Creating the first project in

mikroC PRO for 8051

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

Table of contents

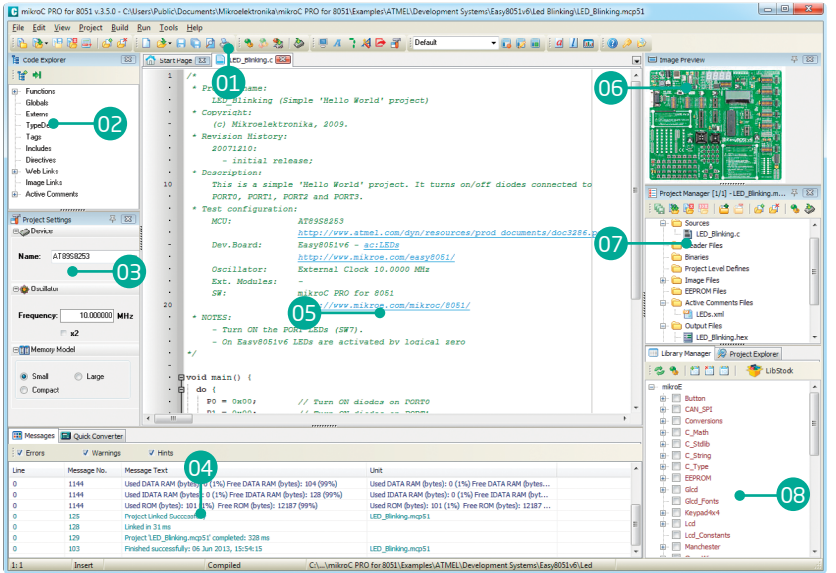
1. Introduction to mikroC PRO for 8051

mikroC PRO for 8051 organizes applications into projects consisting of a single project file (file with the **.mcp51** extension) and one or more source files (files with the .c extension). The mikroC PRO for 8051 compiler allows you to manage several projects at a time. Source files can be compiled only if they are part of the project.

A project file contains:

- Project name and optional description
- Target device in use
- Device clock
- List of the project source files
- Binary files (*.mcl)
- Other files.

In this reference guide, we will create a new project, write code, compile it and test the results. The purpose of this project is to make microcontroller PORT0 LEDs blink, which will be easy to test.



01 Main toolbar
02 Code explorer
03 Project settings
04 Messages
05 Code editor
06 Image preview
07 Project manager
08 Library manager

2. Hardware connection

Let's make a simple "Hello world" example for the selected microcontroller. First thing embedded programmers usually write is a simple **LED blinking** program. So, let's do that in a few simple lines of C code.

LED blinking is just turning ON and OFF LEDs that are connected to desired PORT pins. In order to see the example in action, it is necessary to connect the target microcontroller according to schematics shown on **Figure 2-1**. In the project we are about to write, we will use only **PORT0**, so you should connect the LEDs to PORT0 only.

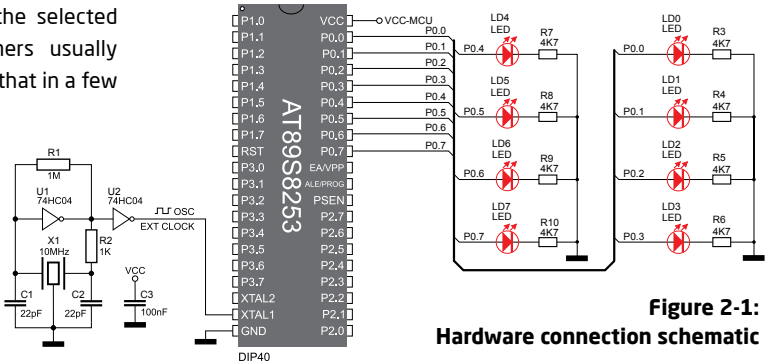


Figure 2-1: Hardware connection schematic

Prior to creating a new project, it is necessary to do the following:

Step 1: Install the compiler

Install the mikroC PRO for 8051 compiler from the **Product DVD** or download it from the MikroElektronika website:

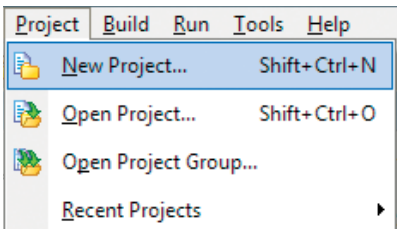
www.mikroe.com/mikroc/8051/

Step 2: Start up the compiler

Double click on the compiler icon in the Start menu, or on your desktop to Start up the mikroC PRO for 8051 compiler. The mikroC PRO for 8051 IDE (Integrated Development Environment) will appear on the screen. Now you are ready to start creating a new project.

3. Creating a new project

The process of creating a new project is very simple. Select the **New Project** option from the **Project menu** as shown below. The **New Project Wizard** window appears. It can also be opened by clicking the **New Project icon** from the **Project toolbar**.



The **New Project Wizard** (**Figure 3-1**) will guide you through the process of creating a new project. The introductory window of this application contains a list of actions to be performed when creating a new project.

01 Click **Next**.

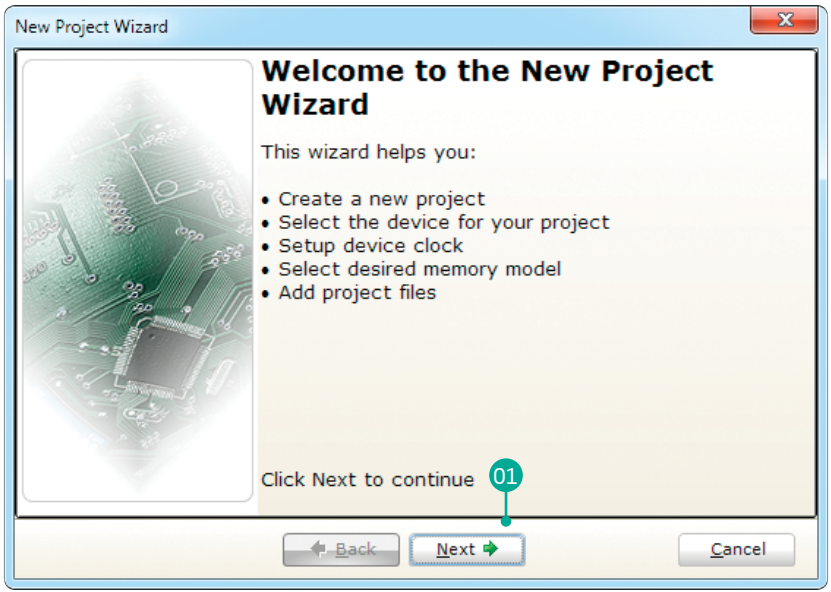


Figure 3-1: Introductory window of the New Project Wizard

Step 1 - Project settings

First thing we have to do is to specify the general project information. This is done by selecting the target microcontroller, it's operating clock frequency, and of course - naming our project. This is an important step, because the compiler will adjust the internal settings based on this information. Default configuration is already suggested to us at the begining. We will not change the microcontroller, and we will leave the default **AT89S8253** as the choice for this project.

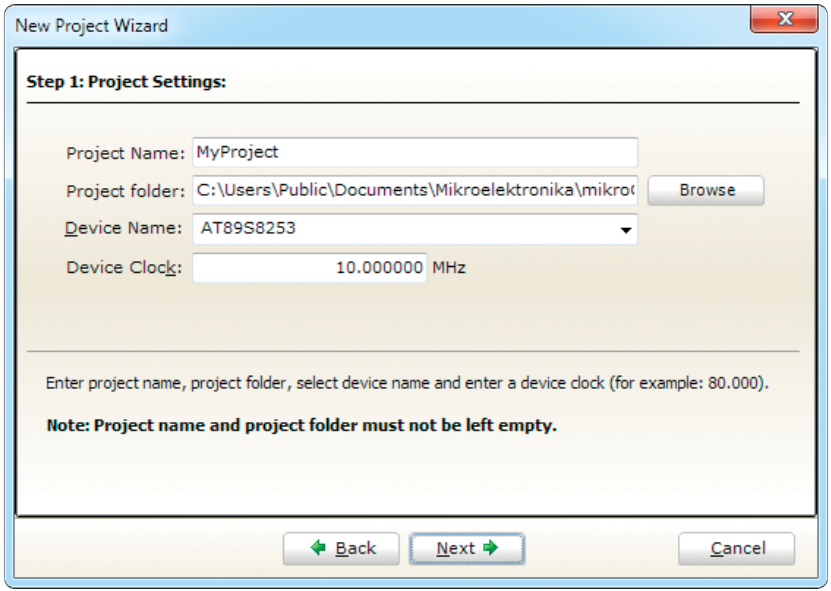


Figure 3-2: You can specify project name, path, device and clock in the first step

Step 1 - Project settings

If you do not want to use the suggested path for storing your new project, you can **change the destination folder**. In order to do that, follow a simple procedure:

01 Click the **Browse** button in the Project Settings window to open the **Browse for Folder** dialog.

02 Select the desired folder to be the destination path for storing your new project files.

03 Click **OK** to confirm your selection and apply the new path.

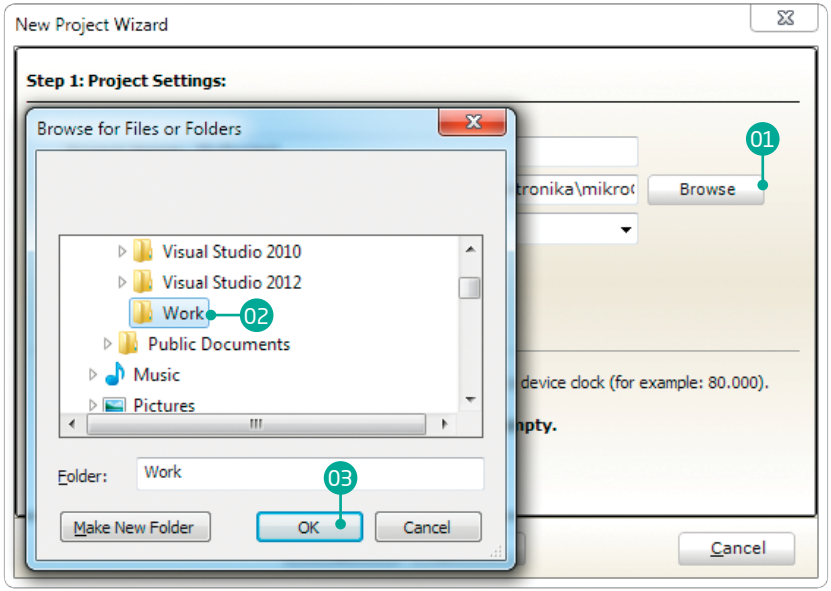


Figure 3-3: Change the destination folder using Browse For Folder dialog

Step 1 - Project settings

Once we have selected the destination project folder, let's do the rest of the project settings:

01 Enter the name of your project. Since we are going to blink some LEDs, it's appropriate to call the project **"LedBlinking"**

02 For this demonstration, we will use the default external crystal **10MHz clock**. Clock speed depends on your target hardware. However you configure your hardware, make sure to specify the exact clock (**Fosc**) that the microcontroller is operating at.

03 Click the **OK** button to proceed.

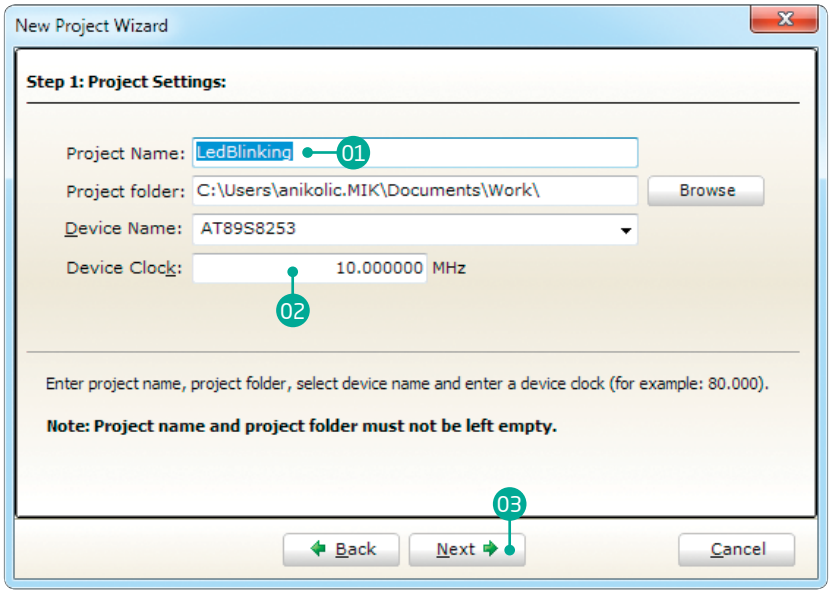


Figure 3-4: Enter project name and change device clock speed if necessary

Step 2 - Memory model

The memory model determines which default memory type will be used for function arguments, automatic variables, and declarations that include no explicit memory type.

Since our project is very simple, we will choose the **Small memory model**. In this model, all variables, by default, reside in the internal data memory of the 8051 system. Variable access in this memory model is very efficient.

01 Select **Small** memory model.

02 Click **Next**.

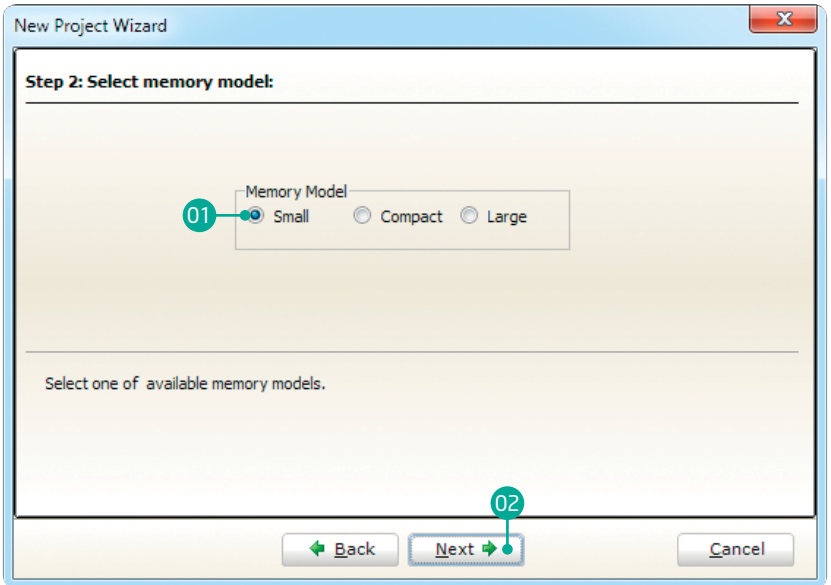


Figure 3-5: Choose the memory model. "Small" is the best choice for our project.

Step 3 - Add files

This step allows you to include additional files that you need in your project: some headers or source files that you already wrote, and that you might need in further development. Since we are building a simple application, we won't be adding any files at this moment.

01 Click **Next**.

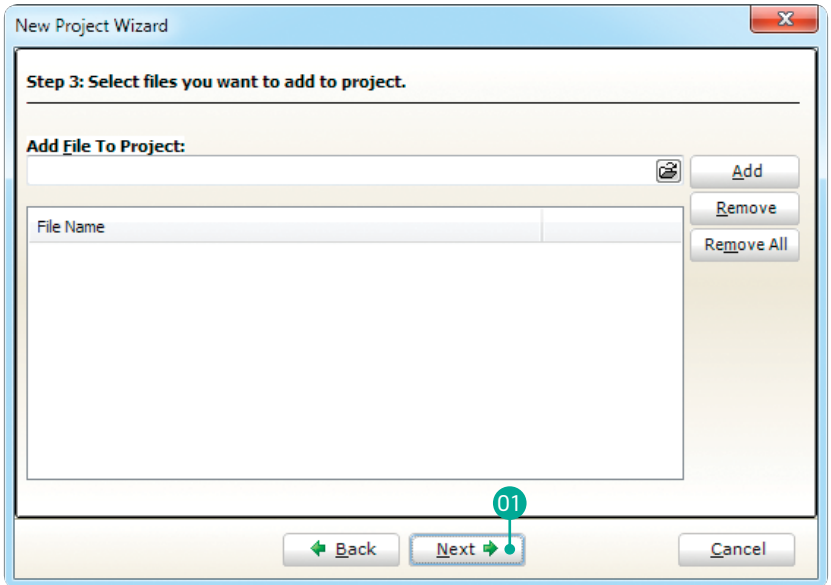


Figure 3-6: Add existing headers, sources or other files if necessary

Step 4 - Include libraries

This step allows you to quickly set whether you want to include all libraries in your project, or not. Even if all libraries are included, they will not consume any memory unless they are explicitly used from within your code. The main advantage of including all libraries is that you will have over **250 functions** available for use in your code right away, and visible from **Code Assistant [CTRL+Space]**. We will leave this in default configuration:

01 Make sure to leave **"Include All"** selected.

02 Click **Next.**

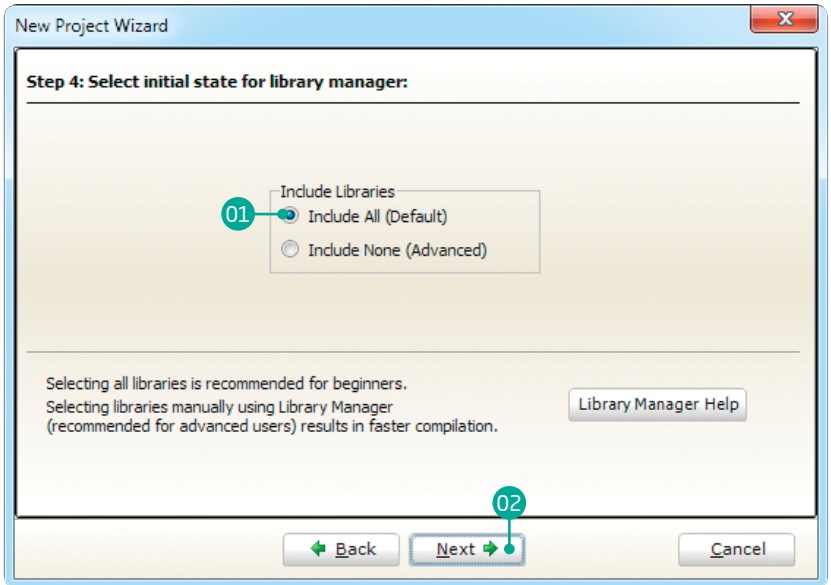


Figure 3-7: Include all libraries in the project, which is a default configuration.

Step 5 - Finishing

After all configuration is done, click “Finish” to complete the New Project wizard. You can still go back and change any settings if necessary.

01 Click **Finish**.

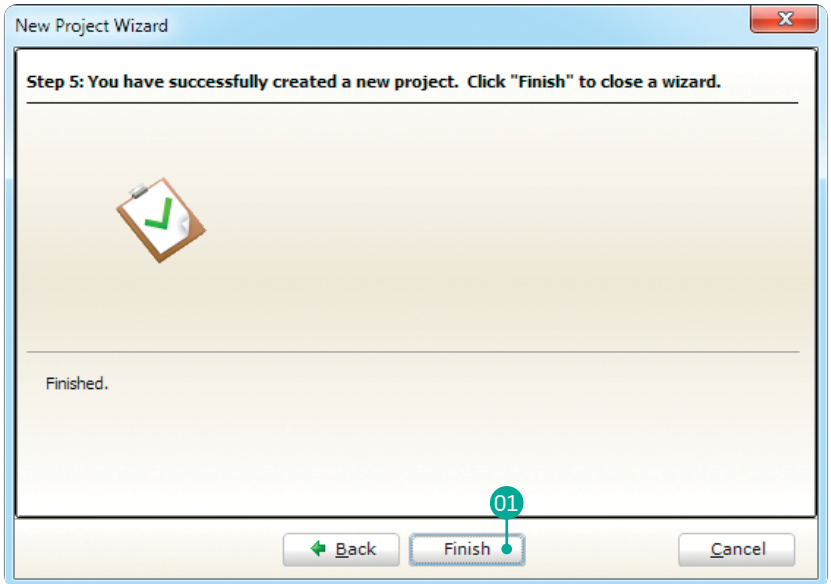


Figure 3-8: Finish the New project wizard, or go back and make some changes.

Blank new project created

New project is finally created. A new source file called **"LedBlinking.c"** is created and it contains the `void main()` function, which will hold the program. You may notice that the project is configured according to the settings done in the **New Project Wizard**.

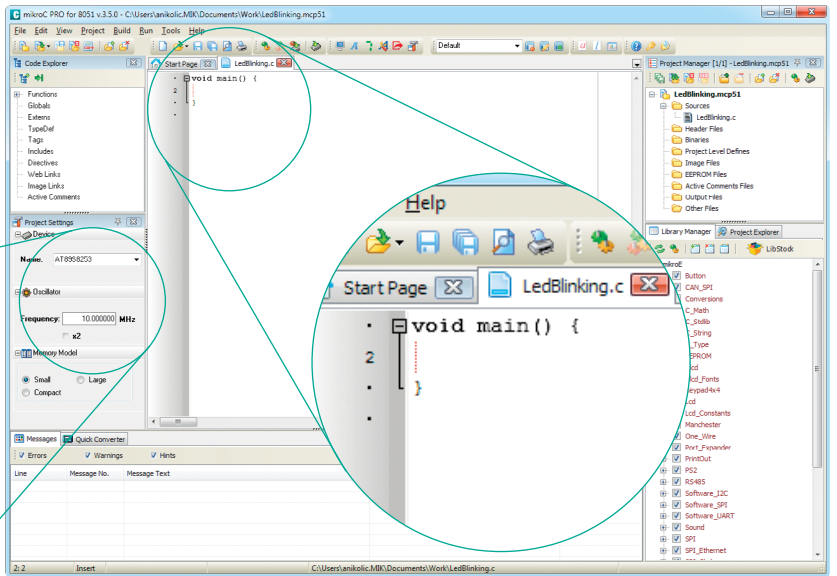


Figure 3-9: New blank project is created with your configuration

4. Code example

It's time to do some coding. First thing we need to do is to put PORT0 LEDs into initial state. For example, let's fill it with logic zeros on every pin:

```
// Turn OFF all LEDs on PORT0
P0 = 0;
```

Finally, in a **while()** loop we will toggle the PORT0 value, and put a 1000 ms delay, so the blinking is not too fast.

```
while(1) {
  // Toggle LEDs on PORT0
  P0 = ~P0;

  // Delay 1000 ms
  Delay_ms(1000);
}
```

LedBlinking.c - source code

```
1  void main() {
2    // Turn OFF LEDs on PORT0
3    P0 = 0;
4
5    while(1) {
6      // Toggle LEDs on PORT0
7      P0 = ~P0;
8
9      // Delay 1000 ms
10     Delay_ms(1000);
11   }
12 }
13
14
15
```

Figure 4-1: Complete source code of the PORT0 LED blinking

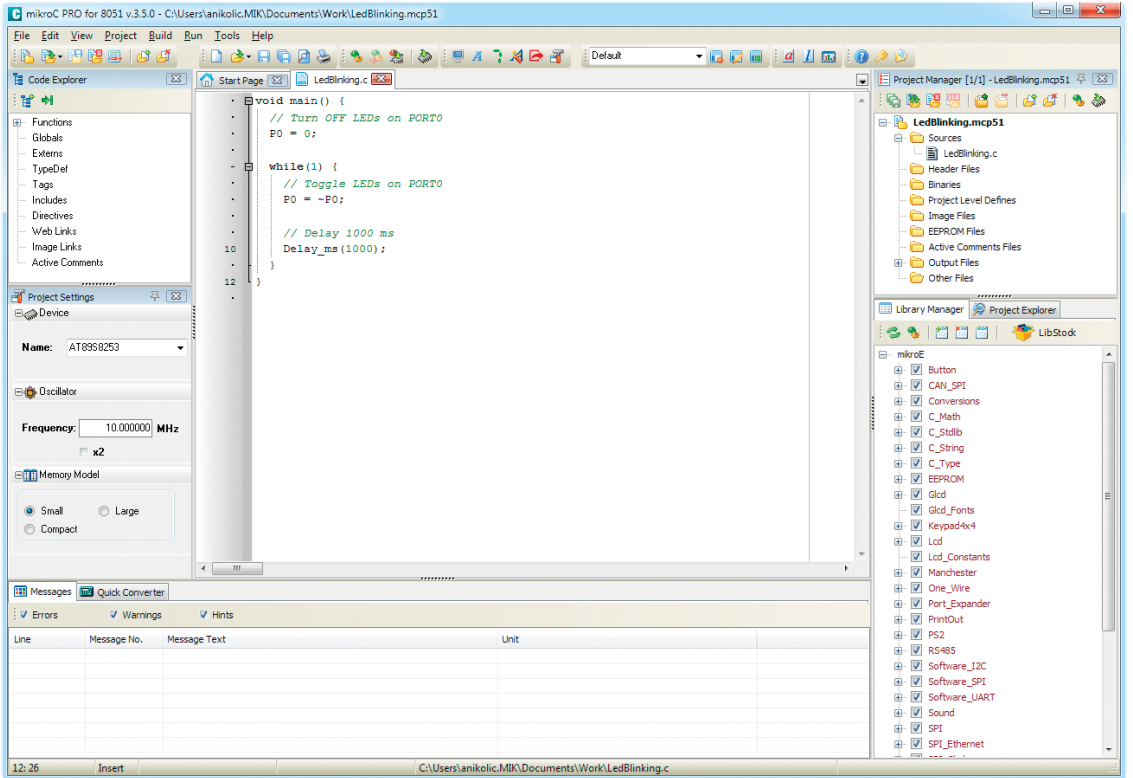
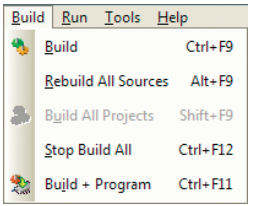


Figure 4-2: This is how the code looks written in the compiler code editor window

5. Building the source

When we are done writing our first LedBlinking code, we can now build the project and create a **.HEX** file which can be loaded into our target microcontroller, so we can test the program on real hardware. "Building" includes compilation, linking and optimization which are done automatically. Build your code by clicking on the  icon in the main toolbar, or simply go to **Build menu** and click **Build [CTRL+F9]**. Message window will report the details of the building process (**Figure 5-1**). Compiler automatically creates necessary output files. **LedBlinking.hex** (**Figure 5-2**) is among them.



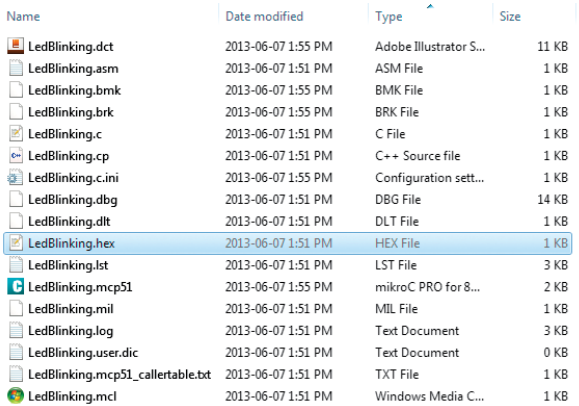


Figure 5-2: Listing of project files after building is done

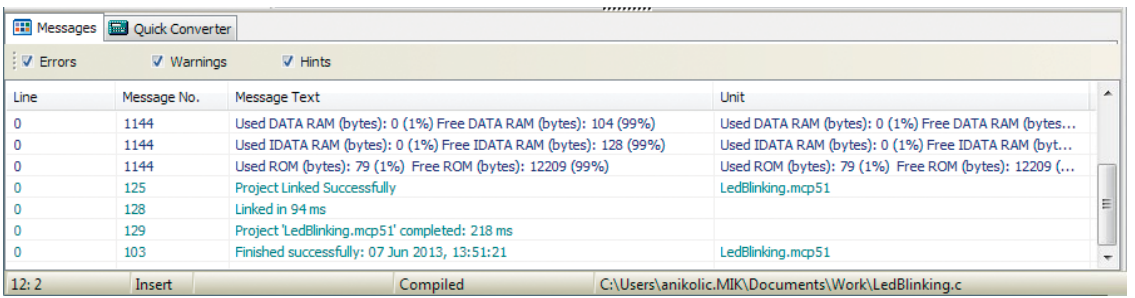


Figure 5-1: After the successful compilation and linking, the message window should look something like this

6. What's next?

More examples

mikroC PRO for 8051 comes with over **90 examples** which demonstrate a variety of features. They represent the best starting point when developing a new project. You will find projects written for MikroElektronika development boards, additional boards, internal MCU modules and other examples. This gives you a head start in development, and you don't have to do it all from scratch. In most cases, you can combine different simple projects to create a more complex one. For example, if you want to build a date, time and temperature semaphore on a 7-segment display, you can combine RTC and temperature sensor examples with the Seven Segment Display example and do the job in much less time. All projects are delivered with working .HEX files, so you don't have to buy a compiler license in order to test them. You can load them into your development board right away without the need for building them.

Community

If you want to find answers to your questions on many interesting topics we invite you to visit our forum at **www.mikroe.com/forum** and browse through more than 200 thousand posts. You are likely to find just the right information for you.

On the other hand, if you want to download more free projects and libraries, or share your own code, please visit the **Libstock** website **www.libstock.com**. With user profiles, you can get to know other programmers, and subscribe to receive notifications on their code.

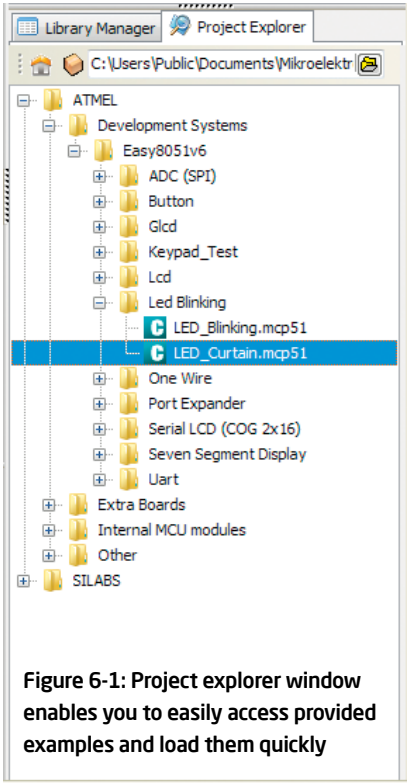


Figure 6-1: Project explorer window enables you to easily access provided examples and load them quickly

DISCLAIMER

HIGH RISK ACTIVITIES

TRADEMARKS

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

Designed by
MikroElektronika Ltd.
www.mikroe.com

Creating the first project
in mikroC PRO for 8051 manual
ver. 1.00a

0 100000 026649