As digital technologies become embedded in our everyday world, and as ubiquitous computing becomes commonplace, we need to consider how to better educate people about computer science concepts, regardless of their (professional) goals in life. Realizing the value of having basic programming skills and an understanding of core computer science concepts, the UK is working to provide its schools with a new product, called the BBC micro:bit (www.microbit.co.uk), that could put pervasive computing at the center of computer science education.

**A NEW CORE SKILL**

Teaching computer science should no longer be viewed as teaching a trade or providing training for coders. In many ways, it’s becoming similar to needing to learn your native language in school. Teachers push pupils to analyze and construct poems, write short stories, or read theater plays. They don’t do this with the primary goal of creating the next generation of writers or poets. Learning a language is seen as a cultural skill that lets you use verbal communication and written text effectively in a variety of contexts—from writing a job application letter to completing your PhD thesis. This language education might encourage some students to become writers, but even those who go on to become, say, financial consultants, will use the skill throughout their lives.

In a similar way, understanding computing technologies, programming, and computational concepts has become a core skill for informed participants in modern society—much like reading, writing, and basic arithmetic are core skills. Without basic computing knowledge, it’s difficult to understand how Google sorts and personalizes your search results (and contemplate whether you’re being manipulated by the results), understand what it means to give an app access to your camera and the network, decide between end-to-end or server-based encryption, or understand how Amazon “guesses” what you like. Allowing illiteracy with regard to computational concepts will create a major digital divide.

**ENGAGING STUDENTS WITH COMPUTING**

Thirty years ago, for many pupils and students, learning programming had a clear purpose: write a new program currently unavailable. Some of you (those between ages 35 and 55) probably wrote your own database for music records, created a tournament planner for your favorite sport, developed a customer database for your neighbor’s small business, or wrote a simple text adventure or a jump-and-run game. Even as a novice programmer with mediocre skills, you could create a piece of useful software that impressed others.

Things are different now. My friends and I sometimes play a game: think of a useful app and guess how many versions of it exist in Android Play or the App store (you can even think of a strange and useless app, and you’ll often find different versions). This highlights how difficult it has become for a novice programmer to create something useful that’s competitive with other apps on the market. Few are motivated to program a poor version of something that’s already available. (Sometimes it works, if the application you model is very simple. In teaching, I have asked students to create their own version of Flappy Bird, with a focus on design and interaction, which typically works well.) However, the solution might be to move into the ubicomp domain.

**Using Pervasive Apps as a Motivator**

Pervasive computing has the potential to create, for the current generation of pupils, the experience many of us had 30 years ago. If you provide school kids with the right tools and platforms, they can build a smart irrigation system for dad’s dying potted tree in the hall, create an online leaderboard for emptying the dishwasher, implement a sensor that tracks the dog’s activity when no one is home, or make a device that automatically tweets to friends when you put on your inline skates. It’s motivating to be able to create something new that demonstrates value, something that your friends haven’t seen before. It doesn’t matter that many of these developments will have the same fate as those programmed 30 years ago—that is, they’ll rarely be used—because it’s the experience itself that’s important. It’s the sense of achievement.
Introducing the BBC micro:bit
In the UK, there’s an initiative to provide all kids in school year 7 (that is, approximately one million school kids ages 12–13) with the BBC micro:bit computing platform, making it easy for students to create ubiquitous computing applications. At the same time, the platform will provide a solid educational base for learning about computational concepts.

The initiative is a partnership between educational authorities, the BBC, and hardware and software companies. The device is constructed to be simple enough to allow a quick start (low threshold for starting) yet powerful enough to create real applications (high ceiling). A core concept is to provide well-curated and well-presented material that makes it easy and fun to get started—for teachers as well as for students.

The computer is a single board that includes a processor, different input and output components, and wireless connectivity (see Figure 1). The processor is a 32-bit ARM Cortex M0, with 16K RAM running at 16 MHz, that also includes a Bluetooth low-energy module (the antenna is on the board). The board can be powered using a micro USB power supply or batteries. It includes two buttons, an accelerometer, and a compass for input and 25 LEDs for output. It also has digital inputs and outputs to connect additional sensors or actuators. The overall size is 4 cm by 5 cm, and one side features a 20-pin connector. The hardware isn’t that impressive, but it has everything you’d expect for tinkering, and it’s optimized for low cost.

There’s basic software running on the board that lets users execute programs written using one of several Web-based development environments. Once the program is written, it can be compiled online, and it’s then available for download from the website. To deploy it to the actual hardware, the BBC micro:bit is connected to the USB port. The program is then available as a drive (much like a USB memory stick), and the compiled program is just copied onto the drive. After a reset of the BBC micro:bit, the last stored file is automatically executed. A wireless programming approach is in the making.

On the website, editors are available that support different programming languages—JavaScript, Python, and Blockly—and another editor, the Microsoft Touch Develop, supports programming on touchscreen devices (see Figure 2). All of these integrated development environments (IDEs) provide visual programming support and offer an emulator to test the code and see the effects on the LED grid. All IDEs make it easy to build on existing examples and share developments. The website offers a variety of tutorials and examples (see Figure 3).

Many countries around the world have yet to make teaching computer science or programming a top priority; these topics are often viewed as electives for “geeks” or are pushed only to those preparing for a job in the field of computer science. I hope that as computer science matures, it becomes better integrated.

Figure 1. The BBC micro:bit hardware, front and back. The computer is a single board that includes a processor, different input and output components, and wireless connectivity. (Source: www.microbit.co.uk; used with permission.)
Innovations in UbI Comp Products

The integration of Ubiquitous Computing (UbI) concepts into the general curriculum, leading to improved literacy for computing concepts. Time will tell how well this ubiquitous computing experiment in the UK will work out in terms of its impact on learning. However, as members of the pervasive computing community, we should recognize that UbI concepts and technologies, combined with practical application, could be the key to motivating students of all ages to engage with computing. Let’s encourage students to create their first system, something never before built, to spark their interest!

Figure 2. There are different development environments that support coding and software development for the BBC micro:bit. They are optimized to make it easy to use resources available on the board. (Source: www.microbit.co.uk; used with permission.)

Figure 3. On the website, many tutorials and project examples are provided, and users are encouraged to add more examples. Many programs can be looked at (the source code) and directly used, tried out, and altered. (Source: www.microbit.co.uk; used with permission.)

Albrecht Schmidt is a professor of human-computer interaction at the University of Stuttgart. Contact him at albrecht.schmidt@vis.uni-stuttgart.de.

Selected CS articles and columns are also available for free at http://ComputingNow.computer.org.