Click to verify

How to program atmega328p with arduino

By following these steps and utilizing the resources linked throughout, you can expand your understanding and effectively program microcontrollers for your embedded systems projects.Let us know how your projects go, and don’t forget to check out our other programming tutorials for more inspiration! Over the past few tutorials, we have mentioned several scenarios where using any of the Arduino board in a project may be an overkill due to the cost, size, and more technical reasons such as high power consumption. Paste the following boards link that will help to download and update the boards field of Arduino IDE. The first problem that may become obvious though is that you can't connect the ATmega328P directly to your computer to upload programs to it. You should now see "ATmega328 on a breadboard (8 MHz internal clock)" as an option in the boards list. What is the difference between … In this Instuctable I'll be covering a step by step guide of programming an ATmega328P IC (The same microcontroller present on Arudino UNO) using … In this repo I'll be covering a step by step guide of programming an ATmega328P IC (The same micro-controller present on Arduino UNO) using Arduino IDE and an Arduino UNO as a programmer to make yourself a custom Arduino, to make your … The ATMega328P is a microcontroller with 23 IO pins, two 8-bit internal clocks, and 32kB of flash memory. Happy Making..!!Thanks,Gudimetla Kowshik Developing a custom PCB for personalized applications with Atmega328P SMD IC and stuck at how to program it or make it work with Arduino IDE like a normal Arduino UNO? Components To flash an ATmega328P chip with an Arduino you only need a few simple components that you likely already have if you use an Arduino including: Arduino Uno ATmega328P microcontroller Breadboard Jumper wires For this example I'll be flashing the ATmega328P with a simple blinking LED program so I also need the following components: 2 x LED 2 x 560 ohm resistor (or any resistor above 200 ohms) Hardware Setup For this project we will setup the standalone ATmega328P on a breadboard with two LEDs connected to seperate pins that will flash back and forth at changing speeds. Schematics 1: USB to Serial Adapter and the Atmega328P Most adapters can be configured to work at either 5v or 3.3v logic level. The ATmega328P microcontroller, commonly found in Arduino Uno boards, can be used as a standalone microcontroller in embedded systems. An ATmega328P straight from the manufacturer is set to run at 1MHz. This difference between what the IDE thinks the clock speed is set to and what the Atmega328P is actually running at will mean the delay times in the code will be incorrect. The go to "Sketch" and select "Upload Using Programmer". To remove this difficulty, the microcontroller is flashed with the Arduino bootloader, which makes it ready for programming using the simpler and easy to use Arduino IDE. This can be changed by modifying the ATmega328P clock speed by what are called the "fuse bits" but that takes some extra work. After successful code upload using any of the approaches described above, the Arduino or USB – Serial/TTL converter can be disconnected and the project connected to a battery to run on standalone as shown in the image below. Check out these tutorials:ConclusionUsing an Arduino as a hardware programmer is a simple and economical way to program the ATmega328P microcontroller. I am attaching a snap of my schematics here for your reference to know how the proper schematics must be drawn for the custom boards.Schematics of the custom board I made (only MCU, clock and reset sections are attached)How to program the custom board with Arduino UNO?The connections to be made between your custom board and Arduino UNO are pretty simple and straight forward. Helpful Resources From Arduino to a Microcontroller on a Breadboard This basic setup will power up your IC and you'll be ready for bootloading. Select the example and click upload. Now, open Arduino IDE . Prototyping is a great use for an Arduino board as it allows for quick and easy iterations of a design, but for completed projects it can often be overkill depending on the features used. Place the ATmega328P in the middle of the breadboard with the rows of pins straddling the centerline as shown below. This is where things can get tricky because the header pin numbers on the Arduino do not directly match the pin numbers on the ATmega328P even though the Arduino is using an ATmega328P itself. All you have you do is to make straight connections between ICSP pins of both boards as shown below.Connections between Arduino UNO and custom board ICSP pinsNote that the RST pin of custom board must be connected to D10 pin on the Arduino UNO (Slave select line). The easier way is to tell the IDE what speed the ATmega328P is at. Whereas on your custom boards, USB to serial adapters are usually not present (At least for the designs I make). You should see the connected LED start blinking after a while. Now, this is not a very traditional method to use but it reduces the number of parts on the PCB greatly saving few bucks to the designer while mass producing the PCB's. The code will be uploaded to the microcontroller. Required Components The following components are required for this approach; Schematic Connect USB to Serial/TTL adapter to the microcontroller as shown in the schematics below. To upload the bootloader, we'll make some extra connections to the basic power connections. Since the ATmega328P that we are flashing does not have the same configuration as the microcontroller on the Arduino, we need to install an additional hardware configuration in the Arduino IDE as outlined below: Download the breadboard-1-6-x configuation here. Preparing and Connecting the Arduino The Arduino Uno itself needs to have a special sketch uploaded to it for it to be used as in-system programmer (ISP). Select Atmega328p from it as we are using it for the current tutorial. Programmed Atmega328p Microcontroller That’s it for this tutorial guys, thanks for following. Luckily, if you have an Arduino, you already have what you need since the Arduino itself can be used as a programmer for the ATmega328P (and many other AVR microcontrollers). However, programming the ATmega328P outside of the Arduino board requires a hardware programmer. The Atmega328p microcontroller, like any other microcontroller, can be quite tasking to use for a beginner. You can also checkout my instructable regarding the same here Developing a custom PCB for personalized applications with Atmega328P SMD IC and stuck at how to program it or make it work with Arduino IDE like a normal Arduino UNO? The first mode is the easiest way to upload code to the microcontroller, as it involves just replacing the microcontroller on the Uno, with the one to be programmed. But the SS on the Arduino doesn't go to the SS on the ATmega328P, instead it is connected to RESET (PC6) on the ATmega328P. This is necessary for the ATmega328P to enter programming mode. Arduino UNO(This is bare minimum requirement to program your custom board) 2. Acknowledgement from Arduino IDE that code has been uploaded successfully.Thanks for the read. IMPORTANT - After bootloading, remove the ATmega chip form the Arduino UNO because now we will be using the … These development platforms, unlike the Arduino IDE usually require high knowledge of C or other programming languages, without the shortcuts and simplified functions which the Arduino provides. Few jumper wires to makes connections. Arduino IDE.That is it..! These are the things you need program your custom board and get it up and running like a normal UNO board.Great! But how did I end up finding this method and what errors did I face with my custom board before trying this method?Well, I went through lot of errors trying out different methods out there on the internet failing at every attempt. Feel free to drop questions and comments under the comment section, I will do my best to respond to them asap. IMPORTANT - After bootloading, remove the ATmega chip form the Arduino UNO because now we will be using the Arduino board just as an ISP Programmer (In System Programmer). The picture below shows how the Arduino pins compare to our ATmega328P for SPI to work. If the ATmega328P is successfully programmed we should now see the LEDs blinking back and forth at faster speeds until the loop completes and starts over. This article shows how to program a standalone ATmega328P using an Arduino and the Arduino IDE. This setup, compared to the second one, is by far the cheapest, as these adapters are usually very cheap. Using a USB to Serial/TTL Adapter USB to Serial/TTL Adapter The first approach involves the connection of a USB to serial adapter to the microcontroller. I have faced errors like followingError while burning bootloaderProgrammer out of sync errorCode flash successful but my custom board is not responding to any of the code I've writtenAt some point I doubted the schematics that are made for the custom board. In this article we will cover: Setting up the Arduino IDE to program a standalone ATmega328P microcontroller Preparing the Arduino Uno to work as an in-system programmer Programming an ATmega328P with a blinking LED program Introduction The ATmega328P is the microcontroller that powers the Arduino Uno development board. Once all the options are properly selected Click on Burn Bootloader option at the end, this will burn the arduino bootloader into the Atmega328p IC which makes it to accept any kind of codes or instruction over ICSP lines.Once this step is successful it is time to write your own sketches and upload them to the custom board. Open up the "doubleLEB_blink" sketch we created earlier. Extract the "breadboard" folder from the "breadboard-1-6-x" zip file and move it into the newly created "hardware" folder. IMPORTANT - This will be a one time process. All you need is 1. If you’re working on inter-microcontroller communication projects, check out this resource on microcontroller communication.Common Issues and TroubleshootingIncorrect Connections: Double-check your wiring against the ATmega328P pinout.Missing Bootloader: If your ATmega328P is not preloaded with a bootloader, you must burn it before uploading sketches.Wrong Port or Board Selected: Ensure the correct port and board are selected in the Arduino IDE.For advanced programming examples, such as working with timers, you can explore this ATmega328P timer programming guide.Bonus ResourcesOnce you’ve mastered the ATmega328P, you might want to explore programming other microcontrollers. The two LEDs will be connected to pin numbers 14 and 15 on the ATmega328P, which correspond to pins PB0 and PB1. This is called a bootloader. Go to the location of your sketchbook folder and create a "hardware" folder if one does not exist already. To test the setup, we will use the Arduino blink example. We are also using the 5V and GND pins on the Arduino to power the ATmega328P, therefore these pins need to be connected to the breadboard power and ground rails as shown. Required components To use this approach, you will need the following components; Arduino Uno Breadboard USB to serial/TTL Adapter 16MHz crystal oscillator 22pf capacitors x2 Jumper Wires 10k resistor 100 ohms resistor LED Schematics Connect the components as shown in the schematics below. At this moment the LED on your breadboard and the default Arduino UNO LED will start blinking in sync. Then you're at the right place. Don’t forget that this procedure will only work if the microcontroller has been flashed with a bootloader according to the procedure described in the last tutorial. An official Arduino Uno costs over $20 (although clones can be found on eBay for $10 or less) while a standalone ATmega328P costs about $2. The SPI pins names are SCK, MISO, MOSI, and SS. Your custom board with Atmega328p IC on it (Note that, the ICSP headers must be extended out to be hooked up with pins of ICSP pins of Arduino UNO) 3. I am going to discuss the easiest method amongst all those and you don't have to purchase any costly stuff to do this. To get the LEDs to blink, we will use the following code. Here's the pinout and connections:Power:Connect VCC and AVCC pins of the ATmega328P to the 5V pin on the Arduino.Connect GND pins of the ATmega328P to the GND pin on the Arduino.Oscillator:Place the 16 MHz crystal oscillator between the XTAL1 and XTAL2 pins of the ATmega328P.Connect a 22pF capacitor between XTAL1 and GND, and another 22pF capacitor between XTAL2 and GND.Reset Circuit:Connect a 10kΩ resistor between the RESET pin of the ATmega328P and VCC.Programming Pins:Connect the following Arduino pins to the ATmega328P pins:Arduino 10 to ATmega328P RESETArduino 11 to ATmega328P MOSIArduino 12 to ATmega328P MISOArduino 13 to ATmega328P SCKFor USART-based programming and communication setups, refer to this programming ATmega328P using USART guide.Step 3: Burning the Bootloader (Optional)To program the ATmega328P with Arduino sketches, it needs the Arduino bootloader.In the Arduino IDE, select Tools > Board > Arduino Uno.Select the correct Port for your Arduino board.Go to File > Examples > ArduinoISP. Learn more about alternative methods in this tutorial on programming ATmega328P with Atmel Studio.Step 4: Uploading a Sketch to ATmega328PNow, let’s upload a sketch to the ATmega328P:Write or load your Arduino sketch in the IDE.Go to Tools > Board > Arduino Uno.Select Tools > Programmer > Arduino as ISP.Instead of clicking the regular Upload button, click Sketch > Upload Using Programmer.The IDE will upload your code to the ATmega328P via the Arduino as ISP. This is usually done via two major ways: Using a USB to Serial/TTL Adapter Using an Arduino board Each of these approaches provides the microcontroller with an interface that enables interaction between the computer and the microcontroller. To begin, notice the notch that indicates the direction of the chip — and the dot that … This article shows how to program a standalone ATmega328P using an Arduino and the Arduino IDE. We will take each of these approaches one after the other and look at the components and setup required to upload code to the microcontroller. Another downside to this is that it only works with the Arduino Uno as all other Arduino boards, use SMD type of microcontrollers which makes replacement impractical and development, expensive. Programming Atmeg328p with the Arduino Uno While using this approach, it is important to remove the microcontroller of the Arduino board to prevent interference. ATmega328P on Breadboard Pin 1 Vcc via 10K resistor Pin 7 and Pin 20 Vcc Pin 8 and Pin 22 Gnd Pin 9 and Pin 10 8 MHz Crystal Oscillator Pin 9 and Pin10 Gnd via 22pF Capacitors each Pin 19 Gnd Via a series combination of 220 Ohm resistor and LED Microcontrollers are usually programmed through a programmer unless you have a piece of firmware in your microcontroller that allows installing new firmware without the need of an external programmer. Upload takes only a few seconds, same as the Arduino board. The Arduino board makes it easy to interface with the pins on the ATmega328P while adding extra features that don't come with the standalone microcontroller, including a USB serial interface and 16 MHz clock. Programming the ATmega328P Before we can flash the program into the ATmega328P, we need to make sure the correct board and programmer are selected in the Arduino IDE by doing the following: Select the correct board in the Arduino IDE by going to Tools > Board and selecting "ATmega328 on a breadboard (8 MHz internal clock)" Select the correct programmer in the Arduino IDE by going to Tools > Programmer and selecting "Arduino as ISP" With the board and programmer correctly set, we can now flash the program into the ATmega328P. Open the "boards.txt" file In the file, find the line that says "atmega328bb.build.f_cpu=8000000L" Change the line to the following "atmega328bb.build.f_cpu=1000000L" Save and close the file Now when the program is loaded onto the ATmega328P it will know the microcontroller is running at 1MHz instead of 8MHz and the delay times will be correct. Using an Arduino Board The second approach involves the use of an Arduino board in either of two similar ways; By replacing the microcontroller on the Arduino Uno with the one to be programmed By using any of the Arduino boards as an In-system programmer. Now remove all the 4 connections made in the bootloading process and do the following connections ATmega Arduino Pin 1 RESET Pin 2 D0 (Rx) Pin 3 D1 (Tx) Now, go to Tools > Programmer > and select AVRISP mkll Go to File > Examples > Basic > Blink Upload change the delays as you wish and upload the Sketch You are now ready with your Minimal Arduino, you can now integrate whatever you want with your Microcontroller and make Custom Arduinos and reduce the size and cost of your projects. The "hardware" folder should now have the "breadboard" folder in it. 4. Hope you guys liked it and saved some time for you avoiding the hard work. You are now ready to Program your ATmega328P IC just like your arduino. It has on-board USB to serial converter IC which translates information coming on the USB from your PC to TTL logic that is understood by the MCU. Determine the location of your Arduino sketchbook folder by opening the Arduino IDE, going to File, selecting Preferences and then looking at the Sketchbook location. Also, while uploading sketches remember to use Arduino Pro or Pro Mini as the Board with Processor as ATmega328P (3.3V, 8Mhz) rather than Arduino UNO as we have used Pro Mini's bootloader because we have connected a 8MHz crystal. After typing in your code, select the port to which your adapter is connected, followed by the board type and hit the upload button. Open your sketch, go to Tools -> Boards -> Arduino Nano For uploading code to Custom board with Atmega328PAfter this step ,finally you need to upload your code using the option Sketch -> Upload Using Programmer or simply press Ctrl + Shift + UUpload code using Programmer (Arduino UNO in our case)You should see a debug message like shown below which confirms that code is successfully uploaded into your custom board and your board must execute that code. #include #include int main(void){ DDRB = 0b00000011; // Sets PB0 and PB1 to outputs while(1){ PORTB = 0b00000001; // Turns on only PB0 _delay_ms(2000); // Delays 2000ms (2s) PORTB = 0b00000010; // Turns on only PB1 _delay_ms(2000); // Delays 2000ms (2s) PORTB = 0b00000001; // Turns on only PB0 _delay_ms(1000); // Delays 1000ms (1s) PORTB = 0b00000010; // Turns on only PB1 _delay_ms(1000); // Delays 1000ms (1s) PORTB = 0b00000001; // Turns on only PB0 _delay_ms(500); // Delays 500ms (0.5s) PORTB = 0b00000010; // Turns on only PB1 _delay_ms(500); // Delays 500ms (0.5s) } return(0); } Setting up the Arduino IDE Before we can program our code into the ATmega328 we need to do some additional setup in the Arduino IDE to get the Arduino Uno to function as a programmer. The USB to Serial/TTL adapter converts data signals from the USB on the computer to serial/TTL for the microcontroller and vice versa. Once the uploading is done you have successfully converted Arduino UNO as ISP programmer.Before we proceed with playing around with the custom board, we need to download the proper core files to bootload the IC. They usually require a certain set of tools, including a programmer (hardware), and a development platform (e.g Atmel Studio) for writing code. Since we are using straight C code for the program, this means we only have to concern ourselves with PORTB for controlling the pins. Luckily, you can use an Arduino board as a cost-effective hardware programmer.In this blog post, we’ll explore how to program an ATmega328P using an Arduino board as the hardware programmer and link to additional resources for those wanting to dive deeper into related topics.What You'll NeedArduino Uno or similarATmega328P microcontrollerBreadboard16 MHz crystal oscillator22pF capacitors (2 pcs)10kΩ resistorJumper wiresUSB cable for the Arduino boardArduino IDE installed on your computerStep 1: Setting Up the Arduino as ISPThe Arduino Integrated Development Environment (IDE) has a built-in feature that allows you to use an Arduino board as an In-System Programmer (ISP).Open the Arduino IDE on your computer.Connect your Arduino board to your computer via the USB cable.Go to File > Examples > 11.ArduinoISP > ArduinoISP.Upload the ArduinoISP sketch to your Arduino board.For more information on configuring external programmers, check out this guide on programming ATmega328P externally.Step 2: Wiring the ATmega328P for ProgrammingSet up the ATmega328P on the breadboard. Type the code to be uploaded or select an example -> select the board type (Duemilanove or Nano W/atmeg328), select the correct port and click upload. So no schematic for this, just swap the microcontroller and hit upload. From the diagram we can see that we need to connect the Arduino pins to the ATmega328P pins as follows: Arduino Pin 13 → PB5 (SCK) Arduino Pin 12 → PB4 (MISO) Arduino Pin 11 → PB3 (MOSI) Arduino Pin 10 → PC6 (SS to RESET) For SCK, MISO, and MOSI we simply connect those pins on the Arduino to the same ones on the ATmega328P. A separate programmer is needed. link to additional boards managers URLsOnce it is pasted, click on Ok and exit the window. Note: when programming the Atmega328p MCU using the Arduino IDE, the matching board type you have to select is the “Arduino Duemilanove or Nano w/ ATmega328” board. Now, upload this ArduinoISP sketch to Arduino UNO to convert it as a programmer for your custom board. Select programmer as AVRISP mkII. In this article we covered how an Arduino Uno can be used as an in-system programmer to program a standalone ATmega328P. This is done by the following: Go to the "hardware" folder we just created when loading the configuration and open the "breadboard" folder, then open the "avr" folder. So do I have to use costly programmers like Atmel ICE or other debuggers to program these custom IC's?One word answer - NO! Fortunately there are multiple other ways to do it. The Code To write the code, we will use the Arduino IDE. So once you have completed prototyping a project with the Arduino, you can transition the project to using a standalone ATmega328P instead. To make sure we are connecting the correct pins we need to compare the Arduino pin numbers to those of the ATmega328P. Ensure yours is configured to work on the 5v voltage level since supply to the microcontroller is 5v. To do this, go to File -> Preferences or press Ctrl + comma on your keyboard. Once uploaded, it will be able to program the ATmega328P. The Arduino is setup to be an ISP by performing the following steps: Open up the ArduinoISP sketch by going to File -> Examples -> ArduinoISP -> "ArduinoISP" Upload the ArduinoISP sketch to your Arduino as you would normally with any other sketch With the Arduino now setup as an ISP, we can use it to program the ATmega328P. Go to File -> Example -> ArduinoISP -> ArduinoISP sketchOpen the ArduinoISP sketchAfter opening this go to Tools -> Boards -> Arduino UNO. We will use the Serial Peripheral Interface (SPI) bus to connect the Arduino to the ATmega328P by properly connecting the SPI pins. Go to Tools -> Boards -> Boards ManagerFind the MiniCore boards file from MCUdude and click on install button to install the boards.Installing core packages for Atmega328P ICOnce this is done, you should be able to see the Atmega328, 168, 88, 48, 8 boards under MiniCore boards section. Restart the Arduino IDE In the Arduino IDE, confirm that setup was completed correctly by going to Tools > Board. However I found a very simple and elegant solution that makes life of hobbyists and makers much more easier. As stated in the name, the configuration we just loaded into the Arduino IDE (ATmega328 on a breadboard (8 MHz internal clock)), assumes that the internal clock speed of the ATmega328P is set to 8MHz. While 8MHz is one of the clock speeds of the microcontroller, it is likely not operating at that speed. This is usually done by the designers because AVR microcontrollers like Atmega328p comes with an programming and debug interface called ICSP(In-circuit Serial programming) which just uses 6 pins on the microcontroller to directly upload the code from any external programmers. Connect pin 7 (VCC) to the power rail and pin 7 (GND) to the ground rail. I have dug so deep such that at one point I was messing up with the fuse bits of the Atmega328p bricking my IC taking it to a state where it no longer accepts any code. To program the microcontroller using the Arduino IDE, the microcontroller must be connected via some sort of hardware to the computer. Put this code in the blank sketch and save it as "doubleLED_blink". Multiple options will be shown for selecting BOD (Brown out detection), EEPROM, Compiler etc., My suggestion is to leave them as it is and select programmer as Arduino as ISP (minicore) (Important step)If this programmer is not selected properly then Arduino IDE will not understand through which the path the code must be sent and it gives back an error. Below we see how the pins should be connected. Delete all the pre-populated code in it since when using C directly to program the ATmega328P we do not need the setup() and loop() routines that the Arduino IDE automatically provides. Now you can prototype and develop custom programs for the microcontroller without needing all the extra features of the Arudino while utilizing a smaller package once your design is finalized. This enables communication from the microcontroller (serial) with the Arduino IDE running on the PC (USB). Atmega Arduino UNO Pin 1 D10 (RESET) Pin 17 D11 (MOSI) Pin 18 D12 (MISO) Pin 19 D13 (SCK) Now open Arduino IDE Go to File > Examples > ArduinoISP Go to Tools > Board > Arduino UNO Select port from the Tools > Port Upload the ArduinoISP sketch to your board After successful uploading of the code go to Tools > Board > and select Arduino Pro or Pro Mini Go to Tools > Processor > and select ATmega328P (3.3V, 8MHz) Go to Tools > Programmer > and select Arduino as ISP (Not ArduinoISP) Go to Tools > Burn Bootloader This may take a while, and you'll be shown Done Burning Bootloader. These serve as the power and ground to the ATmega328P. We will also be using straight C code without any of the built in functions the Arduino IDE provides. Arduino UNO is completely customized to work as evaluation board for the Atmega328p IC. Atmega328p Microcontroller We covered details on preparing the Atmega328p microcontroller for programming by flashing the Arduino bootloader on Atmega328p and today’s tutorial will be a follow up to that tutorial, as we will look at how to program the boot-loaded Atmega328p microcontroller using the Arduino IDE. What is the difference between Arduino UNO and having Atmega328p on custom PCB's? Finally connect the LEDs to pins 14 (PB0) and 15 (PB1) as shown and use a resistor to connect them to the ground rails. The second method involves the use of the Arduino Uno as an In-system programmer. Till next time! Please follow and like us: An Arduino can be used to flash a standalone ATmega328P microcontroller without the Arduino bootloader. Upload Code Code upload process is the same as already described. Click on Tools > Programmer > AVRISP mkII.Click on Sketch > Upload.This step writes the bootloader to the ATmega328P, making it compatible with Arduino sketches. Uploading Code Uploading code to the microcontroller after you are done with the connections, require no additional work asides, what you would have done if you where using an Arduino board. By performing some simple modifications to the Arduino IDE configuration and connecting the Uno to the ATmega328P on a breadboard, we can easily load our programs onto the microcontroller while also powering it directly from the Arduino. However, this may not be the best when prototyping as the move of the chip from the Arduino to the project, back and forth, could lead to the pins of the microcontroller being damaged. In the last tutorial, we discussed an alternative way of using Arduino, i.e. using the Atmega328p microcontroller alone which removes all the downsides of using the Arduino board, while retaining one of the biggest benefits of the Arduino platform; the ease of programming. Open up the Arduino IDE and create a new sketch.