PRACTICE 1: MODERN IOT

1 Create TinkerCADaccount

When you sign up for a Tinkercad account, the login you create becomes your Autodesk ID, too, which means you can use the same login everywhere on all of the Autodesk websites. This is especially useful if you are using other Autodesk applications, such as AutoCAD or Inventor, because it gives you an identity on the Autodesk user forums and on the Autodesk Knowledge Network, or AKN for short.

To create a Tinkercad account:

1. Go to the homepage of Tinkercad via the link: https://www.tinkercad.com/
2. Click JOIN NOW then choose Create a personal account
3. Sign in with your student email address and accept the Tinkercad terms of service.

After creating account successfully, you are able to use not only Tinkercad but also any Autodesk tools. So, let's begin the journey.

2 BlinkingLED

This is a very first project in an embedded platform such as the Arduino board. In contrast to a program running on a PC, print the string "Hello world!!" to the screen is impractical in a low-cost micro-controller platform. Therefore, this first project can be considered as the "Hello world" project in the Arduino board. Moreover, this project uses the built-in LED that are available on most Arduino boards. This LED is connected to a digital pin and its number may vary from board type to board type. To make your program easier, a constant **LED_BUILTIN** is defined to present the name of a pin connected to the LED, and allows you to control the built-in LED easily. Following steps provide the details to implement this project.

Step1: From the main page of TinkerCad, select the **Circuit** and then, **CreateNewCircuit** as following.

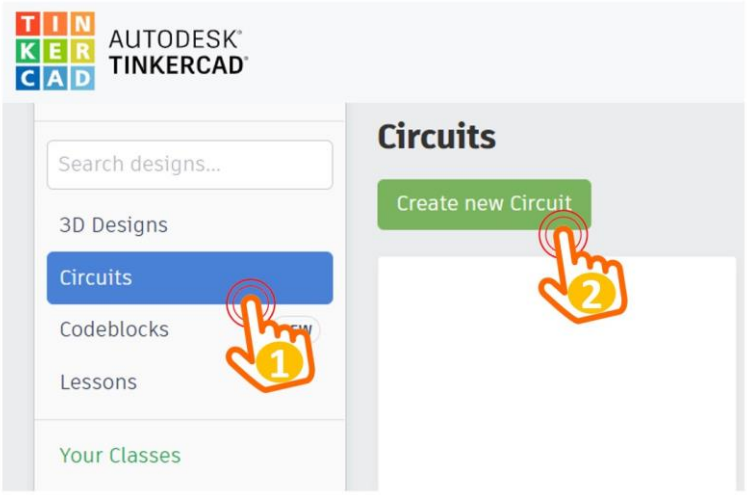

Figure 1.2: *Create a first project on TinkerCad*

Step 2: Choose the Arduino board from the starter list, then **drag and drop** the **Blink** project.

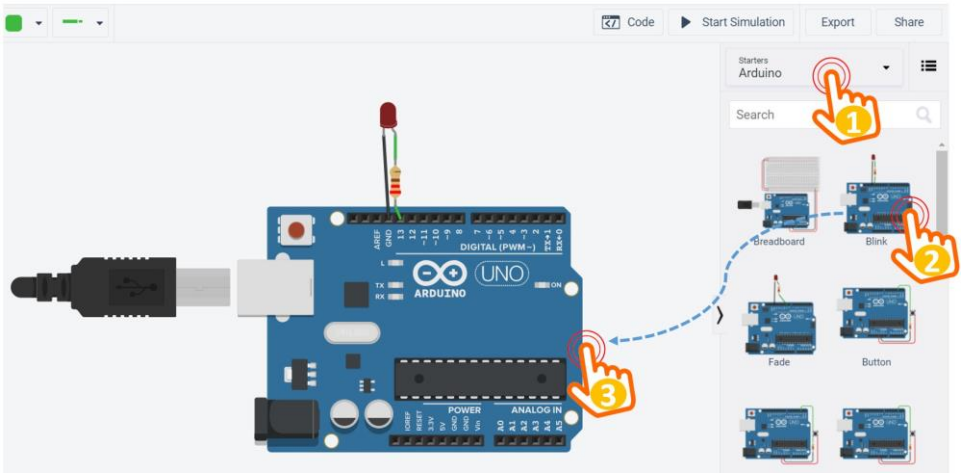

Figure 1.3: *Create the Blinky project*

Step 3: Click on the **Start Simulation** button on the toolbar of TinkerCad (see figure bellow), either the LED connected on the pin number 13 or the build-in LED (notation with letter L on the board) will be blinking. Actually, they are connected in parallel.

Figure 1.4: *Start the simulation on Blinky project*

Click on this button again (now it is **Stop Simulation** to stop the simulation, before clicking on the Code button, to check the source code of the project.

Step 4: Explore the source code of the project by clicking on the **Code** button, and then select the **Text** mode. Following screen is opened.

Figure 1.5: *Explore the source code of the project*

The first thing you do is to initialize **LED_BUILTIN** pin as an output pin with the line:

```
pinMode(LED_BUILTIN, OUTPUT);
```

In the main loop, you turn the LED on with the line:

```
digitalWrite(LED_BUILTIN, HIGH);
```

This supplies 5 volts to the LED anode. That creates a voltage difference across the pins of the LED, and lights it up. Then you turn it off with the line:

```
digitalWrite(LED_BUILTIN, LOW);
```

That takes the LED_BUILTIN pin back to 0 volts, and turns the LED off. In between the on and the off, you want enough time for a person to see the change, so the delay() commands tell the board to do nothing for 1000 milliseconds, or one second. When you use the delay() command, nothing else happens for that amount of time.
Students are proposed to change the delay time and check the simulation again.

3 Exercise

3.1 Two Toggling LEDs

Since schematic design is not in the scope of the course, students are supported by a project in the shareable link bellow:

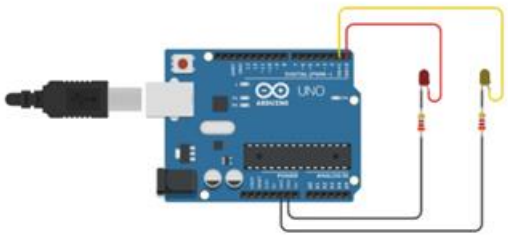

Figure 1.6: *Two LEDs Connections*

By clicking on the **Copy and Tinker** button, the project is clone to your account. The program skeleton is also provided in the code section. In this exercise, **two LEDs are connected to the Pin number 0 and 1**, respectively.

In this exercise, the status of two LEDs are toggled every 2 seconds, as demonstrated in the figure bellow.

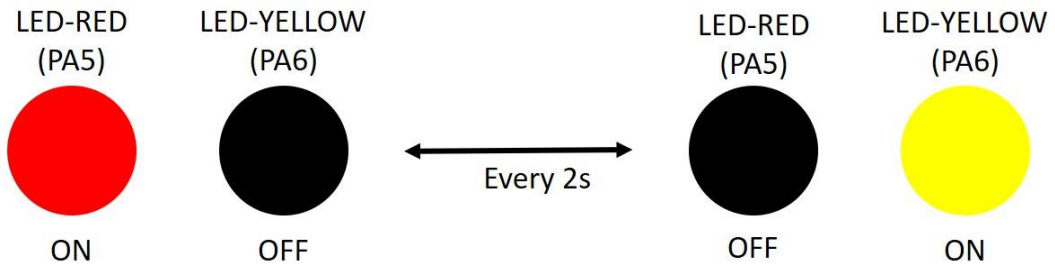

Figure 1.7: *State transitions for 2 LEDs*

Students are propose to implement in the loop function. Your source code is required to place in the following.

```
void   setup(){ pinMode(0,
       OUTPUT); pinMode(1,
       OUTPUT);
}
void
}       loop(){
       //TODO
```

3.2 Multiple Blinking LED

The previous exercise is extended to eight different LEDs to play some animations. An array and FOR statement are used to work with multiple LEDs, as supported in the link bellow.

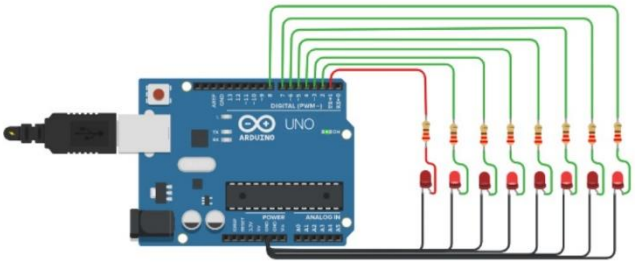

Figure 1.8: *Multiple Blinking LED*

By clicking on the **Copy and Tinker** button, the project is clone to your account. The program skeleton is also provided in the code section. In this program, a simple animations with 2 stages are provided.

Students are propose to implement at least 10 animations for LEDs series. Your source code is required to place in the following.

```
void   setup(){
}
void
}       loop(){
```

3.3 Seven Segment LED

An arrangement of seven different tiny LED in a package form a new component named **7 Segment LED**. This component is widely used to display a digit from 0 to 9, as depicted in the figure bellow.

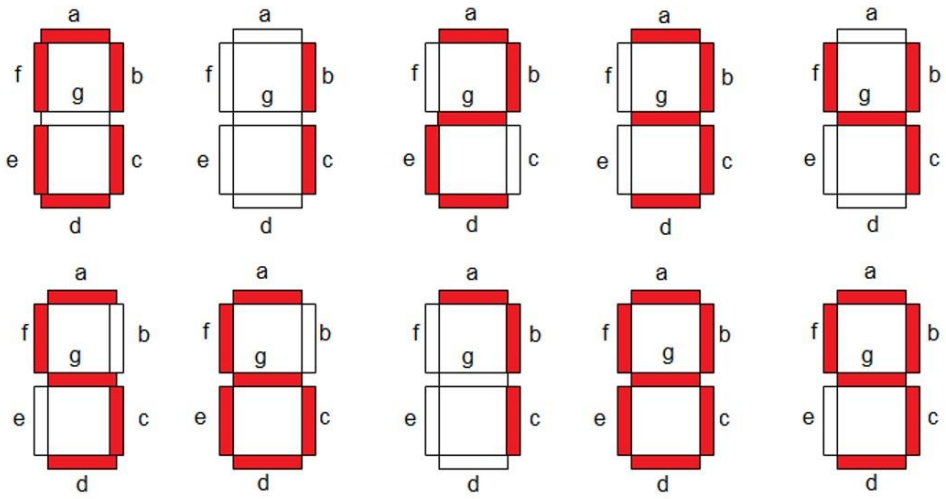

Figure 1.9: *Display number using seven segment LED*

The proposed connection in this exercise is provided in the link bellow.

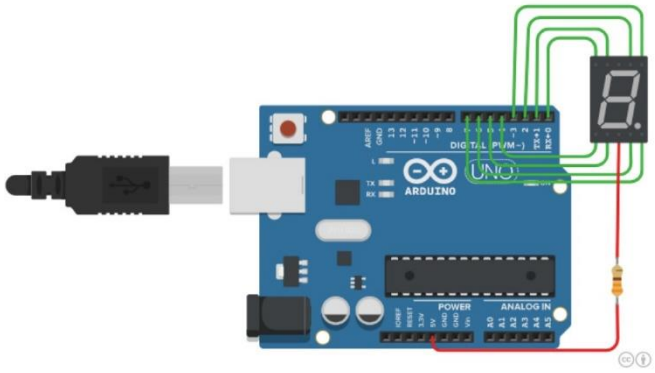

Figure 1.10: *Seven segment LED*

In this schematic, **seven pins from 0 to 6** are used to connect to the device. To turn on a segment of a device, a **LOW** signal is required.

In the example code, number 0 is displayed on the device. **Students are proposed to finalize the function displayNumber(int i) to finalize the project. The source code of this function is required to place in the following.**

```
void displayNumber(int i){ if (i ==
  0){
     digitalWrite(0, HIGH);

     digitalWrite(1, LOW);
     digitalWrite(2, LOW);
     digitalWrite(3, LOW);
     digitalWrite(4, LOW);
      digitalWrite(5, LOW); digitalWrite(6, LOW);

  }else if(i == 1){
     //YOUR CODE HERE }else
  if(i == 2){
     //YOUR CODE HERE }else
  if(i == 3){
     //YOUR CODE HERE }else
  if(i == 4){
     //YOUR CODE HERE }else
  if(i == 5){
     //YOUR CODE HERE }else
  if(i == 6){
     //YOUR CODE HERE }else
  if(i == 7){
     //YOUR CODE HERE }else
  if(i == 8){
     //YOUR CODE HERE }else
  if(i == 9){
     //YOUR CODE HERE
}  }
```

3.4 Two Digit Number

The previous exercise is upgraded to 2 different seven segment LEDs. The project is shared in the link bellow.

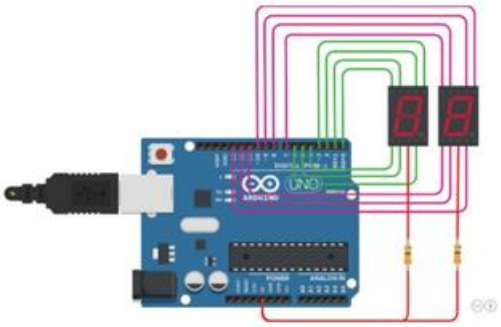

Figure 1.11: *Two seven segment LEDs*

In the source code of this project, a simple unit test is provided. Students can run the simulation to figure out the connections in the circuit.

Students are proposed to display numbers of these LEDs, from 00 to 20 (and then loop back). The updated display period is one second. To provide your source code, do not need to present the functions, which are reused from the exercise before (such as displayNumber1 and displayNumber2).

```
void setup(){
  //TODO: Add your setup here
}
void displayNumber1(int i){
  //REUSED: Do not need the source code
}
void displayNumber2(int i){
  //REUSED: Do not need the source code

}
void loop(){
} //TODO: Add your processing to isplay 2 digit numbers
```

1One Way Traffic Light

In the next three exercises, a traffic light project is proposed and is presented in the link bellow.

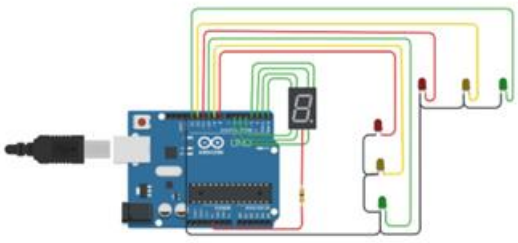

Figure 1.12: Full traffic light with timer

The traffic light has 5 seconds for the RED, 2 seconds for the the YELLOW and 3 seconds for the GREEN. **Please arrange the LEDs in a right order**.

At this exercise, only first way traffic light is required (the LEDs connected to pin number 8, 9 and 10).

```
void  setup(){
    //TODO: Add  your  setup here
}
void
}     loop(){
    //TODO: Add  your  processing  code  here
```

3.5 Two Way Traffic Light

Your source code is upgrade to control the second traffic lights, connected to pin number 11, 12 and 13.

```
void  setup(){
    //TODO: Add  your  setup here
}
void
}     loop(){
    //TODO: Add  your  processing  code  here
```

3.6 Traffic Light with Timer

Finalize the system by a count-down timer using seven segment LED. Five seconds for the RED means that the count-down process is started with 4 and ended by 0.

```
void  setup(){
    //TODO: Add  your  setup here
}
void
}     loop(){
    //TODO: Add  your  processing  code  here
```

3.7 Analog Clock Project

From this exercise, an analog clock project is proposed. Twelve different LEDs are used to simulate a screen of a clock, as show in this link:

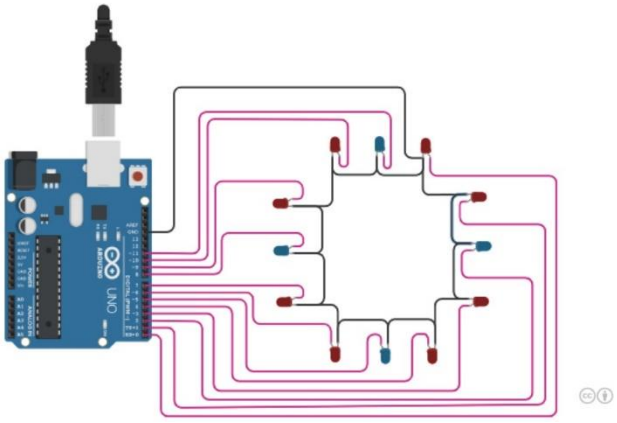

Figure 1.13: *Analog clock with 12 LEDs*

Based on the code skeleton, students are proposed to implement two functions bellow. These functions are invoked in the loop function to perform unit test before implement

```
void displayOnClock(int num){
    if(num <= 12){
        //TODO: Implement your code here
    }
} void clearClock(){
    //TODO: Clear 12 LEDs on Clock
}
```

3.8 Analog Clock with Second

The second information is updated to the LEDs follow exactly principle of an analog clock: if second is between 0 and 4, the number 12 should be indicated. Similar to that, when second is from 5 to 9, number 1 is indicated.

Please use the delay(1000) at the end of the loop function. Present your source code just in the loop.

```
void loop(){
    //TODO
}
```

3.9 Finalize the Analog Clock

Finalize the project with two more information are displayed on the clock. **Present your source code just in the loop**.

```
void loop(){
    //TODO
}
```