

## Lesson 7 – Ciphers, LCD screen and GPS

### Setting the scene

In this project you are going to learn about the importance of ciphers and how they can be used to protect data during transmission. You will learn how to connect an LCD screen to the Arduino using a breadboard in addition to being able to read a range of different GPS data from the MKR GPS shield.

### Success criteria

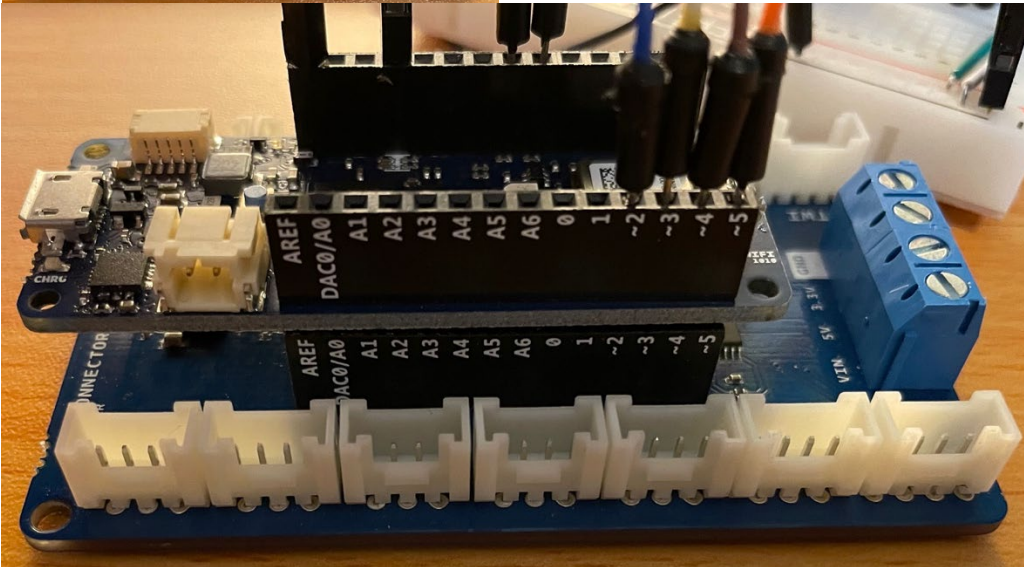
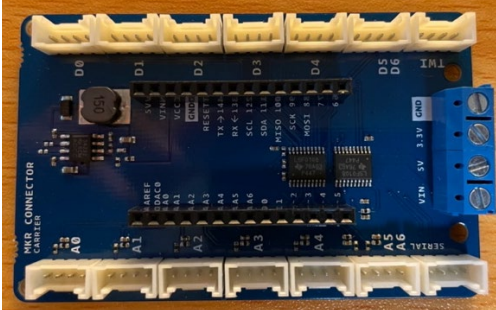
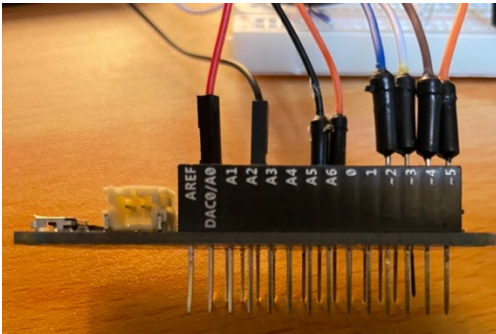
- Create a circuit using a breadboard
- Connect an LCD screen and to output text
- To be able to use a GPS sensor to record a location

### Step 1 – Building the circuit

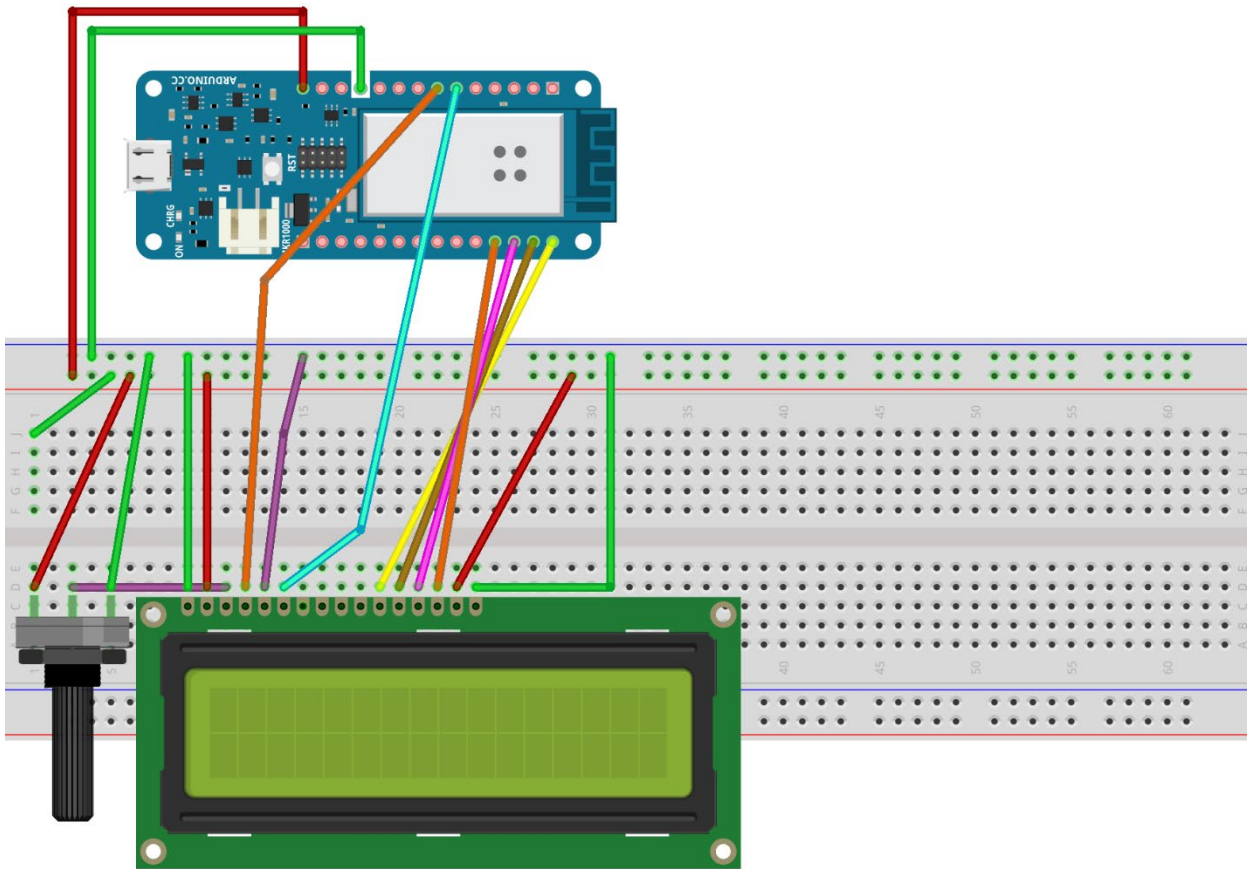
You are going to connect the whole device and then test the LCD screen and GPS sensor separately. You should initially plug the MKR Arduino into the connector carrier.

### Pro-tip

The Arduino has a number of metal connectors exposed on the underside. As part of this project, you will be using your device outside to find a location in a treasure hunt. To help protect the exposed connectors from becoming bent, it is recommended that the Arduino is connected to the connector carrier.



You should then connect the circuit by following the circuit diagram below.

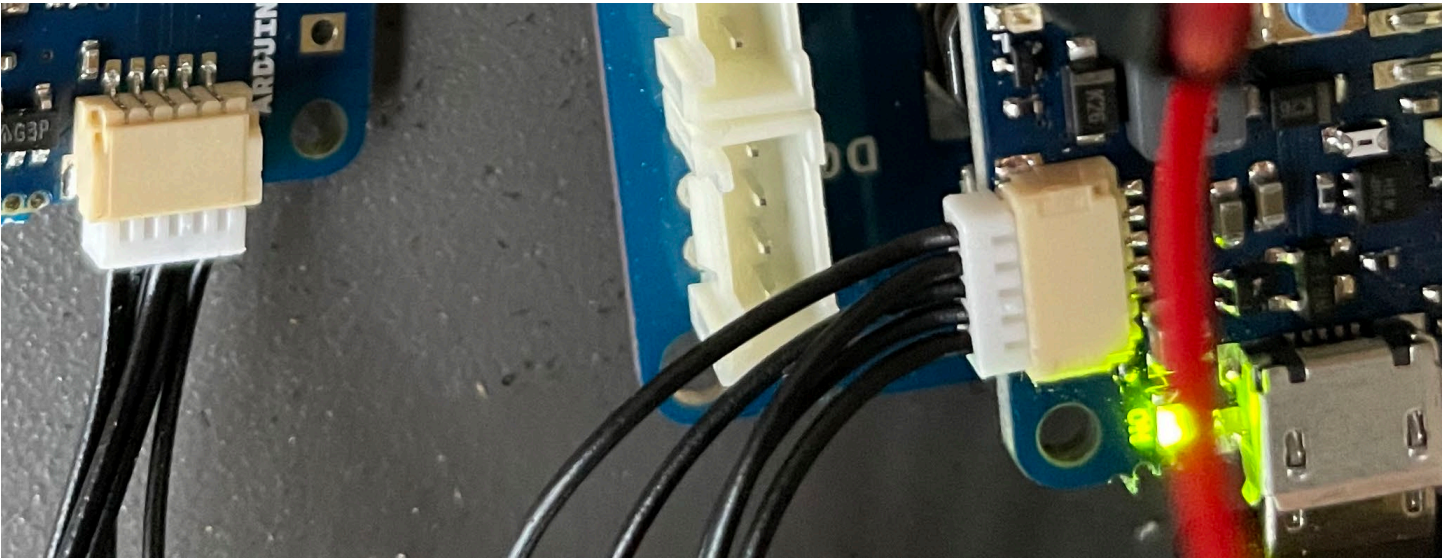


fritzing

### Pro-tip

In order to make the circuit as neat as possible we recommend using shorter length cables.

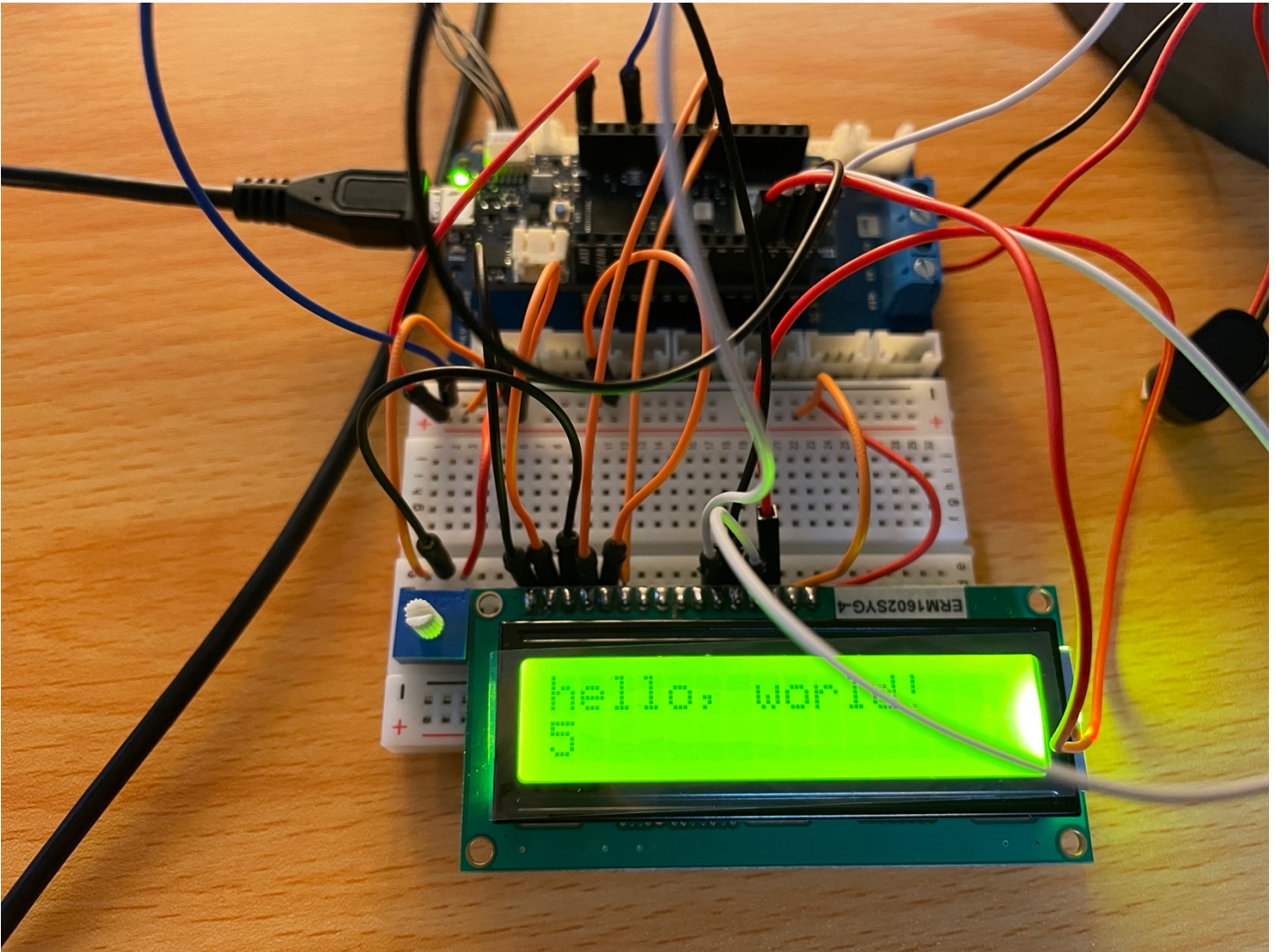
Finally, connect the MKR GPS Shield using the provided cable.



## Testing the LCD screen


To test the LCD screen is working correctly, upload the sample file, `LCD_Screen_Test.zip` to your Arduino. If it is functioning correctly, you should see “hello, world!” printed on the screen. If it is not functioning correctly, check that your cables are connected correctly by following the instructions at the top of the sample file.





## Testing the GPS receiver

It is now time to test that your GPS receiver is functioning correctly. You should upload the sample file `GPSLocation_sample_No_LCD.zip`. If it is working correctly you should see a range of coordinates printed out onto monitor in your IDE.



```
standby
delay .....
wakeup
wait location ... 3514 ms

Location: 51.3175278, -2.1794775
Altitude: 85.30m
Number of satellites: 7

standby
delay .....
wakeup
wait location ... 3885 ms

Location: 51.3176193, -2.1796472
Altitude: 22.20m
Number of satellites: 6

standby
delay .....
wakeup
wait location ... 4911 ms

Location: 51.3174553, -2.1793211
Altitude: 82.00m
Number of satellites: 6

standby
delay .....
```

### Pro-tip

If you do not receive GPS data, you should test your device outside as buildings will often stop the GPS signal from being received via the sensor. Please note that it may take up to 5 minutes for your device to start to receive GPS readings.

## Displaying GPS readings on the LCD screen

You are now going to add to your code to display the GPS coordinates onto the LCD screen. Please note that it will be necessary to remove 'while (!Serial)' loop in your in order to be able to use your device whilst not connected to the computer. The reason for this is that the loop will never exit as a connection with the USB will not be able to be established. The complete code is provided below.



GPSLocation\_sample\_wi

```
1  /*
2   GPS Location
3
4   This sketch uses the GPS to determine the location of the board
5   and prints it to the Serial monitor.
6
7   Circuit:
8   - MKR board
9   - MKR GPS attached via I2C cable
10
11  This example code is in the public domain.
12  */
13
14  #include <Arduino_MKRGPS.h>
15  #include <LiquidCrystal.h>
16
17  LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
18  //const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;
19  //LiquidCrystal lcd(rs, en, d4, d5, d6, d7);
20  // initialize the library with the numbers of the interface pins
21
22  void setup() {
23    // initialize serial communications and wait for port to open:
24    Serial.begin(9600);
25    //while (!Serial) {
26    //    //; // wait for serial port to connect. Needed for native USB port only
27    //}
28
29    // If you are using the MKR GPS as shield, change the next line to pass
30    // the GPS_MODE_SHIELD parameter to the GPS.begin(...)
31
32    if (!GPS.begin()) {
33        Serial.println("Failed to initialize GPS!");
34        while (1);
35    }
36
37
38  }
39
40  void loop() {
41
42    Serial.println("standby");
43
44    delay(5000);
45
46    Serial.print("delay ");
47
48    for (int i = 0; i < 10; i++) {
49        delay(1000);
50
51        Serial.print(".");
52
53    }
54
55    Serial.println();
56
57    Serial.print("Location: ");
58
59    // wake up the GPS
60
61    Serial.println("wakeup");
62
63    GPS.wakeup();
64
65    Serial.print("wait location ... ");
66
67
68  }
```



```
68
69 // wait for new GPS data to become available
70
71 unsigned long startMillis = millis();
72
73 while (!GPS.available());
74
75 unsigned long endMillis = millis();
76
77 Serial.print(endMillis - startMillis);
78
79 Serial.println(" ms");
80
81 // read GPS values
82
83 float latitude = GPS.latitude();
84
85 float longitude = GPS.longitude();
86
87 float altitude = GPS.altitude();
88
89 int satellites = GPS.satellites();
90
91 // print GPS values
92
93 Serial.println();
94
95 Serial.println("Location: ");
96
97 Serial.println(latitude, 7);
98
99 //lcd.println(latitude, 7);
100
101 Serial.println(", ");
102
103 Serial.println(longitude, 7);
104
105 Serial.print("Altitude: ");
106
107 Serial.print(altitude);
108
109 Serial.println("m");
110
111 Serial.print("Number of satellites: ");
112
113 Serial.println(satellites);
114
115 lcd.begin(16,2);
116
117 lcd.setCursor(0,0);
118
119 lcd.print(longitude,7);
120
121 lcd.setCursor(0, 1);
122
123 lcd.print(latitude,7);
124
125 delay(5000);
126
127 Serial.println();
128
129
```

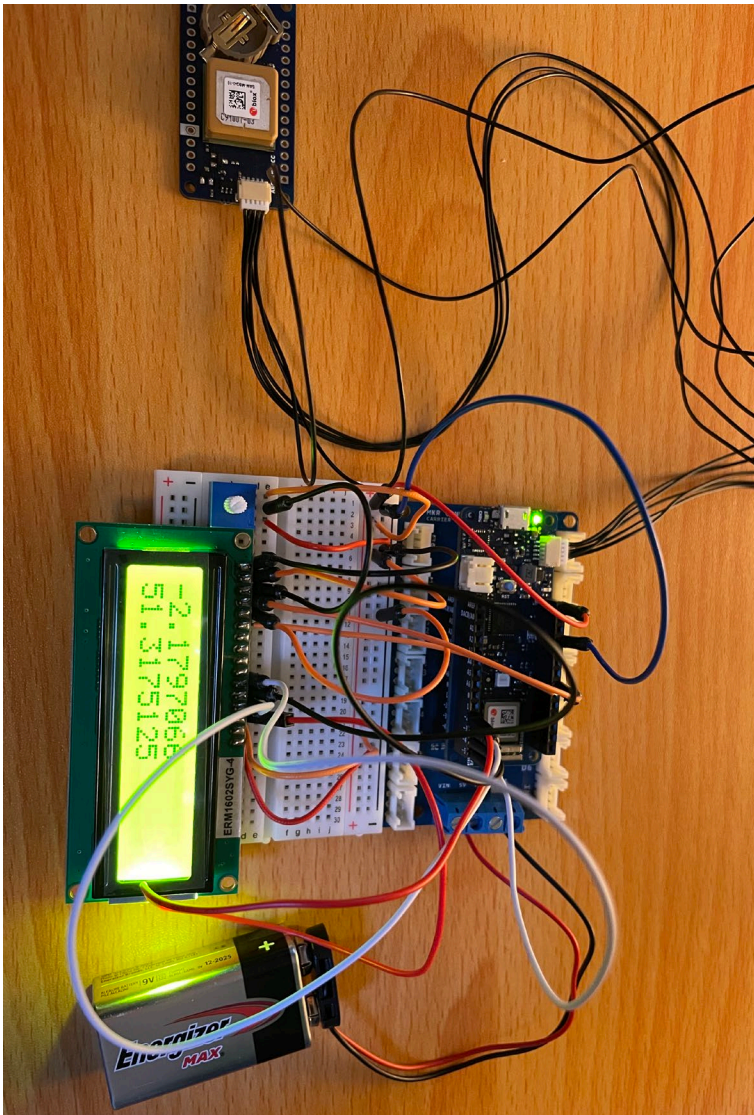


The Serial.print lines have been left in to enable you to check your code with the monitor in the editor. Additional information such as number of satellites and altitude are also output into the monitor. If you wish to, you could add these to your lcd output. The two setCursor lines determine which row the data is output into on the LCD screen.

## Treasure hunt

You should now write one of your friends a secret message using a cipher of your choice and then choose somewhere to hide it. Once you have hidden it, you should make a note of the GPS coordinates. Your friend can then use the device to find the location to uncover your secret message. They should then try and crack the code.

To use your device outside it will need to be powered. You can provide power by connecting a 9v battery to the MKR connector carrier.



## Stretch tasks

Try to write a program using Python to encrypt and decrypt the message.

Adjust your LCD output to include additional location information. Make the information scroll across the screen.

## Final thoughts

In this project you have learned to combine both input and output devices. You have also learned how to use location services with your device which lies at the heart of any GPS navigation system. All data that is transmitted wirelessly should be encrypted to ensure that if someone intercepts it, they will not be able to understand it.