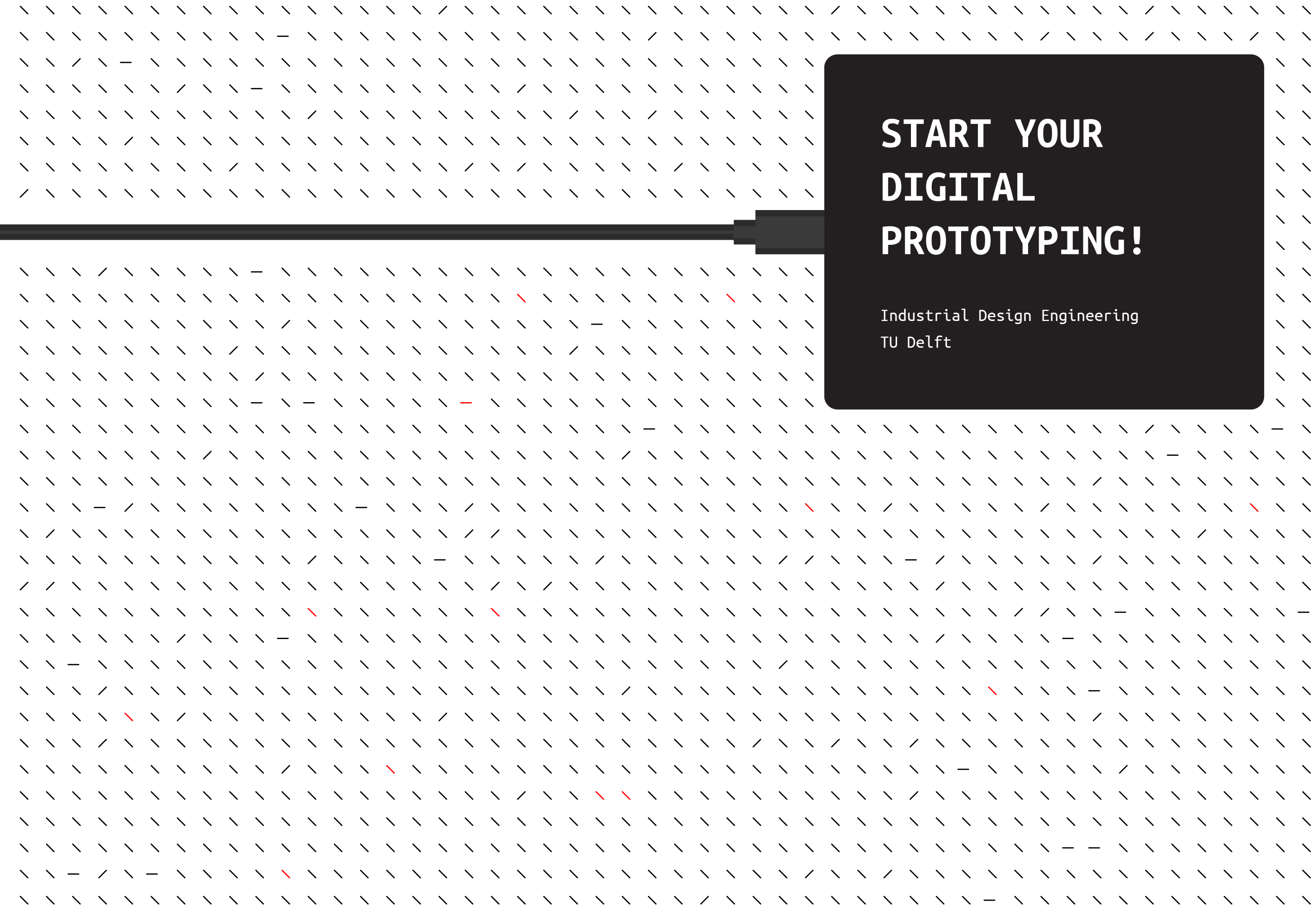


START YOUR DIGITAL PROTOTYPING!

Industrial Design Engineering

TU Delft

TABLE OF CONTENTS

Greeting = 'Hello and welcome!'

This booklet is part of the Connected Interaction Kit designed by students and teachers at the Faculty of Industrial Design Engineering of the TUDelft.

As a design student you have to learn many skills but you have limited time, you cannot be an expert in all. At the same time you want to feel confident when talking to peers and engineers about realising your vision of how technology should influence the daily life of people. With this kit you simply start to build this confidence.

The composition of this kit, the tools and the procedure you use to create technology experiences are carefully selected and tuned to your needs. It is easy to get started, some barriers existing in similar kits on the market are removed. At the same time, the kit is also versatile and extendable, it can grow with you as you build your skills and confidence.

With this kit you can create a great range of technology mediated experiences. You can create anything from a small wearable device to interactive architecture, from a simple interactive lamp on your desk to a connected system of lamps illuminating your friends all over the world.

Have fun and be curious while exploring!

COMPONENT OVERVIEW

Grove Components

01 Chainable RGB LED
Output - typical RGB light can vary colour as well as brightness, can be chained into a LED strip with additional Chainable RGB LEDs

02 Touch Sensor
Input - detects if touched or not, can detect touch through thin material

03 Vibration Motor
Output - vibrates on a scale from soft to harsh, comparable to the one in your phone

04 Piezo Buzzer
Output - produces an alarming beep, can also be used to play simple melodies

Input - detect sound, can hear loud noises relative to silence

05 LED Pack
Output - monochrome light on or off, color depending on the LED in the socket

06 Servo
Output - movement, can rotate it's arm like the windshield wiper of a car

Custom Components

07 Photo Resistor
Input - can detect light intensity on a scale from low light to bright light

08 Button (Tactile Switch)
Input - detect if button pressed down or released

09 Tilt Switch
Input - motion detection, detects between vertical vs. horizontal orientation depending on the placement of the sensor

10 Temperature Sensor
Input - detects temperature variations on a scale from cold to hot

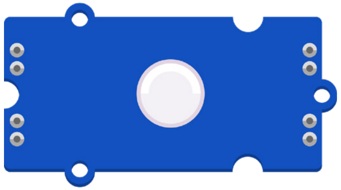
11 Rotation Potentiometer
Input - detect rotation changes between 0 and 270 of the central axis

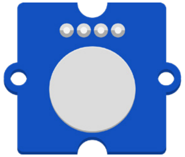
Other

12 Time of Flight Distance Sensor
Input - detects objects in front of the sensor by a distance value between 0.3m and 2.0m

13 ItsyBitsy M4 Express
The micro controller at the heart of you projects, it reads inputs, processes data and controls the outputs (full specs online)

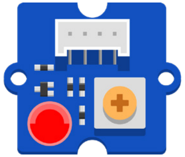
14 Bitsy Expander
Expansion board for the ItsyBitsy M4 Express, it has a number of plug-and-play connectors for quickly interfacing of components and a WiFi module - (more information on the backside of the board)

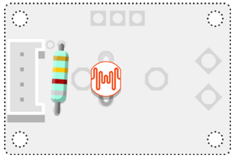

01

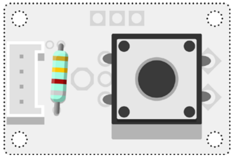

02

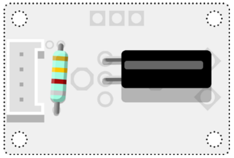
03

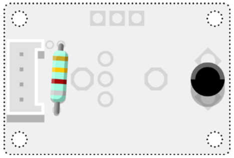
04

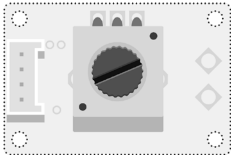

05

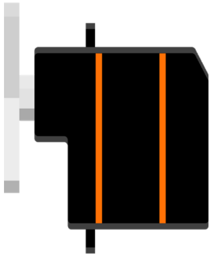

07

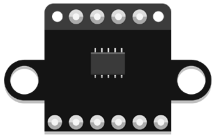

08

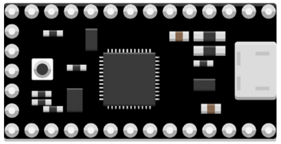

09


10

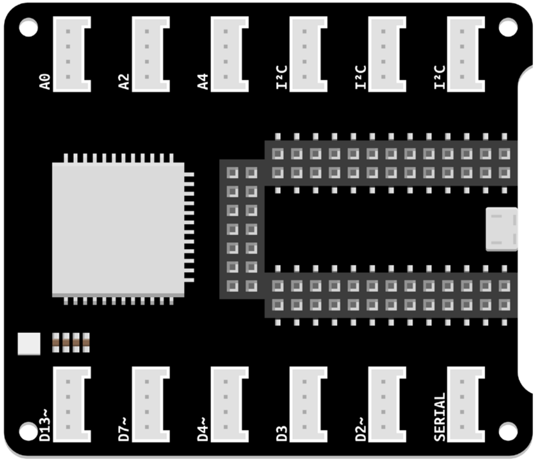

11


06


12


13

14



15 Micro USB Cable – 100 cm
Used to connect the micro controller to your computer for programming and providing power

16 Breadboard
Used for more basic operations (making electrical circuits) interfacing components without soldering

17 Grove-to-pin-cable
Convenience conversion cable to connect a Grove component to the breadboard

18 Grove cable
Connect Grove components to the Bitsy Expander

19 Button caps
Cap fits on top of the tactile with for a nicer haptic experience

20 Resistors (150 & 10k Ohm)
Used for making custom components or to construct electrical circuits on the breadboard

21 Jumper wires
Used to construct electrical circuits on the breadboard and to connect these to the Bitsy Expander double headers

Input though

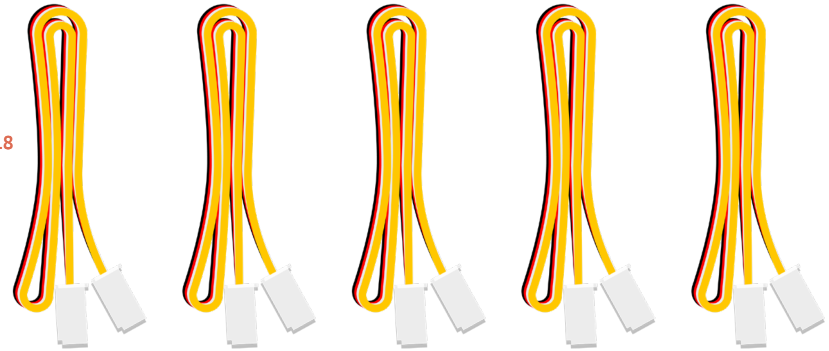
TOUCH PRESS	⟶	02 Touch Sensor, 08 Button (Tactile Switch)
WHOLE BODY GESTURE	⟶	12 ToF Distance Sensor
MOVEMENT	⟶	09 Tilt Switch, 11 Rotation Potentiometer
VISION	⟶	07 Photo Resistor
WARMTH	⟶	10 Temperature Sensor
SOUND	⟶	04 Piezo Buzzer

Output produced

HAPTIC	⟶	03 Vibration Motor
MOVEMENT	⟶	06 Servo
VISION	⟶	01 Chainable RGB LED, 05 LED Pack
SOUND	⟶	04 Piezo Buzzer, 03 Vibration Motor

17

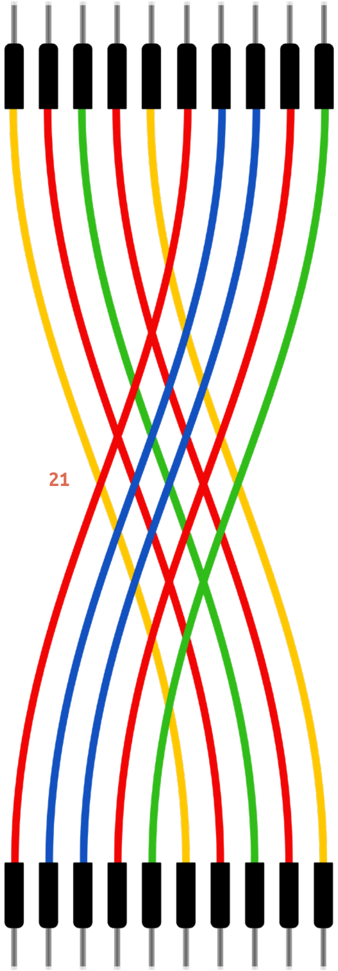
18



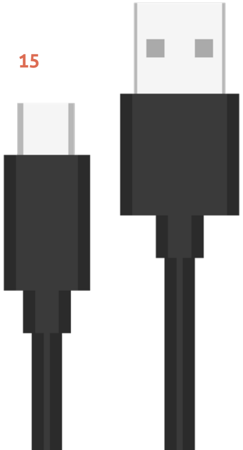
19

20

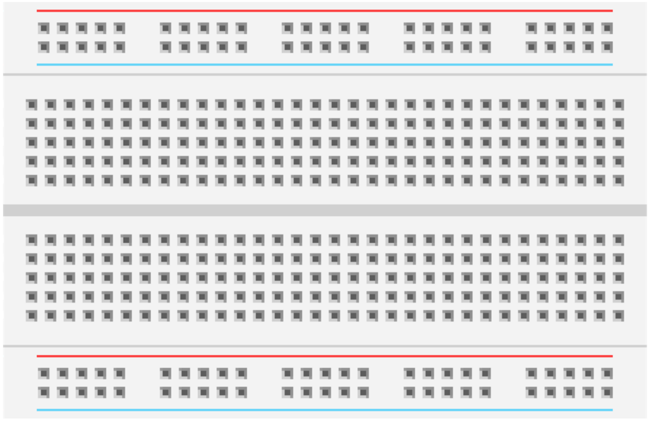
21



15



16

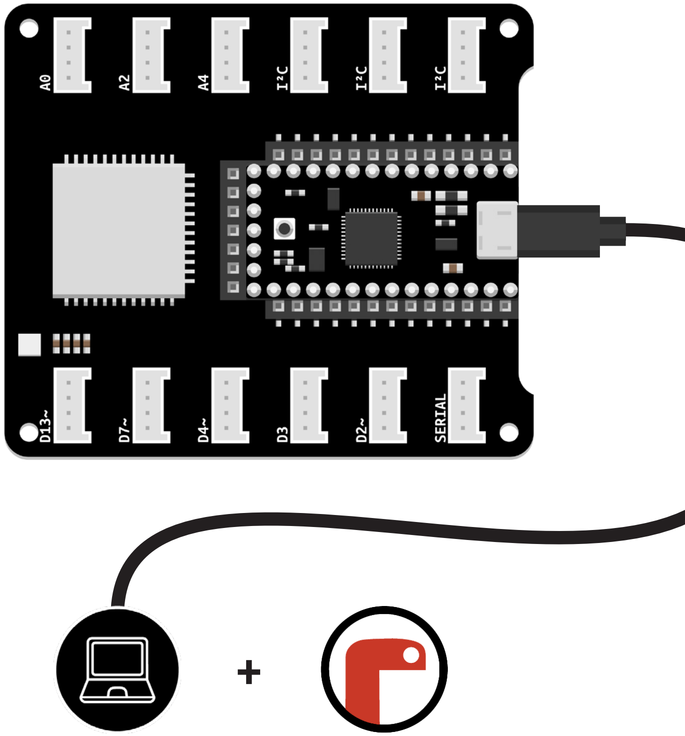


TUTORIAL 1: SET UP AND LET THERE BE LIGHT

In order to make your prototypes behave, you need to create a program and store that on the ItsyBitsy. You need the following equipment to support this:

1. **ItsyBitsy + Expander**
   You connect the ItsyBitsy and Expander to your computer with the USB cable in the box. Once a new program is saved to the ItsyBitsy that program will run. The ItsyBitsy will remember the last saved program even if you disconnect it from your computer.

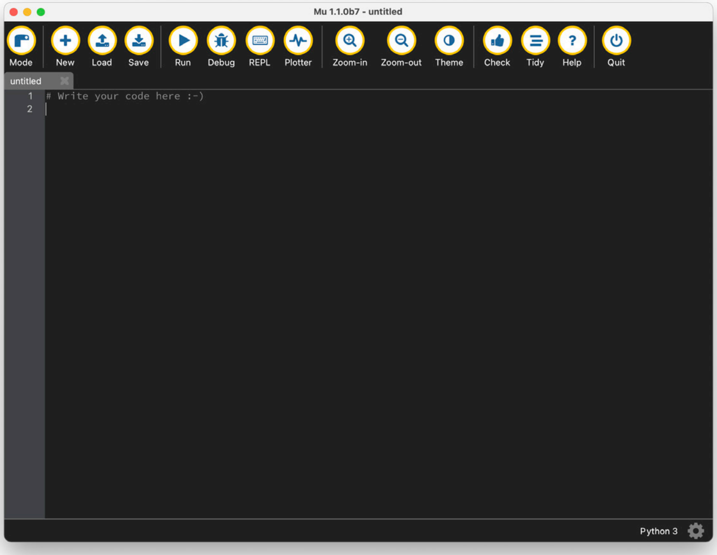
2. **Your computer + program editor**
   You create the program by writing instructions in a program editor. Any text editor software on your computer will do but in this booklet we choose the `Mu Editor`. It originated in education and is open source software available as a free download. The `Mu Editor` also has some nifty features to support you in the process. It helps you to write the instructions correctly and helps you figure out what goes wrong. For sure it will go wrong and that is when you learn ;-)



Step 1: Installing the MU editor

First prepare your computer and set everything up so you can write your first program.

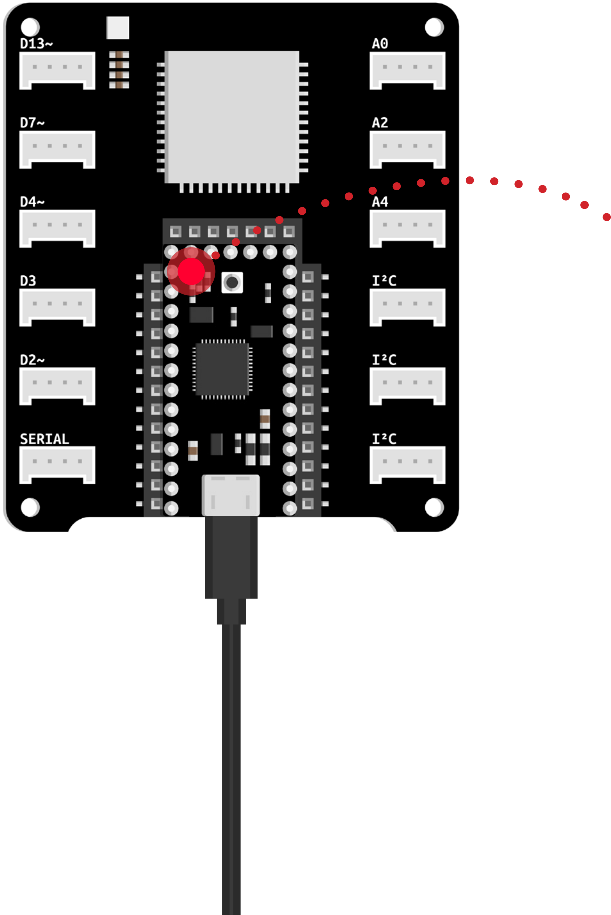
1. Go to the website *codewith.mu*
2. Click on download and select your operating system (Mac OSX or Windows)
3. Install Mu by opening the downloaded file and follow the instructions that appear on your computer screen. If you need help, there is a step-by-step explanation of the installation process on the Mu editor website (click "Instructions" next to the download button for each operating system).



```
# Write your code here :-)
```

Step 2: Connecting the ItsyBitsy to your computer

1. Take the USB cable from the Connected Interaction Kit and connect the ItsyBitsy to your computer. The ItsyBitsy will show up on your computer as a disk volume called `CIRCUITPY`. You should be able to see it in your Finder window (MacOS) or Explorer window (Windows).
2. Open Mu, click on `Load` in the toolbar at the top of Mu's window, select the file `code.py` in the memory volume `CIRCUITPY`. We have already stored this program on the ItsyBitsy when we prepared your kit. This program makes your ItsyBitsy blink its internal LED.



Step 3: Changing the way your ItsyBitsy behaves

1. Take a look at the `code.py` program in Mu's window. Can you understand how the blink behaviour is created by reading the instructions in the program?
2. Can you change the blinking frequency of the internal LED? To do so, play around with the values specified in the parentheses of `time.sleep(1.0)`. Try to make it blink slower or faster. Press the `Save` button in the toolbar of Mu's window to store and run your new program on the ItsyBitsy.
3. Show the `Serial Monitor` panel in the Mu window (click the toolbar button with the label `Serial`). The `Serial Monitor` is an important tool as it can show messages from your program and the `Python Interpreter`.
4. Can you change the message that is now printed every time the LED turns on and off? To do so change the two `strings` in the parentheses of the `print` instructions. Don't forget to press the `Save` button in the toolbar of Mu's window to store and run your new program on the ItsyBitsy.

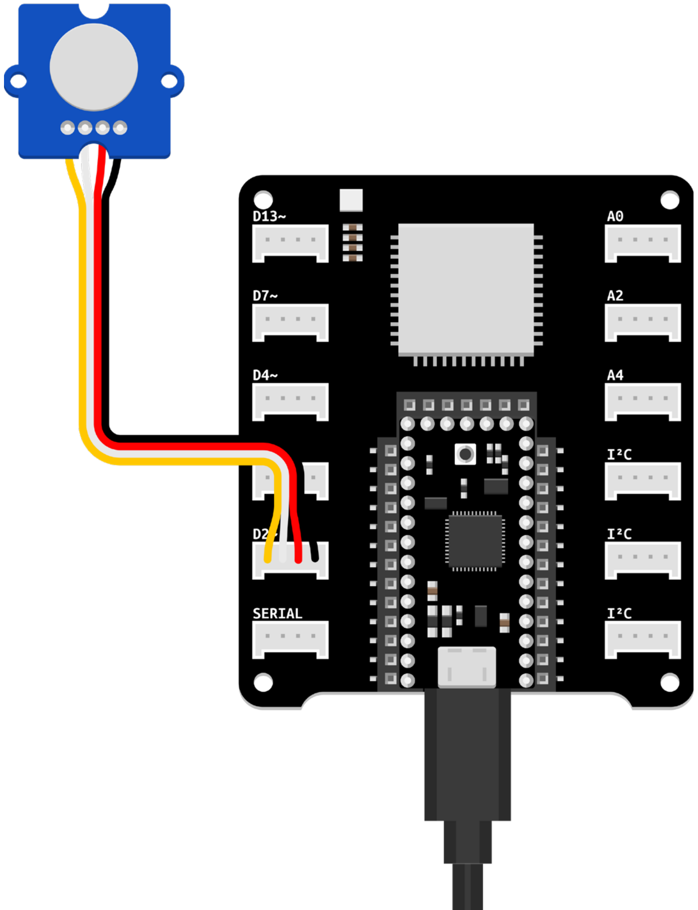
```python
# The essential code for tutorial 1
import board
import digitalio
import time

LED = digitalio.DigitalInOut(board.D13)
LED.direction = digitalio.Direction.OUTPUT

while True:
    time.sleep(1.0)
    LED.value = True
    print("LED is on")

    time.sleep(1.0)
    LED.value = False
    print("LED is off")
```

TUTORIAL 2: ADDING INPUT AND OUTPUT

In Tutorial 1 you changed the blink behaviour of your ItsyBitsy by changing the program. In this tutorial you will create a program with a complete interaction cycle, from "looking at" user actions to effecting changes in the behaviour - your first complete interactive prototype!

Your Connected Interaction Kit comes with a variety of components. Some can read the actions of a user (`inputs` or sensors) others can give feedback (`outputs` or actuators) to effect a change in behaviour of the prototype. Perhaps prompting the user to another action, causing different feedback, etc. Interaction is a process that unfolds over time in a choreography between user and prototype.

The design of the interaction in this case is simple: when a user reaches for the Touch Sensor, the Vibration Motor starts vibrating. When the Touch Sensor is released, the Vibration Motor stops vibrating.



Step 1: Connecting an input component

The program used in this tutorial is similar to the program used in Tutorial 1. This time around you will type it yourself from scratch to better understand what you are doing.

1. Take the Touch Sensor (02) and a Grove cable (18) out of the box and connect it to pin `D2` of the Expander Board, as shown in the illustration on the left.

2. Start Mu and open the file `code.py`. Delete all the program statements so you end up with a blank file. If you want to save the previous code you can copy it out first and paste it somewhere you can always find it. Alternatively you can duplicate `code.py` and rename the copy for example to `code_blink.py` - in this way you create a number of different behaviours ready for use on your `CIRCUITPY` memory volume.

3. In the first three lines of the program you announce to the `Python Interpreter` that you will use libraries created by other programmers. The `board` library makes it possible to conveniently access the pins of your microcontroller. The `digitalio` library makes it possible to work with components that can be in two different states: on or off (`True` or `False`). The `time` library make it possible to use timing functionality in your program.

4. Although the Touch Sensor is already physically attached to the microcontroller, it does not do anything yet. We need to declare a variable point it to the proper pin and define it as an input.

5. Finally you add a `while loop` that continuously reads the state of the touch sensor (`sensor.value`) and prints it to the `Serial Monitor` so you can verify that it is working.

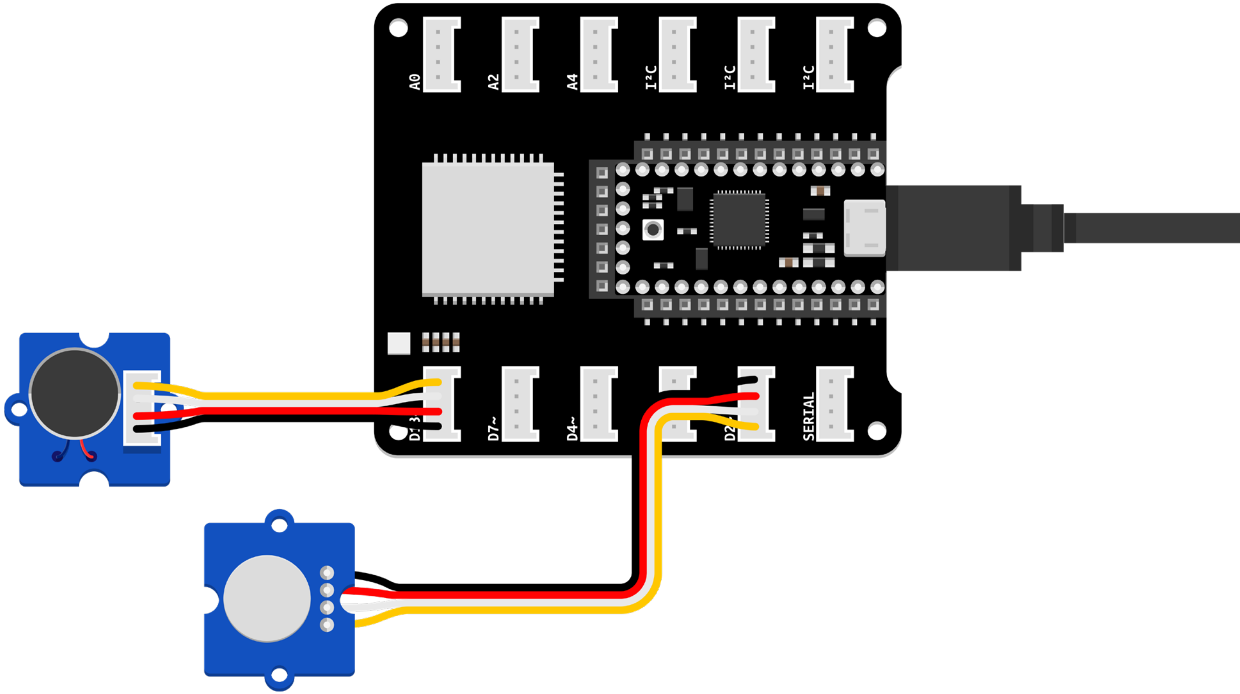
```python
import board
import digitalio
import time

sensor = digitalio.DigitalInOut(board.D2)
sensor.direction = digitalio.Direction.INPUT

while True:
    print(sensor.value)
    time.sleep(0.1)
```

Step 2: Connecting an output component

It is time to add the output component and finish your first interactive prototype!

1. Take the Vibration Motor (06) and a Grove cable (18) out of the box and connect it to pin `D13` of the Expander board, as shown in the illustration on the right.
2. To work with the output component in the program you declare a variable and define it as an output.
3. Now recall the interaction design mentioned on page 10. You need to translate this design into Python statements. Luckily here it is straightforward by using an `if/else` statement. You turn the motor on or off depending on the current state of the touch sensor. In the condition part you read the Touch Sensor state (`sensor.value`) and in the remainder of the `if/else` you set the state of the motor to `True` or `False` (`motor.value`).
4. Save the file and try it out for yourself.
5. Try playing around with the program to change the behaviour of your prototype. You can also replace the output with a Piezo Buzzer (04) or the LED Pack (05) and see how the same program can be applied to them. You do not need to rename the motor variable - both new components operate in the same way.

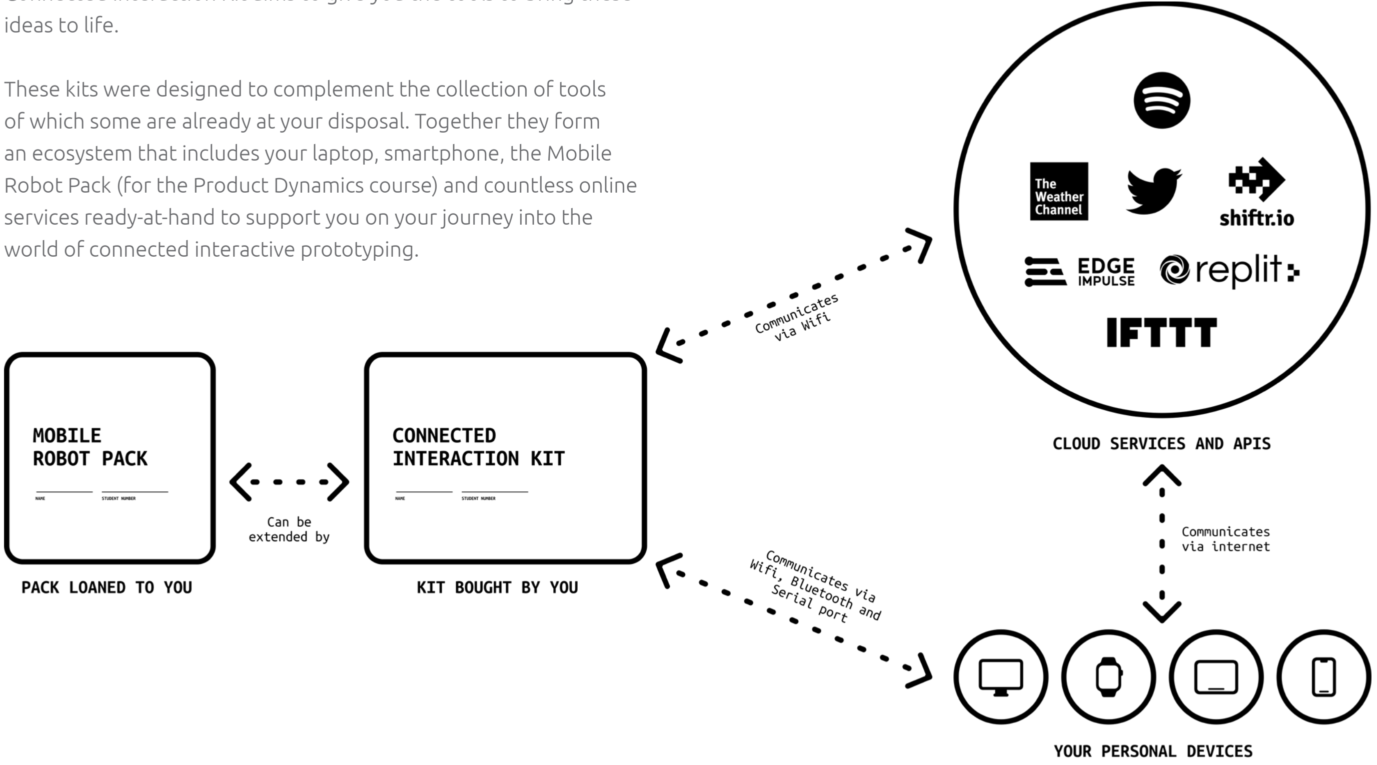
```python
import board
import digitalio
import time

sensor = digitalio.DigitalInOut(board.D2)
sensor.direction = digitalio.Direction.INPUT

motor = digitalio.DigitalInOut(board.D13)
motor.direction = digitalio.Direction.OUTPUT


while True:
    print(sensor.value)
    if sensor.value is True:
        motor.value = True
    else:
        motor.value = False
    time.sleep(0.1)
```


A0
A2
A4
I²C
I²C
I²C
D5
D7~
D4~
D2
SERIAL

DIGITAL PROTOTYPING TOOLS

At the start of your bachelor program you acquired to the Basic Drawing and Modelling Pack. While this pack aims to provide you with the basic tools to draw and model your ideas, the Connected Interaction Kit aims to give you the tools to bring these ideas to life.

These kits were designed to complement the collection of tools of which some are already at your disposal. Together they form an ecosystem that includes your laptop, smartphone, the Mobile Robot Pack (for the Product Dynamics course) and countless online services ready-at-hand to support you on your journey into the world of connected interactive prototyping.

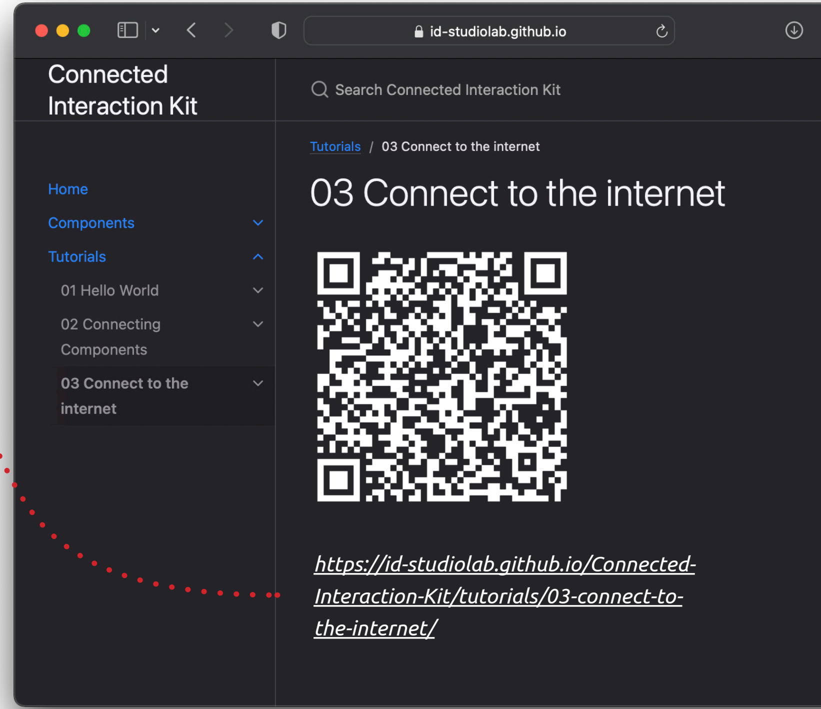


TUTORIAL 3: CONNECTING YOUR ITSYBITSY TO THE INTERNET

In Tutorial 3 you will be able to unlock the connected aspect of your Connected Interaction Kit.

The tutorial is not in this booklet but rather stored online in the accompanying `GitHub repository` that has more tutorials and example programs for working with the kit.

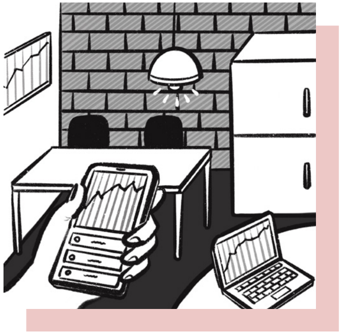
Scan the QR code on the right or enter the link below the code into your internet browser to access the tutorial.


id-studiolab.github.io
Connected Interaction Kit
Search Connected Interaction Kit
Home
Components
Tutorials
01 Hello World
02 Connecting Components
03 Connect to the internet
Tutorials / 03 Connect to the internet

03 Connect to the internet

https://id-studiolab.github.io/Connected-Interaction-Kit/tutorials/03-connect-to-the-internet/

WHERE CAN YOU USE THIS KIT IN THE IDE BACHELOR?



Design Project 2 - Designing Product Services

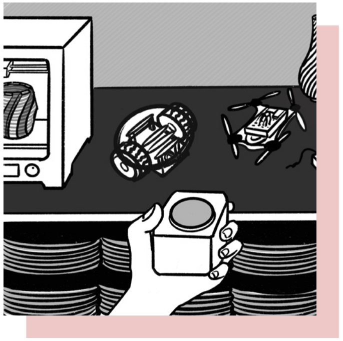
In this course you can employ your digital prototyping tools in the design of Smart Home experiences. In DP2 your will be working in "agile startup" teams to come up with innovative concepts of product-service experiences involving internet-connected home appliances such as colour-changing lamps, “smart" toasters or cat food dispensers. The digital prototyping ecosystem makes it possible to create prototypes that take advantage of laptops, phones, internet services and microcontrollers to deliver your conceptualised product-service experiences, assess their feasibility, and to communicate them using working prototypes.

Digital Product Development

In this course you are introduced to the digital knowledge and skills needed to effectively and responsibly engage throughout the design process of digital products and services. It is about learning to develop software as well as learning about networking technologies. You will learn the basics of Python programming and how organisations implement the software development process. From the digital prototyping ecosystem you will mainly use your laptop and online services like repl.it.

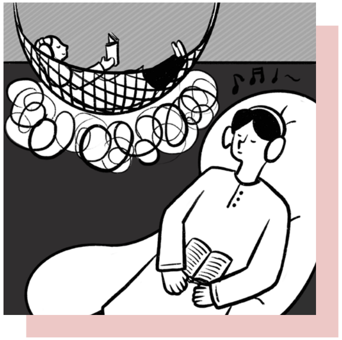
Digital Interfaces

In this course you are introduced to knowledge about Human Computer Interaction. You will learn about the design of interfaces from the perspective of the technology involved. From the perspective of a designer giving form to static and dynamic appearance. And from the perspective of a user interacting with an interface. You will be engaged in several workshops where the digital prototyping ecosystem is employed to build some interfaces examples that you analyse together with coaches and peers.



Product Dynamics

In this course you learn how products involve motion in creating user-experiences and in their functioning. You learn to design with mechanics and electronics, calculate how objects move and understand which forces are applied. From the digital prototyping tools ecosystem you will encounter the Mobile Robot Pack to apply this knowledge for the construction of a robot that moves around to perform a simple task.



Form and Senses

In this course you will learn how to employ multi-sensory experiences in your design process. You can use the tools in the digital prototyping ecosystem to discover how sound design, texture design and light design can influence a user experience. For example the Connected Interaction Kit contains a rich mix of components that enable various multi-modal input and output capabilities.

Interactive Environments minor

In this course you learn about designing interactions at an architectural scale. Conceptualising about meaningful interactions between technology behaviour and the people living in these spatial and interactive environments. You will be engaged in skill building at an individual level as well as engineering a full-scale prototype with a team. In both instances the digital prototype ecosystem offers good starting points. It has enough flexibility to extend to more advanced technology and further enhance the user experience capabilities.

Bachelor End Project

In this course you design for a real-world challenge, a solution showing that you can execute what you have learned. As part of this process you can employ the tools of the prototyping ecosystem to create prototypes to be sure you are getting the design right and communicate to inspire the stakeholders involved about their concept.

Whatever you want to build with it

All these examples of applying tools from the digital prototype ecosystem should make you enthusiastic! You can start with simple prototypes and build skills throughout the bachelor to become a confident designer of interactive concepts that involve technology. Over time your understanding of the technology involved will grow and you will be able to communicate your ideas by demonstrating a working prototype.

The question is: where can't you use these tools?! Once you learned how to work with it, you can use it everywhere in your bachelor and master studies and even apply it to enrich your daily life at home!

TUDelft

Authors
Frederik Ueberschär and Aadjan van der Helm

Layout and graphic design
Frederik Ueberschär and Yeun Kim