Morse Code Transceiver

Project Outline

The main objective of this project is to develop a Morse Code Transceiver (transmitter and receiver) using Arduino Micro-Controller and Python programming language.

Course	GP106: Computing
Batch	E/17
Project Mode	Group - 2/3 members
Duration	4 weeks
Milestones	2 regular and 1 bonus
Marks	15% (regular milestone) + 5% (bonus milestone)
Resources	A complete development kit is given by the department, one per group.

Table 1: Project Outline

Project Specification

0.1 Hardware Setup

This section provides information regarding the device setup. For this, you can refer to Figure (1). Also, please note that this diagram illustrates the **basic component setup** of a transmitter and a receiver without the buzzer circuit.

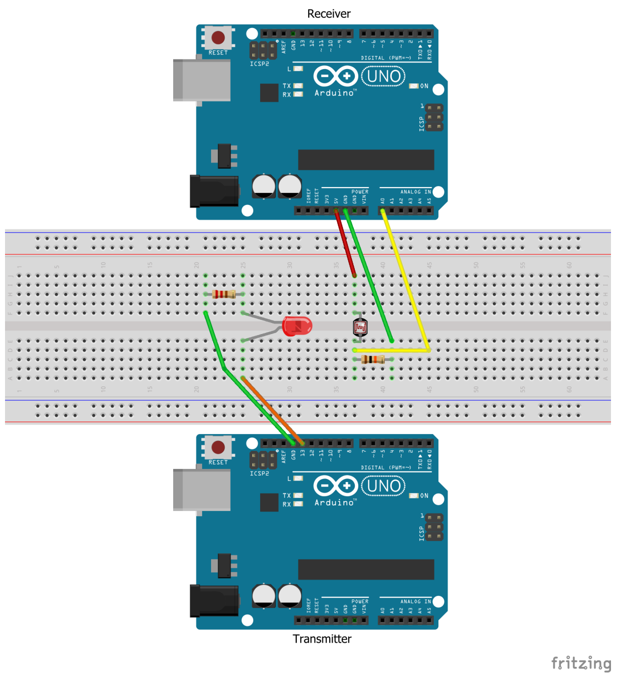


Figure 1: The basic diagram for the setup including a transmitter and a receiver

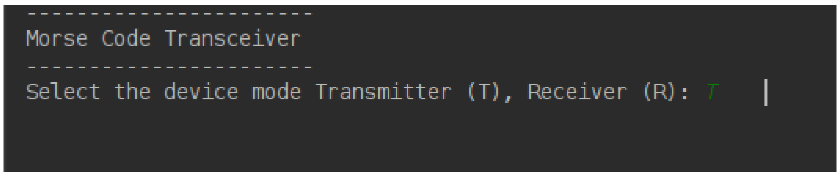
1. **As the transmitting device, you should use an LED** provided in the Arduino development kit.
2. **As the receiving device, you should use an LDR**, which is also provided in the development kit.
3. Furthermore, you should use the **Buzzer from the development kit to emit the received Morse code**.

4. Other than those, if you want additional devices which are not included in the kit, you have to purchase them. (this criteria is applied for the bonus task)

0.2 Algorithm Development

This section provides information regarding the device Algorithm development. For the basic device setup, you should use the Python terminal as a command-line interface. This will be the basic configuration setup. The algorithm for the device (transceiver) should be developed using Python language.

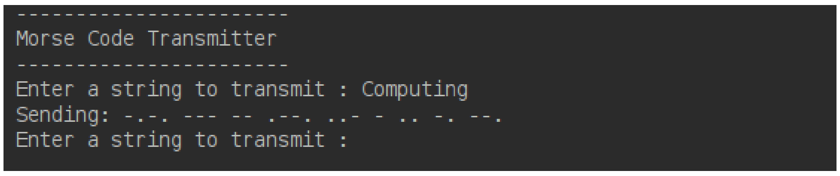
1. The **device operation mode** (whether it's going to be the transmitter or the receiver) should be configurable via the Python command-line interface.



```
------------------------
Morse Code Transceiver
------------------------
Select the device mode Transmitter (T), Receiver (R): T
```

Figure 2: Configuration interface

2. The **transmitting message** provided by the user should be also taken as input via the command-line interface. This can be a single letter, a word or a sentence. Figure 3 shows how the command-line interface should behave in the basic setup. This includes a message entered by the user (*computing*) and the encoded message in Morse code.

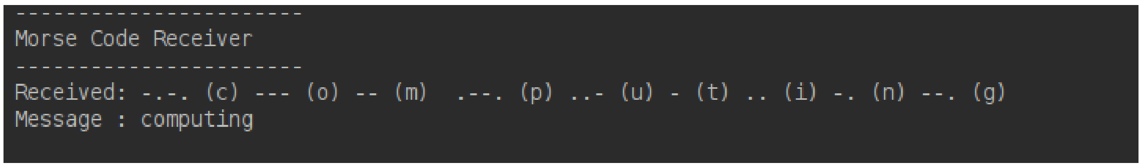


```
-----------------------
Morse Code Transmitter
-----------------------
Enter a string to transmit : Computing
Sending: -.-. --- -- .--. ..- - .. -. --.
Enter a string to transmit :
```

Figure 3: Command line interface for the transmitter

3. While the device is transmitting the data as a Morse code, you should use a Buzzer (which is also provided in the kit) to **emit the transmitting data as a sound** (i.e as a real Morse code device).

4. The **message received via LDR** should be also displayed to the user as shown in Figure 4. For this step, you can use a command-line interface.



```
-----------------------
Morse Code Receiver
-----------------------
Received: -.-. (c) --- (o) -- (m)  .--. (p) ..- (u) - (t) .. (i) -. (n) --. (g)
Message : computing
```

Figure 4: Command line interface for the receiver

5. Please note that these two figures are an example of how your device should interact with the user in basic setup. Therefore, you can have your own interface (maybe even a GUI) with the core functionalities of the system as the final product.

6. For the algorithm development, you should **only use Python language and its libraries** (inbuilt or other). You are **only allowed to use Pyfrimata related Arduino scripts** to configure the device.

7. As the **unit time interval** for a **dot** (.) in Morse code, all groups should use **0.2 seconds** (read the Wikipedia link given in resources). This value should be consistent because in the final evaluation we will be testing your projects by combining two different groups (one as the transmitter one as the receiver).

8. Some resourcefull links for the project:

   - Morse codes: Wikipedia : https://en.wikipedia.org/wiki/Morse_code
   - Python GUI: Tkinter, PySimple GUI
   - Example Projects: Demo Project 1: https://pysimplegui.readthedocs.io/en/latest/
     Demo Project 2 : https://www.instructables.com/id/Morse-Code-Communication-Transmitter-and-Receiver/

0.3 Bonus

Other than these specifications, we have allocated 5% bonus marks for the project (from the final grade). This mark can be achieved by a significant improvement to the project compared to the mentioned setup. Also, please note that this will be evaluated as the third milestone of the project (see: next section). Following list shows some ideas for getting bonus marks.

1. User inputs and outputs using a GUI interface.
2. Configuration of the device using a push-button (mode: transmitter or receiver).
3. Soldered circuit for the device.
4. Or any other impressive approaches/developments.

Milestones

1. Every group should complete project milestones 1 and 2. As mentioned, the 3rd milestone is **an additional one**, and it will be assessed with milestone 2.

2. Since this is a group project, an individual viva session will be conducted to evaluate the contribution of each group member (contribution factor %). Please note that the final marks for the project will be allocated depending on the individual contribution factor (**individual_mark** $= project\_marks \times contribution\_factor$).

3. For the first milestone, you have to develop a simple algorithm for the core functionality (specified in Algorithm development section) of the device. At this stage, you **don't have to implement/design** the bonus part. Bonus milestone will be evaluated at the end as the third milestone.

4. Furthermore, note that the bonus mark will be allocated **only for a significant improvement** to the project.

Milestone	**Deliverables**	**Deadline**	**Marks (%)**
1	Flow chart for the Algorithm (4%), Circuit Diagram (1%)	1st Friday Nov, 4th Monday Nov	5
2	Completed Working Device (5% for each operation mode)	15th Friday Nov, 18th Monday Nov	10
3	Bonus: Modified circuit diagram and the algorithm	15th Friday Nov, 18th Monday Nov	5

Table 2: Project Milestones