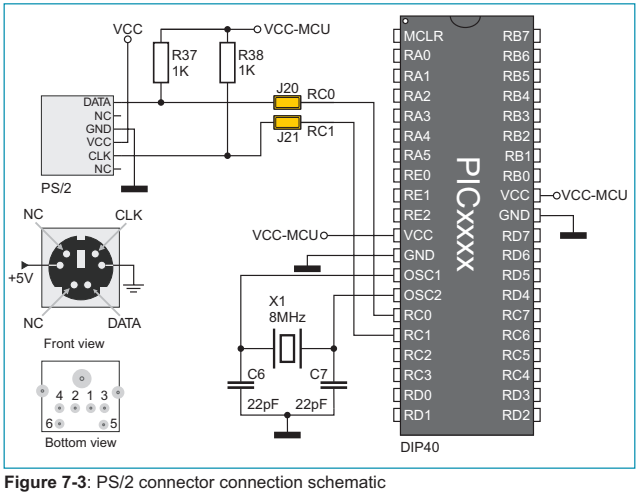


Figure 7-3: PS/2 connector connection schematic

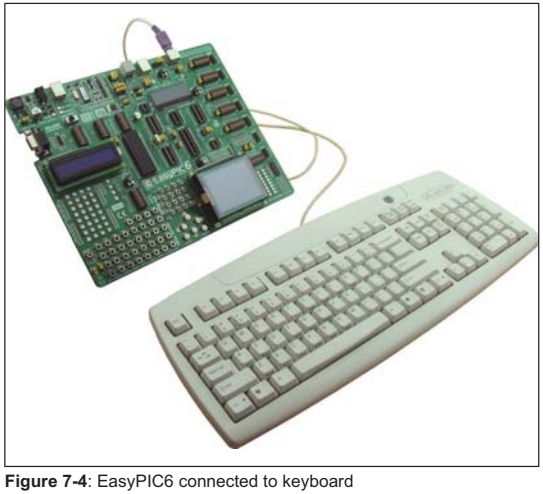


Figure 7-4: EasyPIC6 connected to keyboard

8.0. ICD Connector

ICD (In-Circuit Debugger) connector enables the microcontroller to communicate with external ICD debugger (ICD2 or ICD3)* from MICROCHIP. Jumpers J8 and J9 are placed in the same way as when using the *PICflash* programmer with mikroICD designed by MikroEektronika.

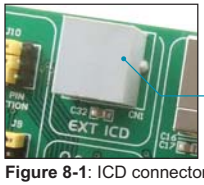
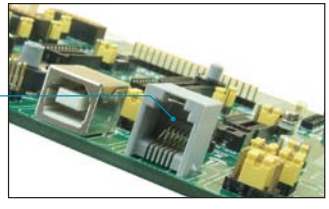


Figure 8-1: ICD connector



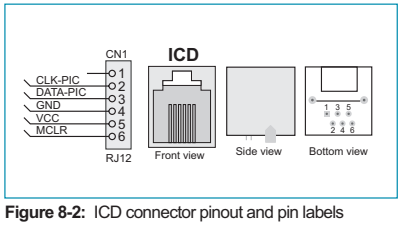


Figure 8-2: ICD connector pinout and pin labels

*ICD2 and ICD3 are registered trademarks of MICROCHIP®

9.0. USB Communication

The **USB** connector enables PIC microcontrollers with a built-in USB communication module to be connected to peripheral components. In order to enable USB communication, it is necessary to change the position of jumpers J12 from left-hand to right-hand, thus connecting the USB DATA lines (D+ i D-) to RC4 and RC5 microcontroller pins and the RC3/VUSB pin to capacitors C16 and C17. If USB communication is not used, jumpers J12 should be left in the left-hand position. The status of USB communication (OFF/ON) is indicated by LED. Figures 9-3 and 9-4 show schematics of the most commonly used microcontrollers with integrated USB module.

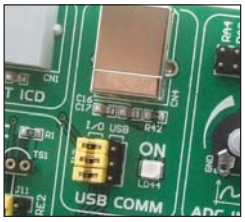


Figure 9-1: USB communication disabled (default position)

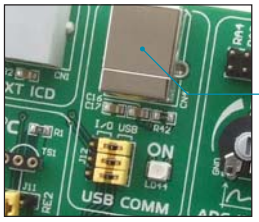
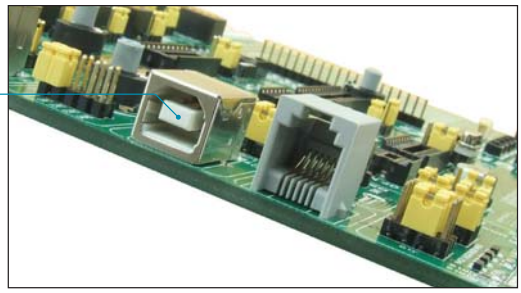


Figure 9-2: USB communication enabled



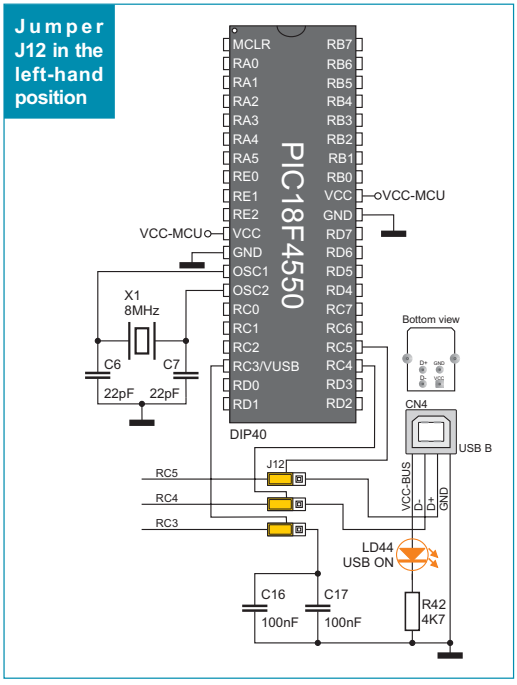


Figure 9-3: PIC18F4550 USB communication schematic

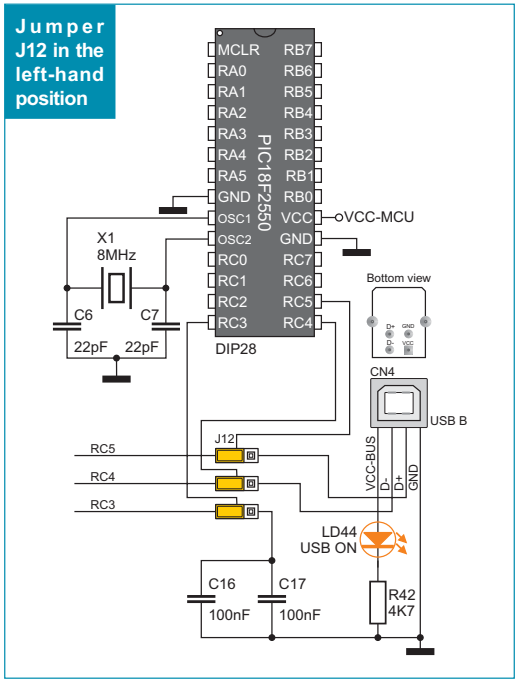


Figure 9-4: PIC18F2550 USB communication schematic

10.0. DS1820 Temperature Sensor

1-wire® serial communication enables data to be transferred over one single communication line while the process itself is under the control of the master microcontroller. 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 interface.

DS1820 is a temperature sensor that uses 1-wire® standard for its operation. 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. Power supply voltage of 3V to 5.5V is required for its operation. It takes maximum 750ms for the DS1820 to calculate temperature with 9-bit resolution. The EasyPIC6 development system provides a separate socket for the DS1820. It may use either RA5 or RE2 pin for communication with the microcontroller. Jumper J11's purpose is selection of the pin to be used for 1-wire® communication. Figure 10-5 shows 1-wire® communication with microcontroller through the RA5 pin.

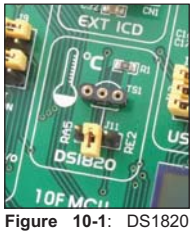


Figure 10-1: DS1820 connector (1-wire communication is not used)

Figure 10-2: J11 in the left-hand position (1-wire communication through the RA5 pin)

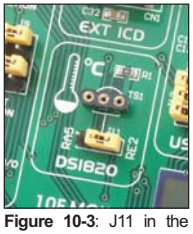


Figure 10-3: J11 in the right-hand position (1-wire communication through the RE2 pin)

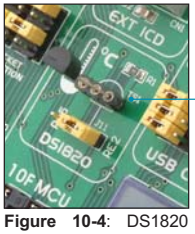


Figure 10-4: DS1820 plugged into appropriate socket

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

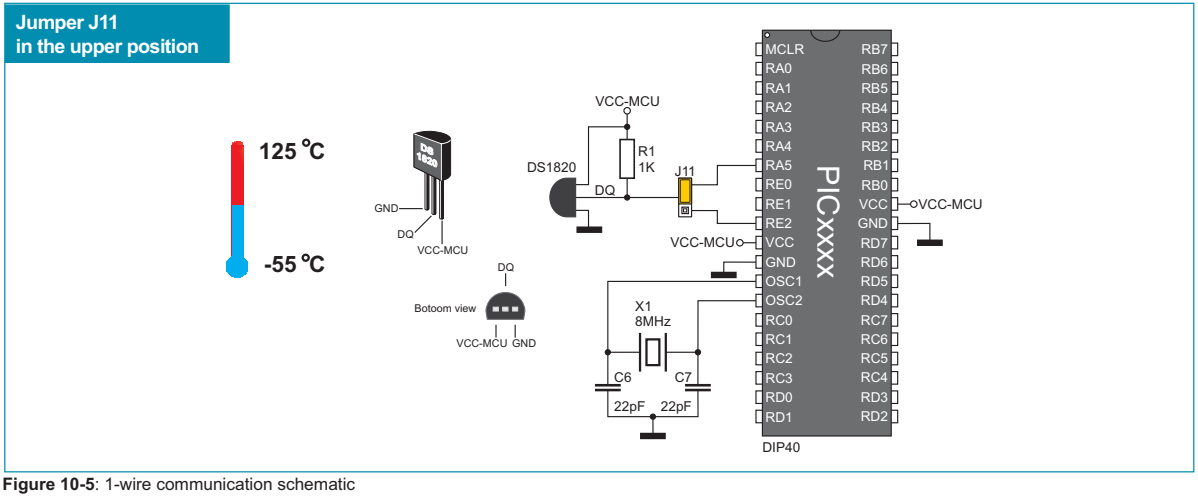


Figure 10-5: 1-wire communication schematic

11.0. A/D Converter

An A/D converter is used for the purpose of converting an analog signal into the appropriate digital value. A/D converter is linear, which means that the converted number is linearly dependent on the input voltage value.
The A/D converter built into the microcontroller provided with the EasyPIC6 development system converts an analog voltage value into a 10-bit number. Voltages varying from 0V to 5V DC may be supplied through the A/D test inputs. Jumper J15 is used for selecting some of the following pins RA0, RA1, RA2, RA3 or RA4 for A/D conversion. The R63 resistor has a protective function as it is used for limiting current flow through the potentiometer or the microcontroller pin. The value of the input analog voltage can be changed linearly using potentiometer P1.

Figure 11-1: ADC (default jumper positions)

Figure 11-2: The RA0 pin used as A/D conversion input

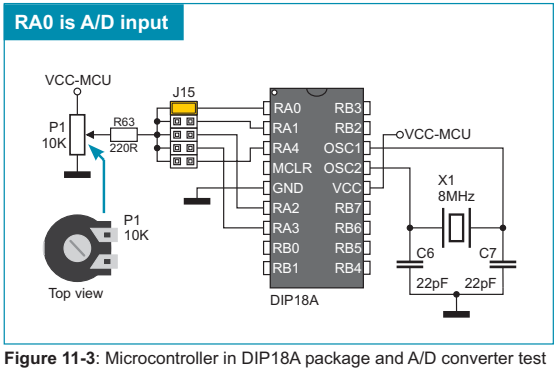


Figure 11-3: Microcontroller in DIP18A package and A/D converter test inputs connection

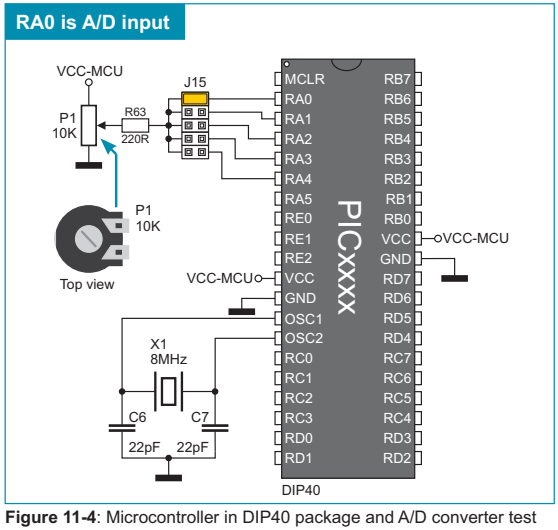


Figure 11-4: Microcontroller in DIP40 package and A/D converter test inputs connectiion

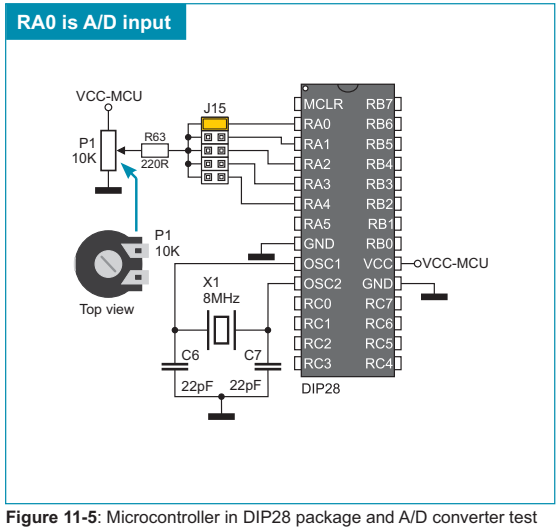


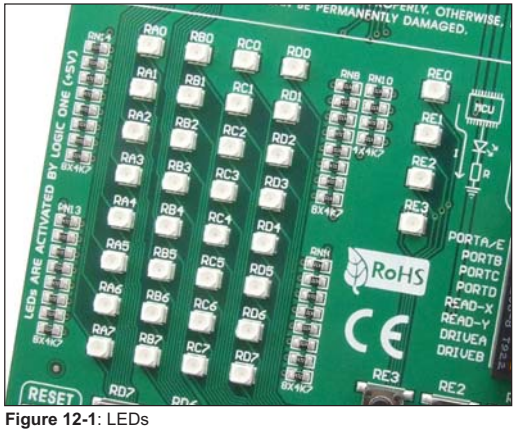
Figure 11-5: Microcontroller in DIP28 package and A/D converter test inputs connection

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

12.0. LEDs

LED diode (Light-Emitting Diode) is a highly efficient electronic light source. When connecting LEDs, it is necessary to place a current limiting resistor the value of which is calculated using formula R=U/I where R is referred to resistance expressed in ohms, U is referred to voltage on the LED and I stands for LED diode current. A common LED diode voltage is approximately 2.5V, while the current varies from 1mA to 20mA depending on the type of LED diode. The EasyPIC6 development system uses LEDs with current I=1mA.

The EasyPIC6 has 36 LEDs which visually indicate the logic state of each microcontroller I/O pin. An active LED diode indicates that a logic one (1) is present on the pin. In order to enable LEDs, it is necessary to select appropriate port PORTA/E, PORTB, PORTC or PORTD using the DIP switch SW9.



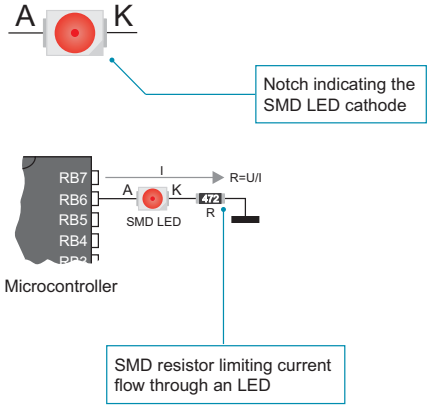


Figure 12-1: LEDs

SW9: PORTB = ON

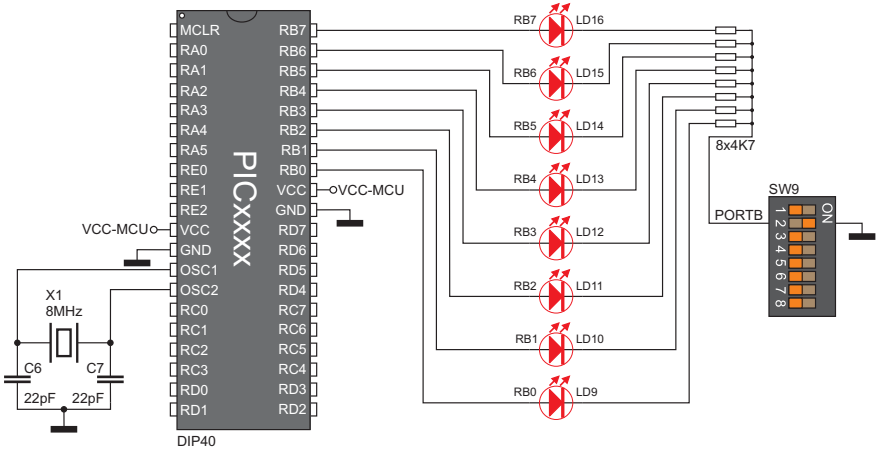


Figure 12-2: LED diode and PORTB connection schematic

13.0. Push Buttons

The logic state of all microcontroller digital inputs may be changed using push buttons. Jumper J17 is used to determine the logic state to be applied to the desired microcontroller pin by pressing the appropriate push button. The purpose of the protective resistor is to limit maximum current thus preventing a short circuit from occurring. Advanced users may, if needed, disable such resistor using jumper J24. Just next to the push buttons, there is a RESET button which is not connected to the MCLR pin. The reset signal is generated by the programmer.

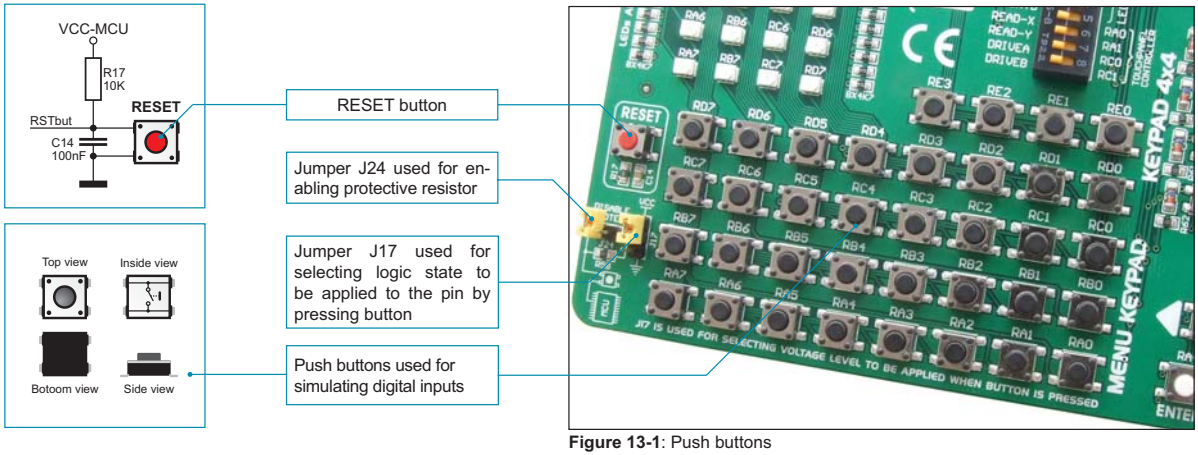


Figure 13-1: Push buttons

By pressing any push button (R0-R7) when jumper J17 is in the VCC-MCU position, a logic one (5V) will be applied to the appropriate microcontroller pin as shown in Figure 13-2.

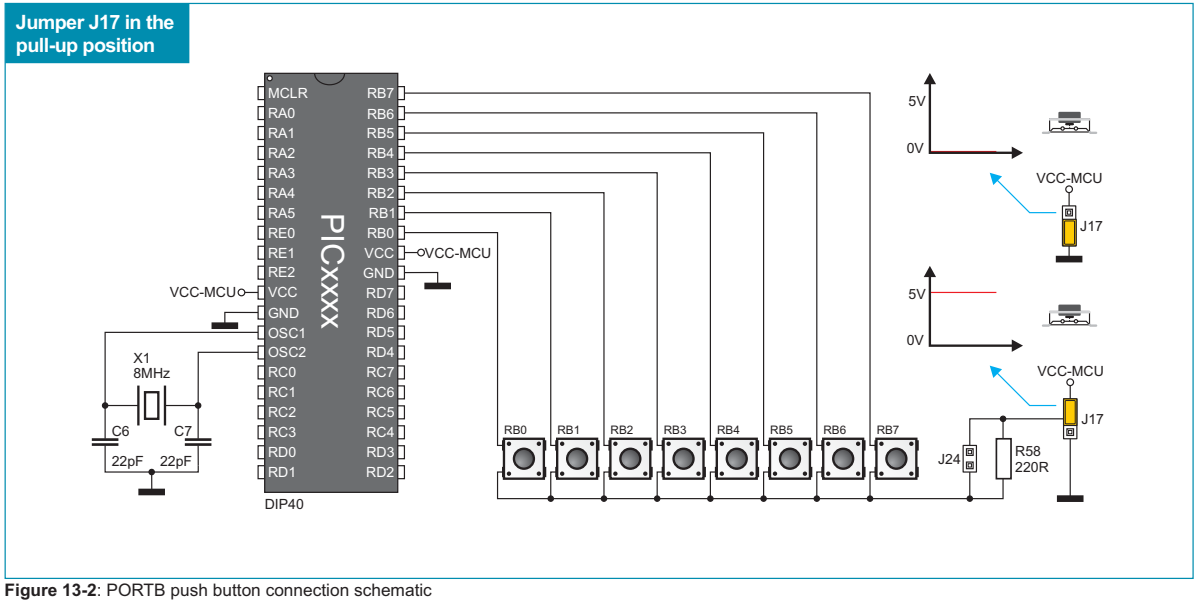


Figure 13-2: PORTB push button connection schematic

14.0. Keypads

There are two keypads provided on the EasyPIC6 development system. These are keypad *4x4* and keypad *MENU*. Keypad *4x4* is a standard alphanumeric keypad connected to the microcontroller PORTD. The performance of such a keypad is based on the 'scan and sense' principle where the RD0, RD1, RD2 and RD3 pins are configured as inputs connected to pull-down resistors. The RD4, RD5, RD6 and RD7 pins are configured as high level voltage outputs. Pressing any button will cause a logic one (1) to be applied to input pins. Push button detection is performed from within software. For example, pressing button '6' will cause a logic one (1) to appear on the RD2 pin. In order to determine which of the push buttons is pressed, a logic one (1) is applied to each of the following output pins RD4, RD5, RD6 and RD7.

Keypad *MENU* buttons are connected in a similar way to the PORTA buttons. The only difference is in the button arrangement. The keypad MENU buttons are arranged so as to provide easy navigation through menus.

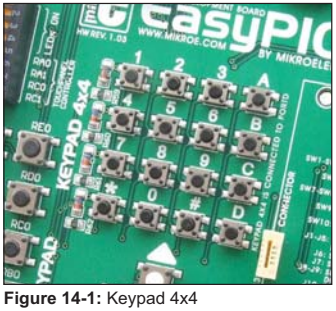


Figure 14-1: Keypad 4x4

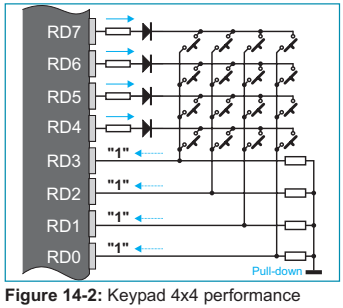


Figure 14-2: Keypad 4x4 performance

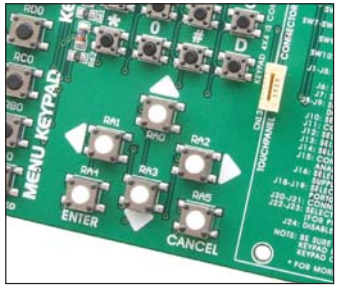


Figure 14-3: Keypad MENU

Jumper J17 is in the pull-up position. Pins RD0, RD1, RD2 and RD3 are connected to pull-down resistors through DIP switch SW4

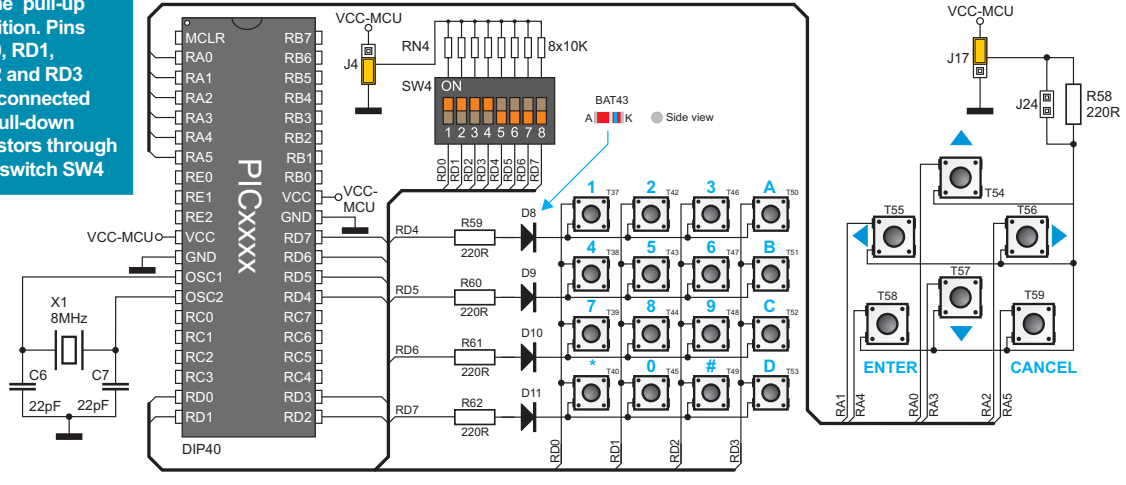


Figure 14-4: Keypads (4x4 and MENU) and microcontroller connection schematic

15.0. 2x16 LCD Display

The EasyPIC6 development system provides an on-board connector to plug alphanumeric 2x16 LCD display into. Such connector is connected to the microcontroller through the PORTB port. Potentiometer P4 is used for display contrast adjustment. The *LCD* switch on the DIP switch SW6 is used for turning on/off display backlight. Communication between an LCD display and the microcontroller is established using a 4-bit mode. Alphanumeric digits are displayed in two lines each containing up to 16 characters of 7x5 pixels.

Figure 15-1: Alphanumeric LCD connector

Figure 15-2: 2x16 LCD display

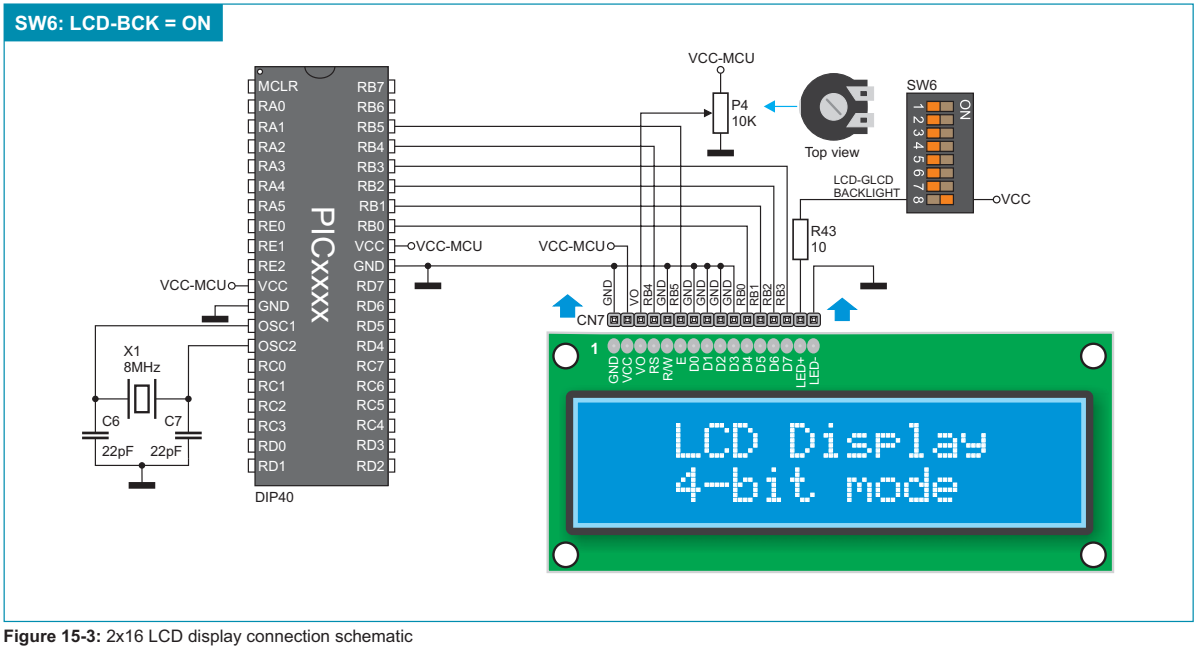


Figure 15-3: 2x16 LCD display connection schematic

16.0. On-Board 2x16 LCD Display

On-board 2x16 display is connected to the microcontroller through a port expander. In order to use this display, it is necessary to set the DIP switch SW10 to the ON position, thus connecting the on-board LCD display to port expander's port 1. The DIP switch SW6 enables the port expander to use serial communication. Potentiometer P5 is used for display contrast adjustment.
Unlike common LCD display, the on-board LCD display has no backlights and receives data to be displayed through the port expander which employs SPI communication for the purpose of communicating with the microcontroller. Similar to standard 2x16 LCD display, the on-board 2x16 LCD display also displays digits in two lines each containing up to 16 characters of 7x5 pixels.

Figure 16-1: On-board 2x16 LCD display

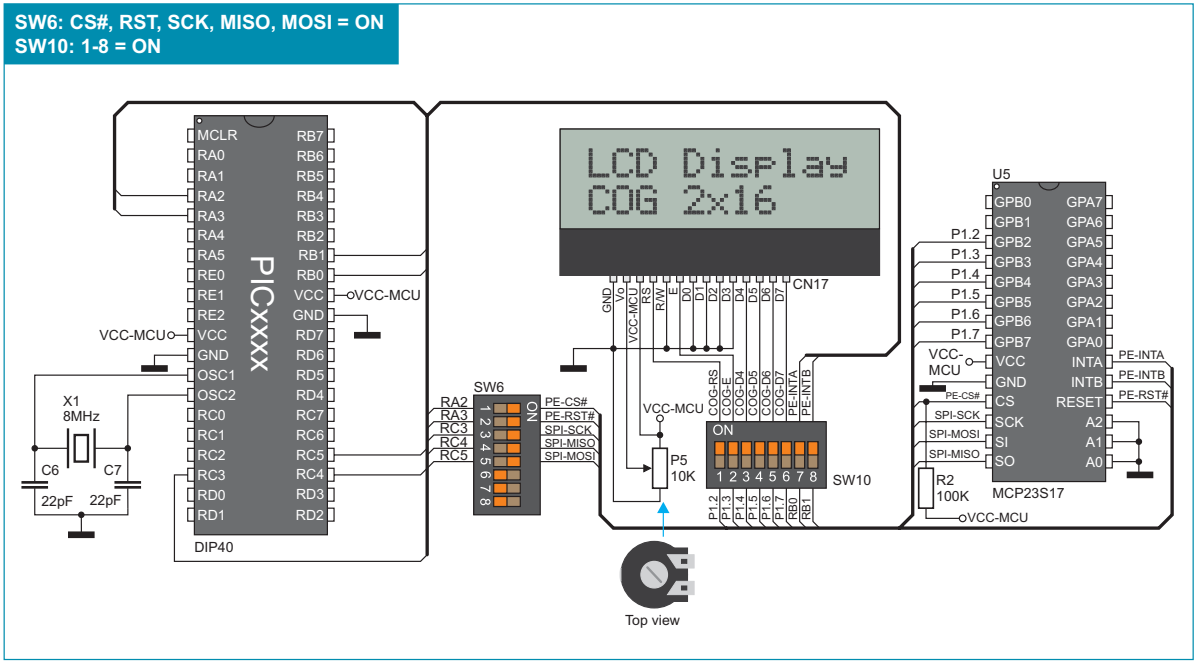


Figure 16-2: On-board 2x16 LCD display connection schematic

page

17.0. 128x64 Graphic LCD Display

128x64 graphic LCD display (128x64 GLCD) provides an advanced method for displaying graphic messages. It is connected to the microcontroller through PORTB and PORTD. GLCD display has the screen resolution of 128x64 pixels which allows you to display diagrams, tables and other graphical contents. Since the PORTB port is also used by 2x16 alphanumeric LCD display, you cannot use both displays simultaneously. Potentiometer P3 is used for the GLCD display contrast adjustment. Switch 8 on the DIP switch SW6 is used for turning on/off display backlight.

.

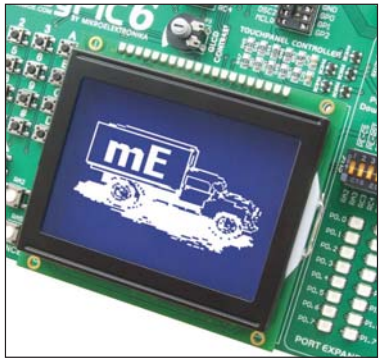


Figure 17-1: GLCD display

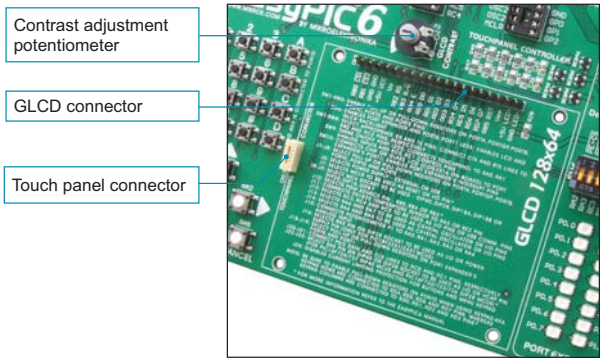


Figure 17-2: GLCD connector

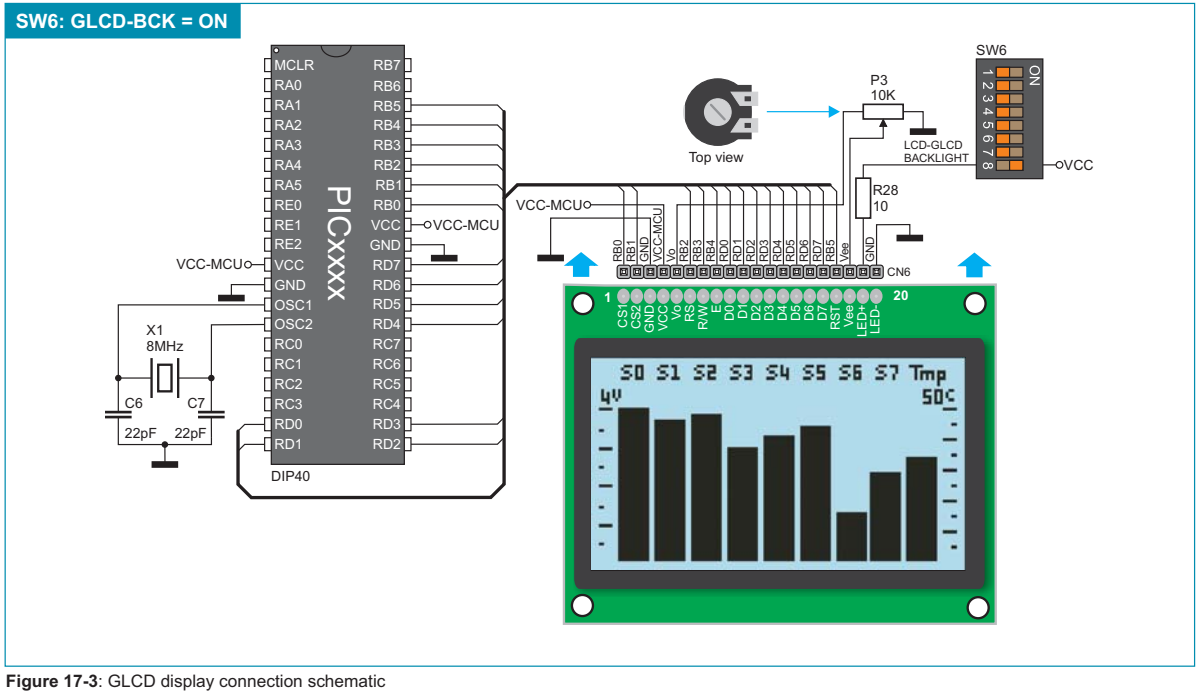


Figure 17-3: GLCD display connection schematic

18.0. Touch Panel

The touch panel is a thin, self-adhesive, transparent panel sensitive to touch. It is placed over a GLCD display. The main purpose of this panel is to register pressure at some specific display point and to forward its coordinates in the form of analog voltage to the microcontroller. Switches 5,6,7 and 8 on the DIP switch SW9 are used for connecting touch panel to the microcontroller.

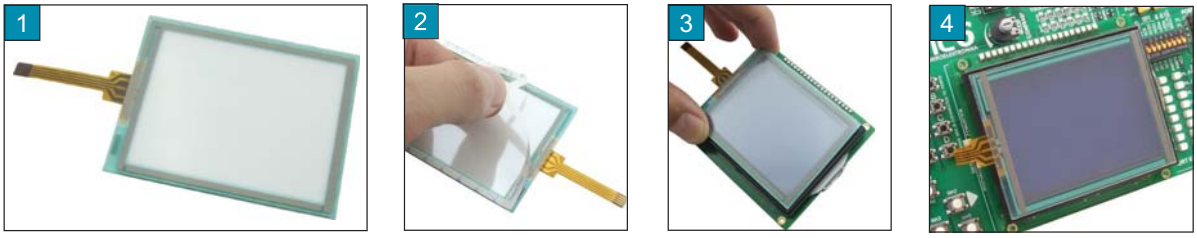


Figure 18-1: Touch panel

Figure 18-1 shows how to place a touch panel over a GLCD display. Make sure that the flat cable is to the left of the GLCD display, as shown in Figure 4.

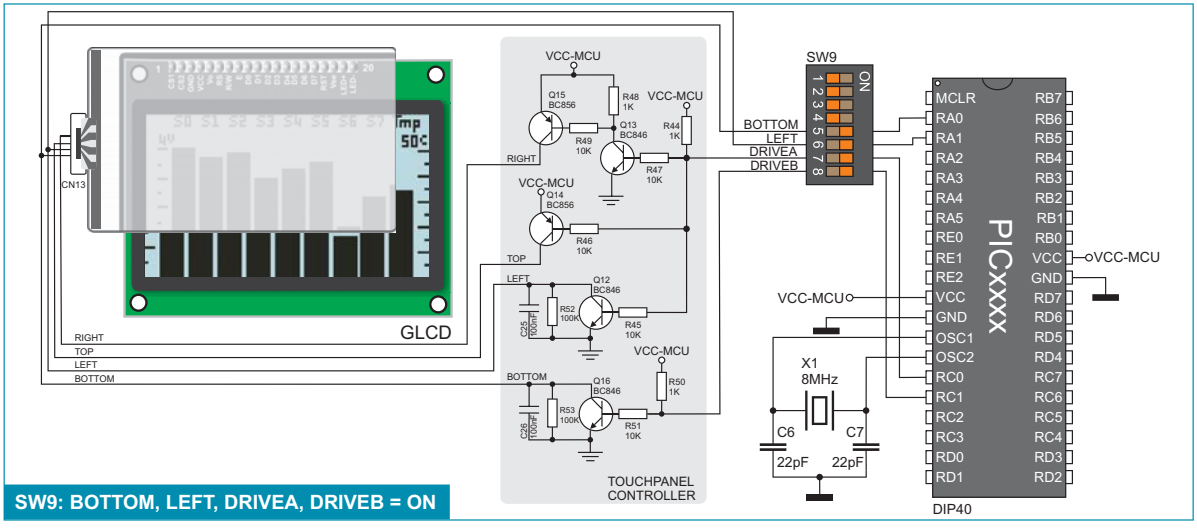


Figure 18-2: Touch panel connection schematic

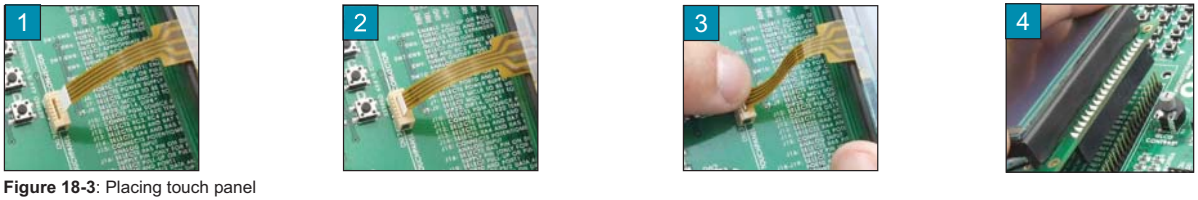


Figure 18-3: Placing touch panel

Figure 18-3 shows in detail how to connect a touch panel to the microcontroller. Bring the end of the flat cable close to the CN13 connector as shown in Figure 1. Plug the cable into the connector, as shown in Figure 2, and press it easily so as to fit the connector, as shown in Figure 3. Now you can plug a GLCD display into the appropriate connector as shown in Figure 4.

NOTE: LEDs and pull-up/pull-down resistors on the RA0 and RA1 pins of the PORTA port must be turned off when using a touch panel.

19.0. Input/Output Ports

Along the right side of the development system, there are seven 10-pin connectors which are connected to the microcontroller's I/O ports. Some of the connector's pins are directly connected to the microcontroller pins, whereas some of them are connected using jumpers. DIP switches SW1-SW5 enable each connector pin to be connected to one pull-up/pull-down resistor. Whether port pins are to be connected to a pull-up or pull-down resistor depends on the position of jumpers J1-J5.

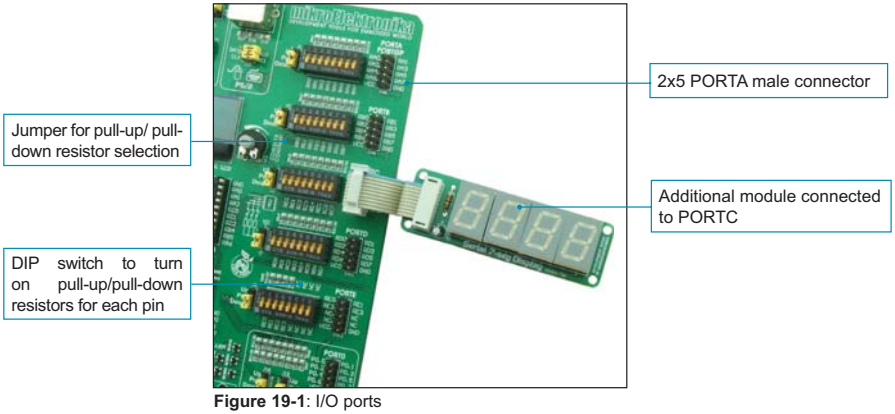


Figure 19-1: I/O ports

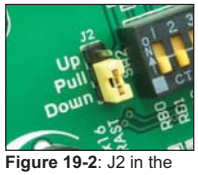


Figure 19-2: J2 in the pull-down position

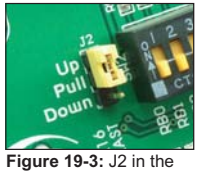


Figure 19-3: J2 in the pull-up position

SW2: 1-8 = ON
Jumper J2 in the pull-down position
Jumper J17 in the VCC-MCU position

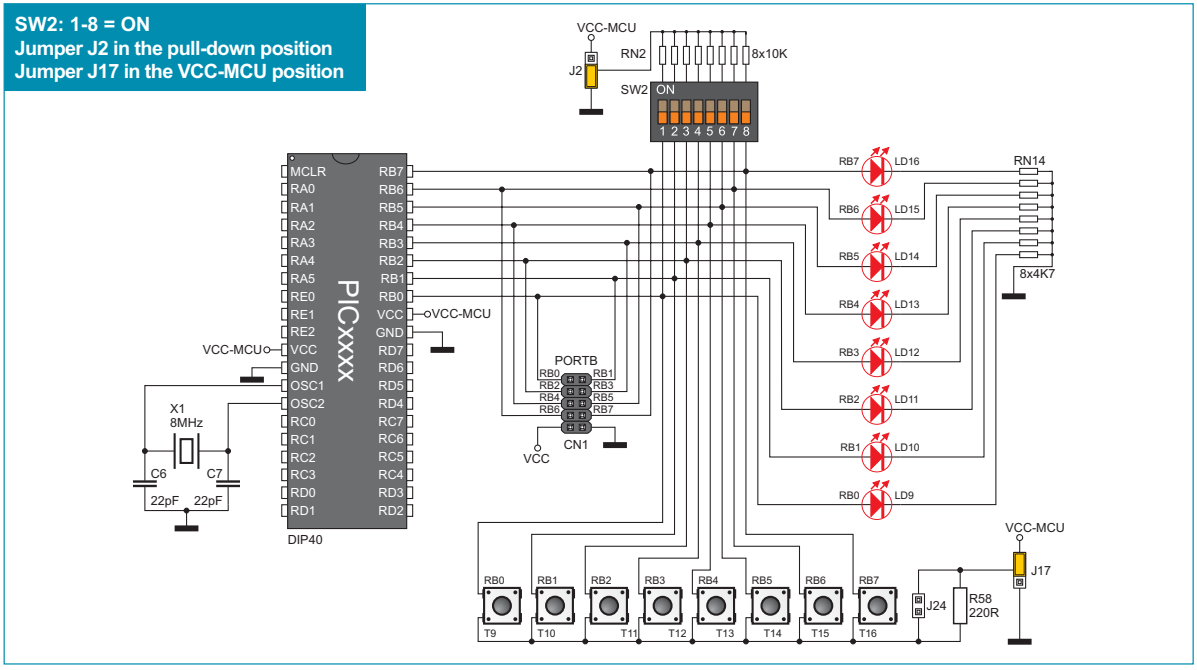


Figure 19-4: PORTB schematic connection

Pull-up/pull-down resistors enable voltage signal to be brought to the microcontroller pins. The logic level at pin idle state depends on the pull-up/pull-down jumper position. The RB0 pin along with the relevant DIP switch SW2, jumper J2 and RB2 push button with jumper J17 are used here for the purpose of explaining the performance of pull-up/pull-down resistors. The principle of their operation is identical for all the microcontroller pins.

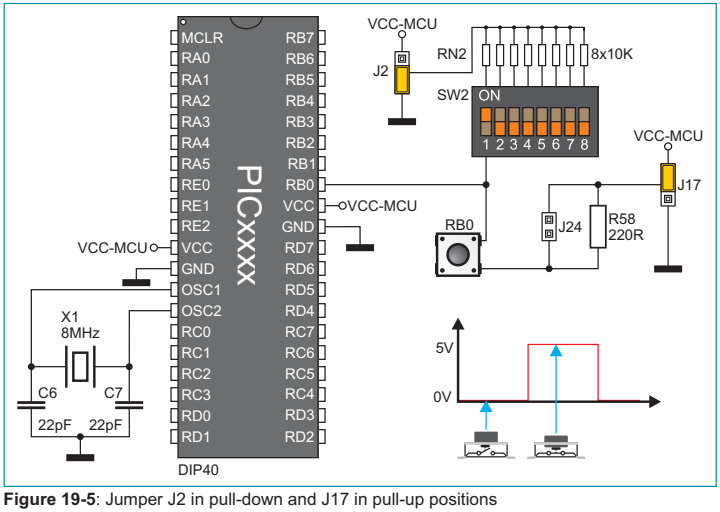


Figure 19-5: Jumper J2 in pull-down and J17 in pull-up positions

In order to enable PORTB pins to be connected to pull-down resistors, it is necessary to set jumper J2 in the lower position, thus providing 8x10K resistor network with a logic zero (0V). To bring a signal to the RB0 pin, it is necessary to set switch 1 on the DIP switch SW2 to the ON position. This will cause the microcontroller RB0 pin to be 'pulled down' to the low logic level (0V) in its idle state.
Jumper J17, used to determine the pin logic state provided by pressing push-buttons, should be set in the opposite position of jumper J2.

Accordingly, every time you press the RB0 push button, a logic one (1) will appear on the RB0 pin.

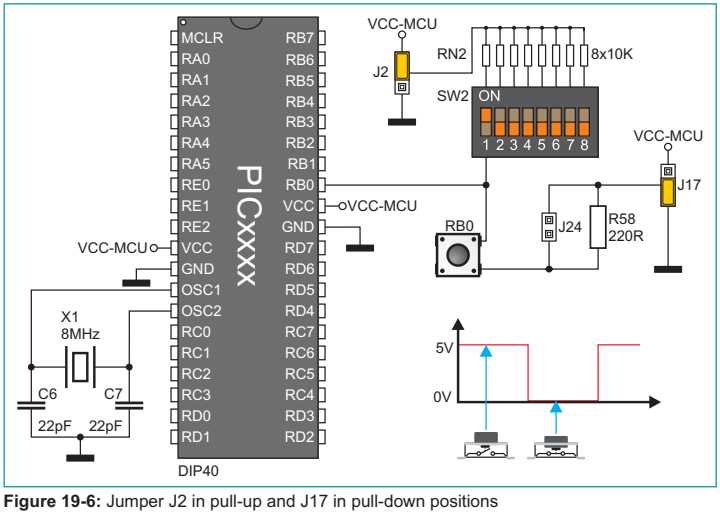


Figure 19-6: Jumper J2 in pull-up and J17 in pull-down positions

In order to enable PORTB pins to be connected to pull-up resistors, it is necessary to set jumper J2 in the upper position (5V) and jumper J17 in the lower position (0V). This enables each PORTB pin to be 'pulled up' to the high logic level (5V) in its idle state. In order to do this, it is necessary to set appropriate switch on the DIP switch SW2 to the ON position.

Accordingly, every time you press the RB0 push button, a logic zero (0) will appear on the RB0 pin.

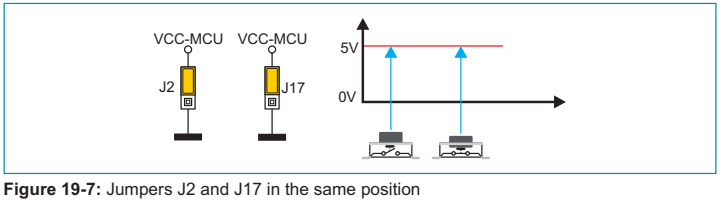


Figure 19-7: Jumpers J2 and J17 in the same position

In this case, jumpers J2 and J17 have the same logic state which means that pressing push button will not cause any pin to change its logic state.

20.0. Additional I/O Ports

The SPI communication lines and MCP23S17 circuit provide the EasyPIC6 development system with a means of increasing the number of available I/O ports by two. If the port expander is connected over the DIP switch SW6, the following pins RA2, RA3, RC3, RC4 and RC5 will be used for SPI communication and thus cannot be used as I/O pins. Switches INTA and INTB on the DIP switch SW10 enable interrupt. MCP23S17 enables 16-bit parallel expansion and may be configured to operate in either 16- or 8-bit mode.

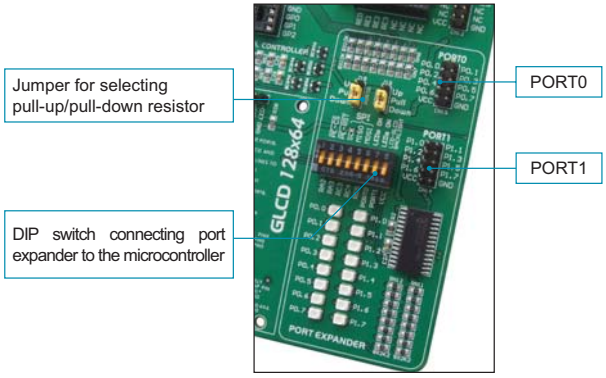


Figure 20-1: Port expander

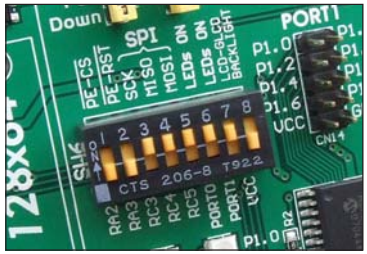


Figure 20-2: DIP switch SW6 when port expander is enabled

SW6: CS#, RST, SCK, MISO, MOSI = ON
SW10: INTA, INTB = ON
Jumpers J18 and J19 in the upper position

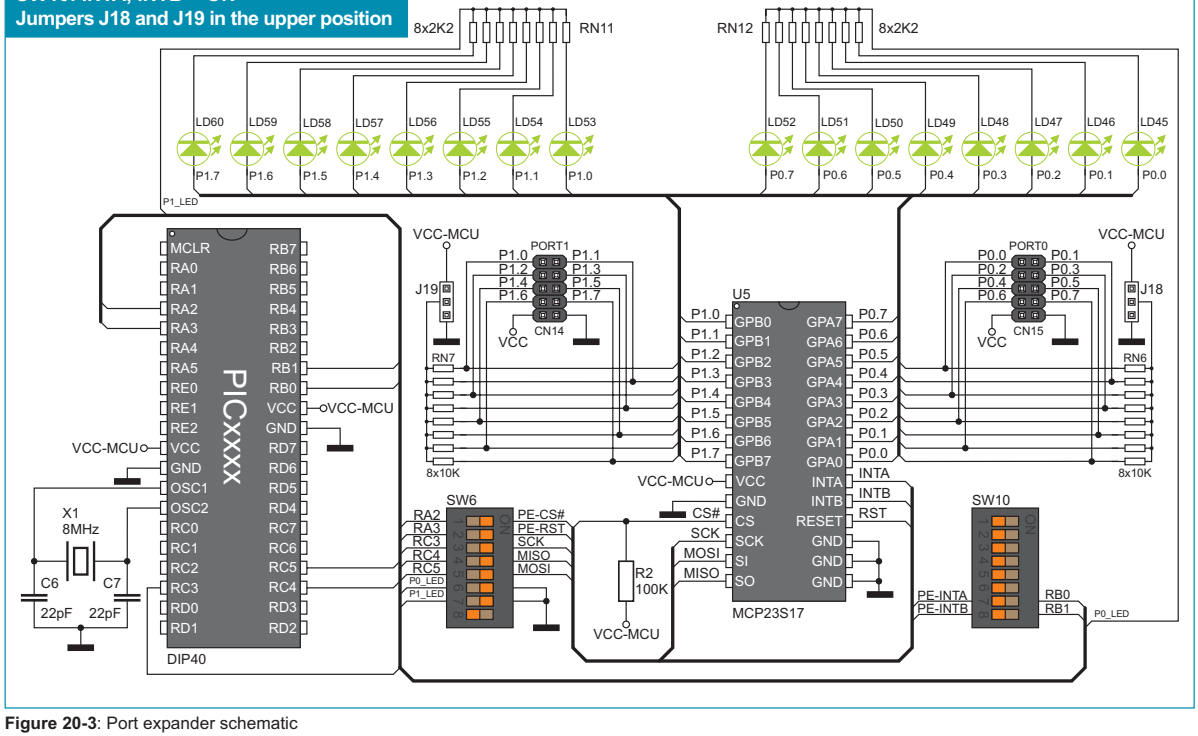


Figure 20-3: Port expander schematic

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.

All the 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.

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.

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