The BBC micro:bit - from the UK to the World

[Extended Abstract]

ARM BBC Lancaster University

Micro:bit Educational Foundation Microsoft

ABSTRACT

The micro:bit rocks!

1. INTRODUCTION

In the early 1980's, the British Broadcasting Corporation (BBC) introduced a whole generation of educators and students in the United Kingdom (UK) to computing through the *BBC Computer Literacy Project*, which featured the BBC Micro, a 6502-based personal computer designed and produced by Acorn Computers Ltd. (referred to at times as the "British Apple"). The project was very sucessful: more than 80% of UK classrooms had a BBC Micro and many of today's computing professionals from the UK first encountered computing through the BBC Micro.

Fast forward to the 21st century, and the BBC sought to again inspire a new generation get creative with coding, programming and digital technology through its 2015 *Make It Digital* initiative, as well as to support the UK's mandate to teach computer science concepts at all grade levels. [3]

As part of this effort, the BBC introduced the micro:bit (see Figure 1), a small programmable and embeddable computer designed, developed and deployed by the BBC and partners (including ARM, Microsoft and Lancaster University) to approximately 800,000 UK middle school students in 2015-2016. Harkening back to its work with the BBC Micro, the BBC described the micro:bit as its "most ambitious education initiative in 30 years, with an ambition to inspire digital creativity and develop a new generation of tech pioneers." [?]

From a local educational experiment in the UK, the micro:bit has become wildly successful globally because of a unique combination of hardware and software, embracing a constructionist approach to computing education. Driving the worldwide expansion is the Micro:bit Educational Foundation (`www.microbit.org`) is a non-profit organization established in September 2016 with the support of its founding partners [1]. The Foundation's Mission Statement is to:

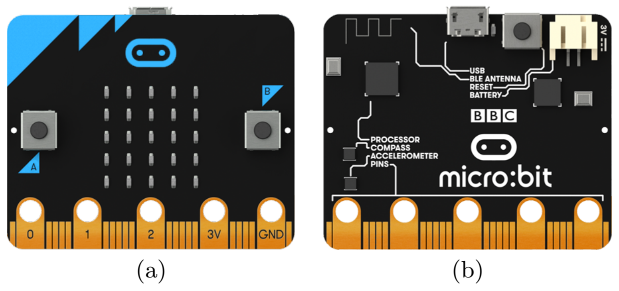


Figure 1: The micro:bit: (a) front, with two buttons, 5x5 LED display, and edge connector (bottom); (b) back, with processor, accelerometer, compass, Bluetooth, USB and battery ports.

- enable and inspire all children to participate in the digital world, with particular focus on girls and those from disadvantaged groups.
- make micro:bit the easiest and most effective learning tool for digital skills and creativity.
- work in collaboration with educators to create and curate exceptional curriculum materials, training programmes and resources.
- build and support communities of educators and partners to remove the barriers to learning digital skills

In this paper, we describe the key decisions and lessons from delivering the BBC micro:bit in the UK and then expanding to reach more educators and students around the world. We draw from two full years of full deployment of the micro:bit in the UK, as well as deployments in Europe, the Americas, and Asia. There are approximately two million micro:bits now in the market and many hardware, content, and education partners participating.

1.1 Hardware

Figure 1 shows (a) the front and (b) the back of the micro:bit, which measures 4cm x 5cm. Like the Arduino Uno [?], the micro:bit is a single-board microcontroller that can be programmed via a host computer (usually a laptop or desktop) and then embedded in projects where it runs on battery power. In contrast to the Uno, which has no built-in sensors, the micro:bit board hosts a variety of sensors (temperature, accelerometer, magnetometer, light level), a 5x5 LED matrix, two user-defined buttons, as well as Bluetooth

[1]ARM, Amazon, BBC, British Council, IET, Lancaster University, Microsoft, Nominet, and Samsung.

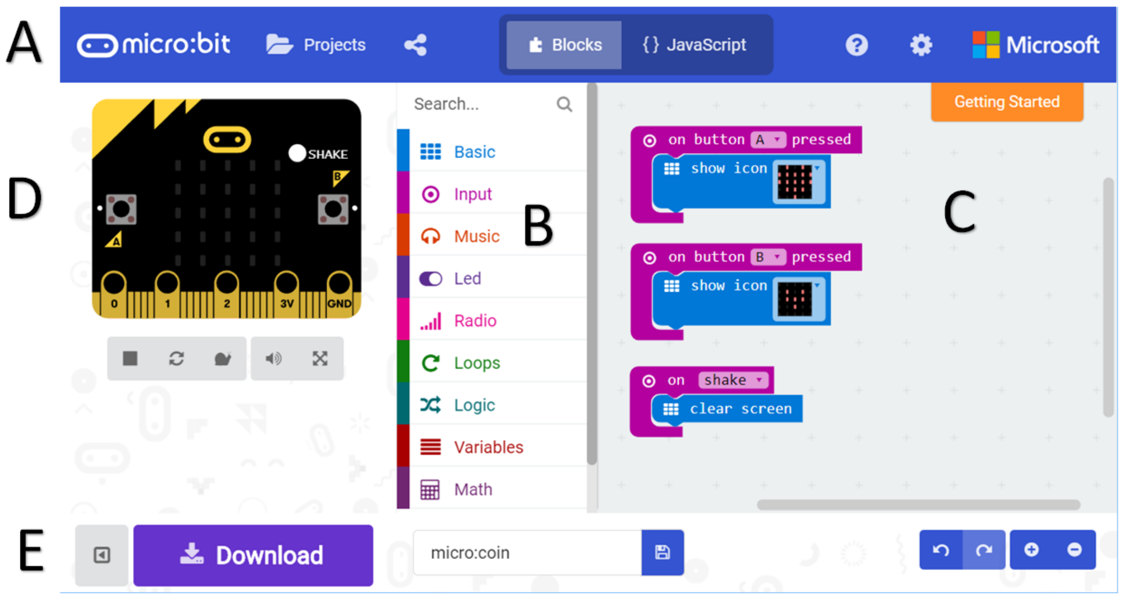


Figure 2: MakeCode web app for the micro:bit (`http://makecode.microbit.org`).

Low Energy (BLE) communications.[2].

The design of the micro:bit hardware was driven by the first two objectives of the BBC micro:bit project: (B1) to provide a simple creative experience for physical computing, wearable and Internet of Things (IoT) projects; (B2) to supply a device that can continue to provide learning opportunities as the user's expertise grows.

On the hardware side, the micro:bit's built-in sensors, buttons and LED display allow many projects to be completed with no additional hardware or wiring. The holes on micro:bit's edge connector allows additional external sensors and actuators to be connected via crocodile clips. The micro:bit's BLE capabilities introduces networking to the picture, and enables streaming of data and command/control operations among the micro:bit, smartphones, laptops, as well as other micro:bits. As with Arduino, an ecosytem of micro:bit shields (hardware peripherals) that accommodate the micro:bit's edge connector expands its capabilities greatly (`http://microbit.org/resellers/`).

1.2 Software

The design of the micro:bit coding tools also was oriented towards a simple starting experience with room for progression. In particular, the coding objectives of the project were: (B3) to give students an exciting, engaging introduction to coding; (B4) to stimulate curiosity about how computing technologies can be utilized to solve problems that students identify.

Based on in-school trials with a micro:bit prototype, the BBC focused on delivering a web app based on the popular Blockly framework [1] to permit students to create scripts via drag-and-drop operations in a web browser, and see the execution of their scripts via a simulator. Text-based coding via scripting languages also was identified as an important feature. As the micro:bit would be incorporated into standalone projects, it also was essential for the user's program to be compiled and installed in non-volatile storage on the micro:bit where it could be run via battery power.

The solution delivered by the BBC's partners evolved from the initial design to include support for Blockly, JavaScript and Python, all via web apps. Figure 2 shows a screen snapshot of the MakeCode web app for the micro:bit, which supports programming via both Blocky and JavaScript. The web app has five main sections: (A) menu bar with access to projects and examples, and switching between Blockly and JavaScript editors; (B) Blockly toolbox of micro:bit API categories; (C) Blockly programming canvas with a simple reactive program; (D) micro:bit simulator for execution of the user's program in browser; (E) download button, which invokes the in-browser compiler/linker to produce a binary executable.

The Python solution for the micro:bit is based on MicroPython (`http://micropython.org`) an implementation of Python 3.0 for microcontrollers. It includes a full Python compiler and runtime that runs on the micro:bit and supports a read-eval-print loop (REPL) to execute commands sent via a terminal, as well as to import and run scripts from the Python web app for the micro:bit (`http://python.microbit.org`).

1.3 Overview

We take a bottom-up approach, starting with a review of physical computing (Section 2), which anchors and defines the micro:bit experience. Section 3 presents a broad set of micro:bit-based projects, ranging from wearable games to environment science, that demonstrate the micro:bit's capabilities. Section 4 describes the rollout of the micro:bit in the UK and other countries in Europe, the Americas and Asia. Section 5 concludes with final thoughts about what has made the micro:bit successful to date and what comes

[2]The micro:bit has a whopping 16kB of RAM and 256kB of Flash memory, compared to the Uno's 2kB of RAM and 32kB of Flash

next.

2. PHYSICAL COMPUTING

As discussed in the Introduction, the micro:bit is a device with similarities to the Arduino family of printed circuit boards. Such *physical computing* devices are designed to be placed in and interact with our physical environment. Physical computing lives in the spaces between computing and many other disciplines: art, industrial design, health, environmental monitoring; it has close ties to cyber-physical systems, embedded systems, and IoT. The National Science Foundation defines cyber-physical systems as those that "integrate sensing, computation, control and networking into physical objects and infrastructure, connecting them to the Internet and to each other."[2]

TODO: this derived from Steve/Sue white paper - need to check carefully. The benefits of using physical computing to introduce beginners to computing systems include:

- *broad reach* because of diverse applications of physical computing – leverage fine arts, music, design, etc. in projects;
- *increased motivation* because of tangible visible outcome (rather than virtual on screen);
- *learning by doing* as there are many ways to achieve goal (no single correct solution)
- *natural division of labor* for more complex projects (design, hardware, software, ...)
- *full system view of computing*: hardware and software working together.

2.1 Wiring and Arduino

To help explain the BBC micro:bit, it's very instructive to understand Hernando Barragan's 2003 Master's thesis, "Wiring: Prototyping Physical Interaction Design", the inspiration for the Arduino system [**?**]. His objective was to make it easier for non-technical creators, such as artists and designers, to leverage electronics in their their work by simplifying the hardware and programming experience. In particular, he said of existing work: "Current prototyping tools for electronics and programming are mostly targeted to engineering, robotics and technical audiences." Of Wiring's design, he identified the following key concepts:

- a simple cross-platform integrated development environment (IDE) to create so-called "sketches";
- simplified application programming interfaces (APIs) to access a microcontroller's resources;
- leverage open source compiler/linker toolchain, transparent to the end user;
- a bootloader to make it easy to upload a compiled sketch to the microcontroller;

Also key to Wiring was openness of both the hardware and software comprising the system.

But, still some issues:

- reliance on the C language and C compiler (needs to be installed)

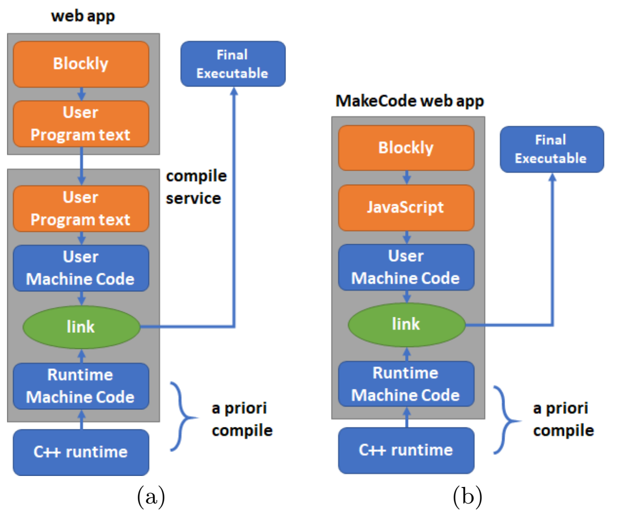


Figure 3: Web and compiler designs: (a) initial BBC design; (b) final design, as implemented in MakeCode.

- very poor experience in IDE
- USB bootloader requires device drivers on some systems

2.2 The BBC micro:bit

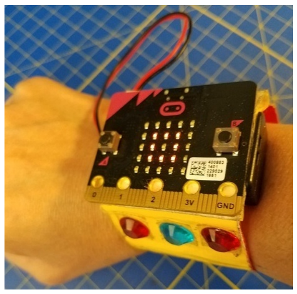
Main points:

- *A Visible Computer*: BBC micro:bit inherits the raw PCB nature of Arduino (everything is visible to the end user).
- *No Wiring*: makes starting easy
- *Small Size*:
- *Scripting via Web App*: XYZ
- *No Install*: XYZ As shown in Figure 3(a), in the BBC design the text of a user's program (whether derived from Blockly or produced directly by the user) is submitted to a compile service that returns a final executable to be copied onto a micro:bit (connected to the host computer by USB) via a specialized loader application. avoiding the need for a compile service for user code (as shown in Figure 3(b));
- *Extensible*: via edge connector and layered APIS (package system too). Note that edge connector is a (male) plug - micro:bit plugs into to peripherals, not the other way around.

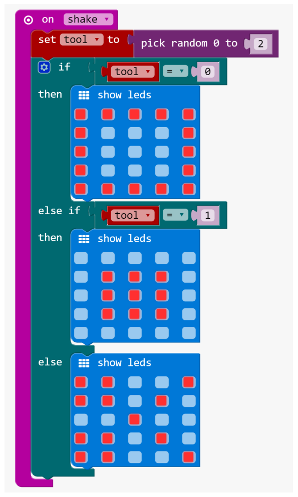
From this perspective, the micro:bit can be seen as a starter device for physical computing, embedded systems and cyber-physical systems, as it has sensing, computation, control and networking capabilities built in. The micro:bit is not properly an IoT device, having no built-in way to connect over IP, but it can be connected to other devices with IP connectivity.

3. PROJECTS

In this section we present a sampling of projects that illustrate the capabilities of the micro:bit.



(a)



```
on shake
  set tool to pick random 0 to 2
  if tool = 0
  then show leds
  else if tool = 1
  then show leds
  else show leds
```

(b)

Figure 4: Micro:bit watch for playing rock/paper/scissors.

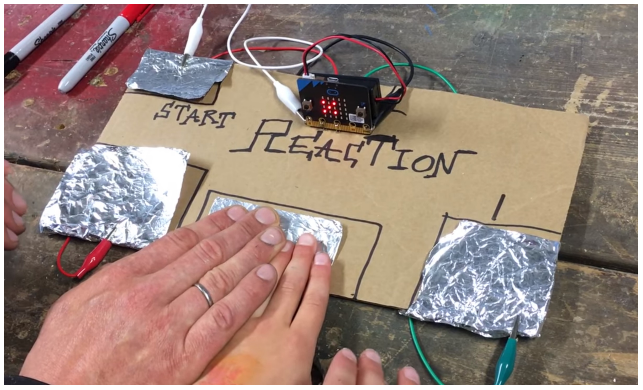


Figure 5: Reaction game.

3.1 Wear and Play

Figure 4 shows one of the most popular micro:bit projects: a watch that plays the rock/paper/scissors game when shaken; the program reacts to a shake event by choosing a random integer between 0 and 2 and displaying a rock, paper or scissor shape on the LED display, based on the number chosen. The user can use this simple app to play the game with themselves or a a friend. The project consists of a making step and coding step, as shown at

`makecode.microbit.org/projects/rock-paper-scissors`

Many micro:bit projects use simple classroom supplies. The reaction game project (Figure 5) uses cardboard, aluminum foil, and crocodile clip connectors to illustrate the use of circuits with a game that measure reaction time. Crocodile clips connected to pins P0, P1, P2 and GND also are connected to aluminum pads. The user completes a circuit by touching the GND pad and one of other pads. The pad labelled “START” begins the game; after a 1-3 seconds (randomly determined), the micro:bit display lights up - the first user to touch their pad wins, and their reaction time is displayed:

`makecode.microbit.org/projects/reaction-time`

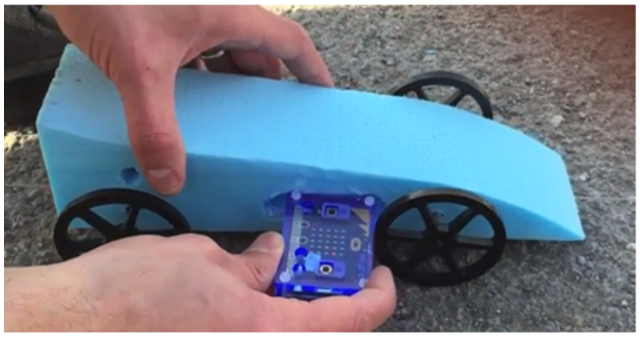
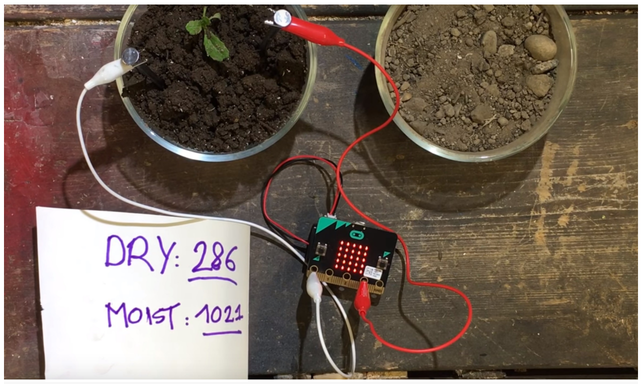
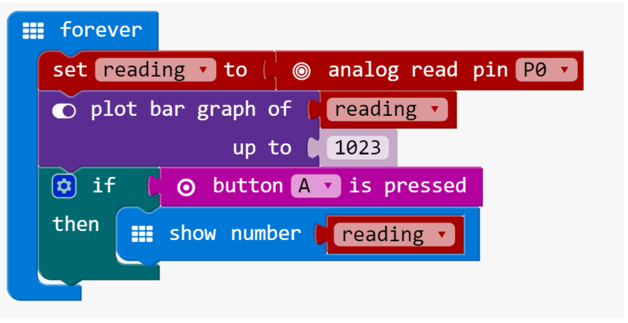


Figure 6: Bloodhound Model Rocket Car with embedded micro:bit for measuring acceleration.





```
forever
  set reading to analog read pin P0
  plot bar graph of reading
                 up to 1023
  if button A is pressed
  then show number reading
```

Figure 7: Measuring soil moisture via micro:bit pins.

3.2 Measure

The micro:bit’s built-in sensors and small size make it perfect for embedding in science and technology projects. The Bloodhound Model Rocket Car is part of the Bloodhound Project, [3] whose goal is to set a new world land speed record and inspire students about STEM subjects. Students design, build and race model rocket cars in competition, learning about physics, aerodynamics, and mechanical engineering. Microsoft worked with the Bloodhound Project to incorporate a micro:bit into the car’s design, as shown in Figure 6; the micro:bit captures the (X,Y,Z) accelerometer data of the rocket car during its race. After the race, students can upload the data from the micro:bit and analyze the performance of their cars.

Figure 7 shows an environmental project that uses the micro:bit to measure soil moisture. The combination of wa-

[3] `www.bloodhoundssc.com`

```
input.onButtonPressed(Button.A, () => {
  radio.sendString("H");
});

input.onButtonPressed(Button.B, () => {
  radio.sendString("S");
});

radio.onDataReceived(() => {
  let data = radio.receiveString();
  if (data == "H") {
    basic.showIcon(IconNames.Happy)
  } else if (data == "S") {
    basic.showIcon(IconNames.Sad)
  } else {
    basic.showString("?");
  }
});
```

Figure 8: . Broadcasting simple messages using the micro:bit radio.

ter and soil nutrients makes the soil have some conductivity. The more water there is in the soil, the greater its conductivity, which can be measured using the analog pin API. In this project, the student first learns to calibrate the measurement readings using dry and wet soil samples. Then, the micro:bit is coded to periodically record the reading. Using the micro:bit's Bluetooth radio, the readings also can be sent to a central source. In this way, the moisture of a set of soil samples (in a classroom) can be recorded and reported. For more about this project, see:

`makecode.microbit.org/projects/soil-moisture`

3.3 Network

Using a lower level of the Bluetooth stack, the micro:bit supports a simple radio broadcast protocol that can be used to send short messages to a set of micro:bits. Figure 8 presents a simple example in JavaScript that shows how to use a micro:bit to communicate your "mood" to other micro:bits in the vicinity. Note that the micro:bit that sends a message does not receive that message.

The following two projects use the micro:bit radio to illustrate how fireflies synchronize their blinking and how infections spread:

`makecode.microbit.org/projects/fireflies`

`makecode.microbit.org/projects/infection`

3.4 Control

The micro:bit can be attached to external actuators, such as servos, to create systems that respond physically to their environment.

4. COUNTRY DEPLOYMENTS

localization - language but also curriculum standards

5. CONCLUSION

6. ACKNOWLEDGMENTS

7. REFERENCES

[1] N. Fraser. Ten things we've learned from blockly. In *Proceedings of the 2015 IEEE Blocks and Beyond Workshop (Blocks and Beyond)*, BLOCKS AND BEYOND '15, pages 49–50, 2015.

[2] NSF. Cyber-physical systems. 2018.

[3] S. Peyton Jones. Computer science as a school subject. In *Proceedings of the 18th ACM SIGPLAN International Conference on Functional Programming*, ICFP '13, pages 159–160. ACM, 2013.