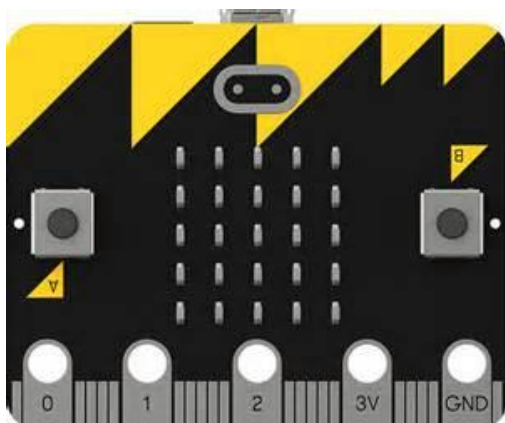


**Background:** During the school year 2015/6 about 1 million BBC micro:bits (m:b for short) were delivered to UK state schools for distribution to 11-year olds. After this distribution was completed in July 2016, the m:b was released for general sale. Several distributors now sell m:b starter kits including USB cable, battery holder and 2 AAA batteries for around £15 – for example [Kitronic](#), [Maplin](#), [Pimoroni](#), [Techwillsaveus](#). The background for the project from the BBC is [here](#) and in this [blog](#). The heart of the m:b is a standard ARM cortex M0 microprocessor with BlueTooth Low Energy BLE. There is a technical spec [here](#) and [here](#). If you are technically minded the inner details are provided by Lancaster University [here](#), [here](#) and [here](#).

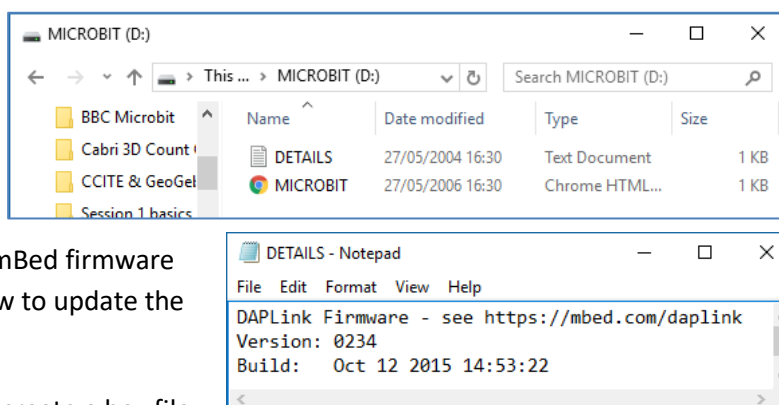


**Getting started:** When you connect the m:b to the USB port of a computer it will open up as an external storage device like a USB stick or SD card.

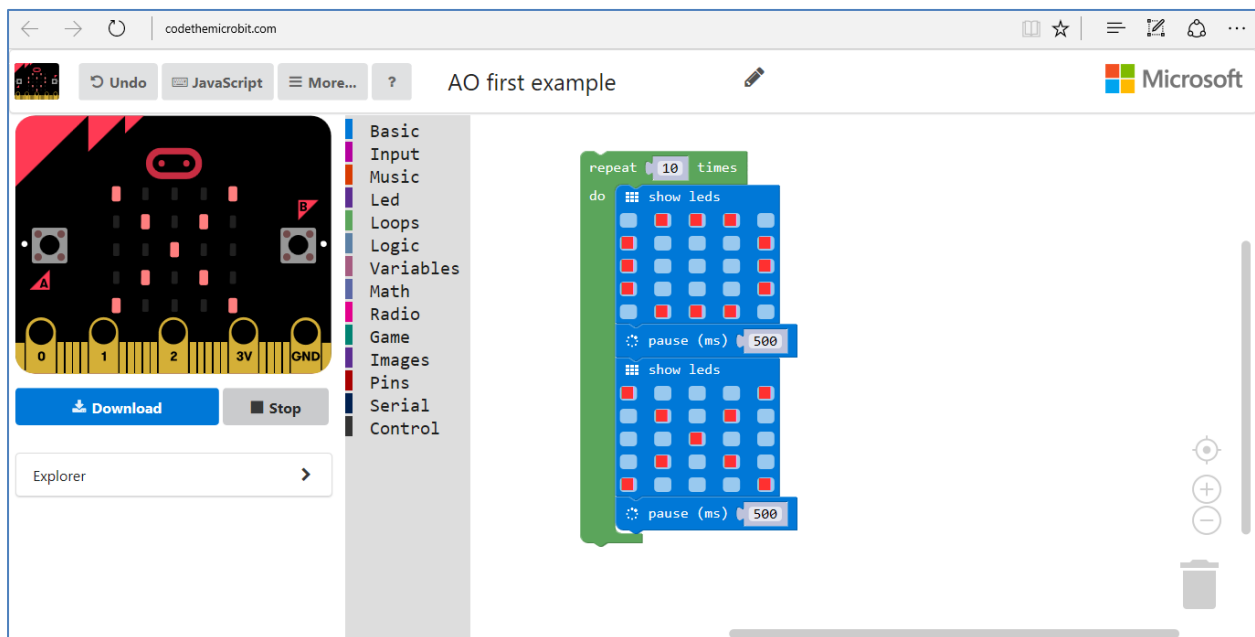
The DETAILS file is a text document containing information about the ARM mBed firmware currently installed. Later we will see how to update the firmware.

To program the m:b you need to find or create a hex file which you drag or copy to the MICROBIT folder. With the m:b connected, the yellow LED on the underside of the board will flash as the file is uploaded to the m:b's memory. When the upload is complete, the program will automatically run. In order for you to be able to test this I have created a hex file for a very simple program which just flashes between 0 and X ten times on the m:b's array of LEDs. The file is in my Dropbox [here](#). Once the file has been transferred to the m:b you can run it while it is attached to the computer, when the board is powered from the USB port. You can connecting the m:b to its battery box and disconnect it from the computer. It is then an autonomous (aka embedded) device and the m:b will continue to run the program.

**Creating a hex file:** The "Ready Set Go" leaflet which comes with the BBC micro:bit gives this site as the source of support: [www.microbitworld.me](http://www.microbitworld.me) – so let's start there. As you scroll down the home page you will see an introductory video [here](#). Then there is a choice of editors to use – (1) Code Kingdoms JavaScript, (2) Microsoft's Blocks and JavaScript Editor or (3) Python (strictly MicroPython). I am starting with (2) which takes you the editor page [here](#). The screen shot below shows the layout of that page.

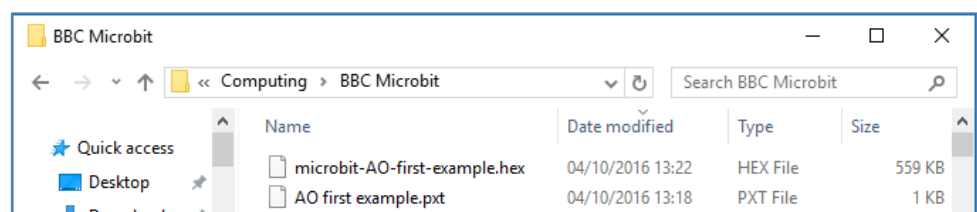


**Using the Microsoft Blocks and JavaScript editor:** the basic layout will look familiar if you have used [Scratch](#).



When you open this to create a program you will have an unnamed, blank sheet to work with. I have typed in “AO first example” so I can retrieve my work. The grey column contains the items on the blocks menu. I am going to use a counted loop, which I can find on the 5<sup>th</sup> choice “Loops”. These building blocks are coloured green. Drag out the “repeat n times” block and edit the default 4 to another value e.g. 10. I want to display a large O symbol, so click on the top “Basic” choice and select the blue “show leds” block and drag it between the green jaws of the loop. Click on the individual leds you want to light up in red. Next I want a delay, so find the blue “pause (ms)” block. Drag this below the blue “show” block and before the bottom of the green “repeat” loop. Edit the time to show e.g. 500 ms (or half a second). Now put in a second “show leds” and create the “X” pattern. Finally put in another “pause” block and change the duration as required.

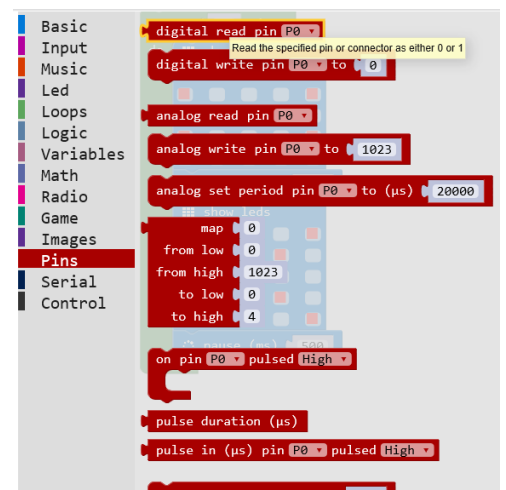
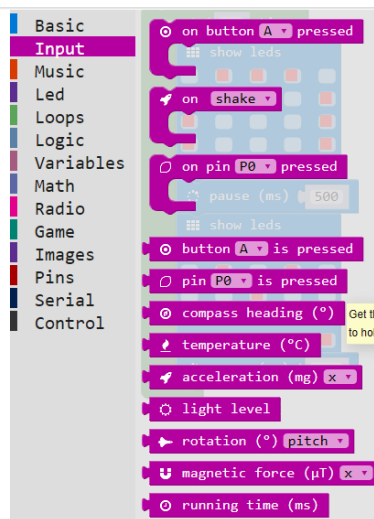
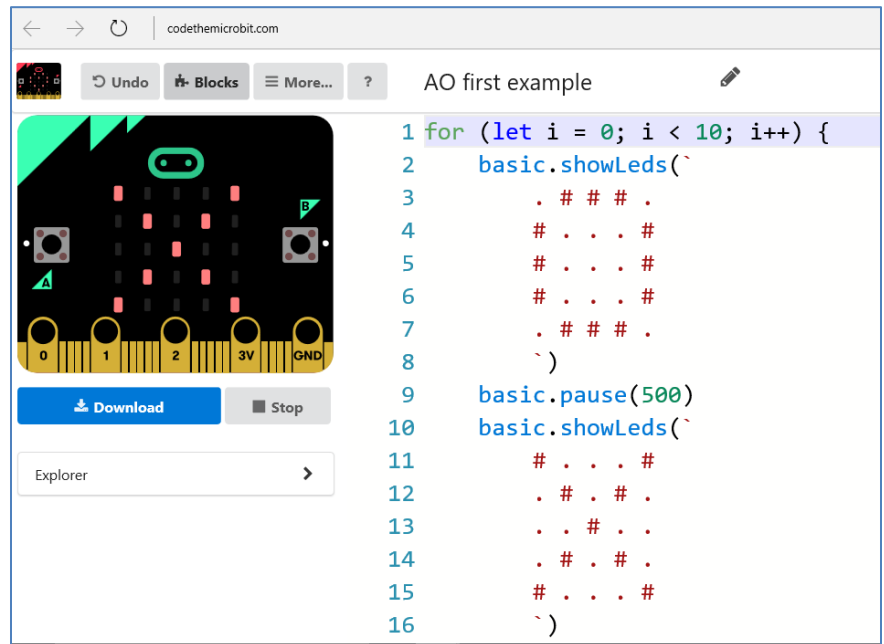
Now you have created a program, you test it with the built-in emulator. Under the image of the m:b there is a STOP/START button. If it shows STOP it is already running, so press it. Now press START and you should see that the emulator produces our desired output. Now is the time to save your work on your own computer. Click on the “More” button and select “Save As” to save your project e.g. as “AO first example.pxt”.



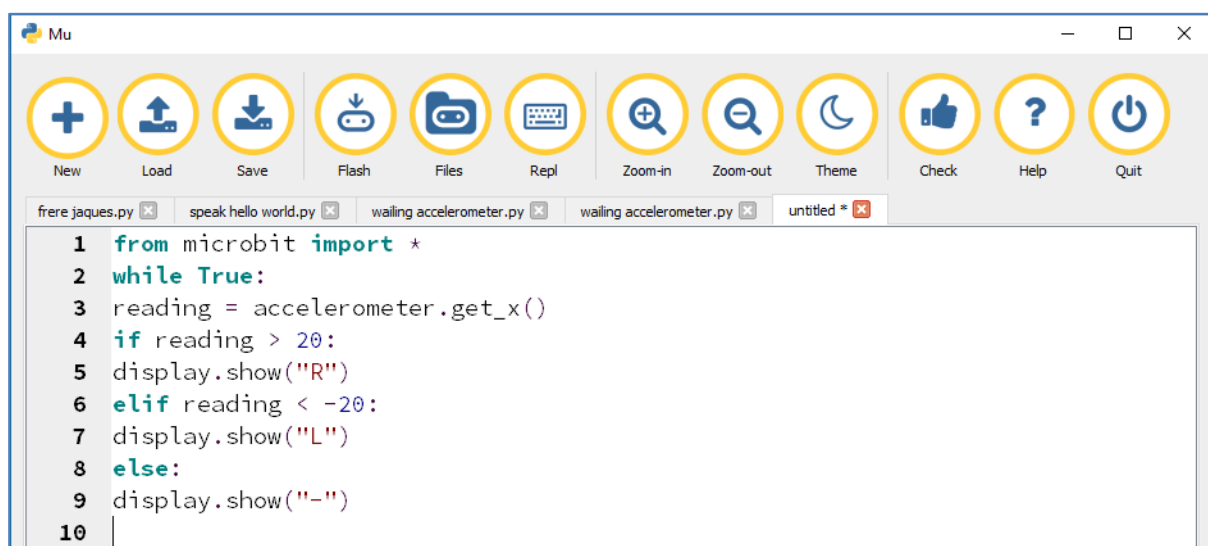
With the m:b attached to the USB port, click on the “Download” button. You can select in which folder you want to save the resulting hex file “microbit-AO-first-example.hex”. Open that folder and right-click on the file’s name. Select “Send to” and you should a list of places which includes the m:b device e.g. as MICROBIT (D:). If you now click on the “More” button, you can select “Embed project”. This gives you the option to upload your project to be publicly published, and creates a link to it. My link is [here](#). If you need help click on the little image of the m:b next to “undo”.

If you click on the “JavaScript” button you will see the text version of the program.

The “JavaScript” button has now flipped to “Blocks”, so you can toggle between graphical and textual representations of the same program. Have a look at the different “Blocks” menus to see some of the powerful features available in the BBC micro:bit. These Microsoft editing tools are bespoke versions of the Microsoft Research [Touch Develop](#) system developed originally to aid app development for Windows 8 mobile phones.



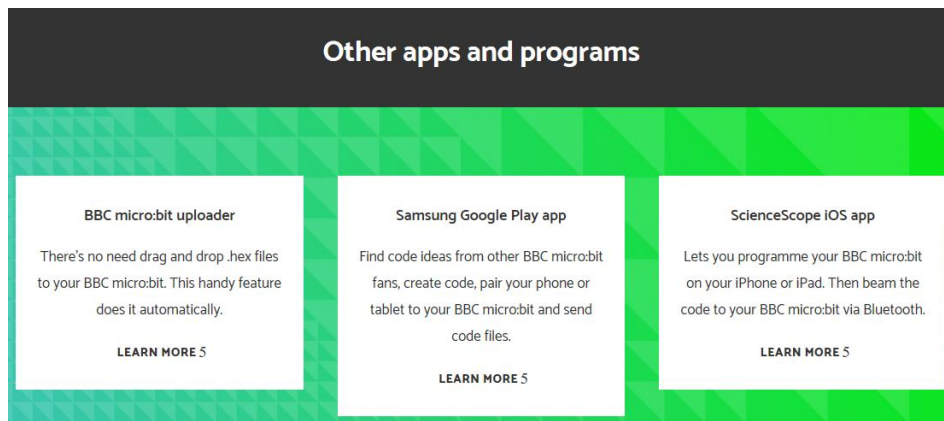
**Python:** if you would like to experiment with using Python follow this [link](#). There is an extensive MicroPython editor and help system specifically for the BBC micro:bit [here](#). This is best used with the [mu](#) editor, which also supports Raspberry Pi. There are some very powerful libraries which you can easily install on the m:b, such as for music and for blue-tooth. You can download a very detailed 118-page get-you-started guide [here](#). Here is a sample program from that guide (p. 25) to simulate a spirit-level.



Towards the bottom of the introductory [page](#) are links to an automatic micro:bit uploader – well worth installing. There are also Android and Apple Apps for use with smartphones and tablets.

**Blue tooth pairing for mobile devices:** both the free [Android](#) (Samsung) and [Apple](#) (Sciencescope) apps require the m:b unit to be paired using a very [simple system](#).

The [Samsung site](#) includes an introductory [pdf](#). Every m: b has a unique 5 character name, such “GITIG”. Both these apps allow you to develop your own programs for the m:b in a similar fashion as we have seen with the block and JavaScript editors. The Samsung system also

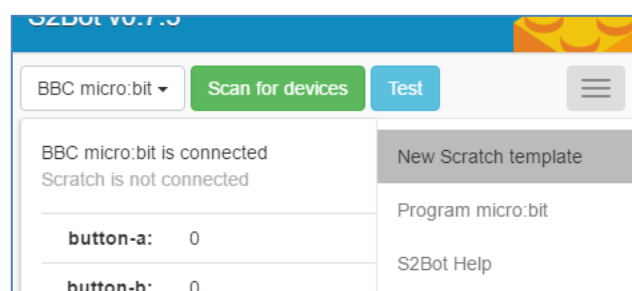
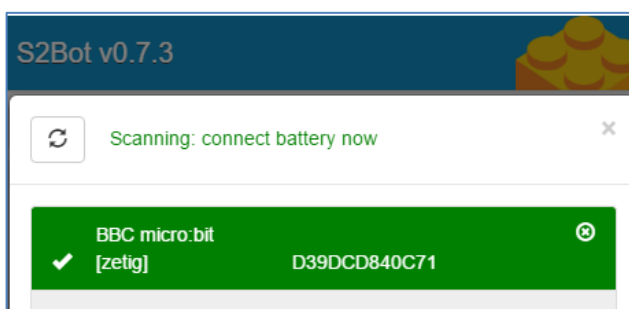
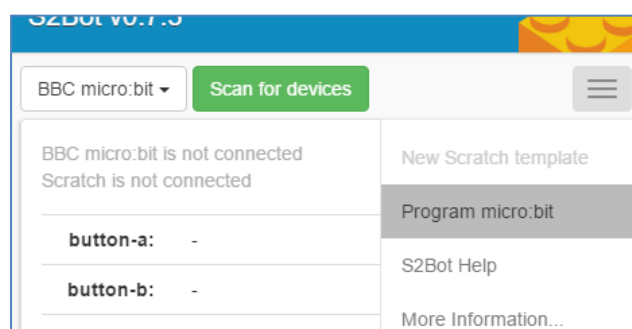


allows you to interact with some of the devices built into the phone, such as the ring-tone and camera.

**Scratch controlling the BBC micro:bit:** Clive Seager of [Revolution Education](#) has developed a very powerful tool to enable [Scratch](#) to communicate with external devices. The Chrome App called [S2Bot](#) now supports many wireless devices, as well as USB ones. I am using a Windows 10 laptop running the [Chrome](#) web-browser. Documentation for the S2Bot App is [here](#) and you can install it from [here](#). I have also purchased a £10 BLE Bluetooth BLED112 USB dongle from the [Picaxe](#) store.

From the S2Bot App menu select “Program micro bit” and save the “microbit-s2bot.hex” file to your computer e.g. to the Desktop.

Connect your m:b to the computer with a USB cable and drag the file from your folder to the folder showing the m:b device. The m:b’s display will show “BLE” and then ask you to describe a circle with the m:b so that the outer LEDs form a large letter O. If you do this correctly you will see a smiley face on the m:b display (briefly). Now you are set up to use the m:b (called “Zetig” in my case) with Scratch. You can run Scratch online from the MIT website, or install the [offline editor](#). Before we can use S2Bot to pick up data from the m:b we must send it a program to enable the wireless broadcasting with BLE. The instructions are [here](#). As before, you will need to save your current Scratch project. Then use the S2Bot menu to create a “New Scratch template” to save to your

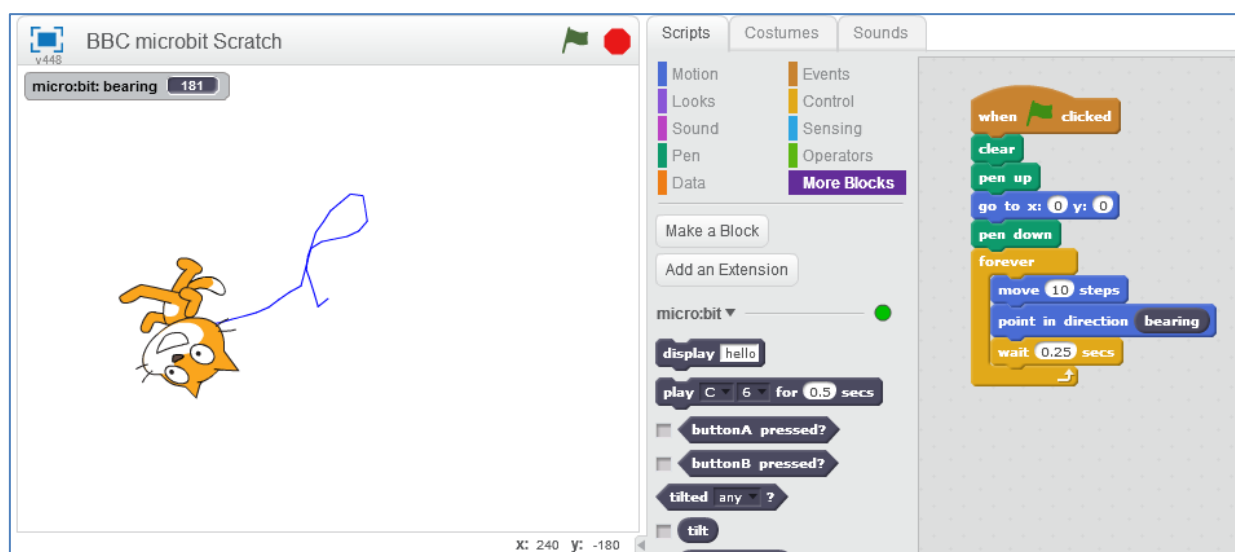
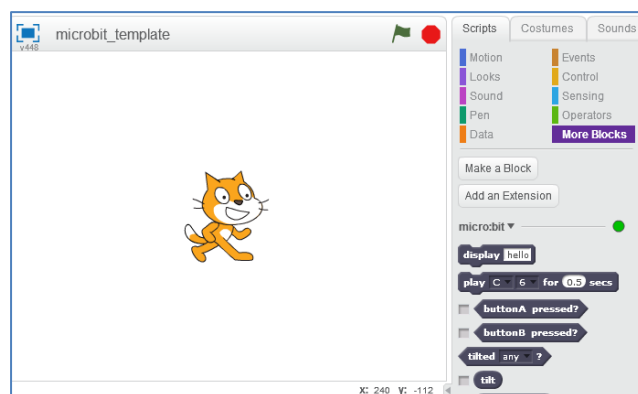


computer. Then you can open this “microbit\_template.sb2” file in Scratch to see if the BBC micro:bit is connected.

You could use the temperature or accelerometer data to plot graph in the way we have already seen.

But you can also turn your m:b device into a real-time game controller as shown in the program below.

Here we use the “bearing” sensor to steer the Scratch icon around the screen. Can you think of some more interesting things to do? A fuller article is [here](#).



**Getting down to the nitty gritty:** at the heart of the BBC micro:bit is an ARM mBed microprocessor. ARM provides a compiler tool for programming this on their [mBed site](#).

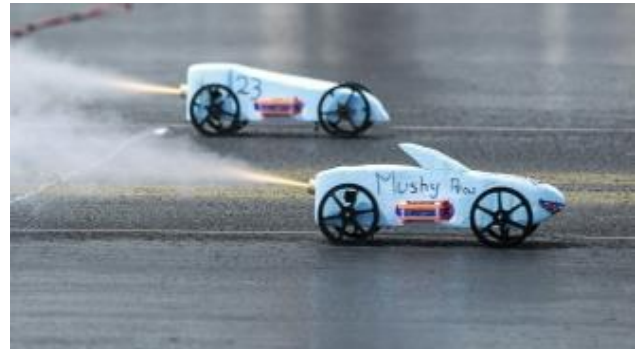
So we have seen that there is support for programming m:b in Microsoft’s Block editor, in JavaScript, in MicroPython, in Scratch and in the ARM mBed compiler. You can create code for your m: on a Windows or Apple laptop, and on both Android and Apple mobile mobile devices. Also that there are other ways to interact with the m:b other than uploading a hex file through a USB cable or via Bluetooth. Information about other ways of using BBC micro:bits, such as Internet of Things devices, is now beginning to trickle out. We just need to make sure these are easy to find and share. Next here are a couple of educational developments.

**Lesson plans and ideas:** The main educational support site for the BBC micro:bit is [here](#). Within this site there is a [collection](#) of example files illustrating a wide variety of applications for coding projects. The Computing At School group is an association formed by the British Computer Society and Microsoft Research to support teachers of Computing. Its [website](#) is free for anyone to use, and it contains thousands of resources which anyone who has registered (for free) can access. I am signed up to receive a daily email alert of new postings on the site. A good example is this series of 4 [introductory lessons](#) aimed for Year 7 or 8 students (11-12). There is a BBC micro:bit special interest group on the [STEM Learning](#) site which anyone



registered (for free) can apply to join. You'll find some postings from me there, as well as some [resources](#). The Institution of Engineering and Technology IET is an official BBC partner which has developed extensive Faraday resources and competitions accessed through this [link](#). The Wellcome Trust has a new initiative called [The Crunch](#). Last summer every secondary school received a Crunch pack which includes some BBC micro:bits. These are currently being used for [The Big Food Survey](#) which finishes today.

The Bloodhound SuperSonic Car SSC is due to break the world land speed record at over 1000mph in South Africa in 2018 with test runs in 2017. The main website is [here](#). Government funding for the project was granted on condition that it would have a substantial STEM education impact on schools to inspire a future generation of UK engineers. The Education site is [here](#). There is now a dedicated channel on the Dendrite platform called [Bloodhound Blast](#). Last year there was very successful micro:bit model rocket car competition for school, which is being repeated in 2016/7. Details are [here](#), and entries close on 14<sup>th</sup> October 2016. The rocket car kits are provided free by the British Army, which also provides young rocketeer training session. They have a slot which holds a micro:bit in a protective case with a button battery to record acceleration data.

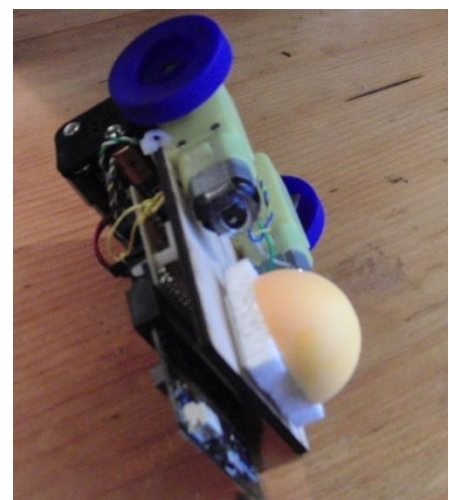
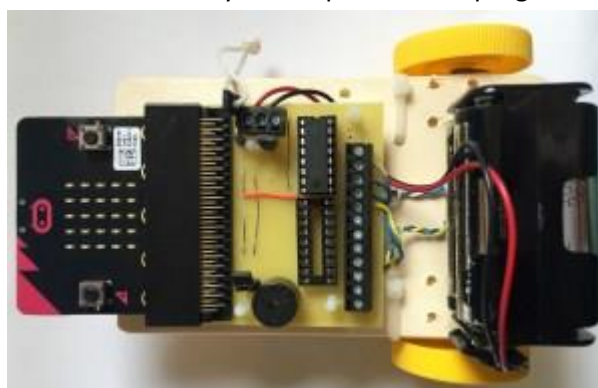


### Sources of gadgets for use with micro:bit

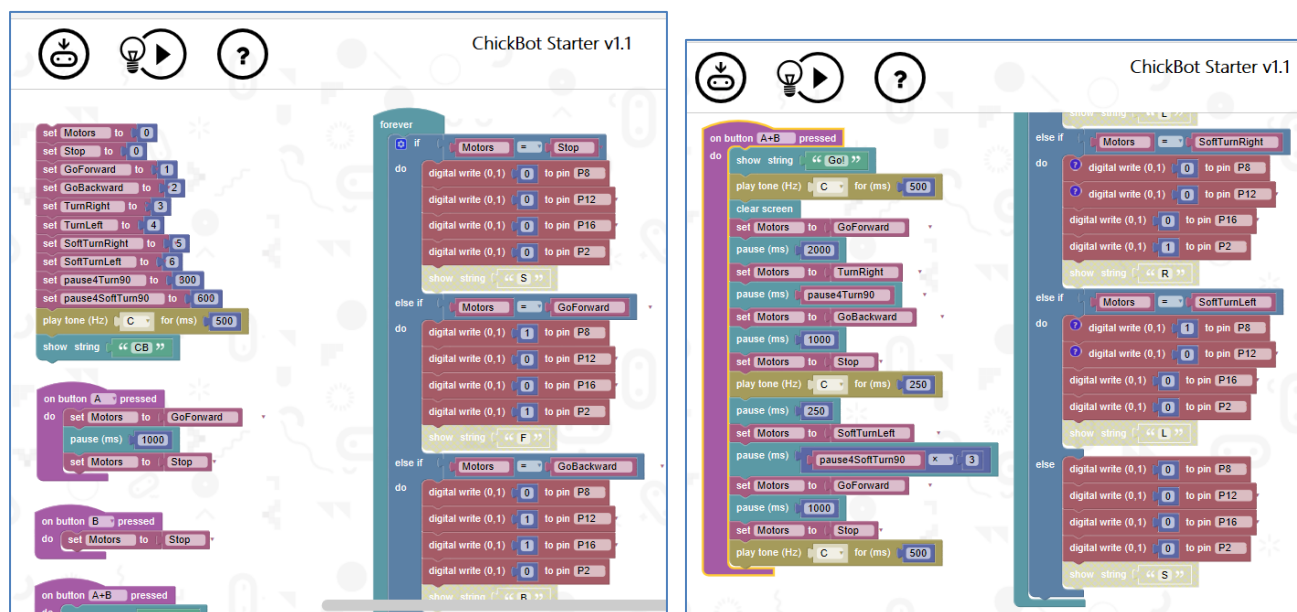
You can make your own connections for an external speaker, buzzer or headphones using leads and crocodile clips, or you can buy a headphone/speaker adaptor very cheaply e.g. [here](#). The black lead is connected to the GND pin and the yellow one to pin 0 on the m:b. The webpage gives a sample program to play a few notes and a table for the frequencies for notes in higher and lower octaves. See the Kitronik link on how to make a [guitar tuner](#)! The Microbit Accessories [shop](#) has many other low cost gadgets for use with m:b such as sensors, buzzers, speakers and leads, as well as experiment kits. One of these is called the [ChickBot Robot Kit](#) which costs £17. There is an accompanying site called the [ChickBot Club](#), with gadgets for micro:bit, Arduino and



Raspberry Pi. I have just bought the micro:bit kit which took less than an hour to assemble, code and test. The building instructions are [here](#). This [link](#) takes you to the editor where you compile the test program and upload the hex file.



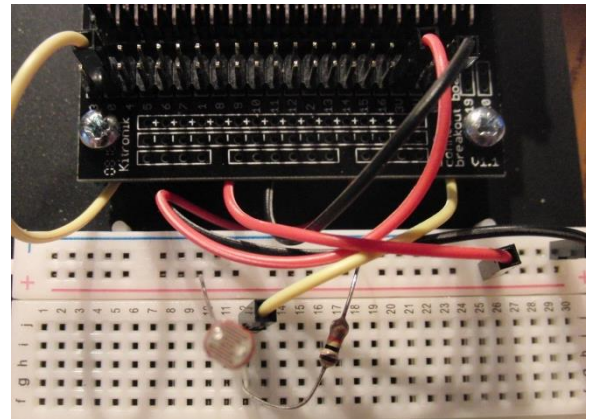
The photo of the top of the robot shows the m:b inserted into an edge connector mounted on a printed circuit board which contains the motor drivers, a buzzer and connections to the battery pack. The photo of the underside shows the two motors and wheels. Instead of a front wheel, there is a jazzy half ping-pong ball. No soldering is required. You also add other components such as a pair of light dependent resistors to sense changes in light in order to make the robot follow a line. As you can see, you can put quite a lot of code into a micro:bit!



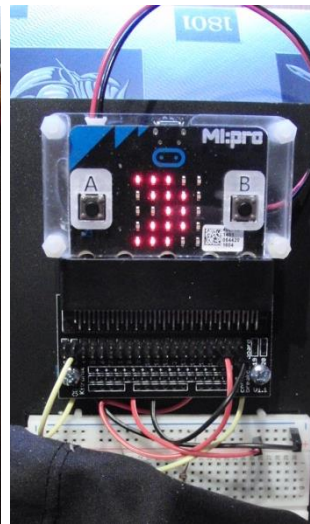
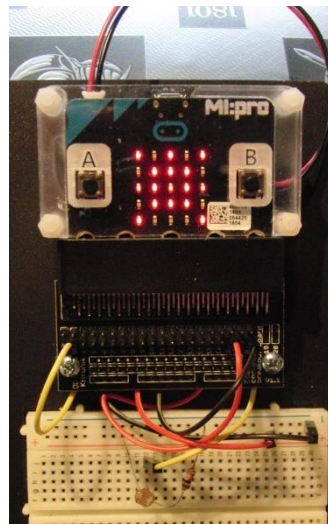
In the next section I will give some examples of the excellent micro:bit resources available from Kitronik. Others can be found [here](#). Lawrence Rogers' [Insight](#) site is always worth a look. He developed the [Apple App](#) for the micro:bit, and has a new project called [Insight Mr Bit](#) to make programming as easy as possible.

**Developing the BBC micro:bit further:** the original intention of the BBC's "Make it digital" project was to recreate some of the sense of enthusiasm for computing that was fuelled by the BBC (and other) programmes in the 1980s following the publication of the late Chris Evan's "The mighty micro". That was originally targeted at families and homes and was supported by the development of the BBC micro by Acorn of Cambridge, which only later found its way into schools. Around 2010 a number of angel investors and academics in Cambridge, some connected with Acorn and its successor ARM Holdings (standing for Acorn Risk Machine), had the idea of rekindling this with a single board computer – the Raspberry Pi. Raspberry was blowing a rude sound at Apple and Pi stood for Python, one of the programming languages it supported. The RPi has been a phenomenal success world-wide selling more than 10m units. But it was never intended as a classroom device. By the time you have added a keyboard, mouse, screen and wifi it is more expensive than many laptops. By contrast, the BBC micro:bit has been designed for both home and school use from the outset. It is much closer to the design of the Italian Arduino engineering education project than RPi. I very much hope we can develop some powerful applications for the m:b as a sensing and control tool for physical computing across the STEM (Science, Technology, Engineering and Mathematics) subjects at all levels. With a little browsing about I have found a number of good sources of both projects and ideas for the m:b.

One source of these ideas are the experiments for the Kitronik [Inventor's Kit](#) for the BBC micro:bit. One [experiment](#) senses the analog input from a Light Dependent Resistor LDR. The Kit costs £25 and its main components are an [Edge Connector Board](#) (available separately for £5) and a [Small Prototype Breadboard](#) (available separately for £3.50). The images below show the BBC micro:bit held in a [Mi-pro case](#) (£4.75) attached to a [battery cage](#) (£1.08). The yellow cable

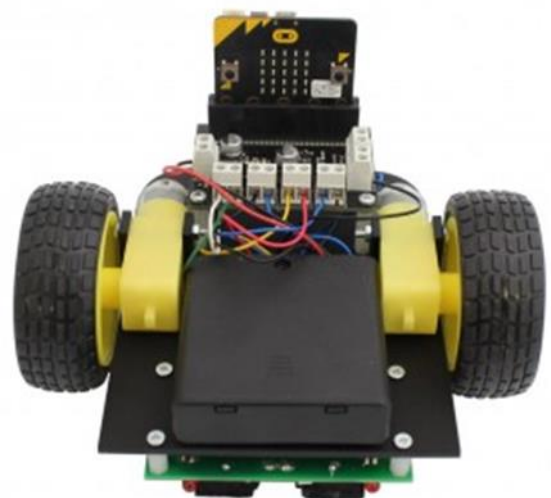


connects pin 0 to a column on the bread-board which also contains one leg of the LRD and one leg of a 10kΩ resistor. The red cable connects the 3V output to the positive row on the bread-board, and the black cable connects one of the 0V outputs to the negative row. The spare leg of the resistor is connected to



the negative row and the spare leg of the LDR to the positive row to complete the circuit. The Block Editor is used to create the very simple program which just reads the state of the analog pin P0 and stores it in the variable called Light. If this exceeds the threshold value of 512 then the LED array displays an impression of the sun. When the LDR is covered up the value of Light is below the threshold and the LED array switches to an impression of the moon. An article developing these ideas further is [here](#).

Here is an attempt to use the hardware and software from the £26.20 [Kitronik](#) line-following buggy kit as the basis to develop code for standard Logo style procedures for Forward, Back, Left, Right etc. The buggy uses a standard Micro:bit [motor control board](#) which has an edge connector to take all the pin connectors. Each motor is controlled by 2 pins to go forward (1,0), back (0,1), coast (0,0) or brake (1,1). Digital pins 8 and 12 control one motor, and 0 and 16 the other. This simple block of code just makes the buggy roll forward for a second, stay still for a second, roll back for a second and stay still for another second. The leds display an appropriate symbol for each of the four sections so it can be tested without the buggy. From this experiment we find that the buggy moves 30 mm in 1 second. The length of the buggy is 15 mm, so it will move forward one length in 500 milliseconds. A fuller article is [here](#).





**script** floor turtle experiments

**function** main ()

**for** 0 ≤ i < 5 **do**


basic → show leds( , 1000)

pins → digital write pin(P8, 1)

pins → digital write pin(P12, 0)


pins → digital write pin(P0, 1)

pins → digital write pin(P16, 0)

basic → show leds( , 1000)

pins → digital write pin(P12, 1)

pins → digital write pin(P16, 1)

basic → show leds( , 1000)

pins → digital write pin(P8, 0)

pins → digital write pin(P0, 0)

basic → show leds( , 1000)

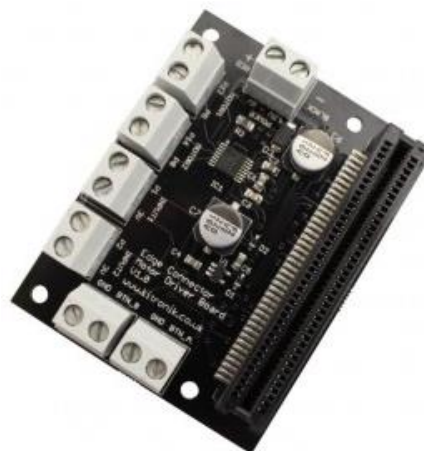
pins → digital write pin(P8, 1)

pins → digital write pin(P0, 1)

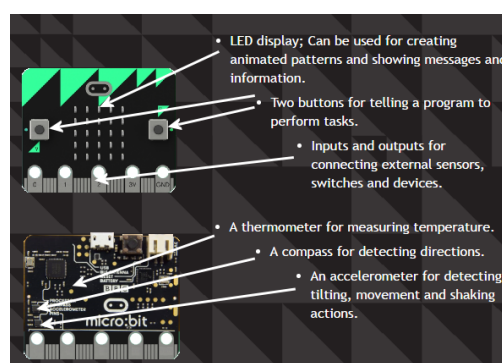
**end for**

**end function**

Further ideas about sensing and control can be found [here](#).



The diagram of sensors below was found on the [Insight](#) page:



**Bluetooth and radio ideas:** a significant advantage of the BBC micro:bit is the inclusion of BlueTooth Low Energy BLE communication. The person leading this development is an engineer called Martin Woolley and here are some [links](#) to his [blogs](#) and the [Kitronik](#) site. There is also a video [here](#). An example of use of the Radio commands in MicroPython is [here](#). The full documentation is [here](#) in Chapter 9. It would be really nice to be able to use the m:b as a real-time sensor transmitting data, like telemetry in a F1 car, to a laptop. There are some clues [here](#).

**Maintenance – upgrading firmware:** In order to take part in the Crunch food survey it is necessary to update the firmware on the BBC micro:bit. This is a surprisingly easy operation! The instructions are [here](#).

1. Download the newest firmware: [234\\_microbit\\_if.hex \(97.0 KB\)](#)
2. Unplug the micro:bit.
3. Press and hold down the reset button next to the USB connector.
4. The micro:bit will power on. Your computer's file browser will list it as a mass storage device called "MAINTENANCE".
5. Copy the firmware hex file you downloaded in step 1 to the "MAINTENANCE" device.
6. Wait until the drive dismounts and the system (orange) LED is flashing continuously.
7. Unplug the micro:bit and plug in again, and the board should enumerate as a MICROBIT.