4CCS1CS1 Computer Systems

Coursework Assignment 2

Josh Murphy and Matthew Howard
Questions? CS1 Stack Exchange

Coursework

This is the second of two coursework assignments in CS1. It is worth 7.5% of the module. You will need to write an assembly program, which will then be submitted to KEATS. There is a base task that most students will be able to complete. The challenge task is more complex and will likely require you to use instructions that we have not explicitly covered in the labs. Both base and challenge tasks will be graded. There is also a 'Just For Fun' task that will not be graded and is worth zero marks. You must work on the assignment individually.

Base Task: Simple Morse Code

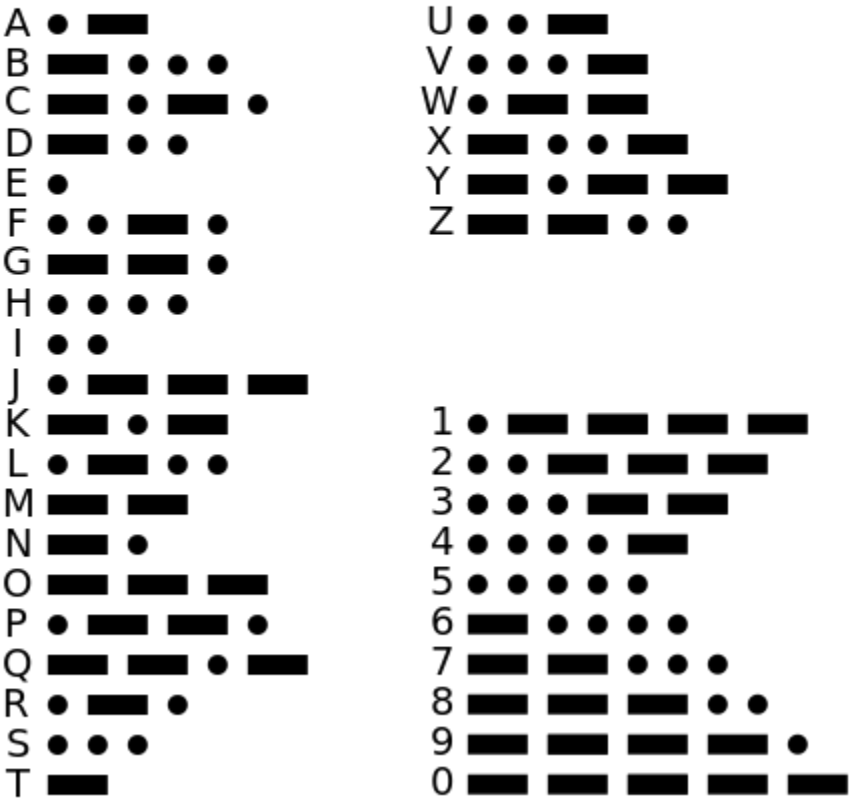
For the program that you will turn in, you will use the circuit that you made in lab 4 to broadcast a short message via Morse code on the LEDs.

What is Morse code? Morse code is a method of transmitting text information as a series of on-off tones, lights, or clicks that can be directly understood by a skilled listener or observer without special equipment. (`https://en.wikipedia.org/wiki/Morse_code`).

Below is the International Morse Code Roman alphabet.

International Morse Code

1. The length of a dot is one unit.
2. A dash is three units.
3. The space between parts of the same letter is one unit.
4. The space between letters is three units.
5. The space between words is seven units.



Your base program will blink a 3 letter sequence in Morse code on the LEDs. You three letter sequence is the first three letters of your first name.

- For expample, Charles Babbage's code would be `CHA`.
- If your first name is less than three characters, you should use the first 3 characters of your first name concatenated with your surname.

So that we can perceive the Morse code, we will use a unit length of **200 milliseconds (ms)**. This means the duration of a *dot* is **200 ms**, and a *dash* is **600 ms**.

Make sure that:

1. Your program is called `morse.s`.
2. Your program writes the Morse code sequence to the `PB0` pin of `PORTB`. This is the same set-up as in the other programs you wrote in Lab 4.
3. Your program will continually loop through the 3-letter sequence.
4. Assume that there is **no** word break after your 3 letters, just repeat the 3-letter sequence.

For example, if your sequence was `ABC`, then your program would run as follows:

1. Turn ON the LED for **200 ms** for the first *dot* of the letter `A`
2. Turn OFF the LED for **200 ms** for the inter-part space of the letter `A`
3. Turn ON the LED for **600 ms** for the first *dash* of the letter `A`
4. Turn OFF the LED for **600 ms** for the inter-letter space between the letters `A` and `B`
5. Turn ON the LED for **600 ms** for the first *dash* of the letter `B`
6. ... and so forth
7. Until the last *dot* of letter `C`
8. Turn OFF the LED for **600 ms** for the final inter-letter space.
9. *Loop back to the beginning of the Morse code sequence*

Challenge Task: Odd, Even, modulo 5

Extend your `morse.s` program as follows.

- The morse code sequence should loop 50 times (1–50).
- On odd iterations (1, 3, 5, ..., 49) your three characters should be displayed in their normal order.
  - *e.g.* `ABC`
- On even iterations (2, 4, 6, ..., 50) your three characters should be displayed in reverse order.
  - *e.g.* `CBA`

Using comments, you should explain how you have implemented the check of whether the iteration is even or odd.

Once you have this behaviour working, you should again extend your program as follows.

- On iterations that are divisible by 5 (5, 10, 15, ..., 50) your program should display a '5' after what would normally be displayed on that iteration.
  - *e.g.* `ABC5` or `CBA5`

Just for fun: LED blinking

Only attempt this challenge if you have completed the previous tasks. This section is worth **no marks**, but it is your chance to practice and show off your skills!

Once the morse code sequence has terminated, your LEDs should display a repeating pattern.

It is up to you to create a pattern for the LEDs. The more technically impressive the pattern the better!

For example, you could implement a ping-pong like pattern, where only a single LED is on at a time, and it appears to move back and forth across the LEDS.

- $1000 \rightarrow 0100 \rightarrow 0010 \rightarrow 0001 \rightarrow 0010 \rightarrow 0100 \rightarrow 1000 \rightarrow ...$

You should not submit your work for this task.

Submission instructions

- You should submit a single `.s` file named `morse.s` via KEATS.
- **DO NOT** put your program in a .zip, .7z, or any other archive, **DO NOT** submit your program as a `.doc`, `.pdf`, or any other format other than `.s`. Doing so will result in losing marks for your submission.
- You must submit by 9/12/2019 by 5pm.
- Your submission will be marked for correctness, organisation, style, and readability.

Returning your Arduino kit

- Receiving a grade for this submission is dependant on you returning your CS1 lab kit. **If we do not receive your lab kit you risk getting zero for this assignment.**
- Your lab kit should be returned with all the components you have been given.
- When you return the kit, your circuit does not have to be assembled, nor does your program have to be loaded onto the Arduino.
- If there are any missing or damaged components please include a paper note in the kit to let us know what is missing or broken.
- **We will arrange drop-off sessions in Semester 2 for you to return your lab kit.** This means you can keep your lab kit over the Christmas break.