Space Concordia – Spacecraft Division

Command & Data Handling (CDH) Intro Task

September 2022

The task consists of two parts, a practical section and a research section. The purpose of this task is to make you familiar with some of the concepts and types of problems you will encounter during your time as a CDH member. Following the completion of the task, we will schedule a review meeting with you to go over your submission and answer any questions you may have.

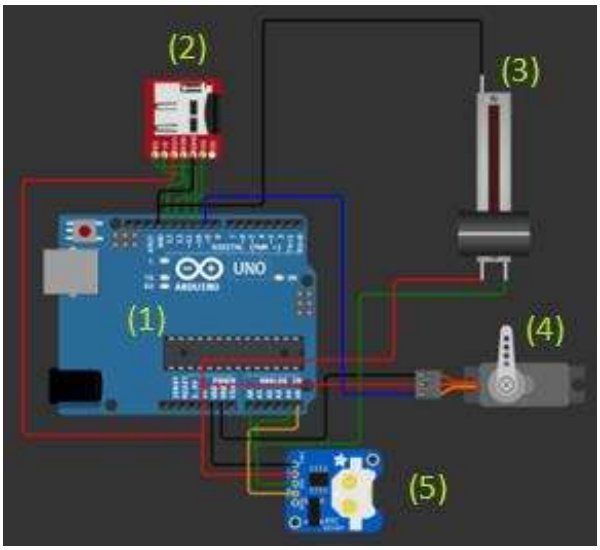
You do not need to have any knowledge of Arduino or embedded software development to be able to complete this task. Follow the resources for the documentation and examples at the bottom of this document. If you have a question, do not hesitate to send me an email and I will answer you as soon as possible.

NOTE: This task does NOT require you to have or purchase any of the hardware components listed below, you are able to complete the entire project using the Arduino simulator linked below. The wiring and selection of the component is already done for you, the only thing you need to do is to implement the code. Before you start writing your code, make sure you create an account on the website and save a copy of the project.

Arduino Simulator: https://wokwi.com/projects/342357001622258260

Practical Section

You are given the following circuit, using an Arduino Uno:



It is composed of the following components:

1. Arduino UNO x1
2. microSD Reader x1
3. Potentiometer x1
4. Servo x1
5. RTC x1

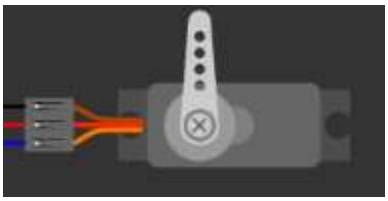
Your task is to control the servo using the potentiometer. When the potentiometer lever is at 100% (all the way to the top), the servo should be pointing up. When it is at 50% (in the middle), the servo should be pointing all the way to the right. Finally, when it is at 0% (its default state, at the bottom), the servo should be pointing downwards.

Once you have completed this, you will want to log (to the console and on a file on the microSD), an integer value for where the servo is facing (see examples below).

Example 1: Potentiometer is at 100% (all the way to the top).

Expected output: `[2022/09/09 13:00:00] Angle: 90 degrees`

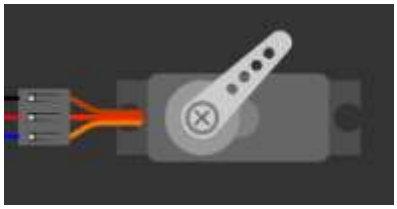
Expected servo:



Example 2: Potentiometer is at 75%.

Expected output: `[2022/09/09 13:00:00] Angle: 45 degrees`

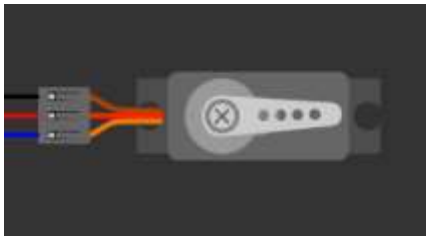
Expected servo:



Example 3: Potentiometer is at 50% (in the middle).

Expected output: `[2022/09/09 13:00:00] Angle: 0 degrees`

Expected servo:



Example 4: Potentiometer is at 0% (all the way at the bottom).

Expected output: `[2022/09/09 13:00:00] Angle: -90 degrees`

Expected servo:

This output should be logged to both the file and the console (serial monitor) every second.

In order to print to the console, you can use the following reference.

[*https://docs.wokwi.com/guides/serial-monitor*]

In order to get the timestamp, you can use the Real-time Clock (RTC) using the RTCLib library.

[*https://docs.wokwi.com/parts/wokwi-ds1307*]

In order to control the servo using the potentiometer, you can use the following reference.

[*https://docs.wokwi.com/parts/wokwi-slide-potentiometer*]

In order to read / write to the microSD card, you can use the following reference (use the SdFat library).

[*https://docs.wokwi.com/parts/wokwi-microsd-card*]

NOTE: The microSD module is not yet complete, so we are not able to see the contents of the file from the Wokwi UI, however you can still use the write functions available to you with the SdFat library.

Research section

1. In the practical section, you were introduced to components that use different types of serial communication protocols. One of which is Pulse-Width Modulation (**PWM**), which is used with the Servo component. Describe how **PWM** works and draw (approximately) what the signal would look like when the potentiometer is at:
    a. 75%
    b. 25%
    c. 100%

2. Another serial communication used in the practical section is **I2C**, which is used with the RTC component. Describe how **I2C** works, and what the advantages are of using I2C are compared to other protocols.
