# Lesson plan - Lesson 3: Energy meter part II

## Using the MonkMakes solar experimenters kit

## Introduction

In lesson two you had time to introduce the energy meter to the students and ensure that they understood the relationship between the energy store and the micro:bit. They had previously discussed the overall aim of the program and begun to investigate the code, using the code conversation template. During this lesson the students will complete the code conversation and, time allowing, determine if they can make effective modifications to the program. The final lesson in the whole scheme focuses more on modification and making, so there will be a good chunk of time available should the students not get time during this lesson.

## Learning objectives

* To successfully set up a micro:bit/energy meter circuit and flash code to the micro:bit
* To be able to read and interpret a program containing user defined functions
* To be able to understand iteration and apply it effectively in a functioning program
* To successfully use trace tables to test a program

## Keywords

Solar store, voltage, super-capacitor, terminals, bulb, harvested, circuit, GPIO, pins, iteration, loops, trace tables, testing

## Preparation

**Subject knowledge:**

This lesson is suitable for a class who already has a basic knowledge of programming using python, having previously used iteration in programs. The coding for the micro:bit is written in a reduced version\* of python called micropython but for practical purposes the syntax will appear the same to the user. In addition the students should have already used the micro:bit, practising simple set up and smaller micro:bit specific programs so that they are used to building and flashing programs. Initially, building the circuit for the “energy meter” is a question of copying a diagram so a practical demonstration of handling the components gently and clipping with alligator clips would suffice. However, we would strongly recommend the teacher accessing the instruction booklet and trying out each of the circuits to get experience of how long charging takes in the available light levels.

**Pedagogical approach:**

The lesson is planned using the PRIMM pedagogy which stands for:

P - Predict

R - Run

I - Investigate

M - Modify

M - Make

For this reason it is important that the class have quick and easy access to the coding of the program which can be accessed via: https://tinyurl.com/skxfbrjw

The very first step is for the students to view the code and attempt to predict what the code does. We use a tool called a **code conversation** which provides the teacher with a conversation style **talkthrough** of the micropython code for you to support your students’ developing knowledge. Occasionally daggers (††) are used to clarify things in the code conversations, instead of asterisks, which could be confused with the multiply symbol.

**Practical set-up and development environment:**

We recommend either using the on-line Python editor (https://python.microbit.org) or Mu (https://codewith.mu). Or you could manage your coding in micro:bit’s own environment <https://classroom.microbit.org/> which enables you to push code to the whole class and manage their activities.

**Resources that you need:**

* Presentation
* Starter worksheet questions
* Starter worksheet answers
* Code conversation full solution
* Code conversation template (level 1 - basic)
* Code conversation template (level 2 - intermediate)
* Plenary worksheet
* Plenary worksheet answers
* Hardware Per pair - 1 micro:bit, 1 Solar panel, 1 Solar store, 1 USB/micro USB cable, 8 alligator clip leads, 1 motor, 1 fan
* Access to appropriate development environment (the on-line Python editor, Mu or micro:bit classroom - Mu is needed for the last lesson in the scheme)

## How the students’ progress is assessed

**Energy meter II**

The presentation itself begins with a slide showing the labels on the micro:bit. The students then copy the significant labels onto their worksheet and answer some hardware related questions. The lesson is interspersed with pair or whole class discussion opportunities. As this is predominantly a practical lesson, observing the students’ success in flashing the program, compiling the circuit and making small successful modifications to the program will also enable you to assess their progress. The learners also complete a worksheet at the end of the lesson which demonstrates their understanding of key principles conveyed during the lesson. Because of the complexity of the circuit and needing to leave some time for charging the lesson is split into two with the code conversation started in part I and completed in part II with time left for the modification part of the PRIMM methodology.

## Plan (with approximate timings)

| Starter activity 8 mins | **Iteration/Loops diagnostic worksheet**  The slide that greets the students instructs them to complete the iteration/loops diagnostic worksheet. It is specially designed using diagnostic questions to test understanding by having the wrong answers uncover specific misunderstandings. A very typical misunderstanding in this case would be the position of the print statement...is it inside or outside a loop?  After the students have completed the worksheet, quickly go through the solutions and ensure that they correct any wrong answers on their starter worksheets. |
| --- | --- |
| **Objectives and homework**  4 mins | **Objectives and homework review**  Review the objectives and last week’s homework with the students to put the lesson in context. |
| **Activity 1**  9 mins | **Trace tables**  Explain the purpose of trace tables to the students i.e. to test algorithms in order to make sure that no logical errors occur while the calculations are being processed. The slide transition produces an animation effect so that you can take the students through the trace table line by line, showing how each of the variables change as you go through the loops within the sample program. |
| **Activity 2**  7 mins | **Run: students compile the circuit, flash and run the code**  Share the slide with instructions on the board to enable the students to build the circuit, flash and run the code. In the previous lesson they have already completed the predict phase, so a quick recap about the functionality of the circuit may be all that is needed initially. |
| Activity 3 13 mins | **Investigate (part II): teacher and students use correct terminology to identify the syntax and features of the program**  Using the A4 worksheet, the learners attempt to complete the second part of the code conversation (from line 12 to the end of the program) matching the correct explanation to each line of code.  Using the slides talk through the different steps in the program. At this stage it is very important to stress the “why” of the code as well as the “what”.  What  - sets a variable called override which is initially set to the boolean value False.  Why  - The override variable ensures that the fan\_needed and fan\_not\_needed functions are overridden by the button pressed functions to allow the user to ignore the readings of storage and temperature and simply switch the fan on and off. |
| Activity 4 10 mins | **Modify: students modify the code to create a new program**  Students are invited to modify the program by first declaring what their new program should do, then by making the modification, saving the new program and testing it by flashing it to the micro:bit.  Suggestions for achievable alterations:-  Changing the brightness any of the LEDs  Find out what happens if you don’t use display.clear()  Change the name of any of the variables or parameter  If you have any students who want to explore the use of images on the micro:bit in response to the microphone input this page provides a list of images: https://microbit-micropython.readthedocs.io/en/v1.0.1/tutorials/images.html |
| **Plenary**  8 mins | **Energy meter II plenary worksheet**  The students are invited to populate a longer trace table which incorporates a while loop and a for loop. The worksheet also gives students the opportunity to make a small suggestion to the code which would change the output considerably. |
| **Homework**  2 mins | **Trace table template**  The students are given an empty template in order to design their own short program and produce a trace table for the program. It may be advisable to let the students take their corrected plenary sheets home so that they can have a useful exemplar. |

## The Author

This lesson plan and all its parts were created by Dr. Paula Beer of Beer Academic Consultancy in collaboration with Monk Makes Ltd.



Dr Paula Beer has taught Computer Science and IT education to new and established teachers since 2007. Her own research has focused on the use of play and collaboration in computer science. She enjoys supporting practicing teachers by designing accessible lesson planning materials to get students engaged in computer science through play and collaboration. Paula has also produced educational materials for The Raspberry Pi Foundation, been a secondary school teacher, written a successful book (Hello App Inventor!) and has previously worked in IT project management for a media company and for the NHS.