



Pi Wedge B+ Hookup Guide

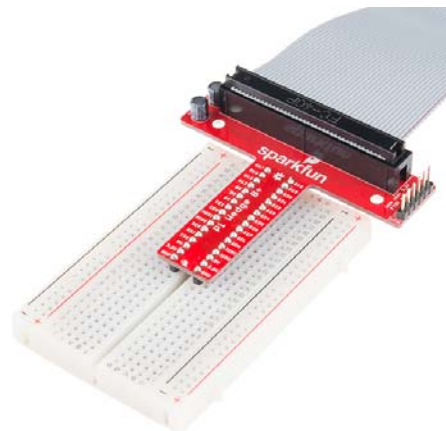
Introduction

The Raspberry Pi B+ is the latest revision of the Raspberry Pi Linux Computer Board. Differentiating it from the older Model B, the B+ features an extended GPIO connector, with 40 pins, up from the Model B's 26.



The Raspberry Pi B+

To match the rearranged connector, we're introducing the Pi Wedge B+, a small board that connects to the GPIO connector, and breaks the GPIO pins out into rows that work with solderless breadboards.



Pi Wedge B+ and Breadboard

If you're using a regular Model B (not the plus version), we also have a Pi Wedge for it.

This guide will show you how to assemble the Pi Wedge B+ and start using it with your Raspberry Pi.

Required Materials

- The Pi Wedge B+ kit
- A Raspberry Pi B+ single board computer.
- A Solderless Breadboard
- An FTDI Basic
- Your favorite I2C, SPI or GPIO-based peripherals
- A soldering iron and leaded or lead-free solder

Suggested Reading

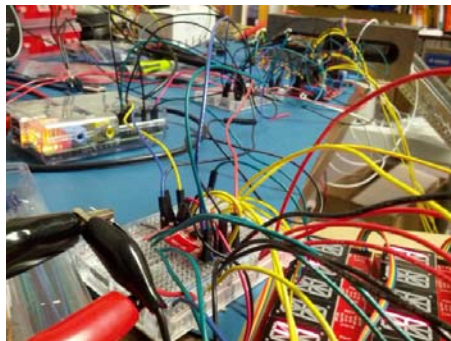
- How To Use a Breadboard
- Setting Up a Boot Card
- Serial Communication
- Serial Peripheral Interface
- I2C

Suggested Viewing

- Getting Started With The Raspberry Pi
 - Part 1
 - Part 2
 - Part 3

Background

In the process of developing projects like the Twitter Monitor and Great American Tweet Race around the Raspberry Pi, we found that we were experiencing some growing pains when trying to expand the Pi into a prototype that involved external hardware.



There's a Pi somewhere in this ratsnest

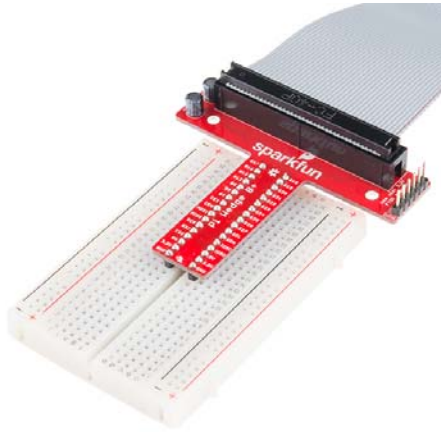
The Raspberry Pi Model B+ has a 40-pin connector that provides access to several communication interfaces, plus GPIO and power. But the connector doesn't have detailed labeling, and the native pin arrangement is somewhat scattershot. Pins used for similar functions aren't always grouped together, and power and ground pins are interspersed with no obvious pattern.

The pins also don't translate to a solderless breadboard very easily. Our first projects used a bunch of F-M jumper wires that we just plugged into the header. They involved a lot of "ratsnest jiggling" when things stopped working.

Bootstrapping

In addition to the physical issues of using the I/O connector, getting started with a brand new Raspberry Pi B+ always seems to involve a chicken-and-egg situation. We just want to SSH into it, so we can use the command line. But in order to SSH to it, we need to know it's IP address...and of course, the IP address is most easily learned by running `ifconfig` on the command line.

The Solution



Meet the Pi Wedge B+

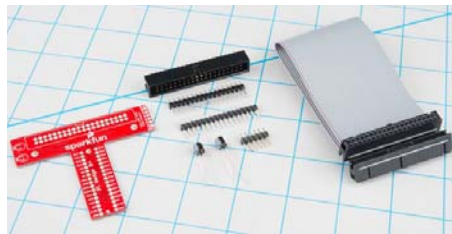
The Pi Wedge B+ connects to the 40-pin GPIO connector, and breaks out the pins in a breadboard-friendly arrangement and spacing. It adds a pair of decoupling capacitors on the power supply lines, and it makes the initial bringup process easier - you can plug an FTDI Basic module into the serial port.

Assembly

The Pi Wedge B+ comes to you as loose parts, which you need to assemble before you can use it. The following steps will guide you through that process.

Kit Contents

Let's start by reviewing the contents of the Pi Wedge B+ kit.



Pi Wedge parts

You should have the following parts.

- Pi Wedge B+ PCB
- 40-pin shrouded header
- Two 17-pin straight headers
- Two 10 uF 25V electrolytic capacitors
- 6-pin straight header
- Ribbon cable

Instructions

Assembly of the Pi Wedge B+ should be mostly obvious and straightforward. One thing to notice is that components are soldered to both sides of the board. The headers that interface with the breadboard are on the bottom of the board, soldered from the top. The other components go on the top, and are soldered from the bottom.

Pin Alignment

To keep the breadboard pins properly spaced and aligned while you solder, you can use your breadboard as an assembly jig. Simply insert the 17-pin headers in the breadboard...



Pins in breadboard

Then place the PCB over them, and solder it in place. Notice that the PCB is placed over the pins so that the signal labels are facing up.

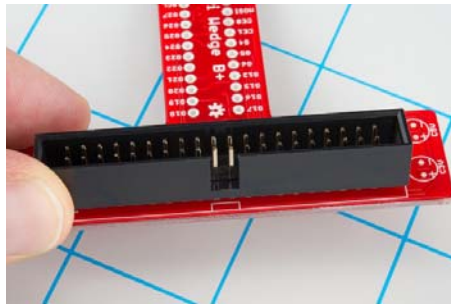


Add the PCB

Be careful not to overheat and melt the breadboard while soldering the pins. Zig-zagging between the rows will help distribute the heat. Also, somewhat counter-intuitively, using a hotter setting on your soldering iron will allow you to work more quickly, minimizing the risk of melting the breadboard.

Shrouded Header

The shrouded header has a notch that keeps the ribbon cable properly oriented. The notch should be lined up with the corresponding rectangle in the PCB silkscreen.



Header Alignment

Capacitors

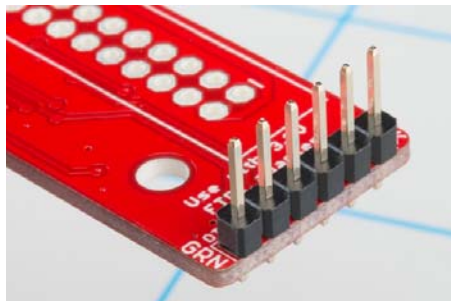
The electrolytic capacitors are polarized. The body has a band of “-” signs near the negative lead, which should be inserted into the PCB hole marked with the - sign. The negative lead is also shorter than the positive lead.



Capacitor Polarity

Serial Adapter Connector

The 6-pin header is soldered onto the PCB across from the capacitors.



Serial Port Connector

If you want a lower profile assembly, you can substitute a right-angle header for the straight pins in the kit.

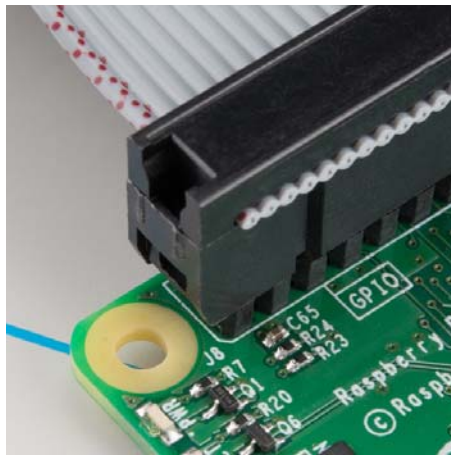
Connection

With the wedge soldered together, you're ready to attach it to your Pi B+, and test it out.

The ribbon cable is polarized. On the Pi Wedge B+, the tooth on the cable will interface with the notch in the shrouded header.



The header on the Pi B+ itself doesn't have anything to help guarantee the alignment. You'll need to take care that it gets connected properly. Pin 1 on the Pi is marked with a dog-eared corner on the silkscreened rectangle. The ribbon cable connector is embossed with (a barely visible) small triangle that marks pin 1. The first pin is also coded on the wire, such as the red markings in the photo below (though it may also be another color, such as black or dark blue).



Proper pin-1 orientation

The FTDI connector also needs to be aligned correctly. Be sure to match up the "grn" and "blk" markings on both boards.



Proper FTDI-basic orientation

In the next section, we'll explore how the signals from the Pi are mapped to the Wedge.

Pin Mapping

Changes With the B+

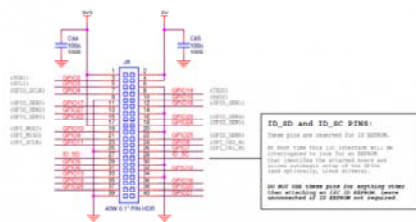
The Raspberry Pi foundation introduced a number of changes with the B+. The changes include

- 40 pin GPIO connector in place of the B's 26 pin connector. It adds
 - Nine more GPIO pins
 - ID_SC and ID_SD pins to identify external peripherals
- Regulated audio power supply
- 4 USB ports
- Micro SD slot, replacing the full-size SD slot on the B.

An Extra I2C bus?

As part of the B+ improvements, the Raspberry Pi Foundation has standardized the specification for add-on boards, in what they call the "Hardware Added On Top" (HAT) specification. It standardizes the physical form factor for add-on boards, and includes a provision for the B+ to automatically identify and initialize HATs at startup. It uses an I2C bus to read a description from an EEPROM on the HAT, similar to cape identification on the Beagle Bone Black.

This I2C bus is implemented with the ID_SC and ID_SD pins (pins 27 and 28 of the 40-pin connector) - but before you get too excited about adding peripherals on that bus, observe the note in the schematic for that port.



This is further clarified in the HAT design guide

On a Model B+, GPIO0 (ID_SD) and GPIO1 (ID_SC) will be switched to ALT0 (I2C-0) mode and probed for an EEPROM. These pins will revert to inputs once the probe sequence has completed.

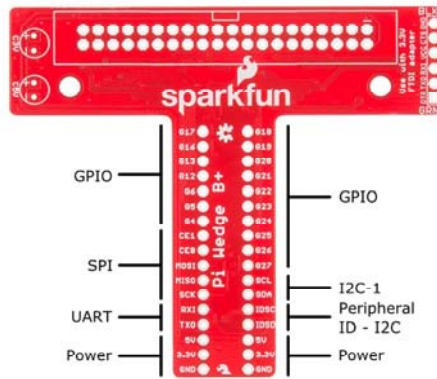
The only allowed connections to the ID_ pins are an ID EEPROM plus 3.9K pull up resistors. Do not connect anything else to these pins!

So if you want I2C on the B+, you'll need to use I2C-1, on pins 3 and 5 of the 40-pin connector, marked SDA and SCL on the Pi Wedge B+.

The HAT specifications and other information are hosted on GitHub. If you're designing a HAT, you'll want to start by reading the HAT Design Guide, and possibly perusing the B+ addons forum.

Signal Location

The Pi Wedge reorganizes the I/O pins on the Pi, putting similar functions on adjacent pins. The SPI, I2C and UART signals are all grouped near each other.



Functional Groupings

The pins are labeled, though the labels are short to fit the space available on the PCB. The UART, SPI and I2C pins are marked with their communication bus functions, but they can also be available as GPIO pins when configured in that mode.

The following table denotes the assignment of signals on the Pi Wedge, including the peripheral and alternate GPIO assignments where appropriate.

Function	GPIO#	Function	GPIO#
	GPIO 17		GPIO 18
	GPIO 16		GPIO 19
	GPIO 13		GPIO 20
	GPIO 12		GPIO 21
	GPIO 6		GPIO 22
	GPIO 5		GPIO 23
	GPIO 4		GPIO 24
SPI CE 1	GPIO 7		GPIO 25
SPI CE 0	GPIO 8		GPIO 26
SPI MOSI	GPIO 10		GPIO 27
SPI MISO	GPIO 9	SCL	GPIO 3
SPI CLK	GPIO 11	SDA	GPIO 2
UART RXI	GPIO 15	ID SC	GPIO 0
UART TXI	GPIO 14	ID SD	GPIO 1
5V			5V
3.3V			3.3V
GROUND			GROUND

Pi Wedge B+ Pin-Function mapping

Power And Logic Levels

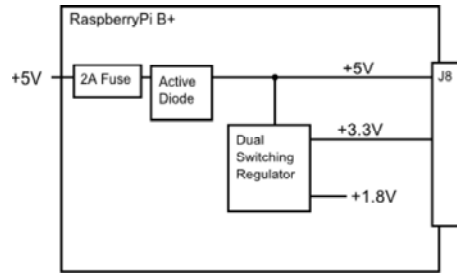
Understanding the Pi's power supply is critical to using it successfully.

The Raspberry Pi B+ is more efficient than its predecessors, as it replaces the former chain of linear power regulators with switching regulators.

Consulting the Raspberry Pi B+ power schematic, we see that 5V comes into the board via connector J1 - it's a micro USB connector, but only the power and ground pins are connected. The 5V coming from this connector passes through a fuse and a transistor circuit that protects against power polarity mishaps, then continues around the board without any further regulation. The 5V connections on the Pi Wedge come straight from this line.

On the B+, the 5V goes to a dual switching regulator that further reduces it

to 3.3V, and 1.8V. The regulated 3.3V is present on the I/O connector.

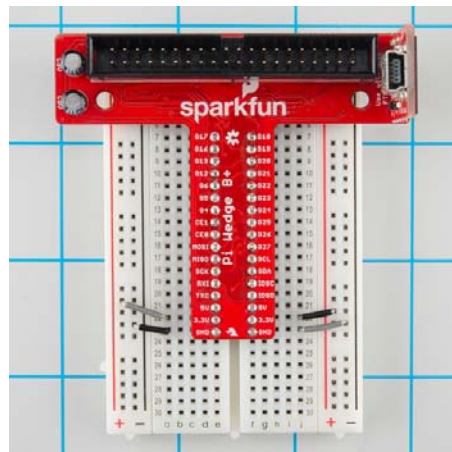


There are several power strategies that can be applied in a Pi deployment, depending on the overall needs and availability.

Power Through the B+ GPIO Connector

The most obvious strategy is to power external circuitry directly from the B+, using the 5V and 3.3V from the GPIO connector.

To power small circuits on your breadboard, you can run jumpers from the 5V or 3.3V and Ground pins on the wedge to the power rails on the breadboard.



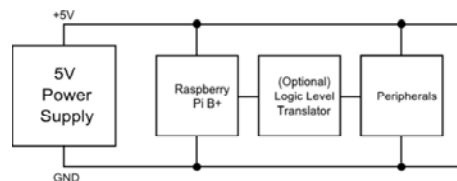
Jumping power to the breadboard rails

While this is the most immediate way to access power, it only extends to small circuits. The B+ itself is limited to 2A total from the 5V line, most of which is needed by the B+ itself. The stated limit for the 3.3V pins is 50 mA.

If you're developing external circuitry, and the B+ resets when you're testing it, you may be exceeding the current limits. We saw this exact situation arise as we added SPI controlled 7-segment LED displays - if we illuminated one too many segments, the system crashed. For circuits with higher power draw, we'll need to explore some alternatives.

Daisy Chaining

The next power option is to connect each section of the circuit directly to the power supply. This means that the peripherals aren't constrained by the current limits of the fuses and regulators on the Pi itself.



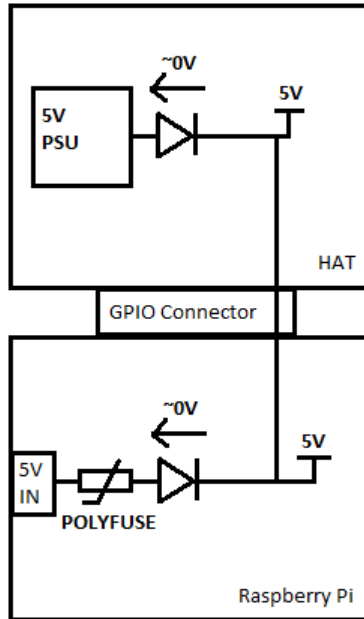
The peripherals are powered directly by the supply directly

For peripherals that use 5V logic, they should also include 3.3V/5V logic

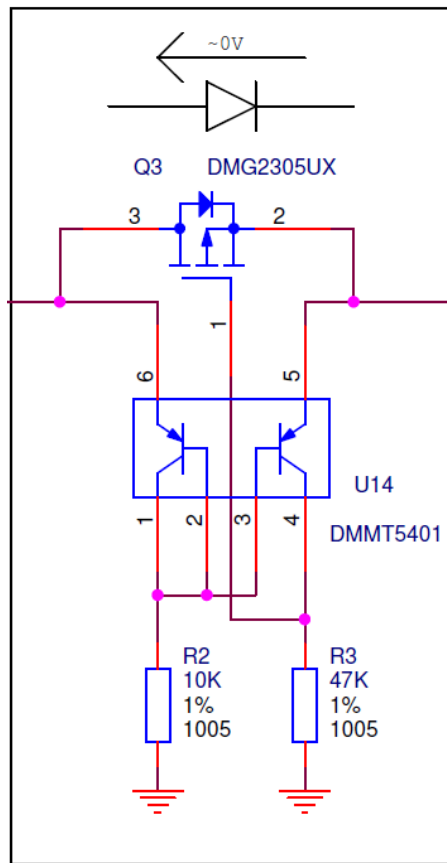
level translation.

Back Power Through J8

As described above, a simple deployment can power peripherals via the 5V and 3.3V pins of J8. But it's also possible to apply power to the Pi via those lines. The Pi Foundation call this "back powering", and they have a number of recommendations for it's implementation.



The first recommendation is to duplicate the fuse and MOSFET + BJT power protection circuit as seen on the Pi itself. This circuit is a variant on the "ideal diode" circuit.



It serves several purposes:

- Under ordinary circumstances, with power coming in via the micro-USB plug, the MOSFET is biased fully on, so there is only minimal voltage drop across it, where a typical Schottky or Silicon diode would drop 0.3V or more.
- Second, it prevents power from flowing if the power polarity at the micro-USB plug is incorrect.
- Third, if the board is powered via J8, it prevents power from being drawn from J1, to prevent contention if two supplies are present at the same time.

The other recommendation is that the HAT needs to be able to provide 5V, +/- 5%, with at least 1.3 A available for the B+.

Logic Levels

The Pi uses 3.3V logic levels, which are not 5V tolerant. Many peripheral devices are capable of running at 3.3V, but in the case that you need to interface with 5V devices, use a level shifter, such as the TXB0104 breakout.

Communications

Finally, the signals on the 6-pin FTDI header are also limited to 3.3v logic levels. Be sure to use it with a 3.3V FTDI module, and not a 5V one.

Software

With the many programming languages available on the Pi, it's hard to know where to start writing software that interfaces with the I/O connector.

For a quick and easy way to check out the connections on the IO connector, check out the WiringPi package. Wiring Pi is a hardware interface library that includes a utility to help you configure and check the status of the GPIO connector pins.

Follow these instructions to install Wiring Pi.

Once you've got it installed, you can use the `gpio` command line utility to quickly check and test hardware connections.

`gpio` takes a number of arguments. The `readall` option prints a handy table that deciphers the configuration and assignment of the pins. It also detects the revision of the Pi, and translates the pin numbers to match.

```

pi@raspberrypi:~$ gpio readall
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
+-----+-----+
| BCM | wPi | Name | Mode | V | Physical | V | Mode | Nam
e | wPi | BCM |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
+-----+-----+
| | | 3.3v | | | 1 || 2 | | | 5
v | | |
| 2 | 8 | SDA.1 | ALT0 | 1 | 3 || 4 | | | 5
V | | |
| 3 | 9 | SCL.1 | ALT0 | 1 | 5 || 6 | | | 0
v | | |
| 4 | 7 | GPIO. 7 | IN | 0 | 7 || 8 | 1 | ALT0 | Tx
D | 15 | 14 |
| | | 0v | | | 9 || 10 | 1 | ALT0 | Rx
D | 16 | 15 |
| 17 | 0 | GPIO. 0 | IN | 0 | 11 || 12 | 0 | IN | GPI
O. 1 | 1 | 18 |
| 27 | 2 | GPIO. 2 | IN | 0 | 13 || 14 | | | 0
v | | |
| 22 | 3 | GPIO. 3 | IN | 0 | 15 || 16 | 0 | IN | GPI
O. 4 | 4 | 23 |
| | | 3.3v | | | 17 || 18 | 0 | IN | GPI
O. 5 | 5 | 24 |
| 10 | 12 | MOSI | ALT0 | 0 | 19 || 20 | | | 0
v | | |
| 9 | 13 | MISO | ALT0 | 0 | 21 || 22 | 0 | IN | GPI
O. 6 | 6 | 25 |
| 11 | 14 | SCLK | ALT0 | 0 | 23 || 24 | 1 | ALT0 | CE
0 | 10 | 8 |
| | | 0v | | | 25 || 26 | 1 | ALT0 | CE
1 | 11 | 7 |
| 0 | 30 | SDA.0 | IN | 0 | 27 || 28 | 0 | IN | SCL.
0 | 31 | 1 |
| 5 | 21 | GPIO.21 | IN | 0 | 29 || 30 | | | 0
v | | |
| 6 | 22 | GPIO.22 | IN | 0 | 31 || 32 | 0 | IN | GPIO.
26 | 26 | 12 |
| 13 | 23 | GPIO.23 | IN | 0 | 33 || 34 | | | 0
v | | |
| 19 | 24 | GPIO.24 | IN | 0 | 35 || 36 | 1 | OUT | GPIO.
27 | 27 | 16 |
| 26 | 25 | GPIO.25 | IN | 0 | 37 || 38 | 0 | IN | GPIO.
28 | 28 | 20 |
| | | 0v | | | 39 || 40 | 0 | IN | GPIO.
29 | 29 | 21 |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
+-----+-----+
| BCM | wPi | Name | Mode | V | Physical | V | Mode | Nam
e | wPi | BCM |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
+-----+-----+

```

Somewhat confusingly, you'll notice that there are three different ways to refer to the GPIO pins. Starting from the center of this table, we see:

- The "Physical" columns that correspond to the pin numbers of the J8 connector on the B+.
- Moving outward from there, the "mode" and "name" columns indicate the presently configured pin function and direction.
- The next column out, "wPi," is a set of contiguous numbers (0 to 31) that WiringPi uses by default (though it can be configured to use other numbering schemes, too - see the GPIO man pages).

- Finally the “BCM” column refers to channels of the GPIO peripheral on the BCM2835 processor.

The Pi Wedge is labeled with the numbers seen in the Name column.

If you run `gpio` without any parameters, it displays the available options. You can use it to configure, read and write pins right from the command line. Similarly, `man gpio` displays more detailed information on using the utility, including a number of sample commands.

Programming

If you're coming from Arduino, and you've got a handle on how it manages I/O, you'll find that the WiringPi libraries present a familiar programming interface. We have a few WiringPi based examples for GPIO, SPI and I2C in the Pi Wedge GitHub Repository that should help you get started.

If Python is more your speed, then take a look at the RPi.GPIO module.

For more detailed information about using both WiringPi and RPi.GPIO, take a look at our Raspberry gPiO tutorial.

Resources & Going Further

Going Further

To take a closer look at programming for the I/O on a Pi, in both Python and C, take a look at our Raspberry gPiO tutorial.

The design files for the PCB, and some WiringPi software examples can be found in the Pi Wedge B Plus GitHub repository.

Resources

For more information about the Raspberry Pi B+ and the software described here, please visit their sites.

- The Raspberry Pi Foundation
- The Pi Foundation's B+ Addons forum.
- The Pi Foundation's GitHub repository for the Raspberry Pi B+ HATs.
- The eLinux.org Raspberry Pi peripherals guide
- WiringPi
- RPi.GPIO module
- Some notes about increasing the available current from the B+ USB ports.

If you have any problems or questions, our technical support department can help. Please don't hesitate to contact us. We also love to hear about your projects!