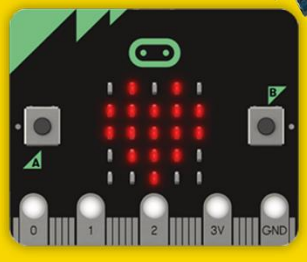
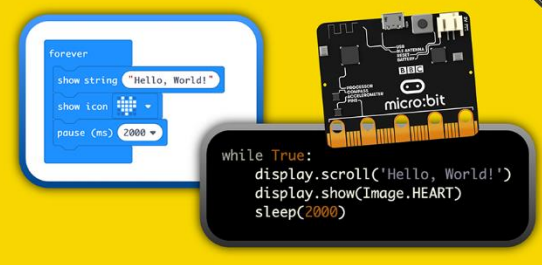
MICRO:BIT CHALLENGE

What will you make?



BUILD A COMMANDER MICRO:BIT DRIVE



CONTROL TECHNIQUES™

1 Introduction

This guide is aimed at helping you build your very own drive to spin a motor. Drives are used all around the world in anything that moves or spins. Elevators, factory machines, cranes, printers (very big ones too), food machines and even cars.

2 What is a drive?

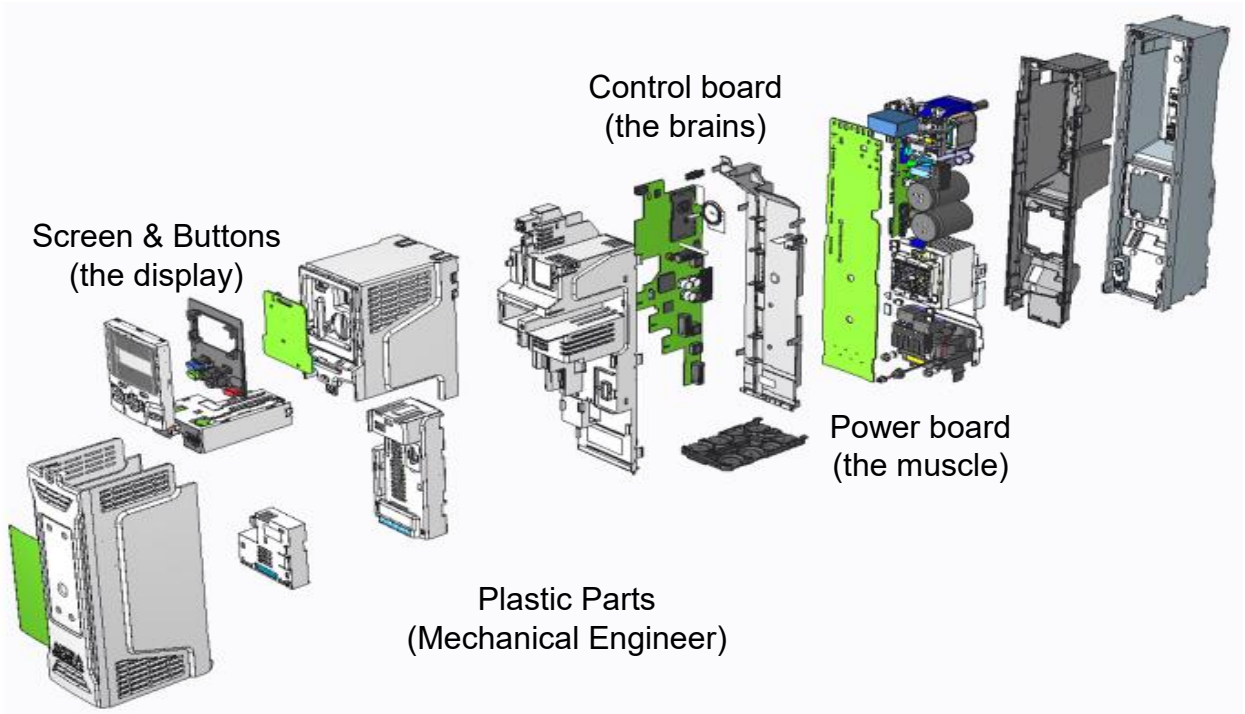
The word drive is used a lot by people who use them all the time. The technical word for a drive is inverter. It changes electricity in a way that can control the speed of a connected motor.

Control Techniques have been making drives in Newtown since 1973 and sell them all over the world. Engineers work together the design new drives all the time. Each drive is designed to turn different types of motors.

Engineers have different skills;

- Mechanical engineers design plastic and metal parts, how the look and connect. They also work out where the heat can escape from the drive as some bits get hot.
- Electrical engineers work with electricity and designing circuit boards that change the way electricity is given to the motor.
- Software engineers write code for small computers called controllers that are on the drive that do a certain job. One drive can have up to 7 controllers to make it all work in a machine.

The picture below shows the bits that make up a Control Techniques drive.

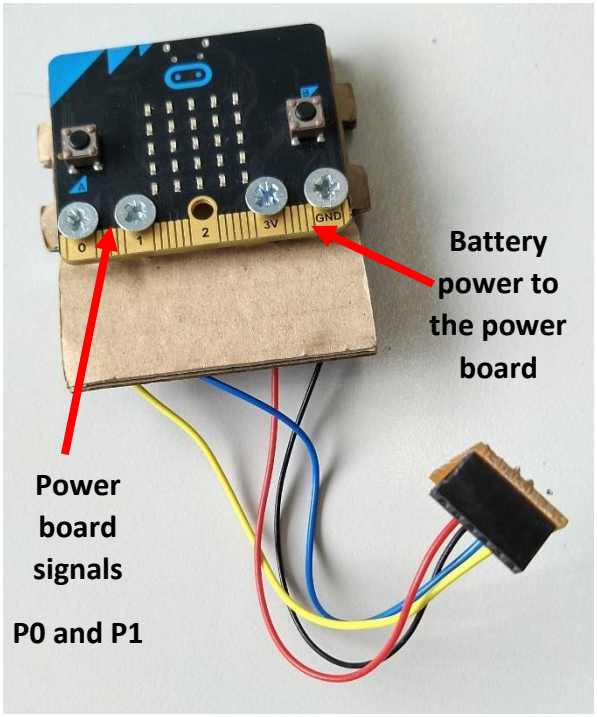


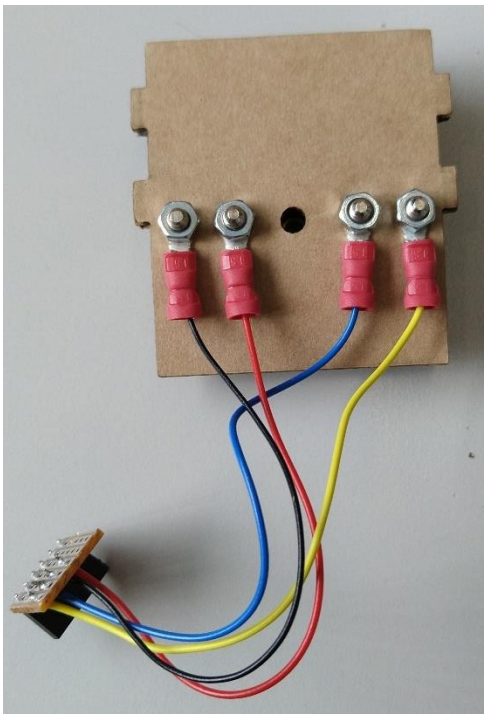
3 Let's build

Use your drive kit and these instructions to build the hardware (name for things you can touch). Some assembly can be touch by yourself, just ask for help.

3.1 Control board

We're going to assemble the control board – this is the brains to the drive.





We need the following:

4x Screws (M4)

4x Nuts (M4)

4x Washer (M4)

1x micro:bit 

1x wire loom 

1x mounting plate 

Step 1. Take the micro:bit and for the two holes on the left and the right we want to place the screws. See the photos above.

Step 2. Put a washer on the back

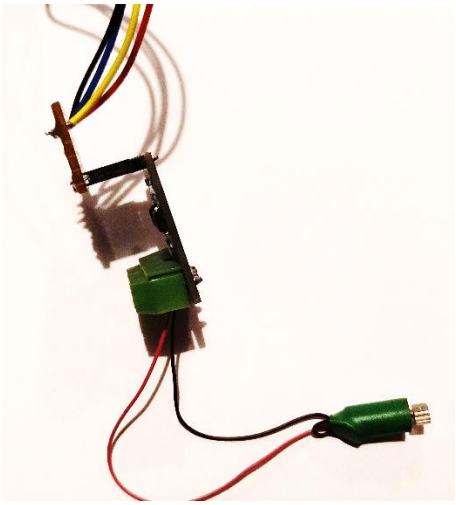
Step 3. Screw on a post onto each screw and tighten

Step 4. Push all the post threads through the cardboard mounting plate

Step 5. Turn over and fit the wire loom and nuts in the colour order shown above.

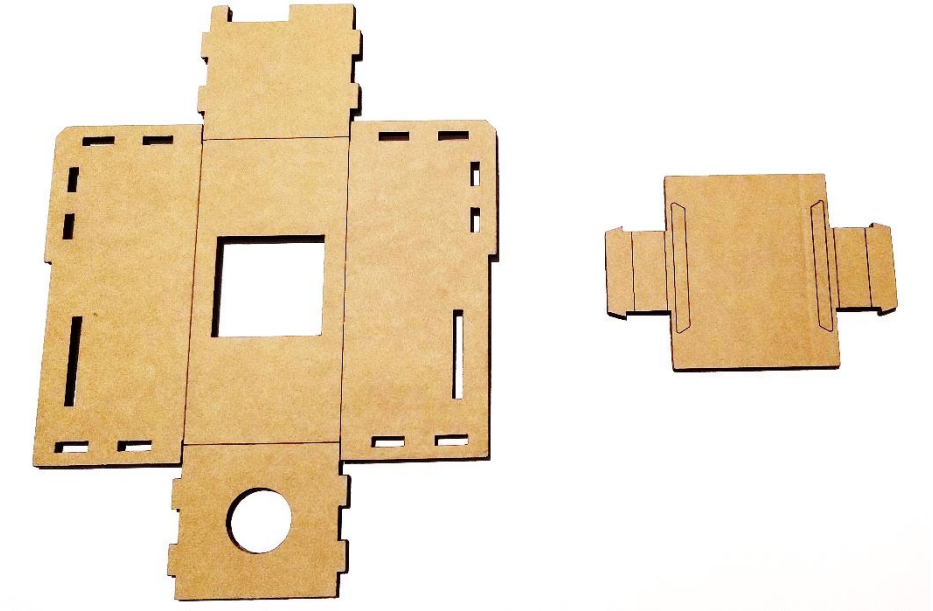
3.2 Power board

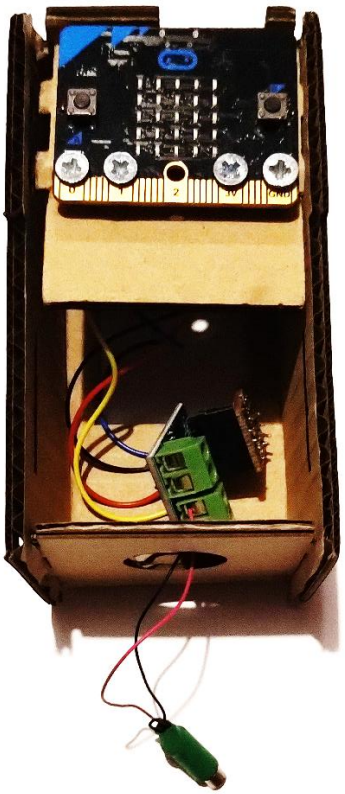
The power board has been fitted to the motor. All that remains is to fit the connector from the wire loom to the power board. Here's a photo of a completed one to copy.



3.3 Put it all together

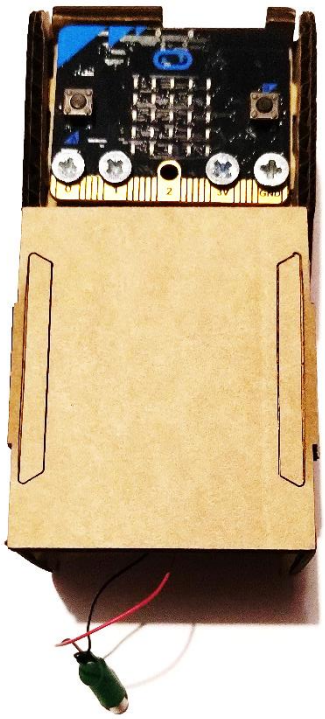
This is like a big origami shape (a Japanese way of folding paper to make things). The picture below shows the two parts that will make up our drive when added to the control board. We bend the parts along the lines so that the lines remain on the outside of the drive.



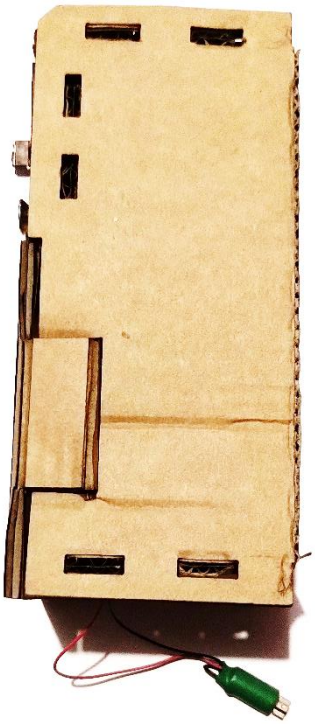


Step 1
Fit the control board and fold up the sides, starting top and bottom. Ask someone to help you here as more hands make it easier.

Top tip: Put the motor through the hole in the bottom.



Step 2
Once fully folded add the front bib that tucks in to the sides keeping everything together.



Step 3
Here you can see how it looks from the side where the tab has been tucked into the slot.

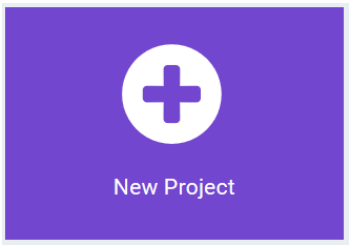
3.4 All done

The hardware build is done! Add your label to the front bid for a drive fit to use.

4 Let's code

The control board and power board can make the motor turn but only when they are given the correct signals. Coding is the way we can check signals from buttons and other sensors, make decisions and give control signals to connected things. We will use the micro:bit makecode editor to write our code.

Visit https://makecode.microbit.org and click "New Project" to get started.



4.1 What do we want to happen?

For engineers, the list of things they want to happen are called ***requirements***. These will be numbered to make it easy to check you have solved all the requirements at the end.

REQ 1. The motor speed can be changed using buttons
REQ 2. The A button will make the speed bigger
REQ 3. The B button will make the speed smaller
REQ 4. The display will show the speed in a simple way

Those were the simple ones, but we need to add a few more. We must think about the things we expect to happen. When we turn the power on should the motor start turning? For safety it must not until it's been told to do so. Let's add that as a requirement.

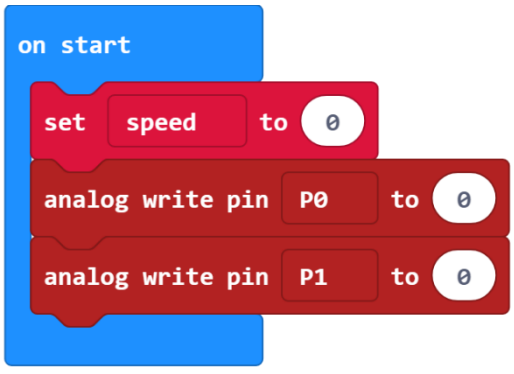
REQ 5. The speed at start-up (power on) must be zero

4.2 Remember the speed we want

With a new project created you will see two blocks in the editor, 'on start' and 'forever'.

To meet our requirement 5, we must set our speed to zero in the 'on start' block.

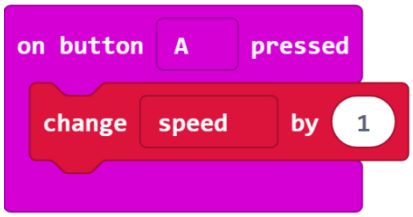
Make your 'on start' look like the picture below. You will need to create a new ***variable*** called 'speed' and set the analogue pins to 0. You will find the pins blocks under the advanced tab in the editor.



4.3 Make speed bigger when A button pressed

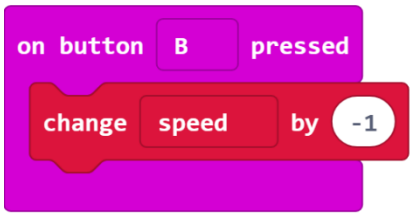
You can make decisions or do actions when sensors change. To do this on a button press go to the 'input' tab and select the 'on button A pressed' block. Change the

speed just like in the picture below. This code will meet our requirement 2.



4.4 Make speed smaller when B button pressed

We've made the speed bigger on pressing A now lets make it smaller with button B pressed. Go to the 'input' tab; there is no block called 'on button B pressed' so drag the A pressed block like you did last time. Once on your editor you can change A to B. That's another requirement done 😊



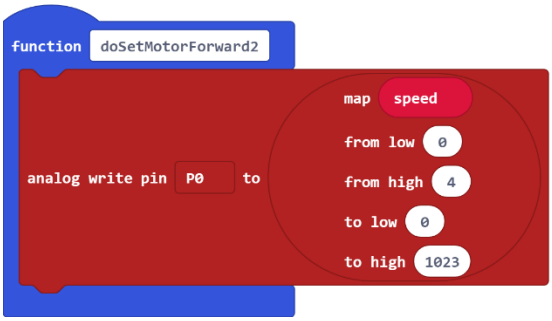
4.5 When things get tricky, make a function

So far, we've made a speed variable go up and down in value when buttons are pressed. We've seen to 3 out of our 5 requirements but still no motor is turning.

We must do some maths here.

To make the speed control simple for our drive we will use a function. This is a block of code that has one job, here it is to make the motor move.

Copy this function into your project. It uses a special bit of maths called 'map'. This changes the ***range*** of the speed value from 0 to 4 to 0 to 1023 which is what the motor control understands.

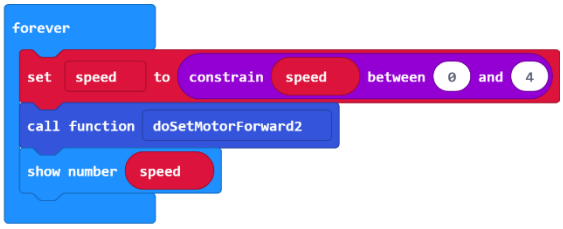


We make the motor move by writing the number from the map to the analogue pin P0.

4.6 Forever and ever

We need to call our function forever more and let everyone know our speed using 'show number'.

But to make sure our drive works smoothly we must make sure the speed can never go too big or too small; we use 'constrain' to do this.



4.7 Here's one I made earlier

You see, edit and download this project here - https://makecode.microbit.org/_D0v897XjD7aF

4.8 Was it too easy

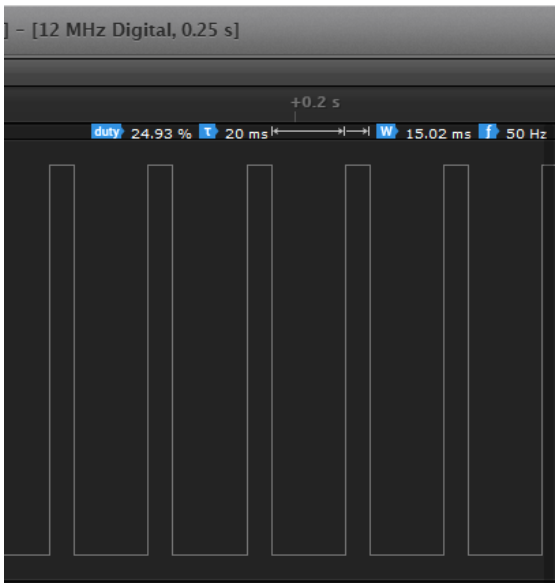
Look at this project; it allows the motor to turn forward and backwards and has a change to the display.

https://makecode.microbit.org/_0XpYA80Egegx

5 So how is the speed controlled?

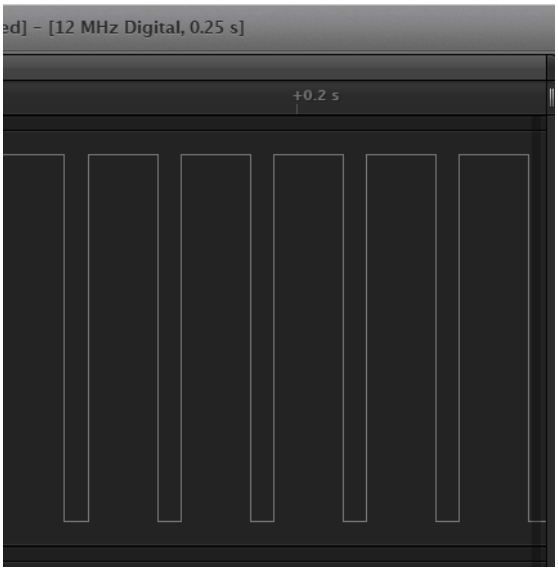
Your drive controls the speed in the same way to Control Techniques drives. It uses PWM control. This stands for some long words, Pulse Width Modulation, but it's all about turning the power on and off very quickly. How quickly changes the amount of energy given to the motor.

More energy gives us higher speed. We can see how this is done with the following pictures. They show the signal P0 which is controlled by our 'write pin' block in the code.

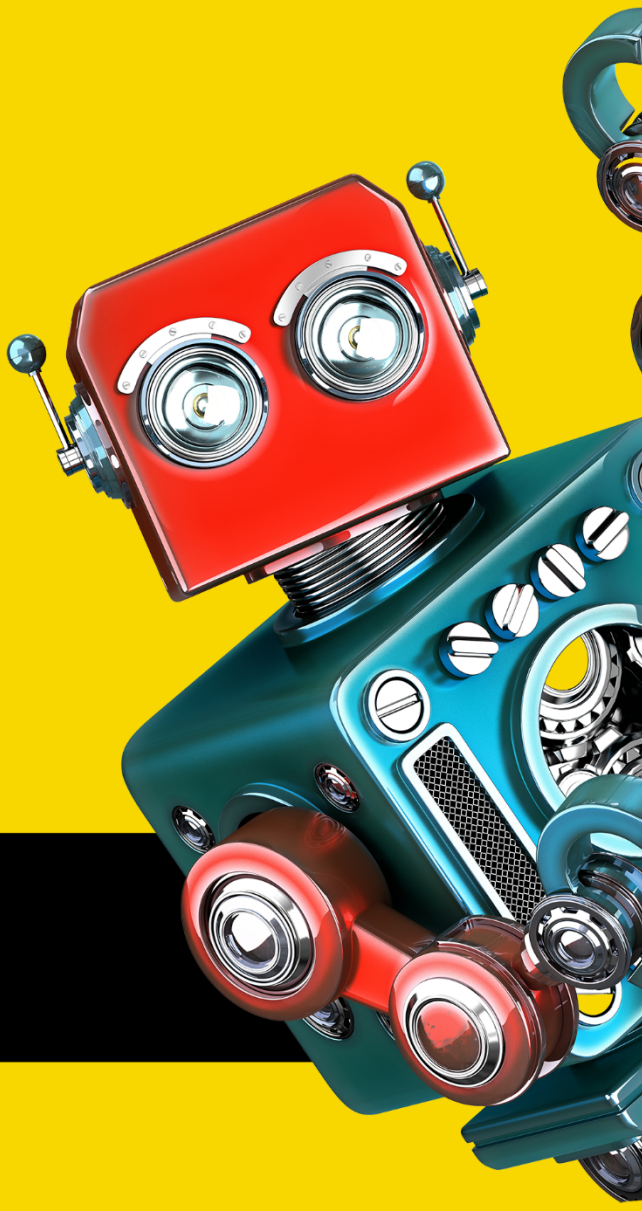


The picture above shows the control signal to the power board when the speed is 1 for our code. You can see the shape of the line that goes up and down is not square, it looks like several rectangles because the signal makes the motor switch off longer than it is switched on. It does this so fast that you can not see it on the motor (we had to use special tools to see it here called an ***oscilloscope***).

As we increase the speed the motor is kept on longer than it is off. The next picture shows the signal when the speed is 3.



A speed of 4 is not very exciting at all as the motor is never switched off. For our drive we were using just one control signal. On Control Techniques drives we use three.



CONTROL TECHNIQUES™