Atmega8

Development Board (v2.0)

User's Manual

starklabz

Projects | Robotics | Embedded systems

About User Guide

Atmega8 DevBoard v2.0 is designed and developed by Stark-labz for students and developers having interest in electronics, embedded system, robotics and industrial-automation. The general design of this board make users easy to learn about AVR microcontrollers. The board also helps in experimenting various real time projects and applications.

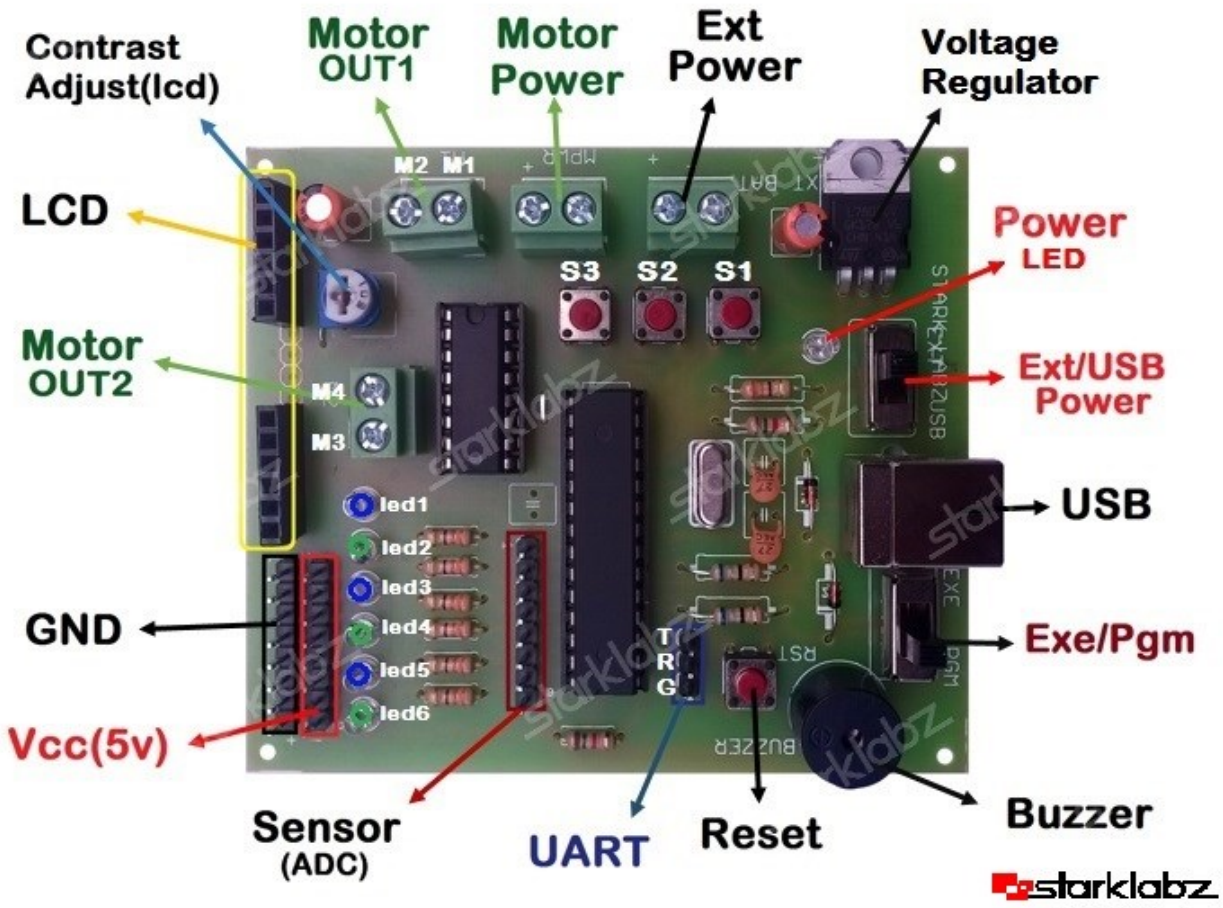
INDEX

1. Product Description:

1.1. Atmega8 Development Board v2.0:

- Includes Atmel's ATmega8 Microcontroller with 8kb flash memory working at 16MIPS.
- On-board LCD interface (it can also be used for any other general purpose application).
- On-board Motor Driver for connecting 2 DC motors or 1 Stepper motor.
- On-board regulated power supply.
- On-board Buzzer.
- 12 MHz external crystal.
- Exposed 7 channel I/O pins for ADC
- Exposed 8 channel I/O pins for servo, sensors and other peripherals with power rails.
- Four tact switches for external input and reset.
- Six test LEDs for status and debugging purpose.
- Supply indicator LED.
- Dual power supply through DC source (6V to 16V) or USB powered.
- On board USB programmer.

2. Parts Identification



2.1 Microcontroller

It is a chip which stores our programs executes them and takes necessary action. The chip used here is Atmel popular AVR micro controller (Atmega8).

2.2 Voltage Regulator

It is a three terminal 5V voltage regulator IC used to provide a constant voltage supply of 5V to the micro controller and other peripherals (i.e. sensors etc.) attached in the main board.

2.3 Motor Driver

This is basically a motor driver IC (l293d) which takes input from microcontroller and is able to drive the DC and stepper motors by using separate power supply.

The motor drivers are used to run the DC motors or stepper motors that may be connected to the board according to the data from the microcontroller. The motor driver's link with micro controller is shown below.

EN1 (Enable 1) – PortB1
M1 - PortB0
M2 - PortB3
EN2 (Enable 2) – PortB2
M3 - PortB4
M4 - PortB5

2.4 Switch

Three tact switches along with a Reset switch are present on the board in order to provide an external input to the board. The switches are connected in the following manner:

S1 – port D5
S2 – port D6
S4 – port D7

RST (Reset switch):

The Reset switch is basically used to reset a running program right to the beginning it is same as the reset switch of a PC.

Ext/Usb Power:

It is basically a toggle switch used to provide power supply to the main board. The power can be supplied either by a battery power supply (through Ext Power) or can be USB powered

Prog/Exe Switch:

It is also a toggle switch for programming the microcontroller using on board USB programmer. For programming mode it should be ON then RESET button should be pressed. For normal operation it should be off.

2.5 LED:

Active high:

LED1 (Blue) – PORTC0
LED2 (Green) – PORTC1
LED3 (Blue) – PORTC2
LED4 (Green) – PORTC3
LED5 (Blue) – PORTC4
LED6 (Green) – PORTC5

RED LED - Power ON indicator

2.6 Buzzer:

It can be easily used to get an audible feedback from the controller. It is connected in an active high mode, i.e., the buzzer beeps when the data given to it is 1. The Buzzer is connected to the pin PORTD0 of the microcontroller

3. AVR Microcontrollers

3.1 Description

The AVR is a Modified Harvard architecture 8-bit RISC single chip microcontroller which was developed by Atmel in 1996. The AVR was one of the first microcontroller families to use on-chip flash memory for program storage, as opposed to One-Time Programmable ROM, EPROM, or EEPROM used by other microcontrollers at the time. Atmel's low power, high performance AVR microcontrollers handle demanding 8 and 16-bit applications. With a single cycle instruction RISC CPU, innovative Pico Power® technology, and a rich feature set, the AVR architecture ensures fast code execution combined with the lowest possible power consumption. Whether you program in C or assembly, the tuned AVR instructions decrease program size and development time. The well-defined I/O structure limits the need for external components and reduces development cost. A variety of internal oscillators, timers, UARTs, SPIs, Pulse Width Modulation, pull-up resistors, ADCs, Analog Comparators and Watch-Dog Timers are some of the features available for creative engineers. The AVR microcontrollers are divided into 4 families tiny AVR, mega AVR, XMEGA and Application specific AVR. Among these 4 families of AVR here we are going to use a microcontroller of mega AVR family “ATmega8”

3.2 Software Installation

The following tutorial covers the steps needed to program the development board using the in-built Usb programmer. The following software are included in the Startup disk or can be downloaded from the internet free of cost.

- **AVR Studio** – to write, simulate, debug and emulate your code
- **AVRDude** – burns your hex code into the MCU

Installation of these software is pretty straightforward. You should not face any trouble in it. During the installation of AVR Studio 5, it might need to install some other stuffs before the actual installation begins.

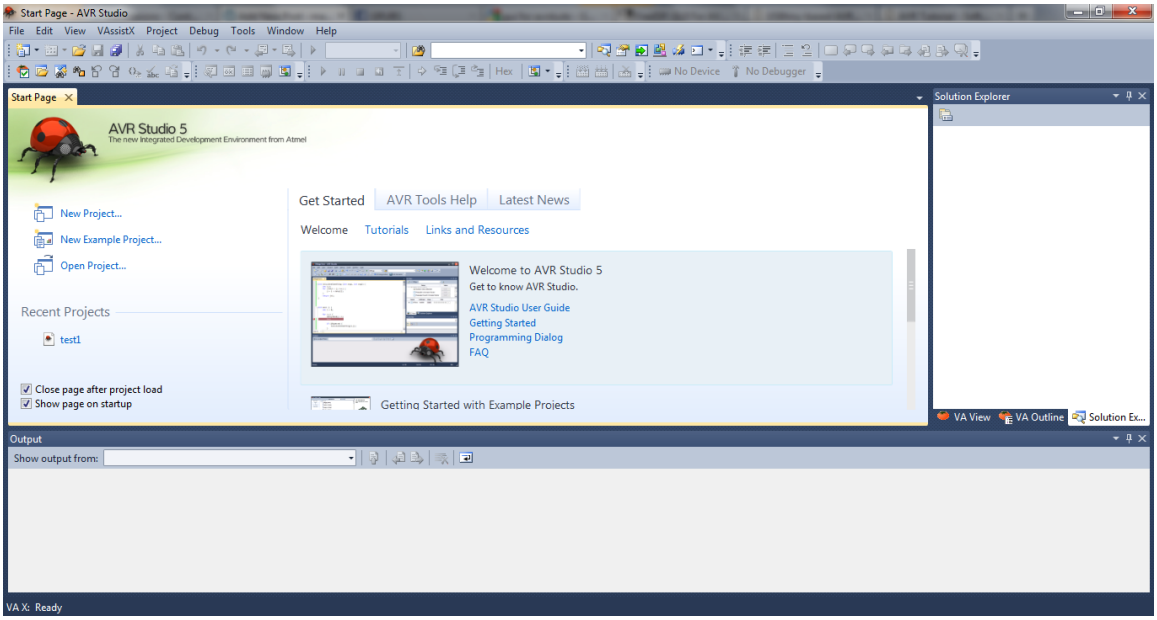
Before starting the actual coding you need to setup the programmer. The following steps guide you to configure AVRdude to your AVR Studio.

First you need to copy the "avrdude.conf" and "avrdude.exe" files into the C:\ drive
(Dont put them in folders,just paste inside C:\ or else change Command accordingly)

Open Avr Studio...
Goto: Tools -> External Tools
Click: Add -> Insert the following in respective columns
Title: Program
Command: C:\avrdude.exe
Arguments: -c usbasp -p atmega8 -U flash:w:"$(ItemDir)Debug\$(ItemFileName).hex":a -q
Select: "Use Output window"
Click: Apply -> OK
Now a new Tool named 'Program' appears inside Tools menu.
Use this to Burn your pgm directly into DevBoard

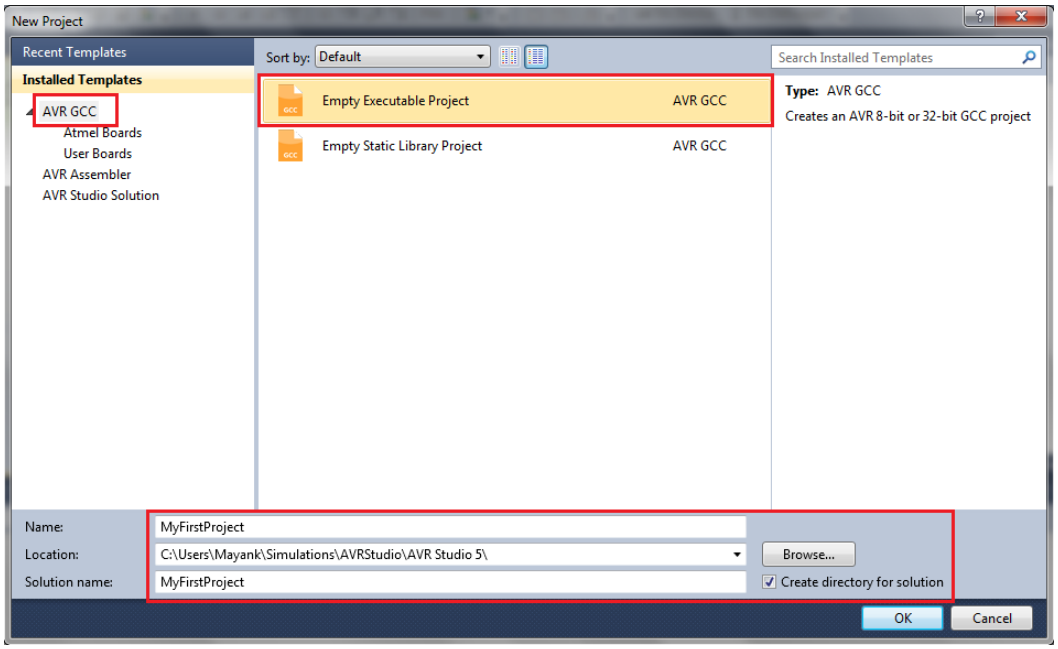
3.3 Creating the First Project

- After installation, open AVR Studio 5 from **Start → All Programs → Atmel AVR Tools → AVR Studio 5.0**
- After opening, you will see a Start Page like this. Click on "**New Project…**".



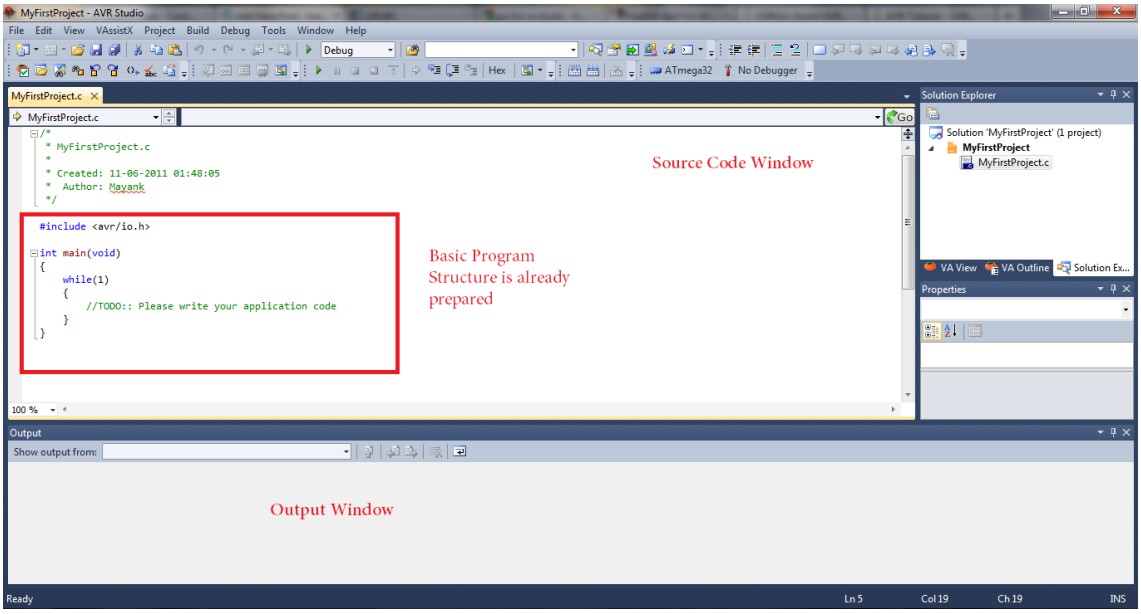
AVR Studio 5 Start Page

- Then, you will see the following dialog box. Choose **AVR GCC** from the 'Installed Templates' pane, and then choose **Empty Executable Project**. Now, you can give any name to it, say MyFirstProject and choose an appropriate location in your hard drive. Check **Create directory for solution**. Click on **OK**.



AVR Studio New Project

- Now, you will see the **Device Selection** dialog box. As you can see, AVR Studio supports *all* the AVR MCUs! The list is huge! Choose your device from this list. We choose **ATmega8**. Click **OK**.
- Now, you will see the following screen. Note that the basic program structure is already prepared for you. You simply need to initialize your variables and place your code inside the while(1) loop.



AVR Studio New Project Start Screen

Stark-Labz
www.starklabz.com
info@starklabz.com