

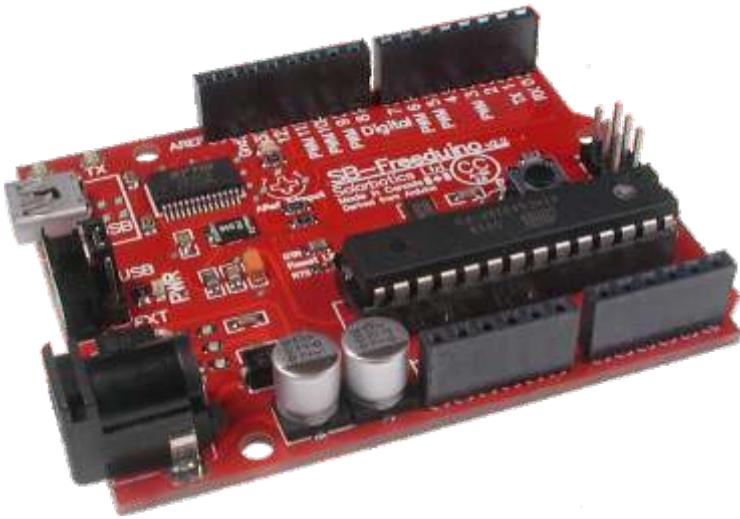


a division of
SOLARBOTICS

Freeduino SB ^{V2.2}

Open-Source
Arduino[™] -Compatible
Semi-Kit

We've done the hard SMD soldering -
you add just the sockets, pins and
connectors!



The Freeduino SB is 100% compatible to the Open-Source Arduino[™] Diecimila project, with the additional features of:

- Mini-B USB jack
- Power switch
- Optional ARef trimpot for Analog input scaling

NEW!

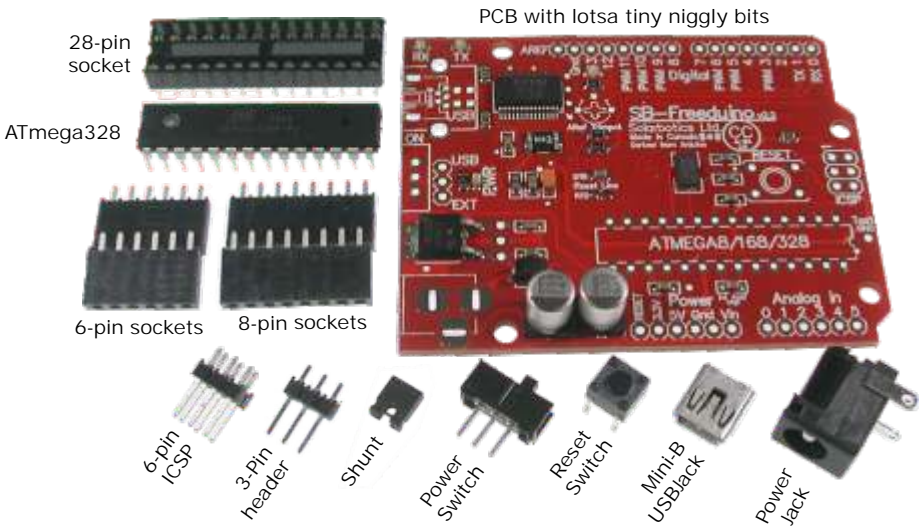
Version 2.2 upgrades:

- ATmega328 at 5 Volts
- FTDI-BitBang Pads
- SMD 16MHz Crystal
- Battery-supply friendly!

This is an intentionally blank page.
Y'know, for notes and pretty pictures and the like.

Parts List

- Printed Circuit Board (with all the tiny bits pre-soldered on)
- Power Jack
- Power Switch
- 3-Pin Header (for USB <--> Jack power selection)
- Shunt (for above header)
- 2x3 6-Pin header (for In-circuit serial programming)
- 2 x 6-socket headers (for shield interface)
- 2 x 8-socket headers (also for shield interface)
- 28-Pin DIP Socket for Atmel Microcontroller
- ATmega-328 Atmel Microcontroller with Arduino™ bootloader
- Pushbutton reset switch
- USB Jack (mini-B)



We strongly suggest you inventory the parts in your kit to make sure you have all the parts listed (c'mon - there's barely a handful of parts, so count them!). If anything is missing, contact Solarbotics Ltd. for replacement parts information.

Disclaimer of Liability

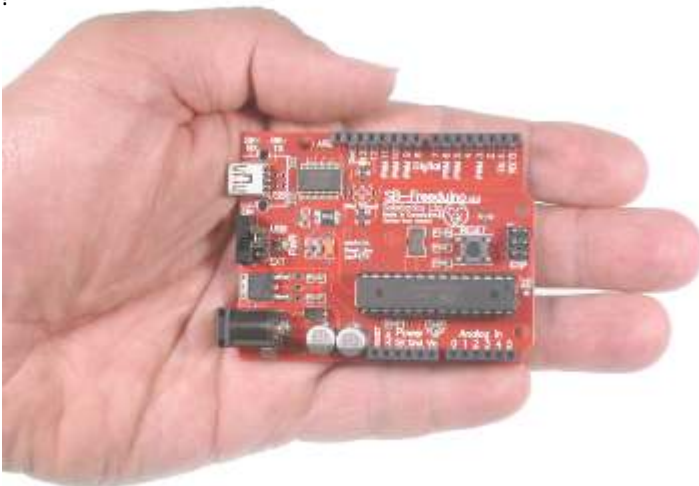
Solarbotics Ltd. is not responsible for any special, incidental, or consequential damages resulting from any breach of warranty, or under any legal theory, including lost profits, downtime, good-will, damage to or replacement of equipment or property, and any costs or recovering of any material or goods associated with the assembly or use of this product. Solarbotics Ltd. reserves the right to make substitutions and changes to this product without prior notice. Sorry - we hate legalese too, but "don't pick on us" doesn't impress the lawyers.

Introduction

Freeduino? Arduino? Draino? By now, you should know the difference. See the back page if you're unclear (note: stuffing this product down a clogged sink is *not* going to be terribly useful).

What makes the Freeduino SB different from the reference Diecimila and original Freeduino designs is:

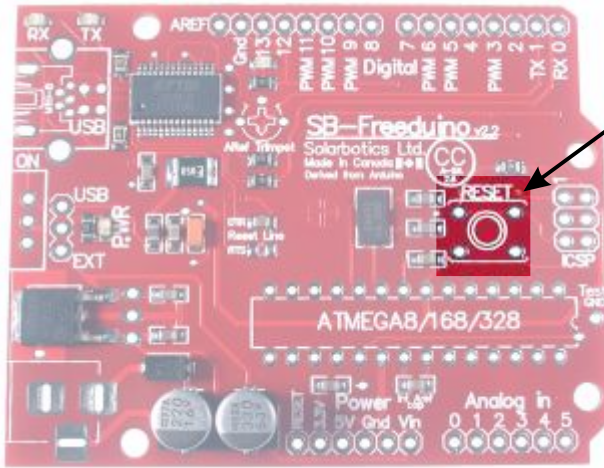
- This is a *semi*-finished kit. We installed all the hard-to-solder surface-mount components. You take care of the simple, large stuff.
- You can use a regular large, boring USB-B jack, or a cool mini-B jack.
- We've added space for a 10k VRef trimpot for easier analog tuning.
- We're using a precision 16MHz SMD crystal - *much* better than a resonator!
- We have a power switch! Funny, but yes, this is a notable improvement over the regular Diecimila!
- Bigger power filter capacitors ("All the better to power you with...").
- Space for a *big* TO-220 power regulator for power hungry circuits. The included one handles up to 500mA, but a TO-220 can power a full ampere (or more) - ideal for projects with multiple motors & servos!
- Data send/receive LEDs near the edge for easy viewing when shield expansions are installed on top. Blinky lights gotta be seen!
- It's a *BLUE* LED on pin 13! All the other guys use boring red. Or green (which is still boring...). Everybody knows blue is the coolest.
- We're 100% RoHS compliant, and made in North America! So is the kit!



Construction!

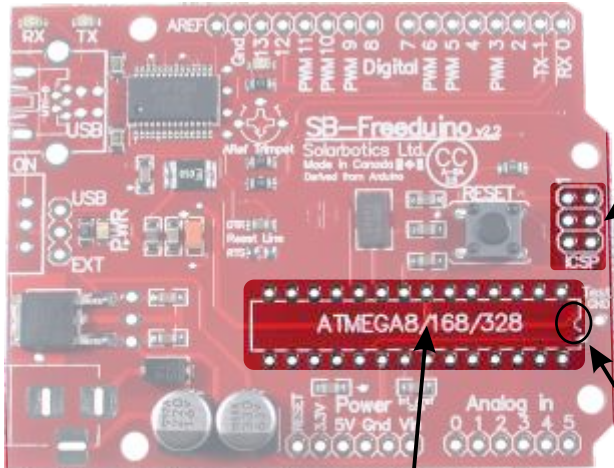
I'm sure you'd rather build something rather than read more fluff. Let's get to it.

Step 1: The Reset Switch - What good is building a project that will take over the world (starting with your workshop) if you don't have a convenient way to knock it over the head? Exactly. Install this, here.



Step 1:
The reset switch

Step 2: The ICSP 6-Pin Header & 28-Pin DIP Socket - Install them where shown. Watch that the notch on the end of the 28-Pin DIP socket matches the picture on the PCB. It's not *critical*, but it's a good idea so you know how to install the chip later.



Step 2a:
2x3
ICSP

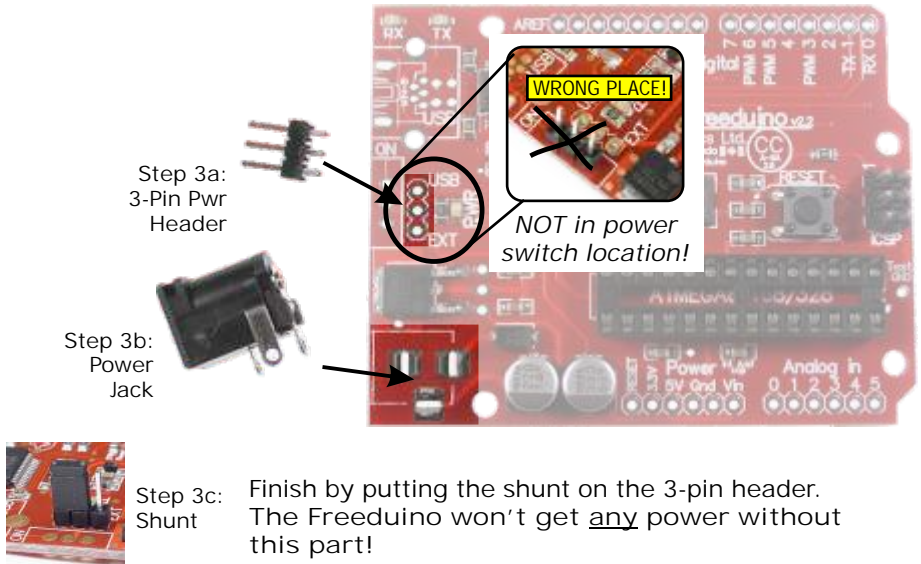
Step 2b:
28-Pin
DIP
Carrier



Note notch position!

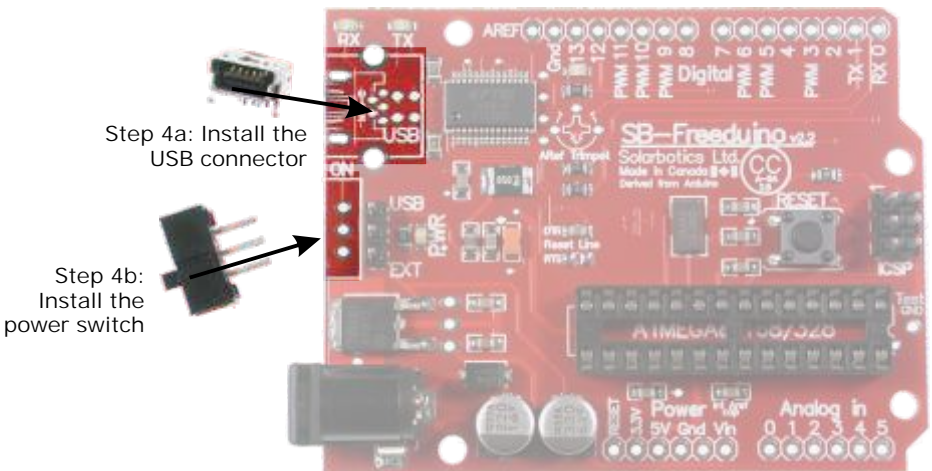
Construction!

Step 3: The Power Jack and 3-Pin Power Selector - Solder the jack into the only place that it will fit, and install the 3-pin header where shown. Do NOT install it in the power switch location! (Bad Freeduino-builder, bad!)



Step 4: The USB Connector and the Power Switch. A small "mini-B" connector is in your kit, but you *can* use a big ugly USB-B jack (guess which we like best?).

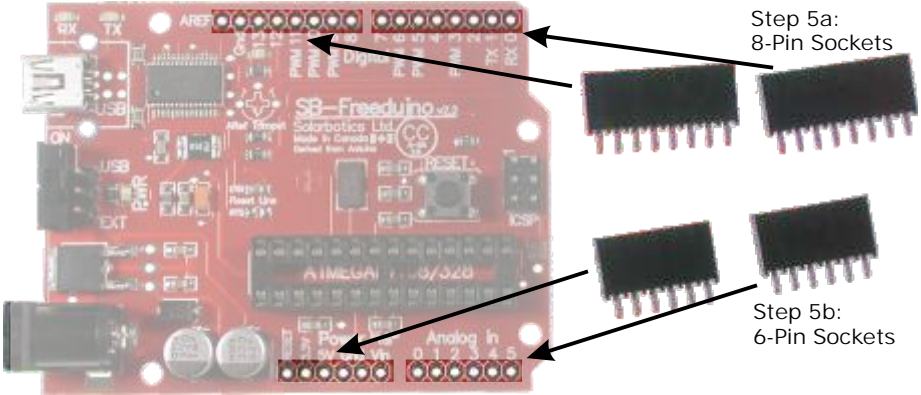
NOTE: If you are planning on using Shield add-ons, you should use the "Mini-B". Our design pushes the USB-B connector up to make room for the switch, and it *will* interfere with some Shield boards. Not so for the Mini-B.



Construction!

Step 5: The 6 and 8-Pin Headers. You don't *need* to install these, but they'll let you add Shield add-on boards, and make it easy to plug wires in for quick prototyping. Install them on the top.

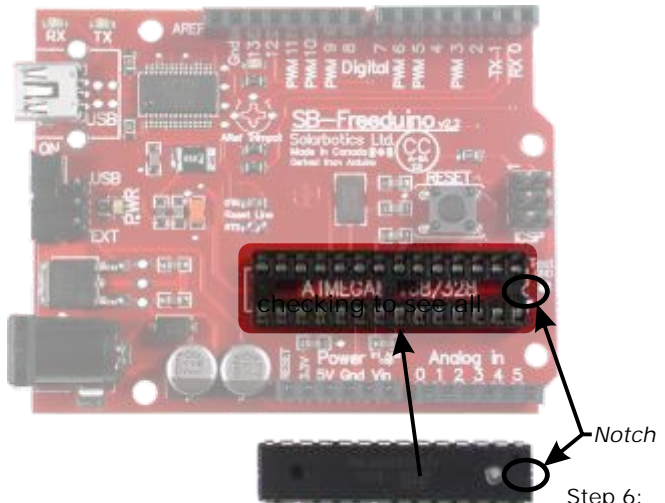
✓ **Tip:** Tack-solder just one leg to make sure you have the socket in straight before soldering in the whole thing.



Step 6: Atmel ATmega328 - Almost time to install the brains! BEFORE touching the chip, touch a metal sink, your computer's USB cable connector or similar to discharge any static you may have built up. Static zaps *will* kill your chip, so try to keep yourself grounded to something that will drain the static charge.

Insert the microcontroller so the end with the notch points *left*; the same side the notch on the carrier.

Gently push the chip into the carrier, pins are in the socket. Bent pins = bad!



See what happens when you're trying to take pictures, and *not* be careful? Don't do this! Bent pins = bad! (Yes, it needs to be said twice)

Step 6: Install the microcontroller. Watch notch orientation!



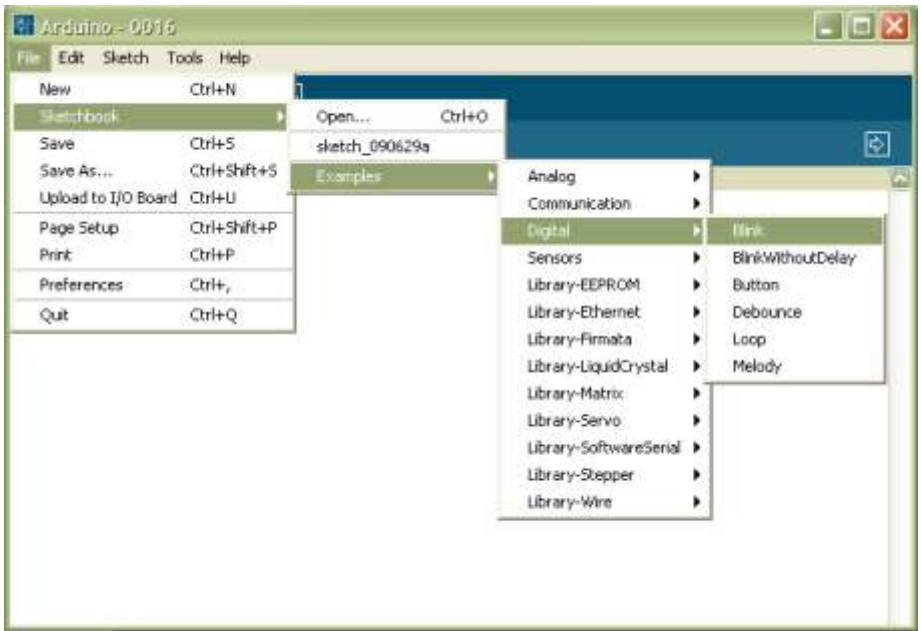
Testing the Freeduino SB



Now you're officially done the assembly of your Freeduino SB microcontroller! Now download the latest *Arduino* software from the Arduino group (<http://www.arduino.cc/en/Main/Software>). It's time to test it!

The easiest way to test your Freeduino SB is to:

1. Put the power jumper on the top two pins so it's powered by the USB port.
2. Plug it into your PC's USB port.
3. Let your computer detect it and install the USB device drivers.
4. Turn on the Freeduino's power switch (the power LED lights up).
5. The blue LED should start blinking. That's a sign the chip is alive.
6. Let's test programming. Load up the "blink" code that comes with the Arduino software (File/Sketchbook/Examples/Digital/Blink).
6. Click the "upload" button (CTRL-U).
7. ...watch the blinky lights on the upper-left corner of your Freeduino...
8. Watch your blue blinky light do it's thing! It takes programs!



Didn't quite work out? This software is pretty straightforward - try looking at Tools/Serial Port for different comm ports to try, or Tools/Board to make sure you have the "Arduino Duemilanove w/ ATmega328" selected.

Still having problems? Check your soldering. Resolder all your connections. REALLY. That fixes a great number of errors we've seen, especially those "Hmm. I thought I soldered that chip carrier in..." mistakes.

If that doesn't do it, give us a quick email, include a nice macro-photo of your board (both sides, make sure the camera's "flower" icon is on for good close-ups), and we'll investigate. At worst, you can return it to us, and we'll repair it free-of-charge for you (contact us for details before shipping).

★ Freeduino SB Extra Features ★

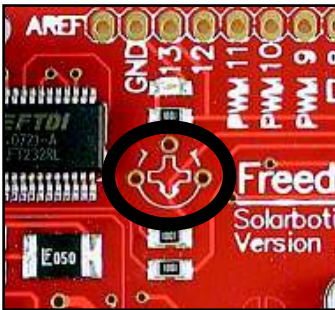
The ARef Trimpot

On a standard Arduino, the analog reference (ARef) is set to 5V, or the “internal reference” voltage (1.1V for the ATmega328). Voltages measured by the analog-input pins are divided against this voltage (to the closest 1/1024 of ARef). At the standard 5V ARef, you can make measurements in 0.0049V increments (5V/1024 steps), which is pretty good. But if your input signal is only 1.9V, you can tweak your ARef with a trimpot to be a good 2V reference, giving you (2V / 1024) increments of 0.002V - over twice as accurate.

We suggest installing a 10k or 100k trimpot (our part RT10k or RT100k) to set your new desired ARef (between 0 and 5 volts). Set the new voltage by measuring the AREF pin (up top, next to the GND pin).

Other Features...

- We went out of our way to make sure we keep the FTDI USB chip happy. We’ve improved on the Diecimila reference design by adding a ferrite bead and refining the capacitor selection to meet FTDI’s specifications.
- The Freeduino SB PCB separates the I/O lines with a ground lead for better noise suppression, which is important for high frequency PWM signals.
- We included pads for using the serial RTS line to reset the Freeduino SB after a download. It’s not essential, but it makes the design future-proof!
- We added a ground test pad near the edge of the board near the ATmega168.
- It’s a blue LED on pin 13! We did mention that, right? Blue is way cooler than red. Or green...



ARef Trimpot mount



RTS Reset Line available



Ground Test Pad



So you've got a Freeduino SB, you know how to turn it on, and make it blink - what do you do now?

The website www.arduino.cc is a very complete resource. The Arduino team has done an excellent job supporting this project, particularly with their personal involvement in the message forums (which we *strongly* suggest you review for support, ideas, and to show off your own work!).

A very active resource for Arduino projects is at Makezine.com, particularly at blog.makezine.com/archive/arduino/. If you're not familiar with MAKE magazine, it's all about using stuff to... make stuff! Which is sorta what we do here at Solarbotics / HVW Technologies. MAKE is a great website and magazine.

Another excellent resource that has been released in the spirit of open source is the "Arduino Programming Notebook" - a book written by Brian Evans that is freely downloadable as a PDF, or purchasable for a very reasonable \$5.33 here:

<http://www.lulu.com/content/1108699a>

Our special thanks to the core Arduino development team (heck, even those on the fringes), Massimo Banzi, David Cuartielles, Tom Igoe, Gianluca Martino, David Mellis and Nicholas Zambetti.

Additionally, we'd like to thank the group at Freeduino.org (Daniel Jolliffe, Tony Kim, Oliver Keller, and William Westfield) for brainstorming with us on how to improve an already very useful piece of equipment. The website www.freeduino.org is indeed an excellent compliment to the main Arduino website. Great thing this open-source stuff - get a little, give a little, and we all benefit!

Lastly, a special acknowledgment to technical specialists Dan Gates and Mark Martens on their work on this project.



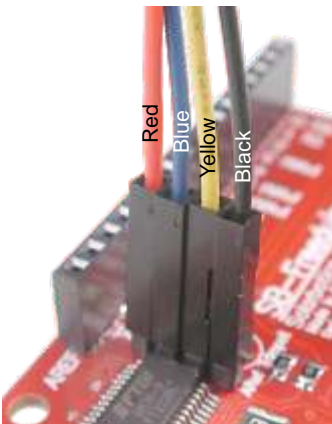
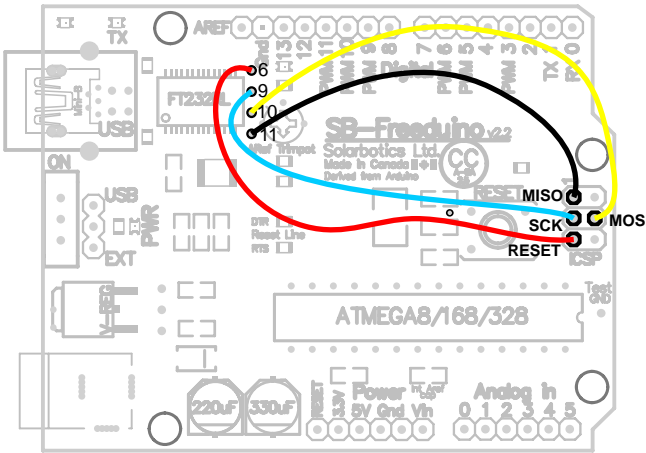


Burn your own Bootloader



Kimio Koaka of Japan worked out a *most* handy hack to the Arduino. He managed to make spare pins on the FTDI USB chip act as a “bit-bang” AVR programmer. This means you can use the extra (unused) functions of the FTDI chip to act as a mini-programmer to program fresh Arduino bootloaders onto new ATmega168/328 chips! Very handy when you want to program up your own Arduino clones and don’t have an AVR programmer!

We labeled the pads on the bottom of the PCB, but ignore them (we, ummm... got the labels wrong). Use *this* wiring diagram, and you’ll be burning your own bootloaders in no time. Follow Mr. Koaka’s instructions - we’ve tested them! http://www.geocities.jp/arduino_diecimila/bootloader/index_en.html



The Freeduino SB: Based on the Open-Source Arduino™ Microcontroller Project

What's an Arduino?

It's an open source electronics prototyping system using popular proven tools and hardware. It's designed to be easy to interface with sensors and actuators, and inexpensive enough you can use many in your projects!

Is that all there is about it?

Not at all! A large part of the appeal of Freeduino/Arduino is the frequently updated software used for programming the hardware. The Arduino programming language is based on a very easy-to-learn open-source language called Wiring, which is similar to C/C++, but streamlined for quicker development!

What is this "Open-Source" ?

Good question! In short, it is a set of principles given to a project by the project's creator that lets everyone use it, and have access to the source design. The creators of the Arduino project allow anybody to use their hardware designs and software, as long as everybody shares back in return! (It's a big topic - do an Internet search!)

What are the hardware specifics?

An Atmel ATmega328 running at 16MHz, connected through an FTDI232R USB-to-serial converter (just plug it into a USB port, and go!). 13 digital I/O pins (6 with PWM) and 6 analog inputs. Power comes from your USB port, or external 6-20VDC power supply. Your PC is protected by a 500mA resettable fuse. LEDs for power, Rx/Tx activity and pin 13 "test" are all pre-installed!

What can I do with it?

LOTS! There's a great many projects documented online. When people use open-source software, they seem to like to share what they've done! A Google search for "Arduino Projects" results with over 1.58 million pages!

Can I build this?

If you can do basic soldering, you bet! We've taken care of the hardest stuff. You just finish it, then download and install the latest Arduino programming software!

"Arduino" is a trademark of the Arduino Team (www.arduino.cc). The Freeduino SB is based off reference designs by the Arduino Team, and is licensed under the Creative Commons A-SA 2.5 license <http://creativecommons.org/licenses/by-sa/2.5/>

Visit us online for more info and neat things:

www.HVWTech.com

Solarbotics Ltd.
201 35th Ave NE
Calgary, Alberta T2E 2K5
Canada



Toll Free: 1-866-276-2687
International: +1 (403) 232-6268
Fax: +1 (403) 226-3741

HVW Technologies is a division of Solarbotics Ltd.
visit us online at www.solarbotics.com



Made in Canada