



# PS/2/You LED Sign

Written By: Immanuel McKenty

## TOOLS:

- [Chisel \(1\)](#)
- [Computer with internet connection and USB port \(1\)](#)
- [Desoldering braid or solder sucker \(1\)](#)
- [Drill and bits: 5/64", countersink \(1\)](#)
- [File \(1\)](#)
- [Glue gun and hot glue \(1\)](#)
- [Hammer or mallet \(1\)](#)
- [Handsaw or chop saw \(1\)](#)
- [Measuring tape or long ruler \(1\)](#)
- [Multimeter \(1\)](#)
- [Needlenose pliers \(1\)](#)  
*(optional) handy for plugging in breadboard jumpers*
- [Screwdriver, medium \(1\)](#)
- [Soldering iron and solder \(1\)](#)
- [Table saw \(1\)](#)  
*(optional)*
- [Wire cutters \(1\)](#)

## PARTS:

- [Make PS/2/You Kit \(1\)](#)  
*from the Maker Shed ([makershed.com/ps2you](http://makershed.com/ps2you)). Our kit includes almost everything you need to build your own PS/2/You sign except for the enclosure.*
- [Dot matrix LED display modules, 8x32, with ribbon cable \(3\)](#)  
*Sure Electronics item #DE-DP106, about \$9, or equivalent module. This item was recently discontinued but is available on eBay.*
- [Computer keyboard with PS/2 connection \(1\)](#)  
*They're readily available at thrift stores. You can also use a USB keyboard with a USB-to-PS/2 adapter.*
- [Ardweeny microcontroller \(1\)](#)  
*This small, cheap Arduino clone fits into a standard 14-pin DIP socket, but it doesn't come with an onboard 5V voltage*

- [regulator or an FTDI USB-serial converter for programming.](#)
- [5V voltage regulator \(1\)](#)  
[You can use a 7805, but the low-dropout LM2937 will make your batteries last longer, especially with lower-voltage NiMH AAs.](#)
- [PS/2 port \(1\)](#)  
[from an old PC motherboard; ask your local computer shop.](#)
- [Breadboard \(1\)](#)
- [Breadboard jumper wires, or solid core 22AWG wire \(20\)](#)  
[\(around 20, multiple colors\) Jumpers are easier to use and well worth the expense.](#)
- [FTDI serial programmer \(1\)](#)  
[such as the FTDI Friend, Maker Shed #MKAD22, \\$15](#)
- [AC wall adapter \(1\)](#)  
[can be found for \\$1–\\$2 at most thrift stores](#)
- [DC power jack \(1\)](#)  
[which ever matches your adapter](#)
- [Power switch \(1\)](#)  
[\(on-off-on\)](#)
- [Capacitor \(1\)](#)
- [Batteries \(6\)](#)
- [Battery holder \(1\)](#)
- [Battery holder \(1\)](#)  
[in a long, flat 2x2 configuration](#)
- [Wire \(1\)](#)  
[4' total](#)

- [Electrical tape or heat shrink tubing \(1\)](#)
- [Acrylic/plexiglass sheet \(1\)](#)  
*[Lexan will work great but is more expensive](#)*
- [Wood screws \(8\)](#)  
*[1 1/4" long](#)*
- [Wood screws \(6\)](#)  
*[1/2" long](#)*
- [Wood screws \(12\)](#)  
*[1/2" long \(optional\)](#)*
- [Dimensional lumber \(1\)](#)  
*[\(3/4" x 3 1/2"\), 4' length, or 1 x 2 \(3/4" x 1 3/4"\) , 8' length](#)*

## SUMMARY

It all started with a small LCD salvaged from an old printer. I recruited my code-savvy older brother, Adam, and we soon had the LCD displaying text from an Arduino microcontroller. This was neat, but it was inconvenient having to plug the Arduino into a computer for reprogramming whenever we wanted to change the text.

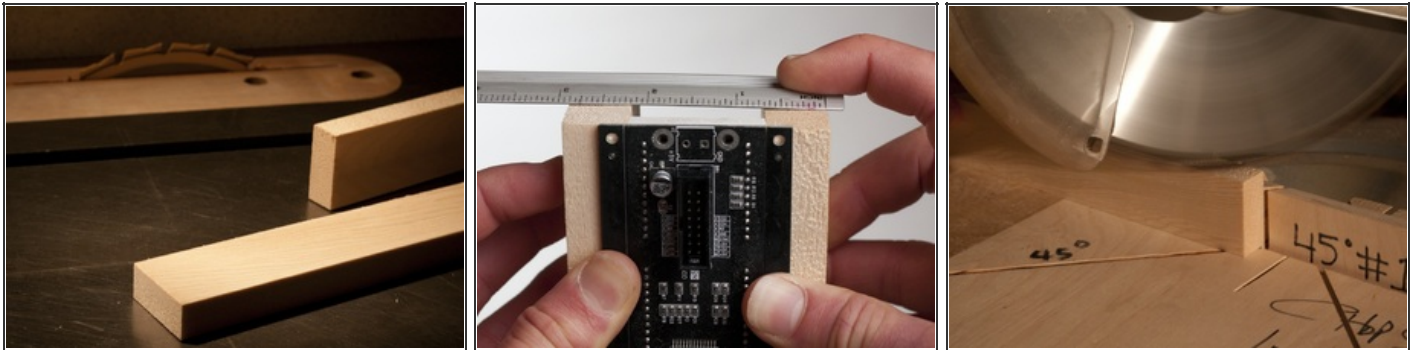
We needed something for inputting the text, and it didn't take long to find a PS/2 keyboard code library for Arduino — which confirmed my observation that anything that communicates with wires has probably been hooked up to an Arduino. I salvaged a PS/2 port from an old computer motherboard, and after some trial and error, we could plug in a common PS/2 keyboard (\$5 new) and type messages directly into the Arduino and out to the LCD.

The LCD was so small, however, that hardly anyone noticed our witty remarks. We needed a bigger display. After looking at many appallingly priced commercial LED matrix products, we found a new and much cheaper offering:

Sure Electronics' 8x32 display boards. They cost \$9 each and you can cascade up to four into one long display. We ordered three, and by the time they'd arrived, the Arduino community had already produced a library to run them. (Our code is based on two open source Arduino libraries: PS2Keyboard, by Christian Weichel; and MatrixDisplay, by Miles

Burton.) The result is our PS/2/You system, which displays keyboard-typed messages in 2"-tall LED letters that can be read from quite a distance. You can store and switch between six different lines of text, and it automatically scrolls through lines that are too long for the display. Power comes from an AC adapter or six AA batteries for portable operation, and the whole thing is housed in a sturdy wooden frame.

### Step 1 — Build the frame.



- Cut the 1×4 lumber in half lengthwise to make 2 strips about  $\frac{3}{4}$ "× $1\frac{3}{4}$ " (a nominal 1×4 is actually around  $\frac{3}{4}$ "× $3\frac{1}{2}$ "). Use a narrow-kerf blade if possible.
- Line up the 2 boards beside each other on a flat surface with their narrow edges up. Place one of the display panels facedown between the boards, so that the flanges on the panel rest on the boards, with the protruding LED matrix between them. Gently squeeze the boards snug against the sides of the LED matrix, and measure between the outside edges of the boards. This measurement is the length of the frame's end pieces. (Remember this measurement for the following steps.)
- Use a chop saw or handsaw to cut a 45° angle on one end of each piece, oriented so the cut goes diagonally across the narrow edge. Measure  $18\frac{1}{4}$ " down the board's length from the inner edge of the cut, and make a second 45° cut that angles back out. Repeat on the second board. These will be the 2 long sides of the frame.

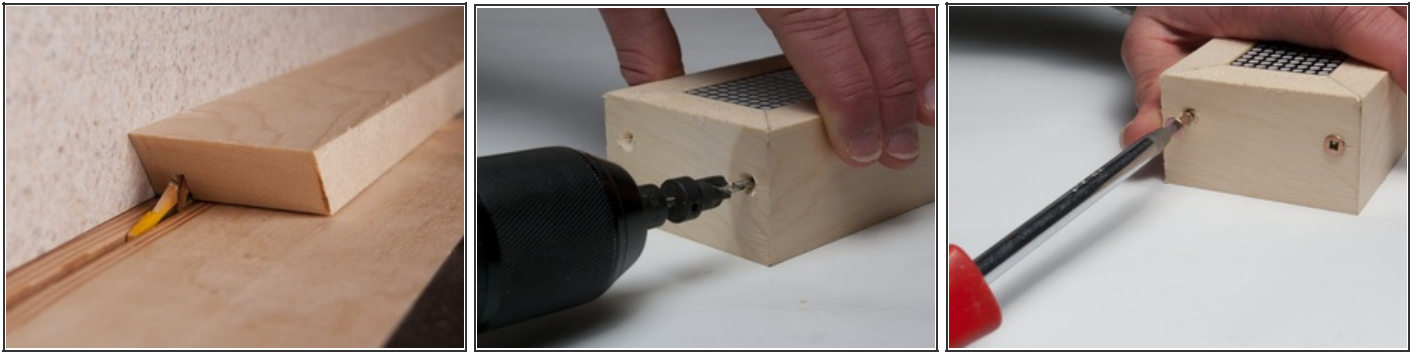
## Step 2



- On each of the leftover board pieces, mark the distance measured measured in Step 1 down from the sharp, outside edge of the miter cut, along the longer face of the board. Cut one at 45° angled in from the measured length (the mirror image of the first cut), which will be each piece's longest dimension. Don't cut the other piece yet.
- Desolder the PS/2 port from its donor motherboard. Line up the DC jack, PS/2 port, and power switch atop the edge of the marked but uncut short piece of wood. Mark out a notch in the edge of the board just wide enough for all of them to fit next to each other and deep enough that the tallest component will sit flush.
- With a handsaw (or chop saw) cut the 2 edges of the notch. Make a few cuts to the correct depth in the middle of the notch, then chisel out the rest of the wood and file the bottom of the notch smooth. The ports and switch should slide into the notch easily but without extra space. Cut the second (notched) end piece just like the first mitered end piece.
- **NOTE:** The frame style isn't crucial, so let your creativity (and materials) have a say in the design. I had a woodshop at my disposal, so I made something like an extra-deep picture frame with mitered corners and a slot cut in the long sides to hold the display panels.



### Step 3

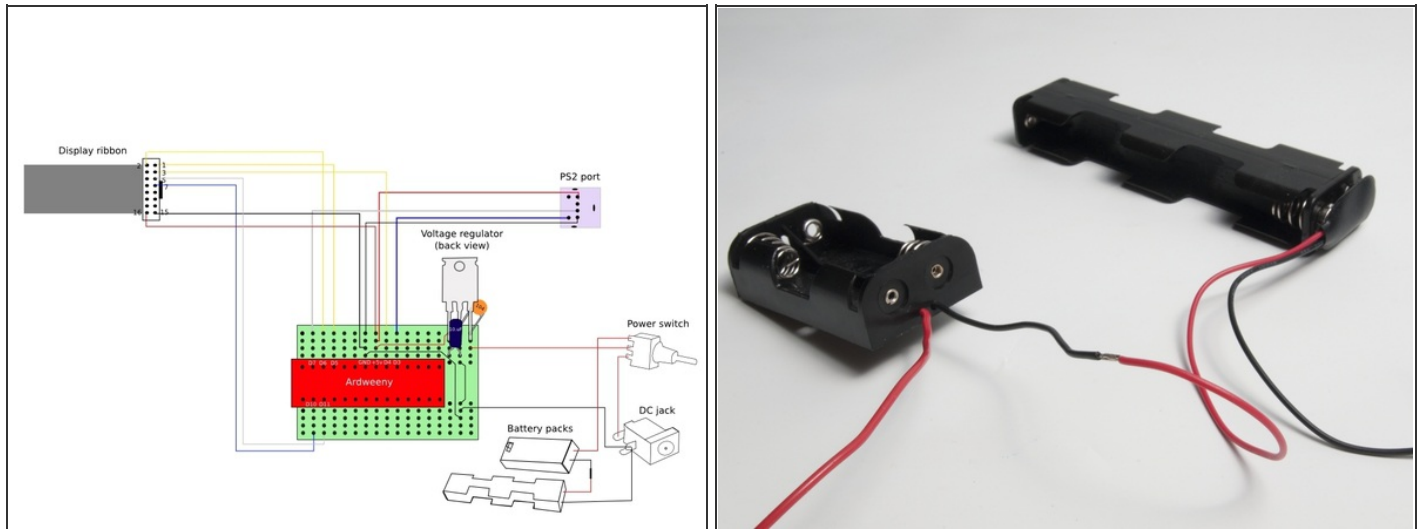


- Set the blade of your table saw to a depth of 5/16" (the size of the flanges on the display panels). Cut a groove lengthwise down the inside face of each long frame piece, 1/4" in from the edge (preferably with a narrow-kerf blade).
- Slide the 3 display panels into the slots in the long frame pieces, all oriented so that the writing on the printed circuit boards reads right side up as you look into the back of the frame. Fit and hold a frame end piece in place at each end, then drill 5/64" pilot holes and countersinks for the 1 1/4" #8 screws that hold the frame together.
- The notched end belongs on the left side of the frame as you read the circuit board backs, which is the right side as you view the front of the display. Put the screws in as you go to keep the frame together. Ensure that your countersinks are deep enough, and (to avoid splitting the frame pieces) don't overtighten the screws. The frame is now finished!

**Step 4 — Cut the back cover.**

- Place the assembled frame on top of the plexiglass sheet. Use a screw or other sharp object to scratch a mark around the edge of the frame.
- Cut out the back cover with a handsaw, table saw, or the cutting implement of your choice. Line up the cut piece on the back of the frame and drill 6 pilot holes through the back cover material and into the frame itself. It's now ready to be closed up once all the electronics are in place and functioning.

## Step 5 — Wire the power and PS/2 ports.

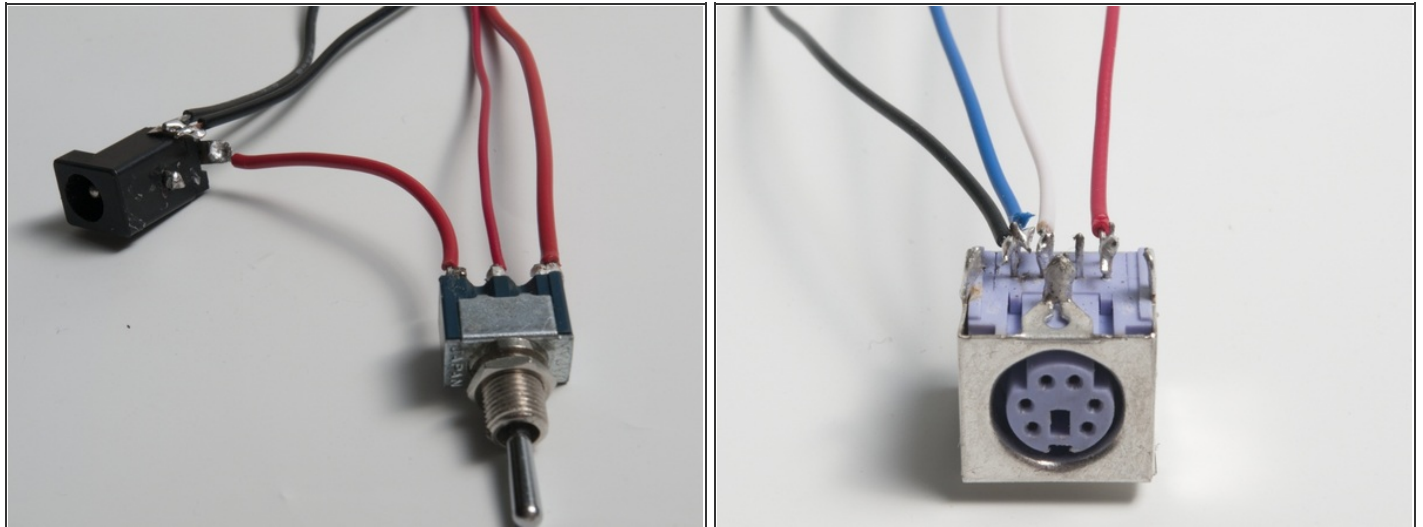


- Wire the 2 battery packs in series by soldering the red (+) lead of one to the black (-) lead of the other.
- Position the battery packs in the frame (I put them at the end opposite the notch). and lengthen the remaining 2 wires if necessary by splicing in stranded wire to let them reach the notch, where you'll connect them to the DC jack and power switch. Insulate connections with electrical tape or heat-shrink tubing.
- **NOTES:** The battery packs need to be removable for battery changes, so you can secure them in the frame using velcro, although the back cover and display panels seem to hold them in place nicely.





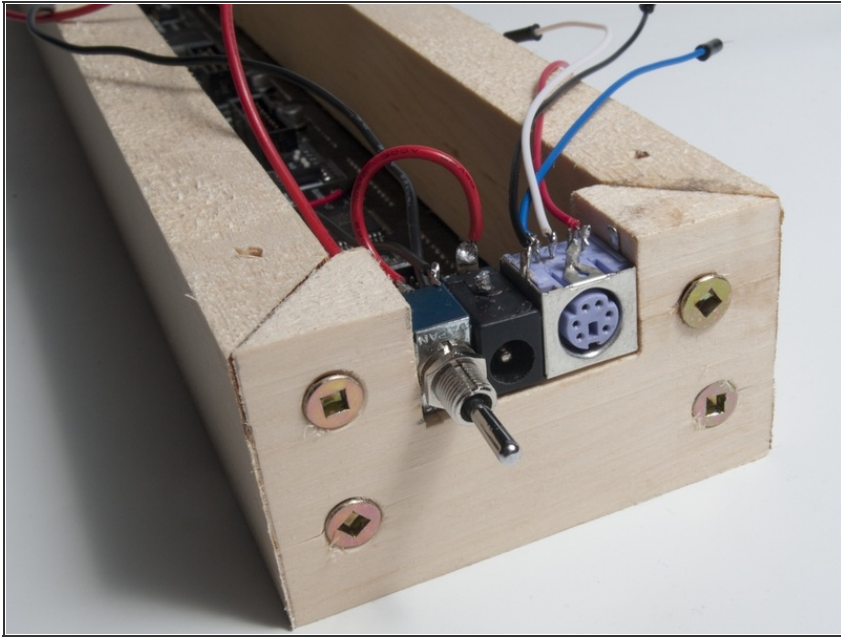
## Step 6



- Cut one end off a black breadboard jumper, and strip and tin the wire. Repeat with a red jumper. Solder the cut end of the black jumper and the black wire from the battery packs onto the DC jack's negative terminal. Solder a short chunk of stranded red wire between the DC jack's positive terminal and one of the outside contacts on the switch. Solder the cut end of the red jumper to the switch's common terminal, and the battery positive to the free outside switch terminal.
- If all went well, your switch should have an off position in the middle, a battery power position to one side, and adapter power on the other.
- Cut one end off 4 more breadboard jumper wires: red, black, blue, and white (or your equivalent). Strip, tin, and solder them onto the positive, negative, data, and clock pins on the PS/2 port (see pin diagram), and use a multimeter set to "continuity" to confirm the pin-wire connections.
- **NOTES:** I used blue for data wires, white for read/write and clock wires, yellow for the display panels' "CS" wires, and red and black for power and ground — although I accidentally switched blue and white here on the PS/2 port. Having a consistent color scheme will make it much easier to figure out what's going on.

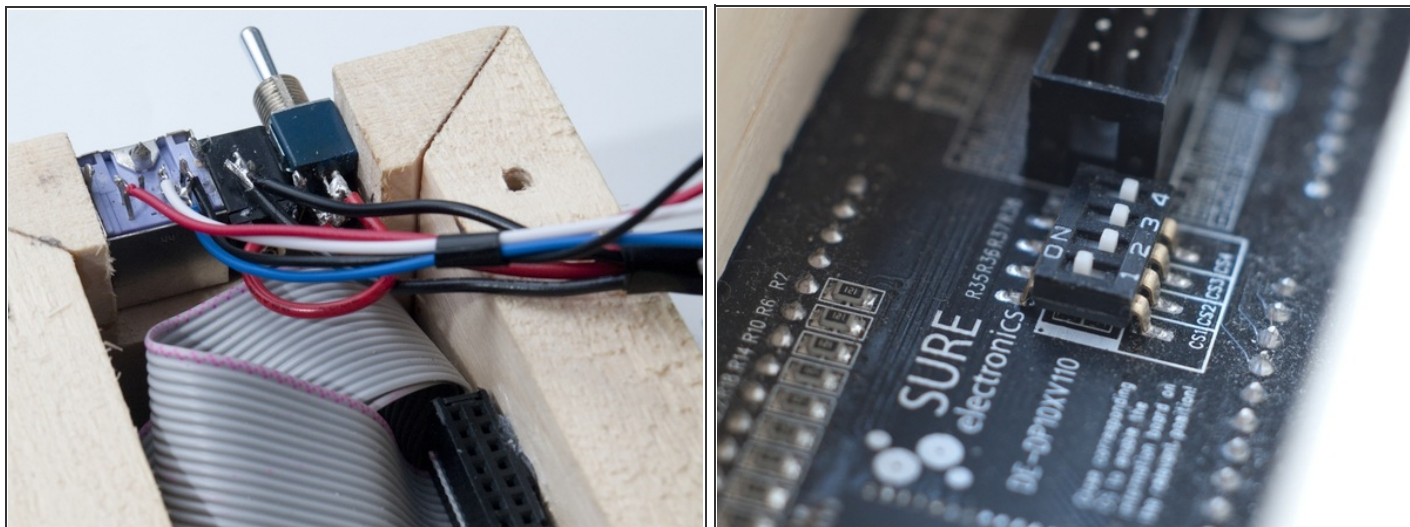


## Step 7



- Dry-fit the PS/2 port, DC jack, and power switch again into the notch at the end of the frame. Heat up a glue gun and dab a bit of glue on each one before quickly pressing it tightly into the notch.
- This is admittedly an unorthodox way of attaching what would normally be PCB or panel-mounted components, but it's very strong and relatively tidy — a good substitute when there's no PCB or mountable panel nearby.

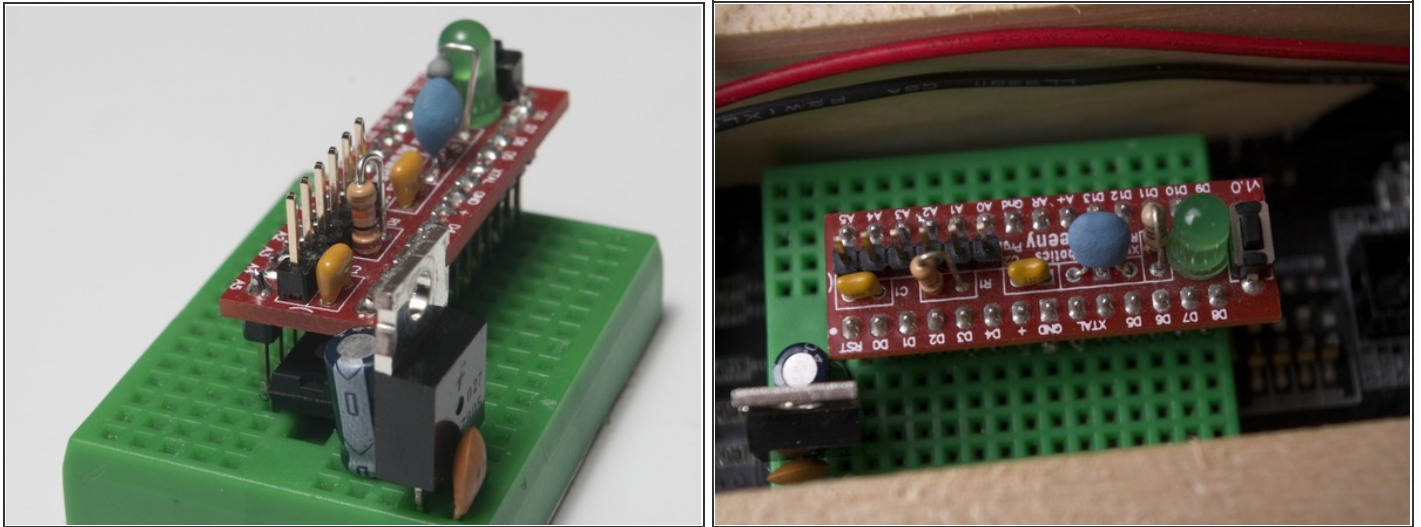
## Step 8 — Configure the display panels.



- Each LED display panel comes with a ribbon cable and has 2 ports on its backside that the cable will connect to.
- Use 2 of the ribbon cables to chain the 3 panels together end-to-end by their adjacent ports. Plug one end of the remaining cable into the port closest to the switch and power jack. Fold this ribbon up tightly and use hot glue to attach its free end plug to an inner side of the frame, with its holes facing up toward the back of the frame.
- Each LED display panel has a block of little DIP switches labeled CS1, CS2, CS3, and CS4. These switches determine how the microcontroller identifies each panel. The PS/2/You code numbers the displays left to right, looking from the front, so turn off all but switch 3 on the panel nearest the notch, all but switch 2 on the middle panel, and all but switch 1 on the panel at the battery end.
- To see what these switches do, set them to some other sequence once you have your display up and running.



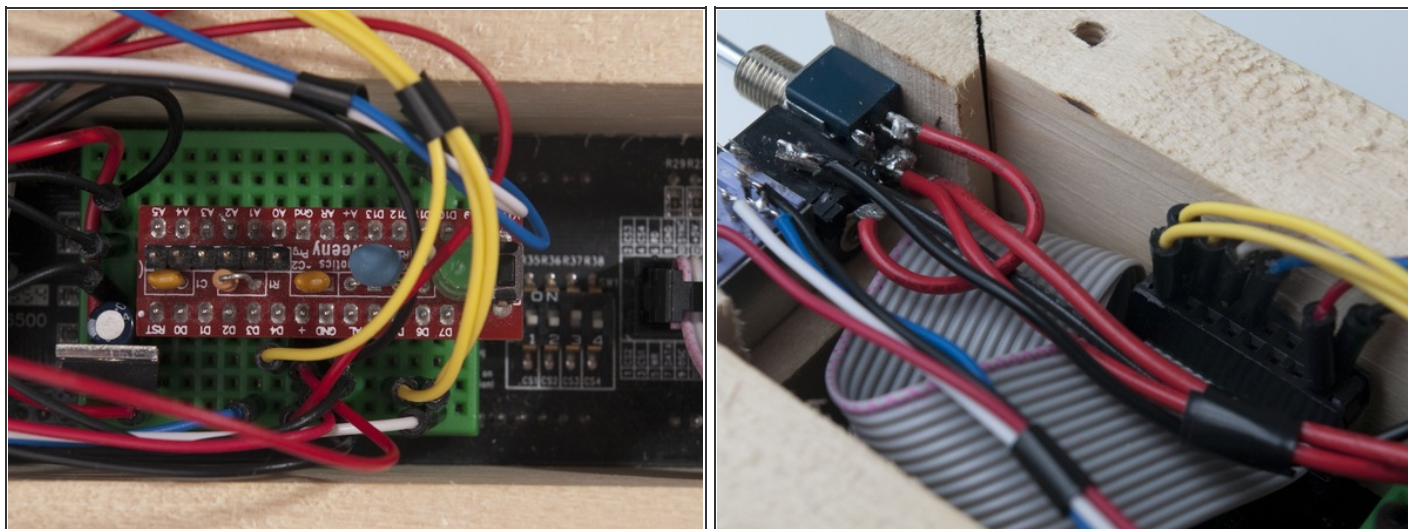
## Step 9 — Add the Ardweeny.



- Plug your Ardweeny into the breadboard straddling the trench, with the green LED near the top, taking up the first 14 rows. Plug the voltage regulator into the bottom 3 rows on one side.
- Plug the 0.1 F capacitor in between the voltage regulator's input and ground legs (typically the sequence is IN-GND-OUT, going left to right looking at the front of the regulator, but check your datasheet to be sure).
- Plug the 10 F capacitor's positive leg in the regulator's output and its negative leg (marked with a stripe) into the regulator's ground bus.
- **NOTES:** The mini breadboard has 17 rows, each consisting of an electrically connected 5-hole bus on either side of a central trench.
- Peel off the breadboard backing (exciting!) and stick it onto the flat, surface-mount Holtek chip on the back of the display panel closest to the switch and ports.

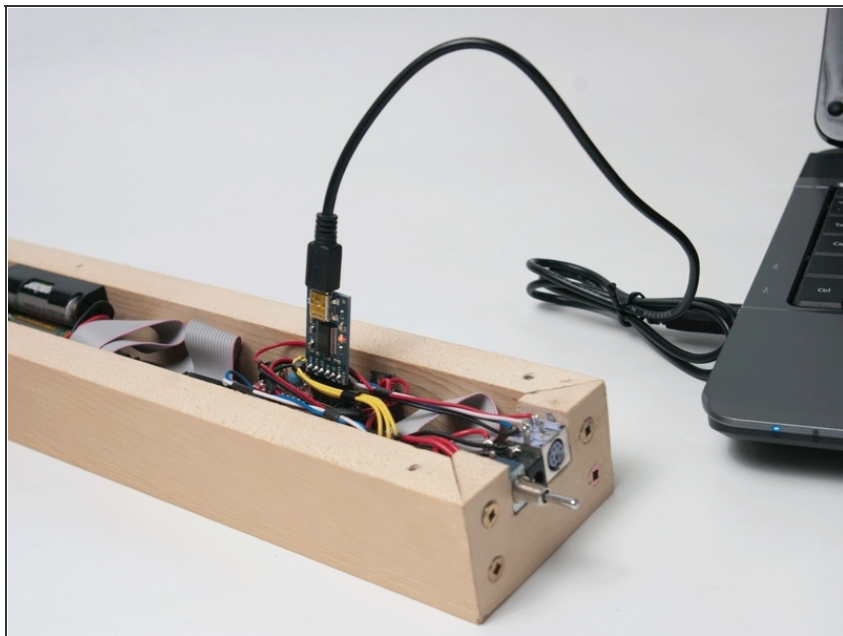


## Step 10



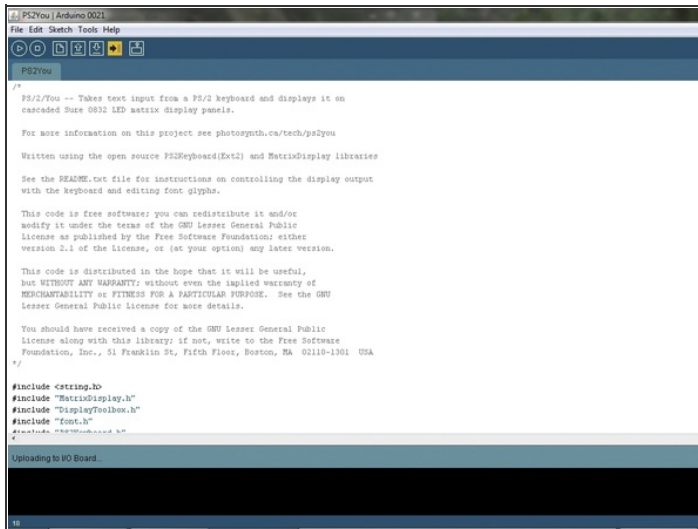
- Since the voltage regulator's ground bus is getting crowded by now, use a small jumper to extend it to an unused bus on the other side of the breadboard.
- Plug the PS/2 port's power wire into the Ardweeny's power and its GND into the Ardweeny's GND bus. Plug the PS/2 port's read/write wire into Ardweeny pin D3 and its data wire into Ardweeny pin D7.
- Use jumpers to connect your components to the ribbon cable plug glued inside the frame. Note that the odd-numbered pins on the 2x8 plug run along the side with the small bump, opposite the side that the ribbon comes in.
- To begin, connect CS2, the first wire on the ribbon (marked in pink) to Ardweeny pin D5. Connect CS3, the second wire, to D6, and ribbon wire 3 (CS1) to Ardweeny D4. For the display's read/write input, connect ribbon wire 5 to Ardweeny D11, and for the data input, connect wire 7 to Ardweeny D10.
- Finally, connect ribbon wire 15 to Ardweeny ground, and ribbon wire 16 to Ardweeny power. (You can also wire power and ground to the buses of the voltage regulator.)

## Step 11 — Program it.



- If you haven't already, download and install the Arduino IDE (integrated development environment) from [arduino.cc/en/Main/Software](https://arduino.cc/en/Main/Software). Launch the IDE.
- To configure it for the Ardweeny, which acts just like an Arduino Duemilanove, click the menu item **Tools** → **Board** → **Arduino Duemilanove** or **Nano w/ATmega328**.
- Then tell it which USB port you'll program the Ardweeny through by clicking **Tools** → **Serial Port** and selecting the highest-numbered COM port (if there's more than one).

## Step 12



```

PS2You [Arduino 001]
File Edit Sketch Tools Help
PS2You
/*
PS2/You -- Takes text input from a PS/2 keyboard and displays it on
cascaded Sure 0933 LED matrix display panels.

For more information on this project see photoguth.ca/tech/ps2you

Written using the open source PS2Keyboard(Mat2) and MatrixDisplay libraries

See the README.txt file for instructions on controlling the display output
with the keyboard and editing font glyphs.

This code is free software: you can redistribute it and/or
modify it under the terms of the GNU Lesser General Public
License as published by the Free Software Foundation; either
version 2.1 of the license, or (at your option) any later version.

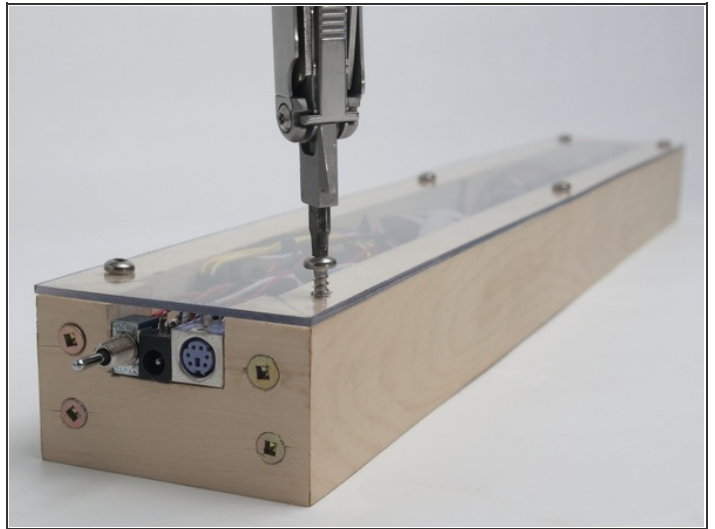
This code is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public
License along with this library; if not, write to the Free Software
Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA
*/

#include <string.h>
#include "MatrixDisplay.h"
#include "DisplayToolbox.h"
#include "font.h"
#define PS2_KEYBOARD_M *PS2Keyboard.h

Uploading to I/O Board.

```



- Download the code package [PS2You\\_code.zip](#) and unzip it.
- Move the PS2Keyboard and MatrixDisplay folders into your Arduino libraries directory.
- Restart the Arduino IDE and open up the code file PS2You.pde. Connect your computer to the Ardweeny with the FTDI programming adapter. Click Verify and Upload, and if all is well, a moment later the display will light up with the default text.
- Unplug the programmer, load some batteries into the battery packs (if you're going to be using battery power), and screw on the back cover. You're all set!

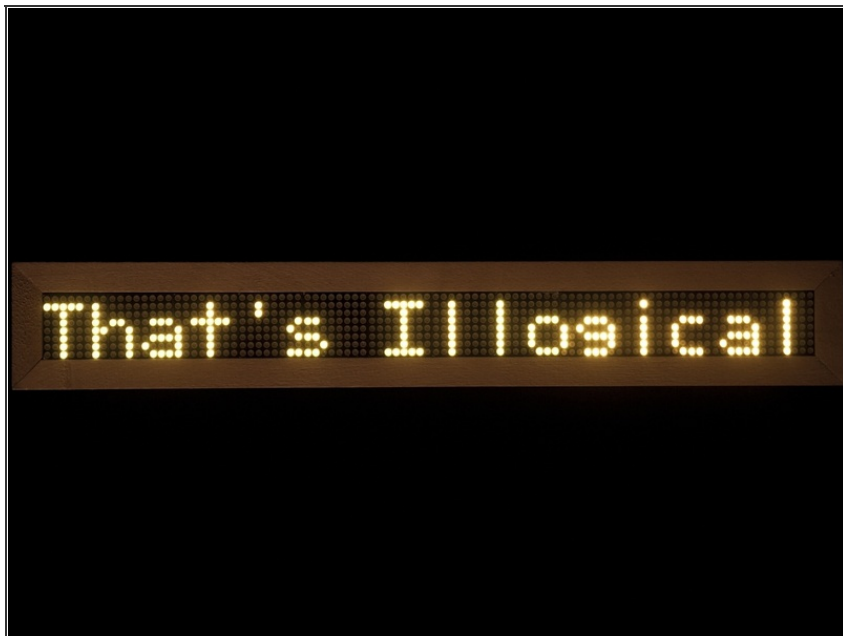
- **UPDATE:** If you are trying to make the PS/2/You with the new version of the LED panels, all you need to do is change line 131 of the MatrixDisplay.cpp file:



Original code: `writeDataBE(8,HT1632_CMD_COMS10,true);` New code for ht1632c LED panels: `writeDataBE(8,HT1632_CMD_COMS00,true);`

- **UPDATE 2:** If you are using the newest version of the Arduino IDE (1.0), you will need to edit MatrixDisplay.h, DisplayToolbox.h, and PS2Keyboard to replace `#include <WProgram.h>` and `#include <wiring.h>` with `#include <Arduino.h>`. In PS2You.pde/.ino, you will also need to change line 141 from `textLines[currentLine] += c;` to `textLines[currentLine] += char(c);`.

## Step 13 — Input and Output Modes



- The PS/2/You has an Input mode for entering text and an Output mode for displaying it. Pressing any alphanumeric key puts the system into Input mode, and hitting Enter gets you back to Output mode.
- In Input mode, the PS/2/You displays a single line of text as you type it in, a maximum of 100 characters long. You can store up to 6 lines of text (this number is settable by changing the value of numLines near the beginning of PS2You.pde). Use the up and down arrow keys to select which line to edit, and backspace over any line of text or hit Escape to delete it.
- In Output mode, the display loops through the stored text lines on its own, displaying each for one second, or if the line is longer than 16 characters, it will scroll across the display before moving onto the next line. If only one line of text is stored, it displays continuously.



## Step 14 — Text Messaging



- The uses for this contraption are many. Plug the keyboard in and enjoy putting your wittiest “wiseclacks” on it in the safety of your home, shop, or office — or use the battery option to take it into the wide world. We like to have the keyboard accessible so that passersby can add a riposte or two to the dialogue. But if monologue is more your thing, you can always keep the keyboard out of reach.
- Add a dowel as a removable handle so you can wander the streets digitally promoting your geekified political leanings. Keep score (or heckle) at sporting events, deliver birthday greetings, advertise your wares at a farmer’s market, beam cryptic messages to your neighbors — the possibilities are endless!

## Step 15 — Further Illuminations



- There's plenty of room for improvement to the code. Try using Ctrl and other keys to modify how the text displays: flashing, sliding in from the top, fading in, or other effects.
- Four display panels can be cascaded together for a longer display, and Sure Electronics sells an identically programmable 8×32 panel with 5mm instead of 3mm LEDs, so a jumbo PS/2/You is almost inevitable.

## Step 16 — Roll your own glyphs.

- The display font is defined by hexadecimal values in the font.h file. It's not user-friendly for editing, but Brent Morse has made a free applet that lets you design your own 5×7 LED display glyphs ([morse-code.com/id89.htm](http://morse-code.com/id89.htm)).
- Use it to modify the font, or make custom smilies or any other pattern you like.

This project first appeared in [MAKE Volume 27](#), page 92.

This document was last generated on 2012-10-31 01:30:33 PM.