EasyAVR6™

User manual

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.

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 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

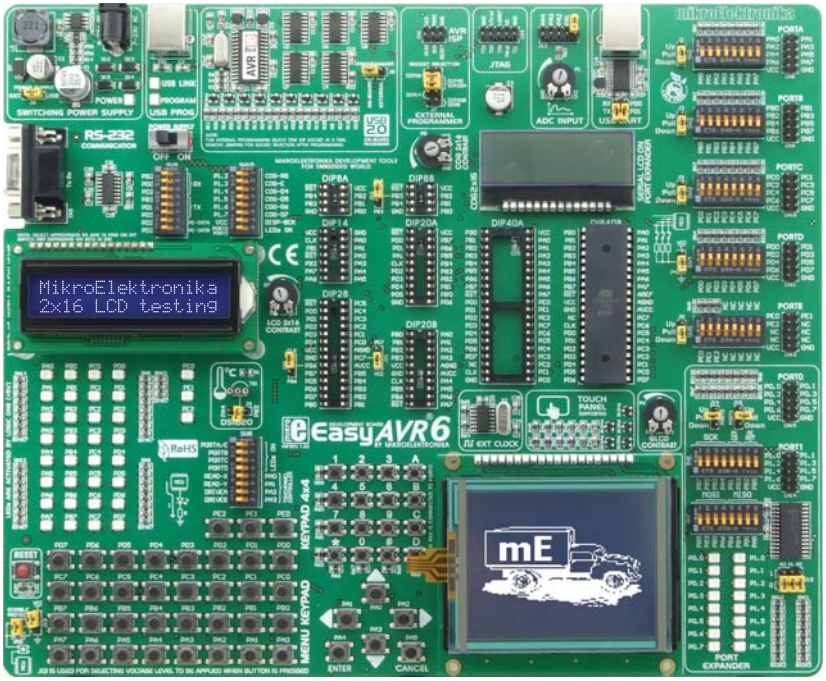
The Atmel name and logo, the Atmel logo, AVR, AVR (Logo), AVR Freaks, AVR Freaks (Logo), AVR Studio, IDIC, megaAVR, megaAVR (Logo), picoPower ®, tinyAVR ® are trademarks of Atmel Coorporation.

TABLE OF CONTENTS

Introduction to EasyAVR6 Development System..... 4
Key Features..... 5
1.0. Connecting the System to your PC..... 6
2.0. Supported Microcontrollers..... 7
3.0. On-Board USB 2.0 AVRprog Programmer..... 8
4.0. External AVRISP mkII Programmer..... 9
5.0 JTAG Connector..... 10
6.0 Clock Oscillator..... 10
7.0. Power Supply..... 11
8.0. RS-232 Communication Interface..... 12
9.0. USB UART Module..... 13
10.0. DS1820 Temperature Sensor..... 14
11.0. A/D Converter Test Inputs..... 15
12.0. LEDs..... 16
13.0. Push Buttons..... 17
14.0. Keyboards..... 18
15.0. Alphanumeric 2x16 LCD 19
16.0. On-Board 2x16 LCD with Serial Communication..... 20
17.0. 128x64 Graphic LCD..... 21
18.0. Touch Panel..... 22
19.0. I/O Ports..... 23
20.0. Port Expander (Additional I/O Ports)..... 25

Introduction to EasyAVR6 Development Board

The *EasyAVR6*™ development system is an extraordinary development tool suitable for programming and experimenting with *AVR*® microcontrollers from *Atmel*®. Such development system includes an on-board programmer providing an interface between the microcontroller and the PC. You are simply expected to write a code in one of the *AVR* compilers, generate a HEX file and program your microcontroller using the *AVRprog*™ programmer. Numerous on-board modules, such as 128x64 graphic LCD, alphanumeric 2x16 LCD, on-board 2x16 LCD with serial communication, keypad 4x4, port expander etc., allow you to easily simulate the operation of the target device.



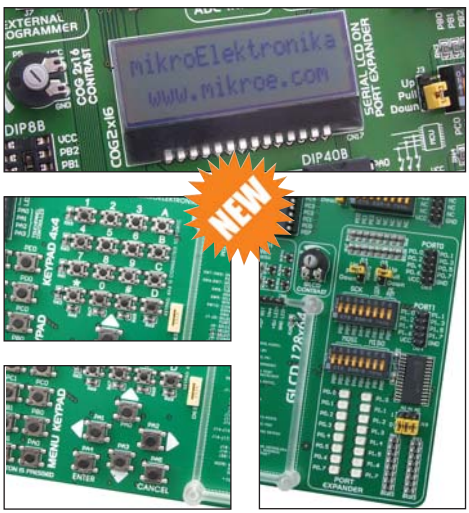
Full-featured and user-friendly development board for AVR microcontrollers

High-Performance USB 2.0 On-Board Programmer

Port Expander provides easy I/O expansion (2 additional ports) using data format conversion

Alphanumeric On-Board 2x16 LCD with Serial Communication

Graphic LCD with backlight



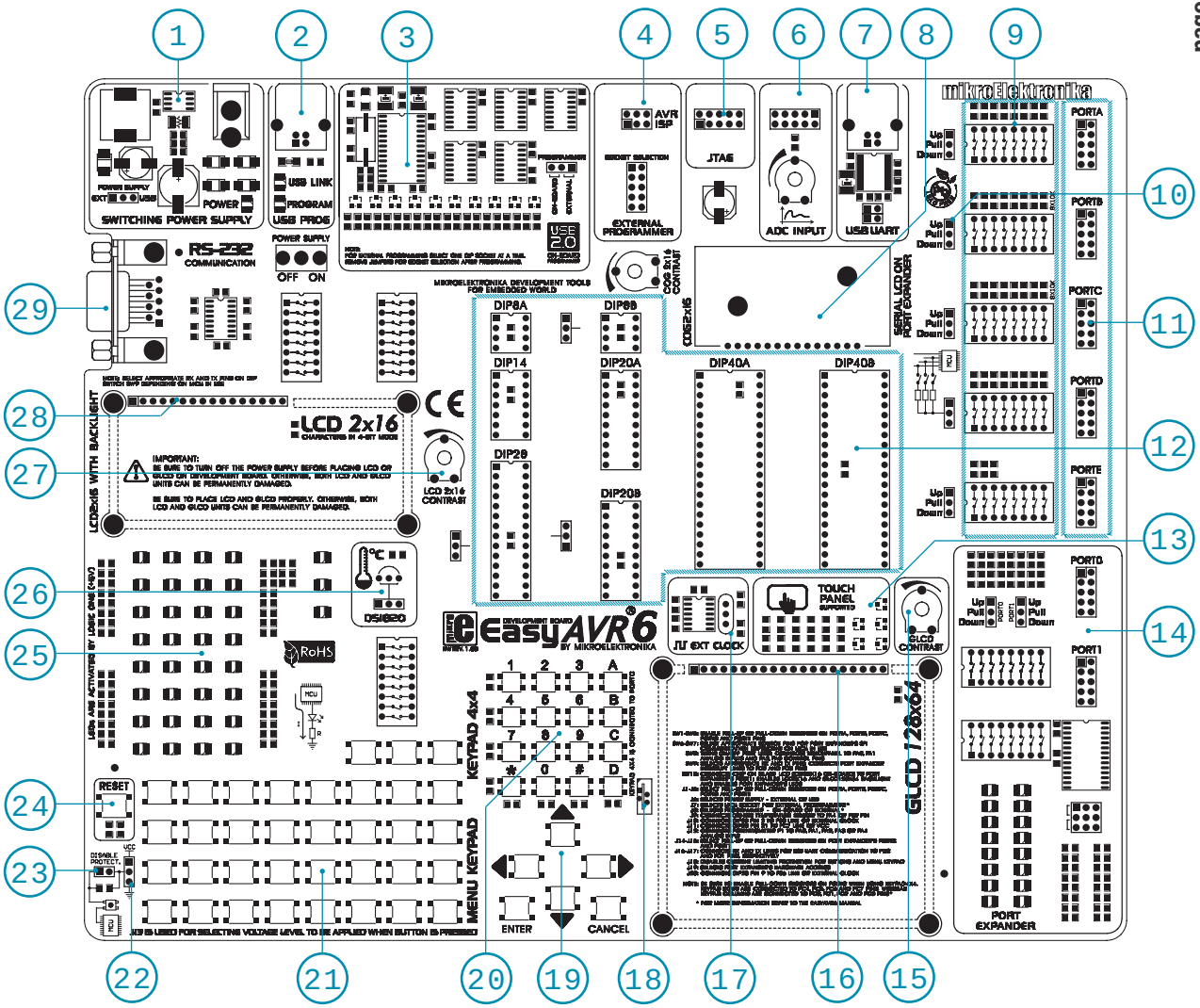
The *AVRflash*™ program provides a complete list of all supported microcontrollers. The latest version of this program with updated list of supported microcontrollers can be downloaded from our website **www.mikroe.com**

Package contains:

Development board:	*EasyAVR6*
CD:	product CD with appropriate software
Cables:	USB cable
Documentation:	*EasyAVR6* and *AVRflash* manuals, *Installing USB drivers* manual and *Electrical Schematic* of the *EasyAVR6* 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:	50mA in idle state (when on-board modules are inactive)
Size:	26,5 x 22cm (10,4 x 8,6inch)
Weight:	~417g (0.92lbs)



Key Features

1. Power supply voltage regulator
2. On-board programmer's USB connector
3. On-boad USB 2.0 programmer *AVRprog*
4. External *AVRISP*® programmer's connector
5. *JTAG*® interface connector
6. A/D converter test inputs
7. USB UART Module
8. On-board 2x16 LCD
9. DIP switches to enable pull-up/pull-down resistors
10. Pull-up/pull-down resistor selection
11. I/O port connectors
12. AVR microcontroller sockets
13. Touch panel controller
14. Port expander
15. 128x64 graphic LCD contrast potentiometer
16. 128x64 graphic LCD connector
17. Clock oscillator
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. 35 LEDs to indicate pins' logic state
26. DS1820 temperature sensor socket
27 Alphanumeric LCD contrast adjustment
28. Alphanumeric LCD connector
29. RS-232 communication connector

1.0. Connecting the System to your PC

Step 1:

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

Step 2:

Use the USB cable to connect the *EasyAVR6* 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.

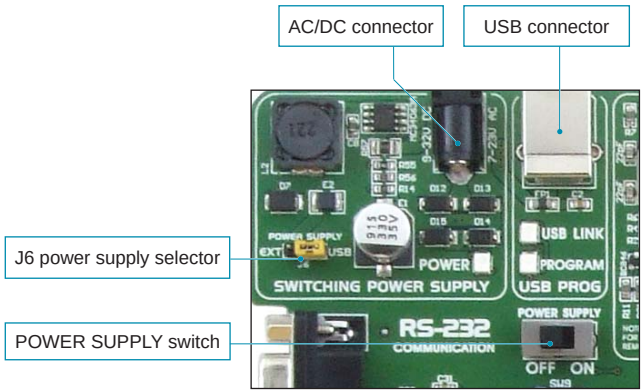


Figure 1-1: Power supply

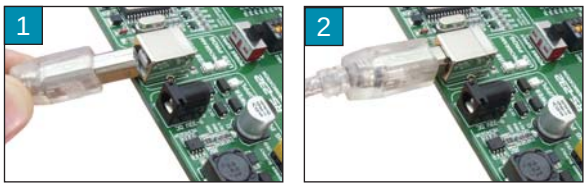


Figure 1-2: Connecting USB cable (J6 in USB position)

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 turned on to indicate that your development system is ready to use. Use the on-board *AVRprog* programmer and *AVRflash* program 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. Refer to Figure 1-3 for their proper placing.

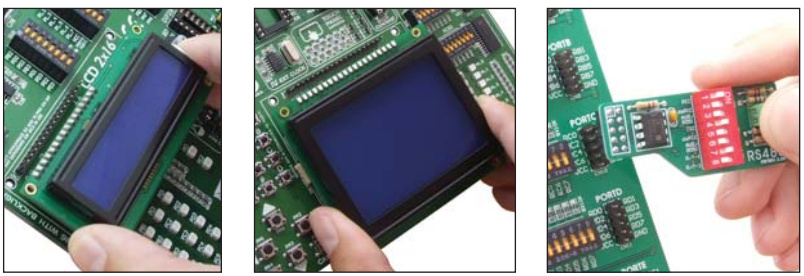


Figure 1-3: Placing additional modules on the board

2.0. Supported Microcontrollers

The *EasyAVR6* development system provides eight separate sockets for AVR microcontrollers in DIP40, DIP28, DIP20, DIP14 and DIP8 packages.The development system EasyAVR6 comes with the microcontroller in a DIP40 package.

Figure 2-1: Microcontroller sockets

There are two sockets for AVR microcontrollers in DIP40, DIP20 and DIP8 packages provided on the board. Which of these sockets will you use depends solely on the pinout of the microcontroller in use. Jumpers J10, J11 and J20 next to the sockets DIP28 and DIP8 are used for selecting functions of the microcontroller pins:

Jumper	Position	Function
J10	PB3	PB3 is an I/O pin
	CLK	Pin PB3 is fed with a clock signal from the on-board oscillator
J11	VCC	Pin is connected to VCC
	PC7	PC7 is an I/O pin
J20	CLK	Pin PB6 is fed with a clock signal from the on-board oscillator
	PB6	PB6 is an I/O pin

AVR microcontrollers can use either built-in (internal) or on-board (external) oscillator as a clock signal source. The clock oscillator provided on the board generates clock signals for most supported microcontrollers.

- Microcontrollers plugged into the DIP8A socket use built-in oscillator for clock generation and are not connected to the on-board oscillator.
- Microcontrollers plugged into the DIP8B socket may use either internal or external oscillator, which depends on the jumper J10 position.

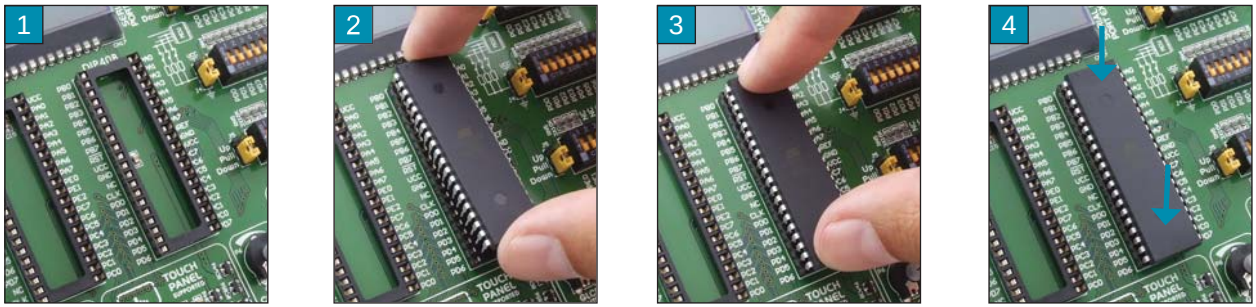


Figure 2-2: Plugging microcontroller into appropriate socket

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 into the appropriate socket. Figure 1 shows an unoccupied DIP40 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 AVRprog Programmer

The *AVRprog* programmer is a tool used for dumping .hex code into the microcontroller. The *EasyAVR6* has an on-board AVRprog programmer which allows you to establish a connection between the microcontroller and your PC. Figure 3-2 shows the connection between a compiler, *AVRflash* program and microcontroller.

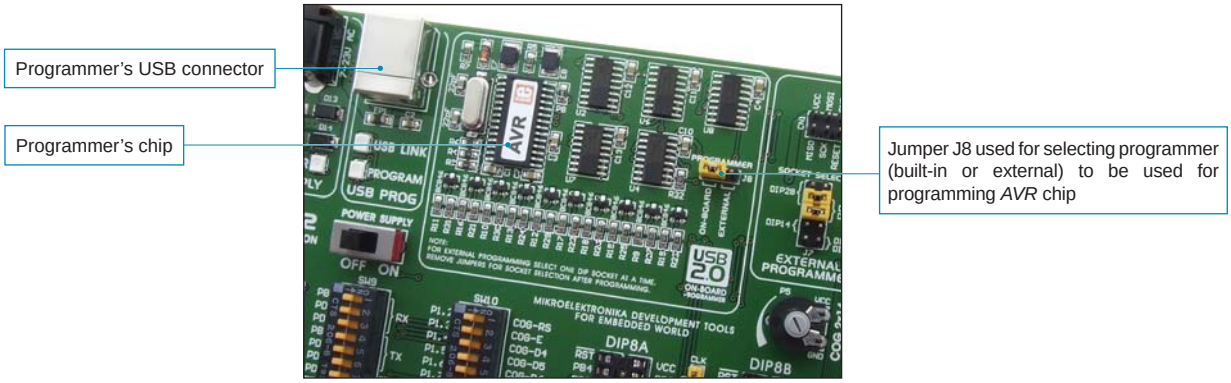
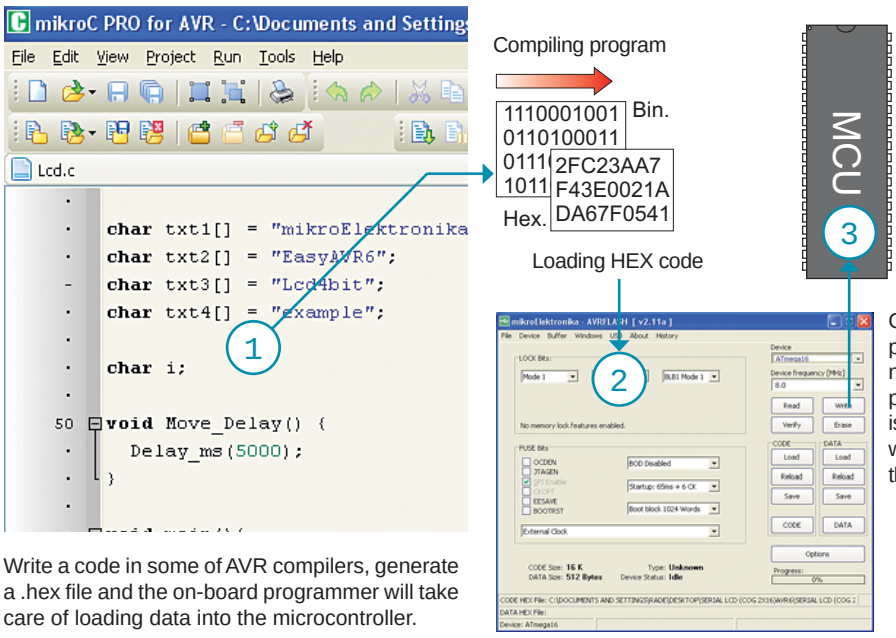


Figure 3-1: *AVRprog* programmer



① Write a program in some of AVR compilers and generate a HEX file;

② Use the *AVRflash* program to select an appropriate microcontroller and to load the HEX file;

③ Click the *Write* button to load the program into the microcontroller.

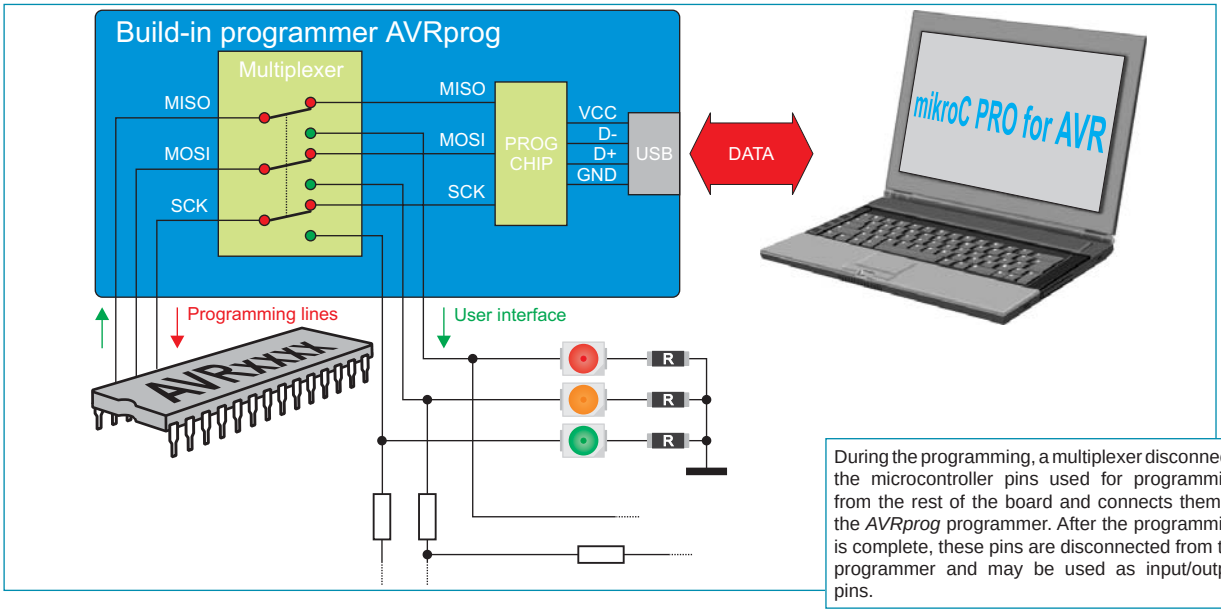
On the right side of the *AVRflash* program's main window there are a number of buttons which make the programming process easier. There is also an option at the bottom of the window which enables you to monitor the programming progress.

Write a code in some of AVR compilers, generate a .hex file and the on-board programmer will take care of loading data into the microcontroller.

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

NOTE: For more information on the *AVRprog* programmer refer to the relevant manual provided in the *EasyAVR6* development system package.

AVR microcontrollers are programmed by means of SPI serial communication using the following microcontroller pins MISO, MOSI and SCK.



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

4.0. External AVRISP mkII Programmer

In addition to the on-board programmer, the *EasyAVR6* development system may also use the external *AVRISP* programmer from *Atmel* for programming microcontrollers. Such programmer is plugged into the *AVRISP* connector.
In order to enable a microcontroller to be programmed using this programmer, it is necessary to set jumper J8 in the EXTERNAL position prior to turning the programmer on. Then use jumper J7 to select the appropriate microcontroller socket.

Figure 4-1: Setting jumper J7

Jumper J8 in the EXTERNAL position enables external *AVRISP* programmer

Jumper J8 in the ON-BOARD position enables on-board programmer

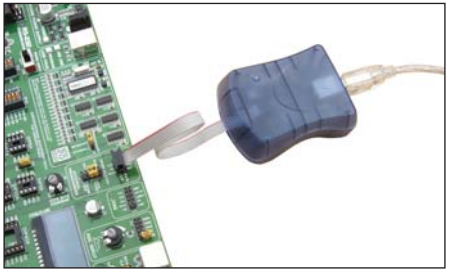


Figure 4-2: *AVRISP mkII* connected to the development system

The position of jumper J7 when the external programmer is used for programming microcontrollers in DIP20B and DIP8 packages

The position of jumper J7 when the external programmer is used for programming microcontrollers in DIP14 package

The position of jumper J7 when the external programmer is used for programming microcontrollers in DIP40 and DIP20A packages

The position of jumper J7 when the external programmer is used for programming microcontrollers in DIP28 package

page

5.0. JTAG Connector

JTAG ICE is an emulator used for AVR microcontrollers with built-in *JTAG* interface (*Mega* AVR microcontrollers). *JTAG ICE* is primarily intended for work with the *AVR Studio* program. The *JTAG* connector built into *AVR* microcontrollers is a modified version of the original *JTAG* interface. It enables contents of internal EEPROM and FLASH memory to be changed (programming microcontroller).

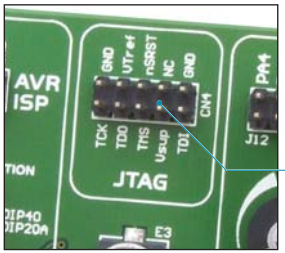


Figure 5-1: *JTAG* connector

JTAG ICE emulator employs a male 2x5 connector to establish connection with the development system

Figure 5-2: *JTAGICE mkII* connected to the development system

The *JTAG* connector is directly connected to the microcontroller pins so that it doesn't depend on jumpers J7 and J8 settings which otherwise have to be performed when using *AVRprog* and *AVRISP* programmers.

6.0. Clock Oscillator

There is a clock oscillator provided on the board used as a clock signal external source. The quartz crystal used for the purpose of stabilizing clock frequency is plugged into the appropriate socket and therefore can always be replaced with another one. Its maximum value depends on the maximum operating frequency of the microcontroller.

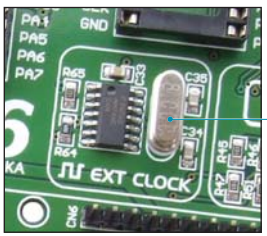


Figure 6-1: Oscillator

Quartz crystal X2 plugged into the appropriate socket, which enables it to be easily replaced

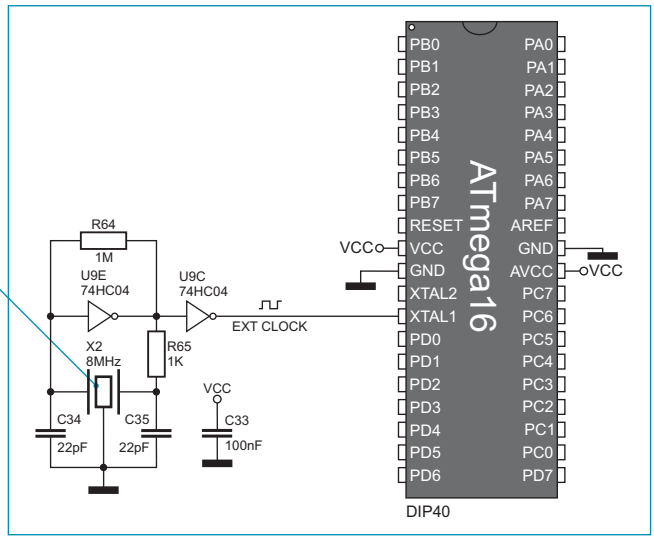


Figure 6-2: Oscillator connection schematic

7.0. Power Supply

The *EasyAVR6* 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 AC/DC connector provided on the development board.

The MC34063A voltage regulator and Gretz rectifier are 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 a 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 on by setting the POWER SUPPLY switch in the ON position.

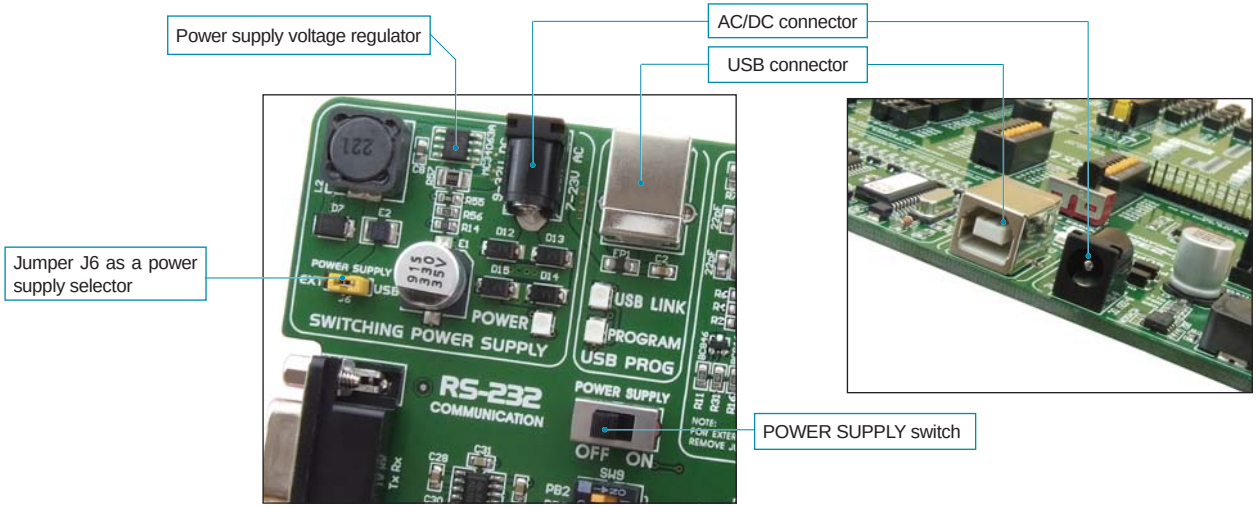


Figure 7-1: Power supply

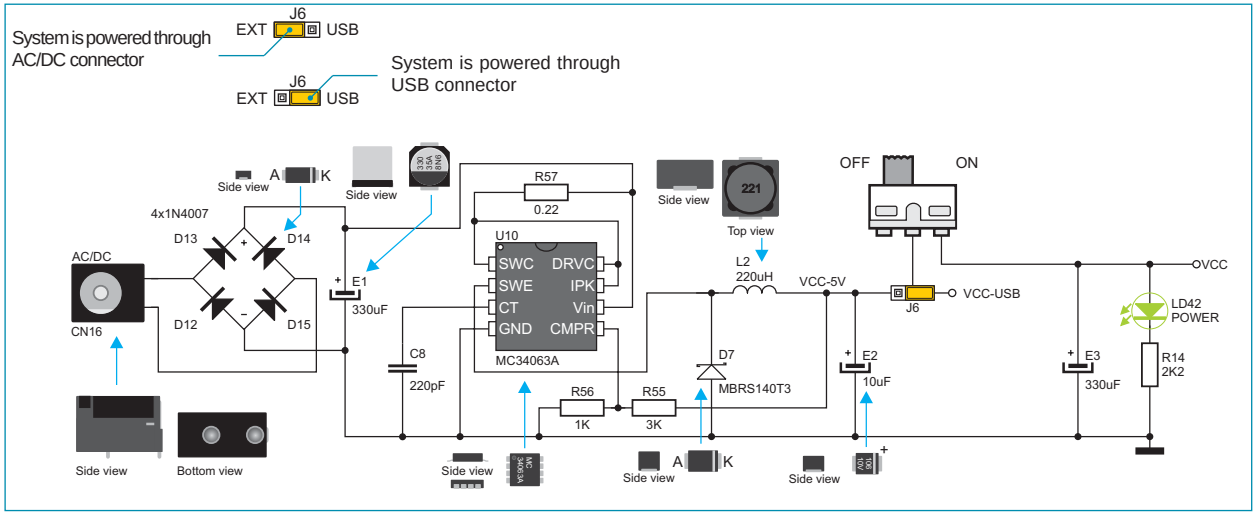


Figure 7-2: Power supply source connection schematic

page

8.0. RS-232 Communication Interface

The USART (universal synchronous/asynchronous receiver/transmitter) is one of the most common ways of exchanging data between the PC and peripheral components. 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 and microcontroller pins provided with USART module using a DIP switch SW9. The microcontroller pins used in such communication are marked as follows: RX - *receive data* and TX - *transmit data*. Baud rate goes up to 115 kbps.
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 MAX202C.

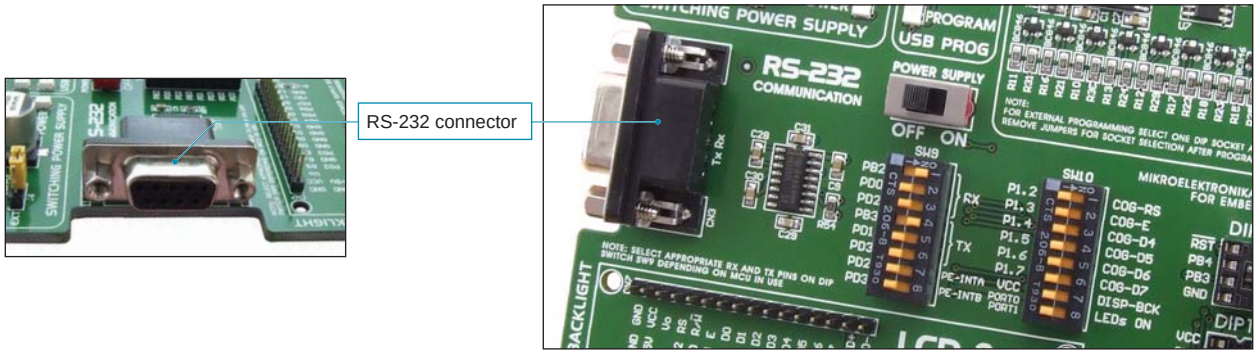


Figure 8-1: RS-232 module

The function of DIP switch SW9 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 8-2 shows the connection between the RS-232 module and the microcontroller in DIP40 package (*ATMEGA16*).

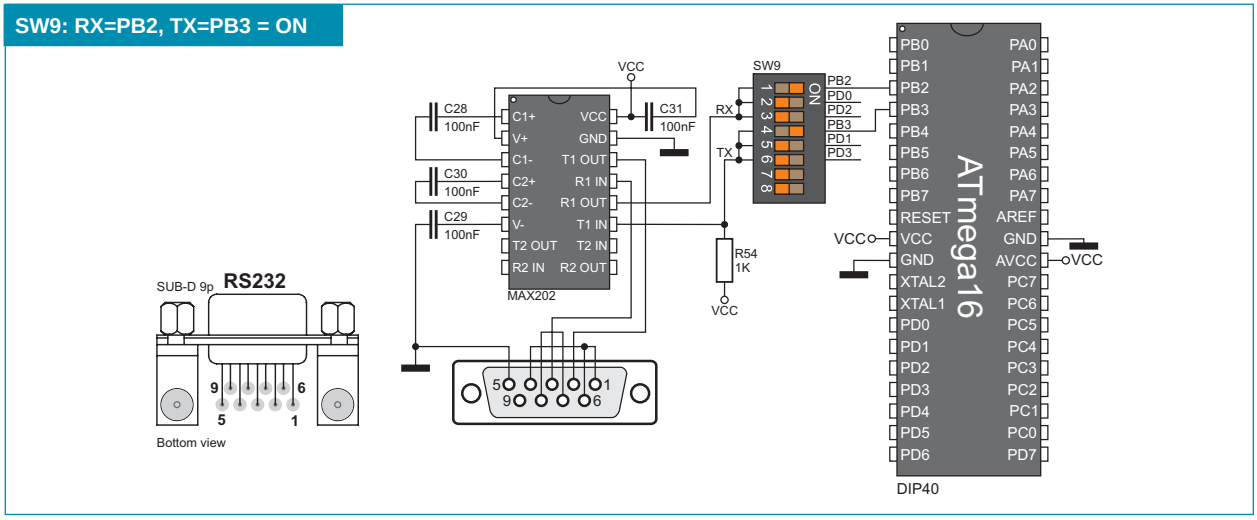


Figure 8-2: RS-232 module schematic

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

9.0. USB UART Module

When the USB UART module is used, it is possible to connect the microcontroller on the development system to the external USB device. The USB UART module features the FT232RL circuit which serves as an interface between a USB device and the microcontroller's serial UART module. In order to establish connection between the microcontroller and the USB UART module it is necessary to place jumpers J16 and J17 as shown in Figure 9-2, thus connecting RX-MCU and TX-MCU pins of the circuit FT232RL to the microcontroller pins PD0 and PD1.
When connecting the UART module to the microcontroller it is necessary to combine the transmit and receive lines, i.e. to connect the UART module's RX pin to the microcontroller pin TX and the UART module's TX pin to the microcontroller pin RX. Otherwise, it wouldn't be possible to connect the UART module's transmit line (TX) to the microcontroller transmit line (TX). It further means that the microcontroller wouldn't be able to receive and transmit data.

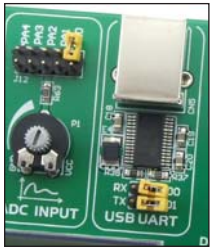


Figure 9-1: USB UART module (J16 and J17 are not placed)

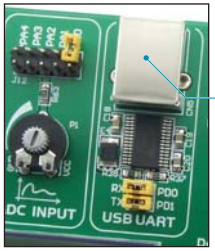
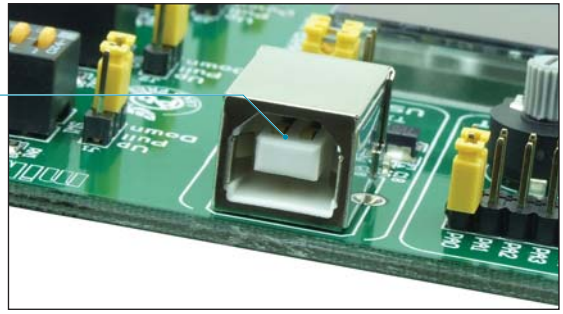


Figure 9-2: USB UART module (J16 and J17 are placed)

USB connector



Jumpers J16 and J17 are placed

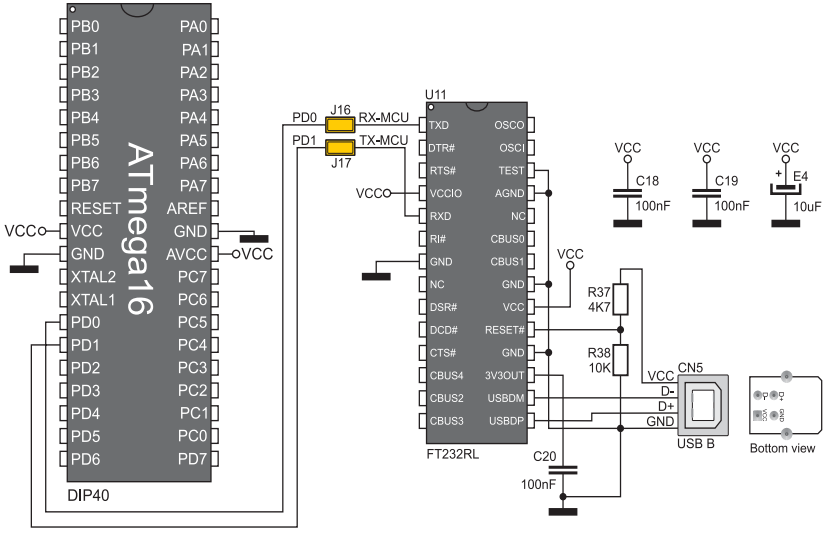


Figure 9-3: USB UART module and microcontroller connection schematic

page

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 *EasyAVR6* development system provides a separate socket for the DS1820. It may use either PA4 or PB2 pin for communication with the microcontroller. Jumper J9's purpose is selection of the pin to be used for 1-wire communication. Figure 10-4 shows 1-wire communication with microcontroller through the PA4 pin.

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

Figure 10-2: J9 in the left-hand position (1-wire communication through the PA4 pin)

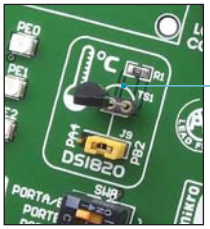


Figure 10-3: J9 in the right-hand position (1-wire communication through the PB2 pin)

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

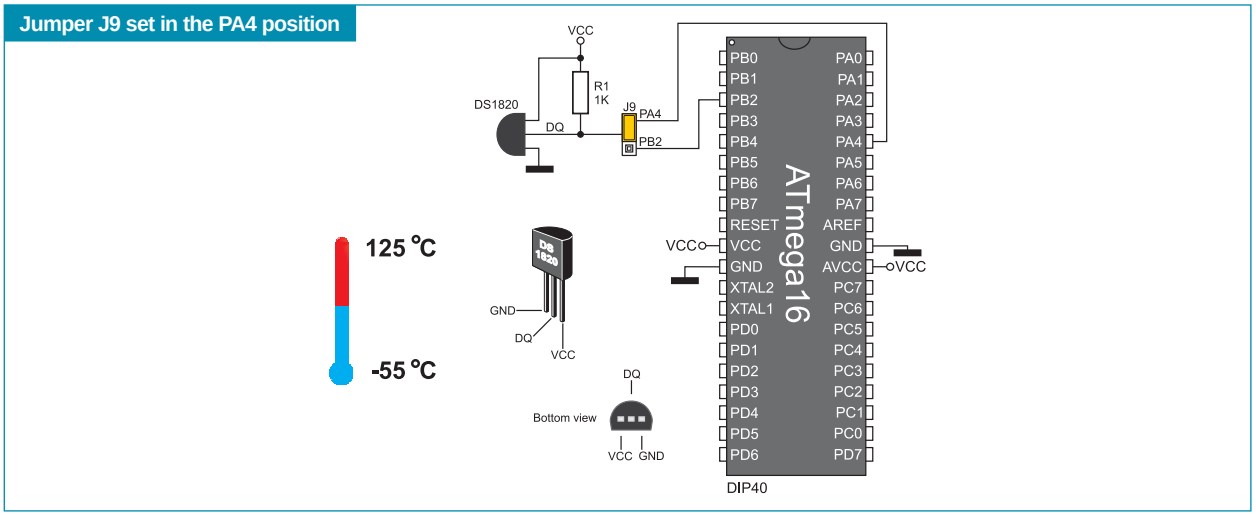


Figure 10-4: 1-wire communication connection schematic

11.0. A/D Converter Test Inputs

An A/D converter is used for 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 within the microcontroller 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 J12 is used for selecting some of the following pins PA0, PA1, PA2, PA3 or PA4. 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 (10k).

Figure 11-1: ADC (default jumper positions)

Figure 11-2: The PA0 pin used as A/D conversion input

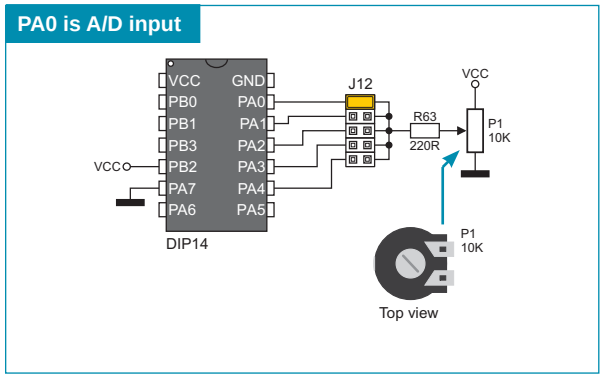


Figure 11-3: AVR microcontroller in DIP14 package and A/D converter test inputs connection

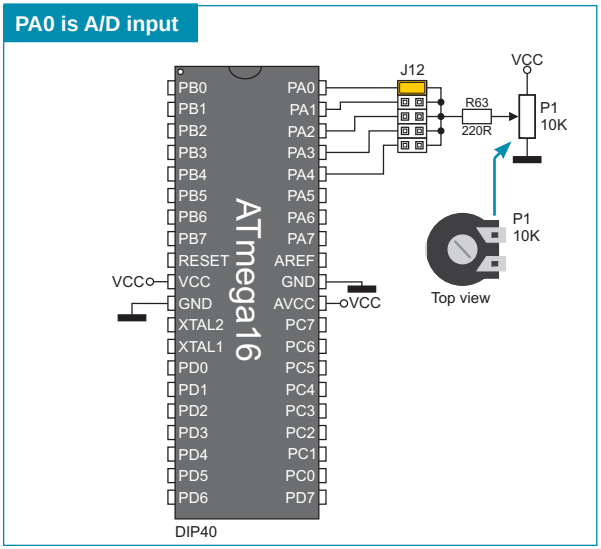


Figure 11-4: AVR microcontroller in DIP40 package and A/D converter test inputs connection

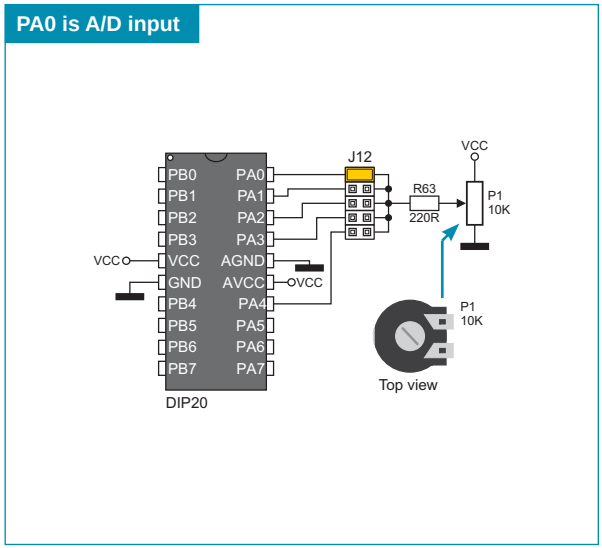


Figure 11-5: AVR microcontroller in DIP20B package and A/D converter test inputs connection

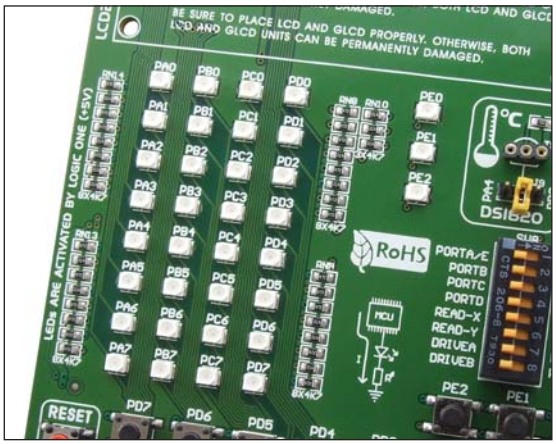
NOTE: In order to enable the microcontroller to accurately perform A/D conversion, it is necessary to turn off LED diodes and pull-up/pull-down resistors on port pins used by the A/D converter.

page

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 *EasyAVR6* development system uses LEDs with current I=1mA.

The *EasyAVR6* has 35 LEDs which visually indicate the 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 the pin state to be shown, it is necessary to select appropriate port PORTA/E, PORTB, PORTC or PORTD using the DIP switch SW8.



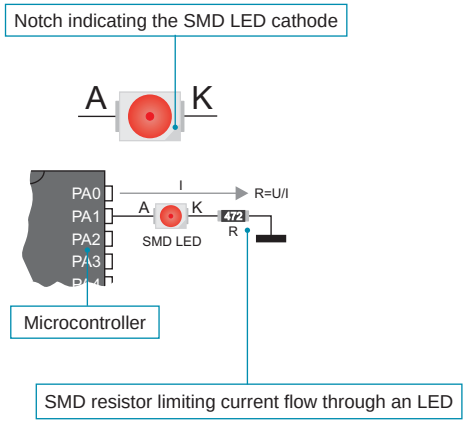


Figure 12-1: LEDs

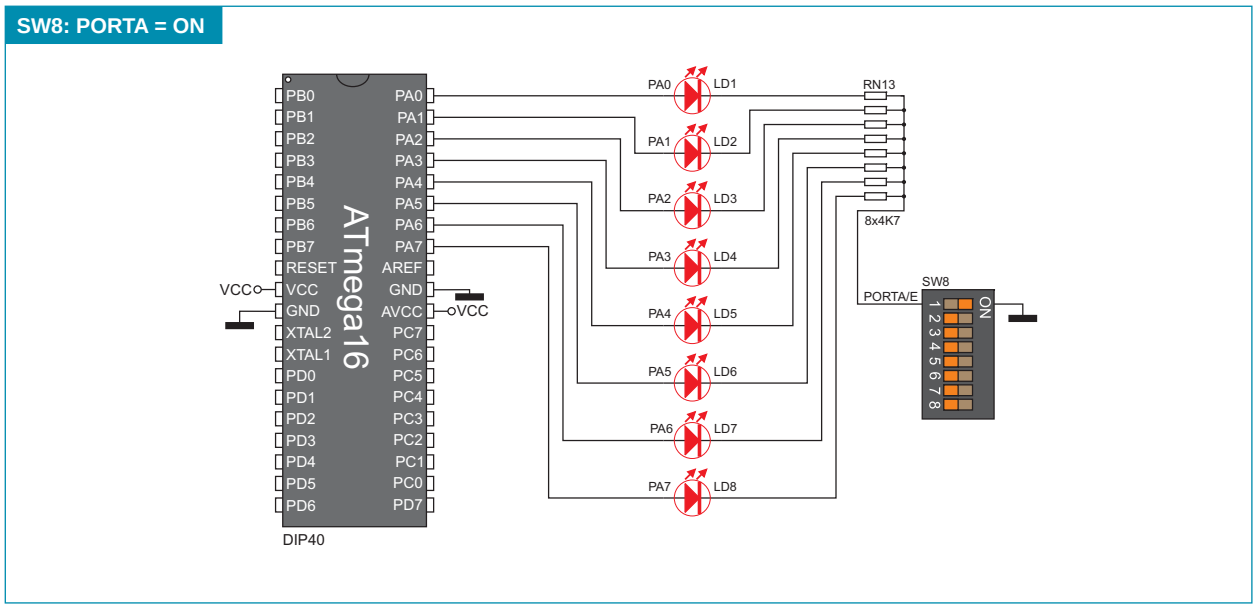


Figure 12-2: LED diodes and PORTA connection schematic

13.0. Push Buttons

The logic state of all microcontroller digital inputs may be changed using push buttons. Jumper J13 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 the maximum current thus preventing a short circuit from occurring. If needed, advanced users may short such resistor using jumper J18. 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.

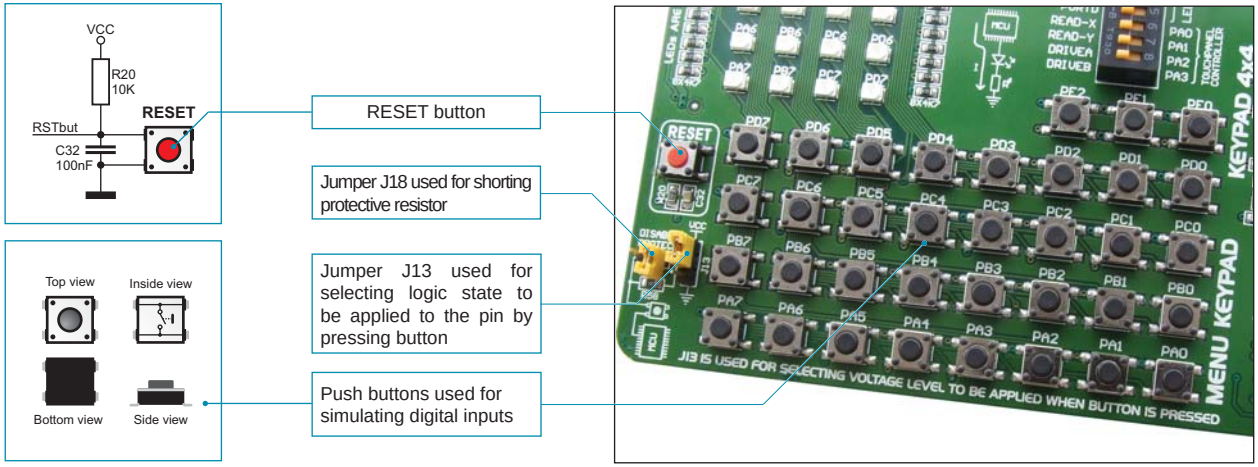


Figure 13-1: Push buttons used for simulating digital inputs

By pressing any push button (PA0-PA7) when jumper J13 is in the VCC position, a logic one (5V) will be applied to the appropriate microcontroller pin as shown in Figure 13-2.

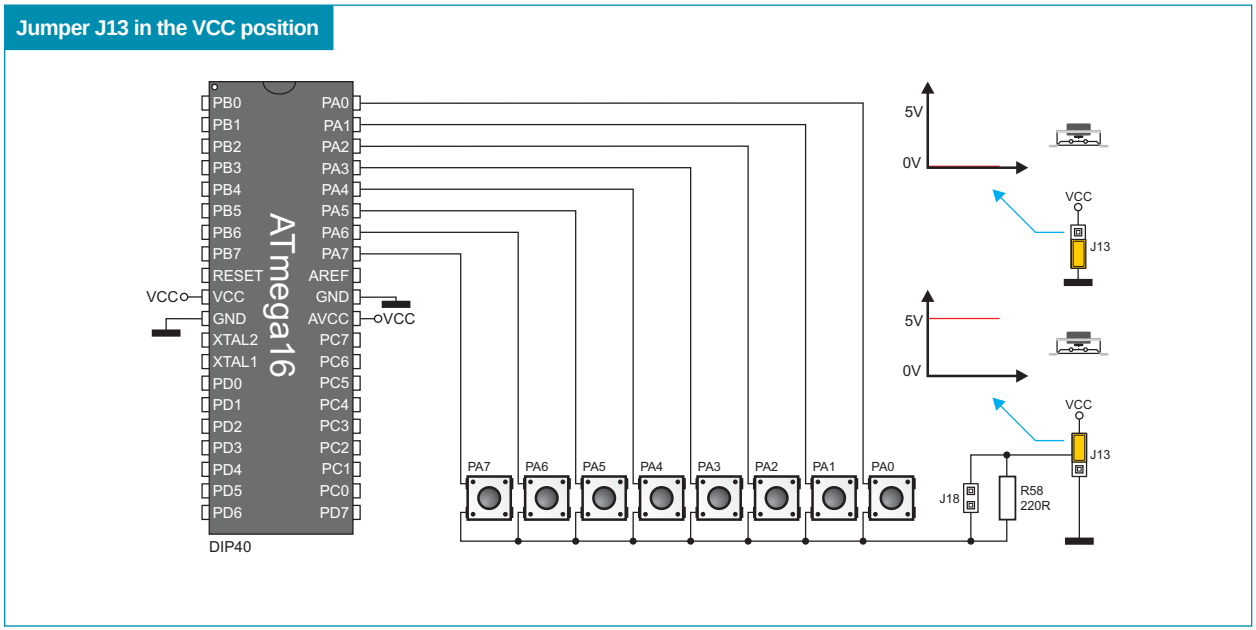


Figure 13-2: PORTA push buttons connection schematic

page

14.0. Keypads

There are two keypads provided on the *EasyAVR6* development system. These are keypad 4x4 and keypad *MENU*. Keypad 4x4 is a standard alphanumeric keypad connected to the microcontroller PORTC. The performance of such keypad is based on the 'scan and sense' principle where the PC0, PC1, PC2 and PC3 pins are configured as inputs connected to pull-down resistors. The PC4, PC5, PC6 and PC7 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 PC2 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 PC4, PC5, PC6 and PC7.

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.

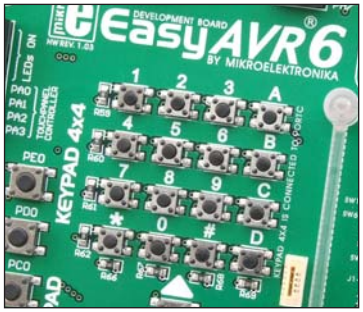


Figure 14-1: Keypad 4x4

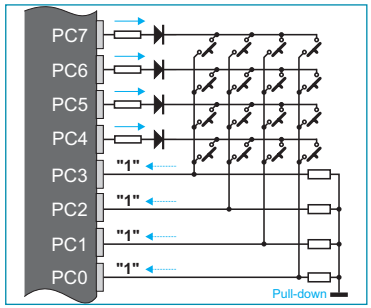


Figure 14-2: Keypad 4x4 performance

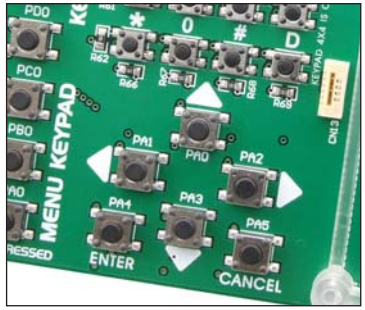


Figure 14-3: Keypad *MENU*

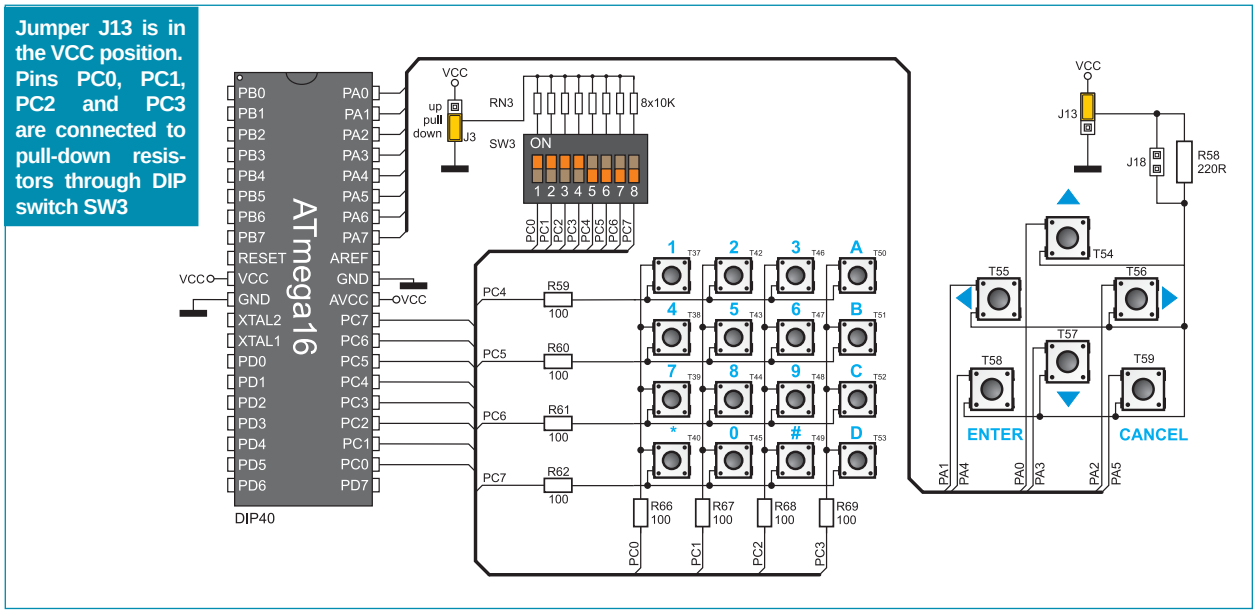


Figure 14-4: Keypads (4x4 and *MENU*) and microcontroller connection schematic

15.0. Alphanumeric 2x16 LCD

The *EasyAVR6* development system provides an on-board connector so that the alphanumeric 2x16 LCD can be plugged in. Such connector is linked to the microcontroller through the PORTD port. Potentiometer P7 is used for display contrast adjustment. The *DISP-BCK* switch on the DIP switch SW10 is used for turning on/off display backlight.

Communication between an LCD 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.

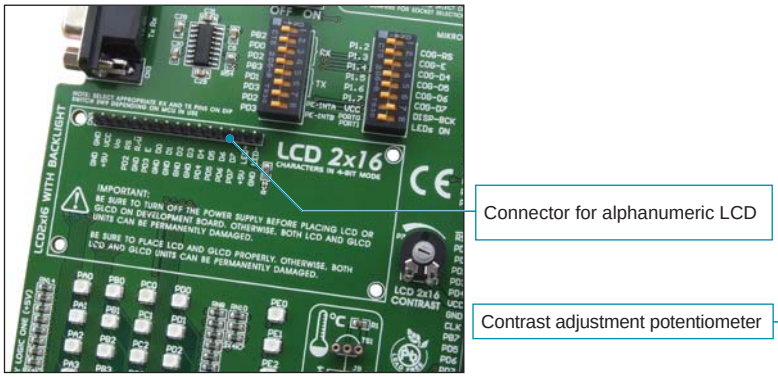


Figure 15-1: Alphanumeric 2x16 LCD connector

Figure 15-2: 2x16 LCD

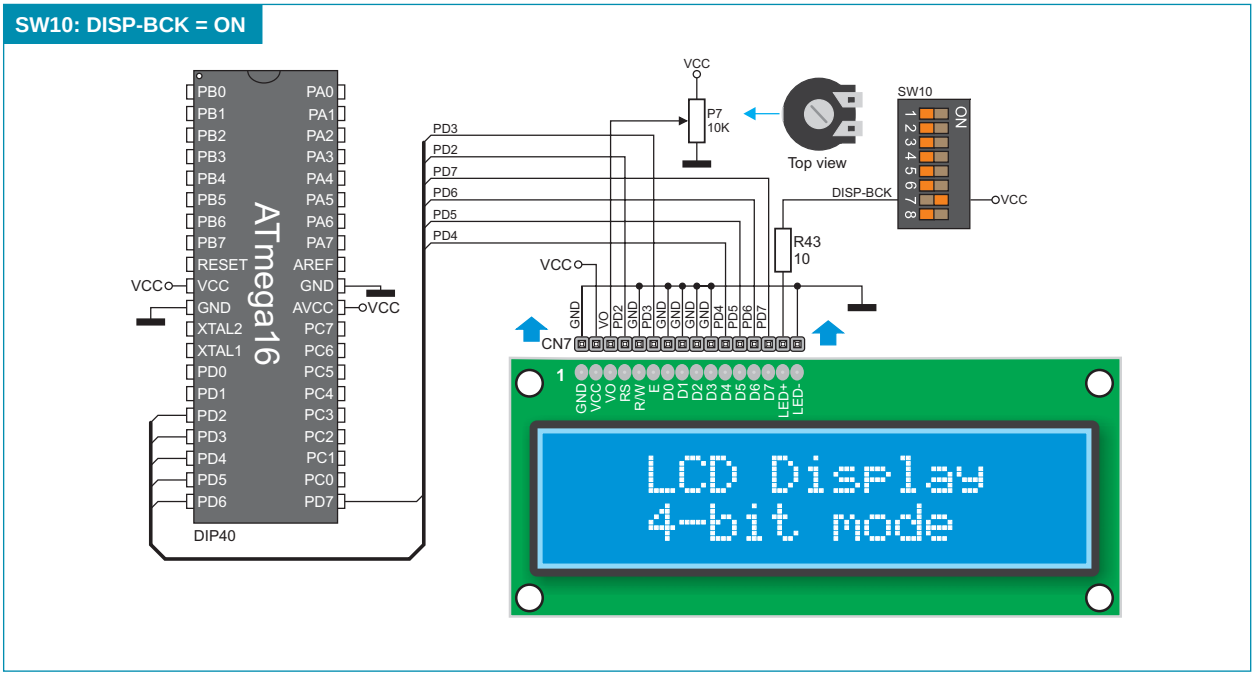


Figure 15-3: Alphanumeric 2x16 LCD connection schematic

page

16.0. On-Board 2x16 LCD with Serial Communication

On-board 2x16 display is connected to the microcontroller through a port expander. In order to use this display, it is necessary to set switches (1-6) on the DIP switch SW10 to the ON position, thus connecting the on-board LCD to port expander's port 1. The following DIP switches SW6, SW7 and SW9 enable the port expander to use serial communication. Potentiometer P5 is used for display contrast adjustment.
Unlike common LCD, the on-board LCD has no backlight and receives data to be displayed through the port expander which employs SPI communication for the purpose of communicating with the microcontroller. Such display also shows digits in two lines each containing up to 16 characters of 7x5 pixels.

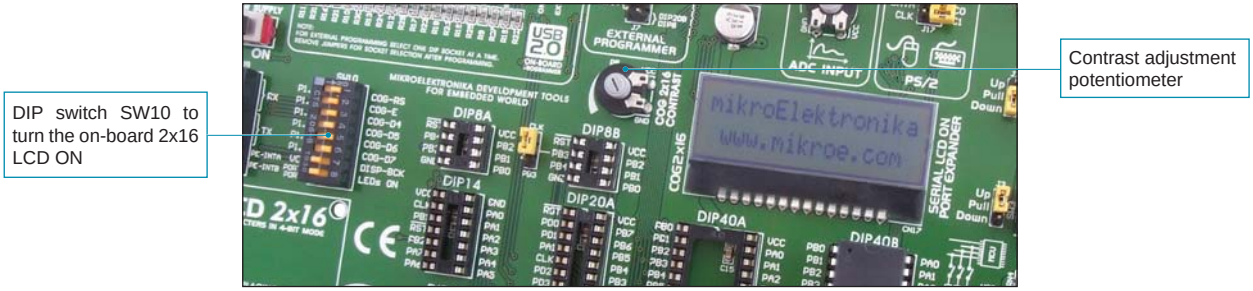


Figure 16-1: On-board 2x16 LCD

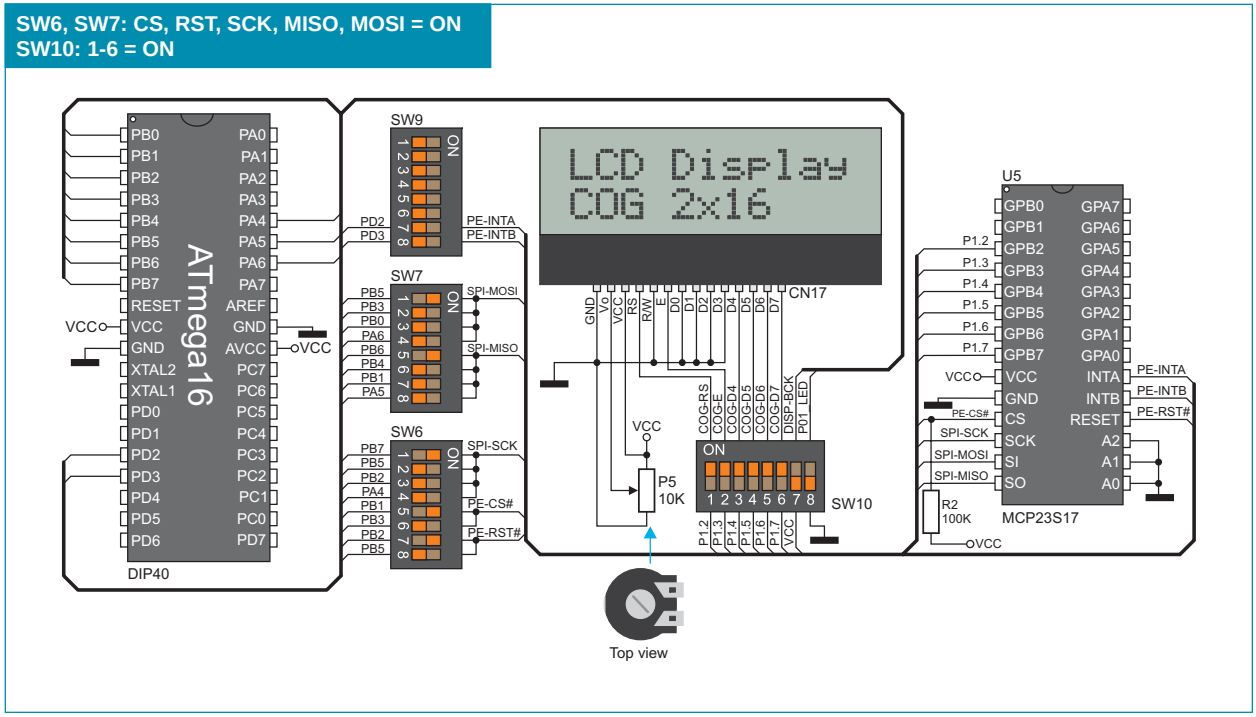


Figure 16-2: On-board 2x16 LCD connection schematic

17.0. 128x64 Graphic LCD

128x64 graphic LCD (128x64 GLCD) provides an advanced method for displaying graphic messages. It is connected to the microcontroller through PORTC and PORTD. GLCD has the screen resolution of 128x64 pixels which allows you to display diagrams, tables and other graphic contents. Since the PORTD port is also used by 2x16 alphanumeric LCD, you cannot use both displays simultaneously. Potentiometer P6 is used for the GLCD contrast adjustment. Switch 7 on the DIP switch SW10 is used for turning on/off display backlight.

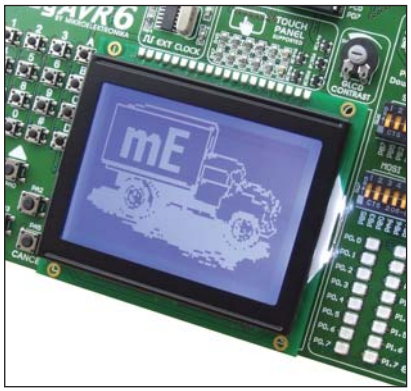


Figure 17-1: GLCD

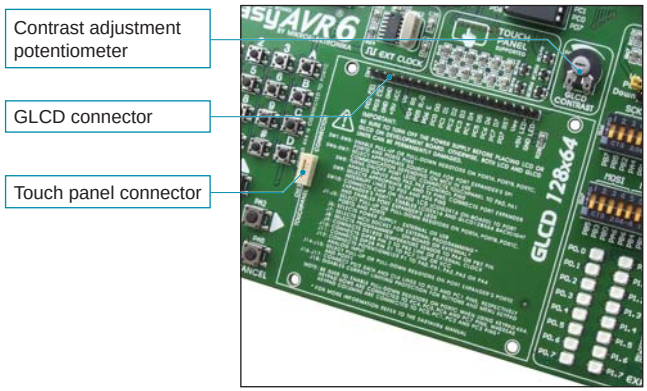


Figure 17-2: GLCD connector

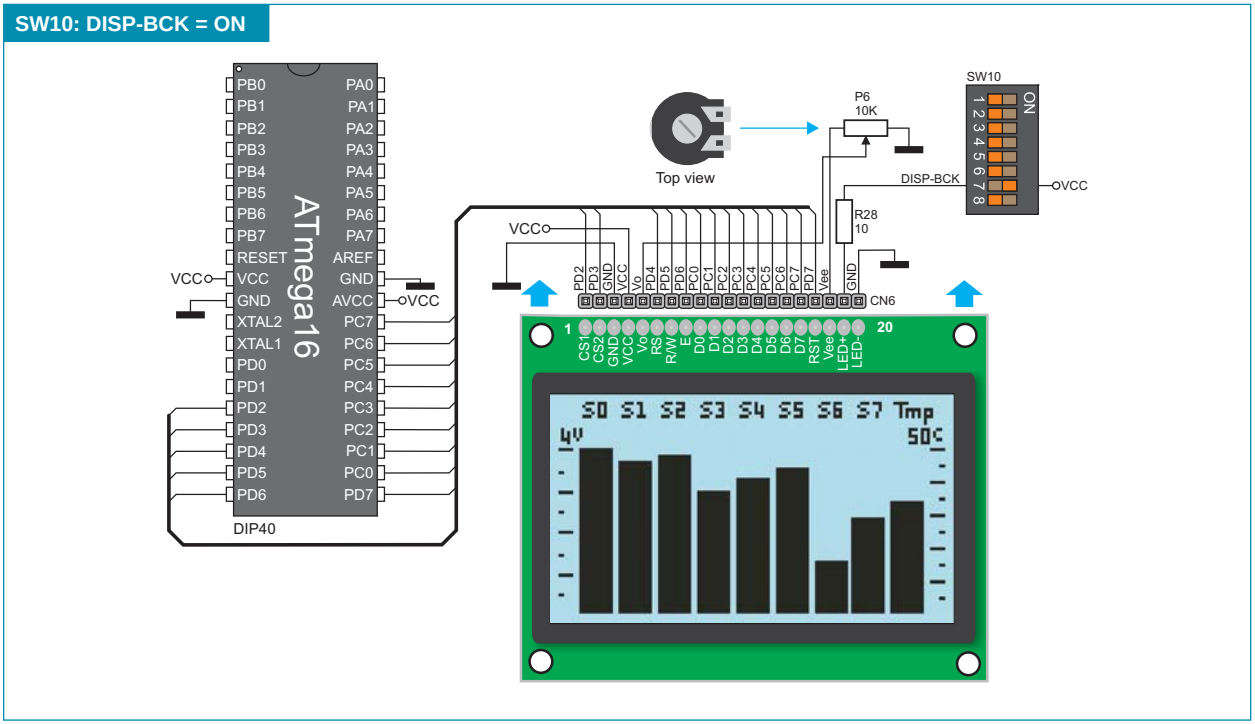


Figure 17-3: GLCD connection schematic

page

18.0. Touch Panel

The touch panel is a thin, self-adhesive, transparent panel sensitive to touch. It is placed over a GLCD. 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 SW8 are used for connecting touch panel to the microcontroller.

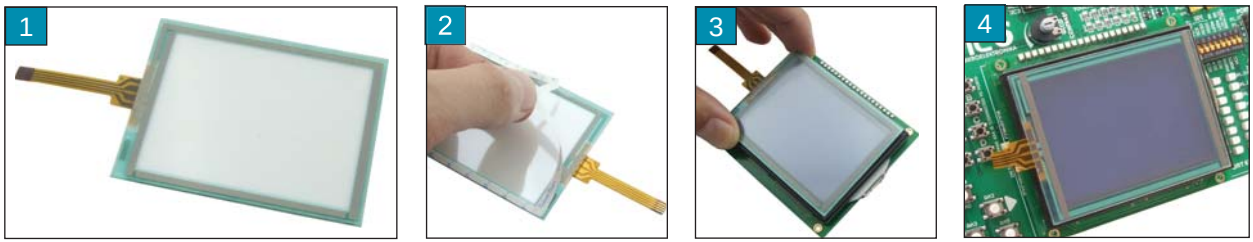


Figure 18-1: Touch panel

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

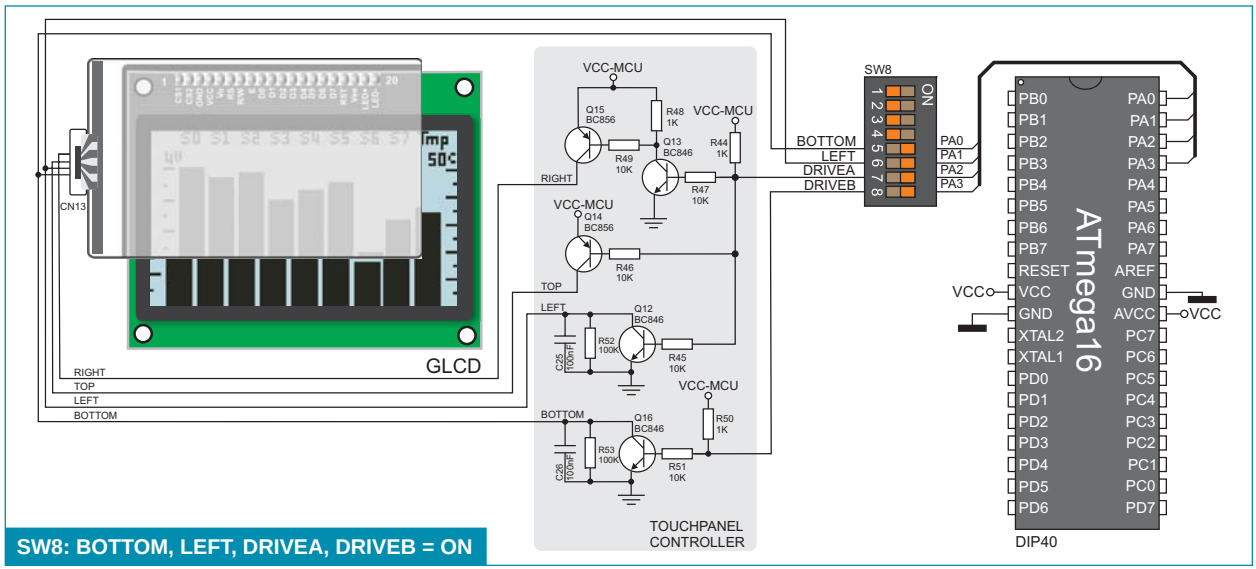


Figure 18-2: Touch panel connection schematic

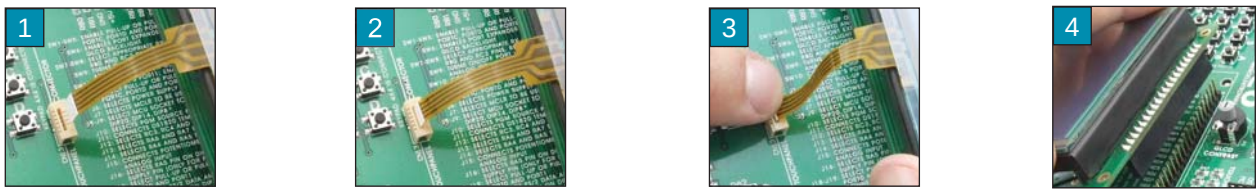


Figure 18-3: Placing touch panel

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 into the appropriate connector as shown in Figure 4.

NOTE: LEDs and pull-up/pull-down resistors on 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 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.

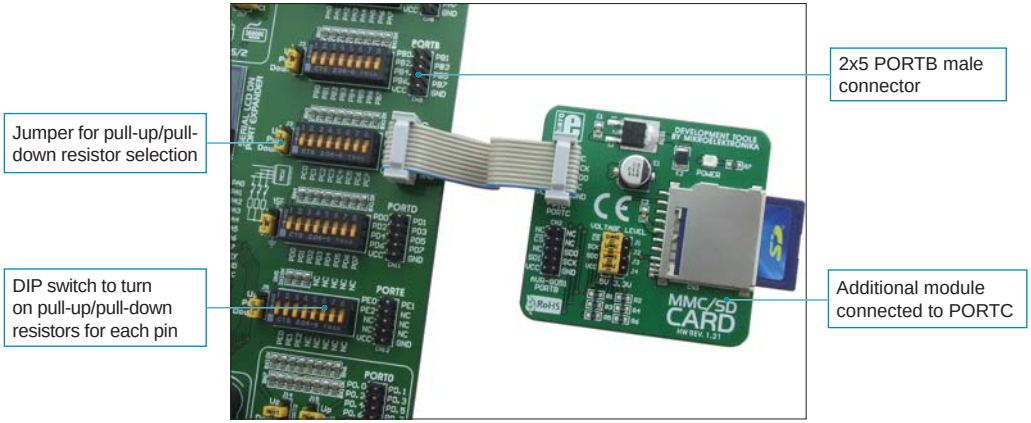


Figure 19-1: I/O ports

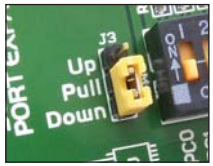


Figure 19-2: J3 in the pull-down position

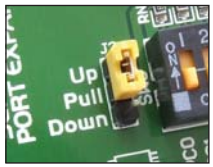


Figure 19-3: J3 in the pull-up position

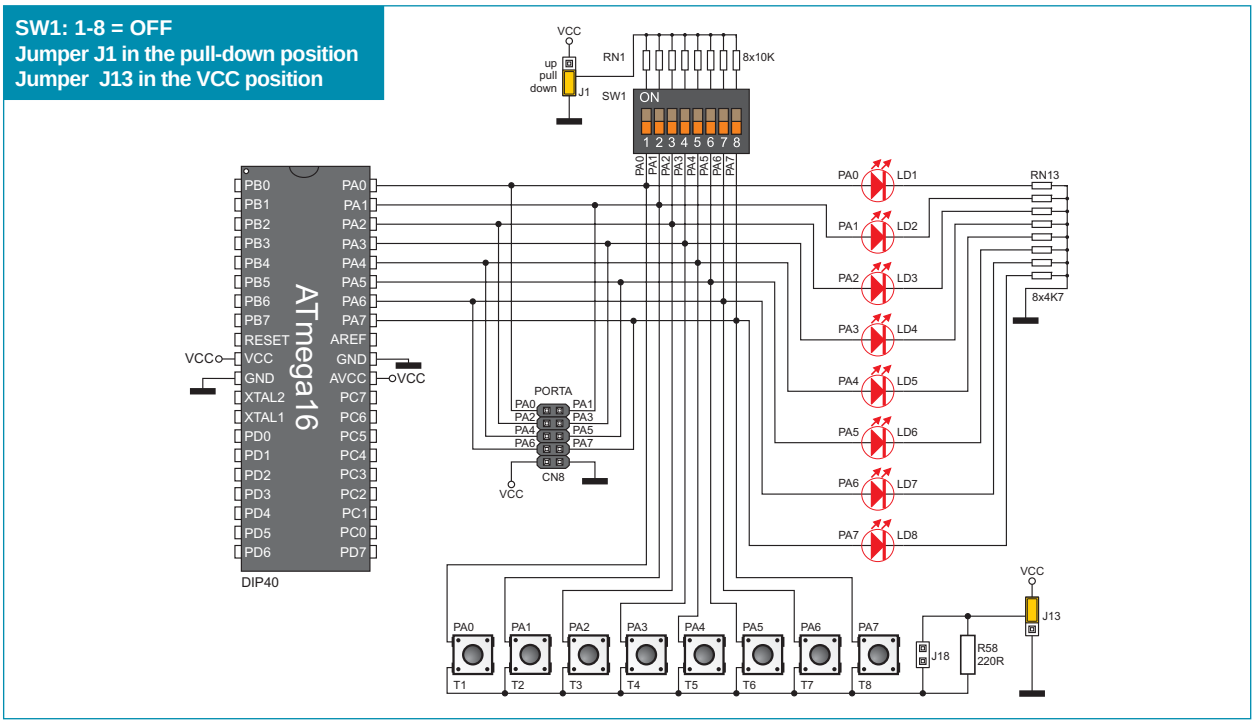


Figure 19-4: PORTA connection schematic

page

Pull-up/pull-down resistors enable you to set the logic level on all microcontroller input pins when they are in idle state. Such level depends on the position of the pull-up/pull-down jumper. The PA0 pin with the relevant DIP switch SW1, jumper J1 and PA0 push button with jumper J13 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.

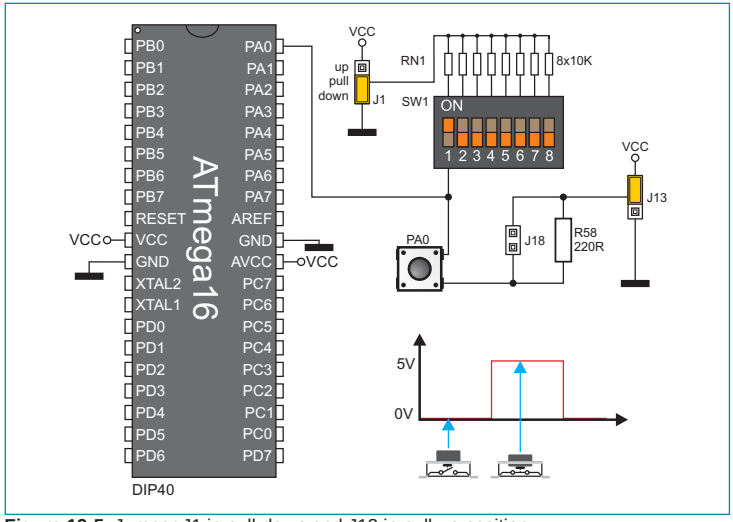


Figure 19-5: Jumper J1 in pull-down and J13 in pull-up position

In order to enable the port PORTA pins to be connected to the pull-down resistors, first it is necessary to set jumper J1 in the *Down* position. This enables any port PORTA pin to be provided with a logic zero (0V) in idle state over jumper J1 and 8x10K resistor network. To provide the PA0 pin with such signal, it is necessary to set switch PA0 on the DIP switch SW1 in the ON position.
As a result, every time you press the PA0 push button, a logic one (1) will appear on the PA0 pin, provided that jumper J13 is set in the VCC position.

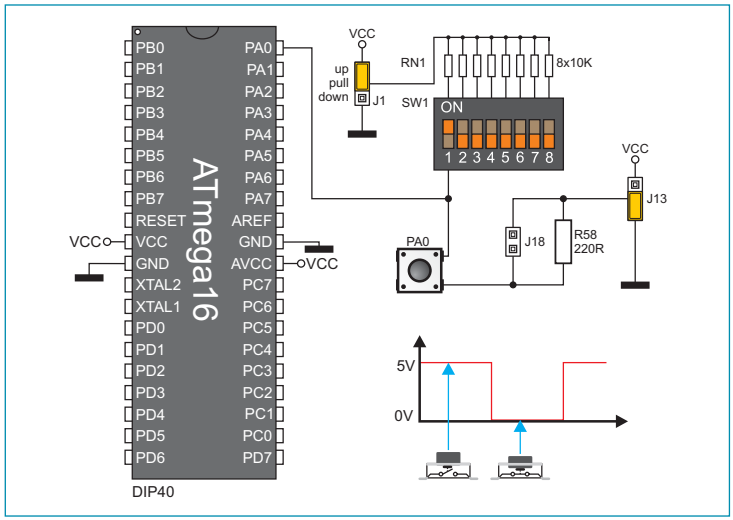


Figure 19-6: Jumper J1 in pull-up and J13 in pull-down position

In order to enable port PORTA pins to be connected to pull-up resistors and the port input pins to be acivated with logic zero (0), it is necessary to set jumper J1 in the *Up* position (5V) and jumper J13 in the GND position (0V). Also, the PA0 pin on the DIP switch SW1 should be set in the ON position so as to enable all port PORTA input pins, over the 10k resistor, to be provided with logic one (5V) in their idle state. The PA0 switch supplies the PA0 pin with this voltage over the 10k resistor.
As a result, every time you press the PA0 push button, a logic zero (0) will appear on the PA0 pin.

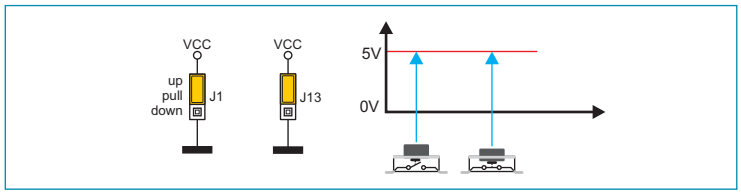


Figure 19-7: Jumpers J1 and J13 in the same position

In case that jumpers J1 and J13 have the same logic state, pressure on any button will not cause input pins to change their logic state.

20.0. Port Expander (Additional Input/Output Ports)

The SPI communication lines and MCP23S17 circuit provide the *EasyAVR6* development system with a means of increasing the number of available I/O ports by two. If the port expander communicates to the microcontroller over the DIP switches SW6 and SW7, then the microcontroller pins used for SPI communication cannot be used as I/O pins. Switches INTA and INTB on the DIP switch SW9 enable interrupt used by MCP23S17.

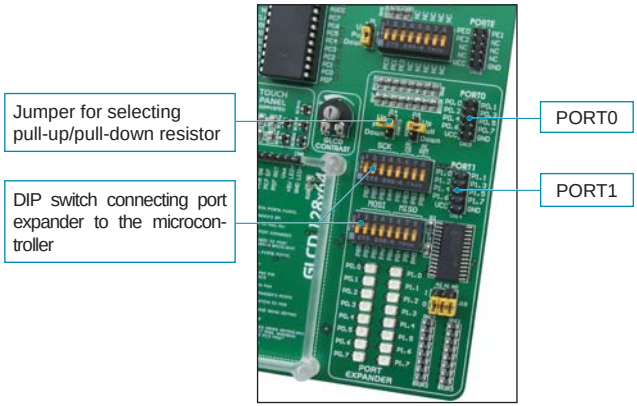


Figure 20-1: Port expander

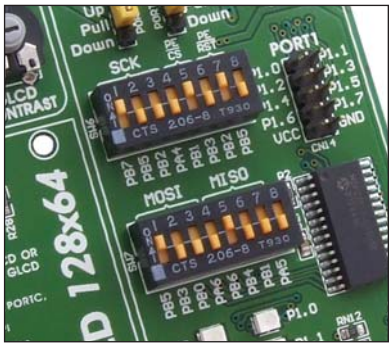


Figure 20-2: DIP switches SW6 and SW7 when port expander is enabled

The microcontroller communicates to the port expander (MCP23S17 circuit) using serial communicaion (SPI). The advantage of such communication is that only four lines are used for transmitting and receiving data simultaneously:

MOSI - Master Output, Slave Input (microcontroller output, MCP23S17 input)
MISO - Master Input, Slave Output (microcontroller input, MCP23S17 output)
SCK - Serial Clock (microcontroller clock signal)
CS - Chip Select (enables data transfer)

Data transfer is performed in both directions simultaneously by means of MOSI and MISO lines. The MOSI line is used for transferring data from the microcontroller to the port expander, whereas the MISO line transfers data from the port expander to the microcontroller. The microcontroller initializes data transfer when the CS pin is driven low (0V). It causes the microcontroller to send clock signal (SCK) and therefore starts data exchange.

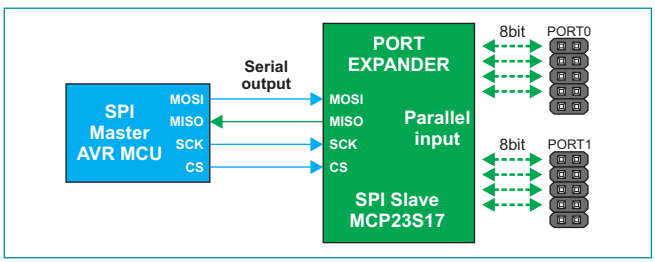


Figure 20-3: SPI communication block diagram

Jumpers J19 are used to determine the port expander's hardware address. They are also used to provide the port expander's pins A2, A1 and A0 with a logic 1 or a logic 0. Jumpers J19 are set in the Down position (logic 0) by default.

The principle of operation of the port expander's ports 0 and 1 is almost identical to the operation of other ports on the development system. The only difference here is that port signals are received in parallel format. The MCP23S17 converts such signals into serial format and sends them to the microcontroller. The result is a reduced number of lines used for sending signals from ports 0 and 1 to the microcontroller.

page

SW6: CS#=PB1, RST=PB2, SCK = PB7
SW7: PB6 =MISO, PB5=MOSI
Jumpers J14 and J15 in the pull-up position

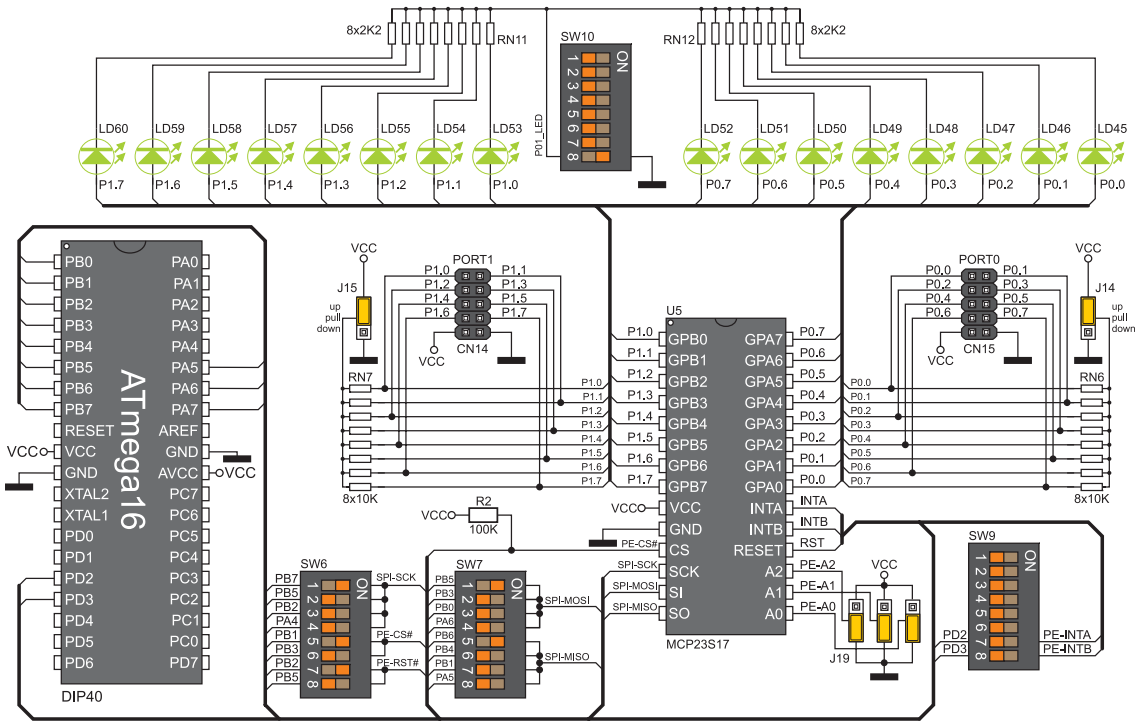


Figure 20-4: Port expander connection schematic

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.

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™, 2010, All Rights Reserved.

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

Mouser Electronics

Authorized Distributor

Click to View Pricing, Inventory, Delivery & Lifecycle Information:

mikroElektronika:

MIKROE-415