PhorsePOV!

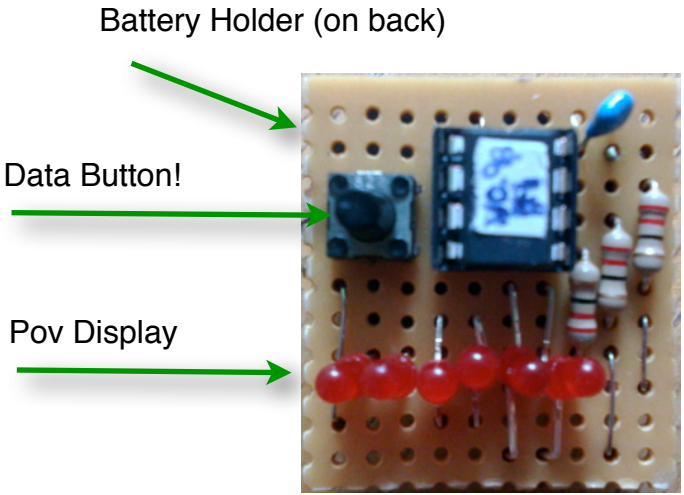
Introduction

PhorsePOV is an easy-to-build scrolling **P**ersistance **O**f **V**ision gadget which uses Phorse code for data entry, and can store messages up to 32 characters in its EEPROM.

The PhorsePOV

Here's a prototype with a battery holder on the back fitted with a Lithium battery, disconnected via a piece of card!

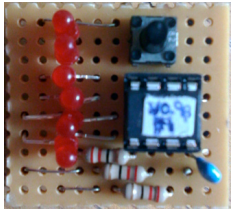
When you switch on the PhorsePOV by removing the card, the LED nearest the button will light up - this is the *data entry mode*.



If you hold down the button for at least 2s you'll get the *POV display mode* showing the current POV message.

How To Make The Display Work

When a POV is displaying a message it usually looks like the LEDs are simply flickering madly. Actually, it's displaying a message as a banner. If you hold it horizontally and then wave it from left to right you'll see the dots separating out.



You need to swing the PhorsePOV as near as possible to 4Hz. If the message seems back-to-front, a quick shake will usually fix it. If the message starts shunting to the right you need to slow down slightly and if it shunts to the left you need to speed up slightly. A few minutes practice should enable you to make it readable!

To turn it off again, re-insert the card into the battery holder as before.

How To Enter Data

Holding down the button for >2s puts the PhorsePOV into data mode from display mode. You can enter characters in a kind of 'morse-code' called PhorseCode.

PhorseCode is an ‘Morse-code’ system which uses a single button to enter data, but organises the dot-dash patterns for a a mobile phone keypad layout to make it easy to learn. Each letter is a 3-bit code followed by nothing, dot or dash.

Some codes are commands like ‘del’, ‘clr’, ‘num’, ‘alpha’ and ‘rst’.

The first thing you’ll need to do is clear the old message. You can do this by typing the clr command:

....- (4 dots and a dash).

Alpha Mode

Number Mode

A dot is entered by pressing the button quickly (<0.15s) and a dash is entered by holding down the key for more than 0.15s, but <2s. PhorsePOV thinks you’ve finished a character if you don’t press a key for more than 0.15s.

As you type the dots, the code is displayed on the LEDs: ‘off’ for ‘.’ and ‘on’ for ‘-’; with a ‘.’ at the end. The pattern starts with a single dot at the button end. So, ‘....-’ will cause the initial dot to be pushed up the LEDs until it gets almost to the resistors and then you tap the final dash.

You can check if you’ve cleared the data by going into display mode and you should see just a single LED flicking at the bottom every 0.25s or so. OK, go back to data mode and now you can enter a message. Let’s try ‘Hello’.

‘H’ is -...

‘E’ is .--.

‘L’ is -.-- (you need to do this twice)

‘O’ is --.-

OK, you can try to display this now by going back to display mode and waving your hand!

You can delete the last character with the command code: “...-.” and restore the original message with “...--”. *Note: I’m still working on improving the visual Syncronisation!*

Building The Phorse POV

This tutorial is for the Veroboard version of the POV. You'll need the following tools:

Tools

- A standard, but fine-tipped soldering Iron. You can get these from Maplin (in the UK) or any electronics shop. A 25w type soldering Iron will work fine. A temperature adjustable iron will be better.
- A wire clipper to clip wires.
- A small pair of pliers - they are really useful for positioning components and bending wires.
- A Spot-face cutter for breaking strips on veroboard or stripboard.
- A cutting tool for cutting PCBs and other materials. I use a hobby knife set.
- A solder sucker is very useful, it'll be almost impossible to do a project without one.
- A simple multi-meter. You'll need to check your connections.

Electronic (in this case Veroboard) projects also need a number of additional materials in general:

- Veroboard or stripboard - you can get this from Maplin. You'll need veroboard with at least 11 strips and 10 columns. Try to pick Stripboard that isn't based on grey fibreglass, but brownish plastic-ish material; it's much easier to work with.

- Solder. In the UK this is pronounced "SoLL - Der", but in the US they seem to pronounce it as "Sodder"! Anyway, I digress. These days you'll use sliver-based Solder. Pick the stuff that has Ag, Sn and Cu in it; the slightly cheaper stuff is much harder to work with (this stuff isn't as easy as traditional Lead solder though). This stuff has a flux core (what they call 'Resin' in the US).

- Wire. I use 0.375mm 28swg bare wire for my projects rather than plastic coated wire, because it takes less space. I have to be careful I don't unintentionally short contacts. A reel of wire will last a long time!

Components
Finally, we get to the components you'll need for this project. I used:

- An Atmel AtTiny25 Microcontroller. If you don't have the means to program these from scratch, then I suggest you order a pre-programmed one from me (along with an entire PhorsePov kit). Otherwise, they can be easily obtained from a number of electronic shops, I bought mine from www.rapidonline.co.uk - right now they cost £1.15 in quantities of 1.
- An 8-pin DIL socket. These cost £1.23 for a tube of 60 from rapid. You probably don't want to buy a tube. You can get smaller quantities from Maplin.
- 6x low power 3mm LEDs. I bought a lucky bag of 10 low-power 3mm LEDs from Maplin, but you can also get them from www.rapidonline.co.uk for about 9p each!
- 3x 22Ω Resistors. Mine came in a general pack of loads of resistors but you can buy 3x 20R resistors from www.rapidonline.co.uk for 17p each.
- A 104 Capacitor (0.1μF). 2p?
- A push-button switch. These are called tactile switches in www.rapidonline.co.uk's catalogue. The cost is 29p / switch - you need 1. I picked a switch which protrudes up so that it clears the IC.

- A 2032 coin cell contact. Again, you can get these from www.rapidonline.co.uk for 13p (it's a good shop!).

- A CR2032 coin cell. These cost 65p or 88p from rapid.

The total cost of the components will be about £5.50. If you order from Rapid, there will be postage (≈£5) and VAT to be added taking it up to £12 or so. However, I sell it at £10.06 (a 60% markup given quantities of 10 giving me a 40% profit).

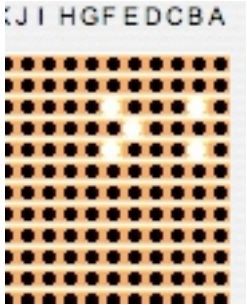
Construction

http://www.kpsec.freeuk.com/ provides some general soldering and veroboard advice:

The general advice is to construct a PCB from the low-profile features and components up to the high-profile components.

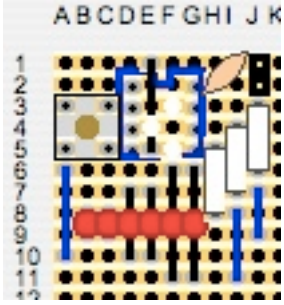
1. Cut the Veroboard to shape. You'll need a board 10x11 holes which means you need to cut down along the 11th column and across along the 12th row. Cutting along the holes is easiest with Veroboard. Use the hobby cutter to do this and be patient, cutting several times (about 10x) along each axis on both sides. If you don't mind wasting some Veroboard it'll be best to cut an entire strip on one axis and then cut along the other axis, otherwise the board may crack at the corner where both meet.

2. Cut the correct tracks. Go to the track-side of the stripboard and use the spot-face cutter to break some of the strips. The grid to break is shown on the right, and there's only 5 holes to break. Count the columns from the right hand side.



3. Solder the wire links. Go back to the front-side (the side without strips on it). You need to solder 4 wires: A6-A10; I8-I11, J7-J9 and E5-E1. I first clip a piece of wire that's a bit longer than I need. Then I bend the wires into shape using my pliers, poking the ends through to the track side. Then I solder the ends and then I clip the ends.

4. Solder the resistors. They go from H5-H8, I4-I7 and J3-J6. They are all the same. I use a similar technique, I bend the ends so they correctly fit in the holes; push them through and then solder and then clip the wires.

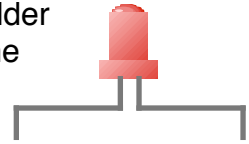


5. Solder the DIP socket. The notch should be over E2, E4. This is fairly easy, just carefully place the socket in the Veroboard and make sure all the pins go through. Then solder in all the pins.

6. Solder the tactile switch. You'll need to check this first though. There are 4 contacts on the switch. Using the multimeter, find two adjacent contacts which don't connect when the switch isn't pressed, but do connect when it is. Then place the switch so that these contacts are at C3 and C5. Then solder in the switch. And clip the ends.

7. Solder the Capacitor. The cap fits over I1 and H2; it smooths out the power input.

8. Solder the LEDs. This is probably the trickest bit since the LEDs must be placed with alternate orientations and you have to bend the leads to the right length. An LED usually has a long lead and a short lead. We want LEDs in columns B,C,D,E,F,G. Leds in columns C, E, G must be fitted with the short lead at the top and the long lead at the bottom. Leds in columns B, D, F must be fitted with the short lead at the bottom and the long lead at the top. Solder them in the order G,F,E..B. To solder LEDs F and G, bend both the legs out a few mm from where they meet the plastic casing. Then place the LED between rows 8 and 9 and then bend each leg down to go through the holes in rows 6 and 11. Then solder them. LEDs D and E follow the same procedure except each leg should be bent out as before and then bent in to go through the holes in rows 7 and 10. Finally LEDs B and C have their legs bent out as before and then bent in to go through the holes in rows 8 and 9. Remember, and double-check the LED orientations! You can verify them from the top side too - you can see in the LED casing, part of the LED looks



like a Flag. The Flag side should be at the top for LEDs C, E, G and at the bottom for LEDs B, D, F!

Testing Part 1

OK. Your PhorsePOV is now complete. Don't place the battery in, nor the AVR MCU. You need to test the connections first! Using the multimeter:

The following connections should connect:

B1 to D5. D5 to C5. C3 to D3. G2 to H2. H1 to B1. G2 to J2. D2 to G2. G3 to J3. G4 to I4. G5 to H5. A10 to J6. The LED leads on F6 and G6, D10 and E10 to J6. J9 to I7. The LED leads on D7, E7, B9 and C9 to I7. The LED leads on B8, C8, F11 and G11 to H8.

The following should not connect:

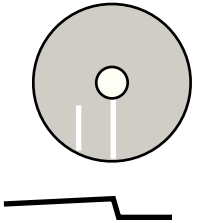
A3 to C3. A5 to C5. D3 to G3. D4 to G4. D5 to G5. A1 to B1. G2 to D5 (this is important, you don't want the power to short; you'll probably hear a little click - that's fine, it's the cap charging). You don't want any of the strips on rows 6 to 11 to directly connect - it's easy to bridge them accidently when soldering.

Finally you'll need to fit the Battery contact, which is tricky as it's on the track side:

1. Solder a wire loop from E5 to C5 on the track side (the chip's GND pin).

2. Cut two slots in the coin cell's plastic cover separated by 0.2" and slide it through the wire loop (this is the -ve contact).

3. Bend the coin cell contact and solder it to B2 on the track-side.



The plastic cover insulates the battery from the tracks and also holds the coin cell in place. Check these connections: The battery contact to G2. The wire loop to D5. D5 to D3 only when the button is pressed. These shouldn't connect: D5 to D6. B2 to B1. B3 to B2.

Programming The AVR

If you didn't buy the AVR from me, you'll need to program it. A binary file for the firmware and EEPROM is available from the Libby8dev website, along with the source code.

Running

After the tests have passed - you're ready to run the PhorsePOV (assuming you have a programmed MCU)! Stick the battery in (the RIGHT way!) and the PhorsePOV should boot into data entry mode. You can slide a bit of cardboard under the contact to turn it off! Now you can go back to the beginning of the guide and play with the PhorsePOV!