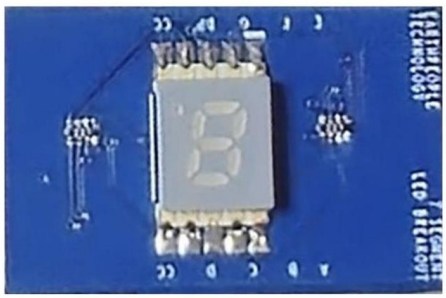
EARTH PEOPLE TECHNOLOGY

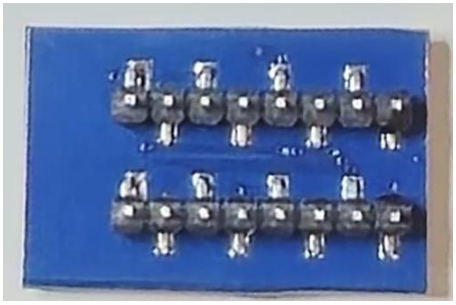
Seven Segment LED Board
User Manual

The EPT-ACSA02-LD-X2 is a breakout board that includes one Seven Segment LED. This Board provides a simple interface to control each segment of the LED Display for a total of 8 LEDs from any MCU (including the Arduino family). It is designed to connect directly to the standard bread board with a reduced footprint. Once connected to an MCU, the MCU can provide a sink for each LED.

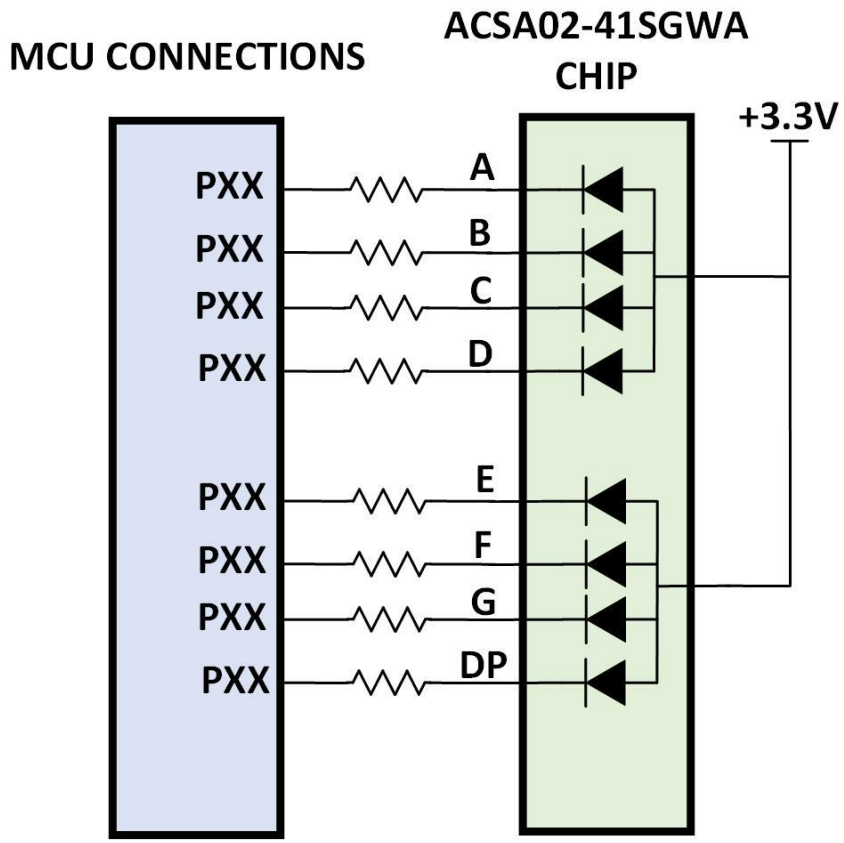
Description

The Seven Segment LED board uses one ACSA02-41SGWA-F01 manufactured by Kingbright. The Display is 0.2 inch digit height. It has low current operation and excellent character appearance. The Super Bright Green source color devices are made with Gallium Phosphide Green Light Emitting Diode.





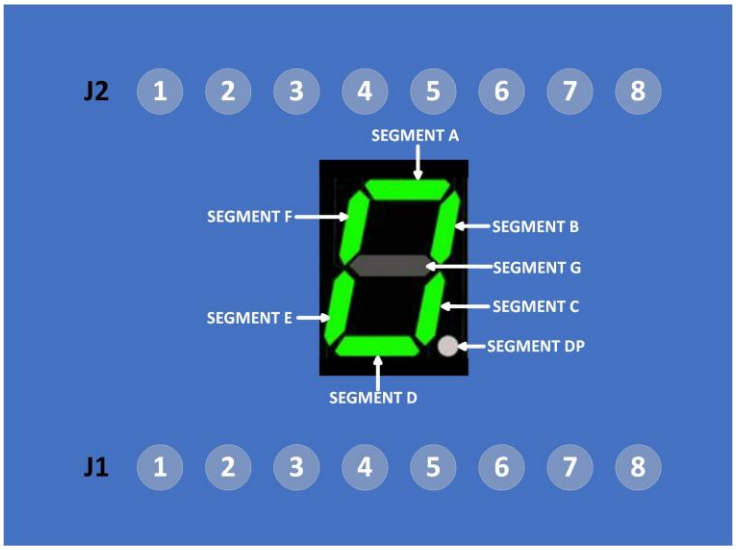
Any GPIO from any MCU (including Arduino family) can be used to control the On/Off of each LED individually. This Breakout board is compatible with both +3.3V and +5V Arduinos. However, the Seven Segment LED Board is for use only with +3.3V.



The ACSA02-41SGWA-F01 chip is a current sink and can be connected to any MCU. The anode should be connected to +3.3V. The reason for this is the Series resistors are calculated for current limiting based on +3.3V. Changing the anode voltage to another source will change the brightness emitting from each LED. Because the ACSA02-41SGWA-F01 chips are current sink, the user can connect to either +5V or +3.3V Arduino (or other MCUs) and control each LED.

Connecting the Arduino

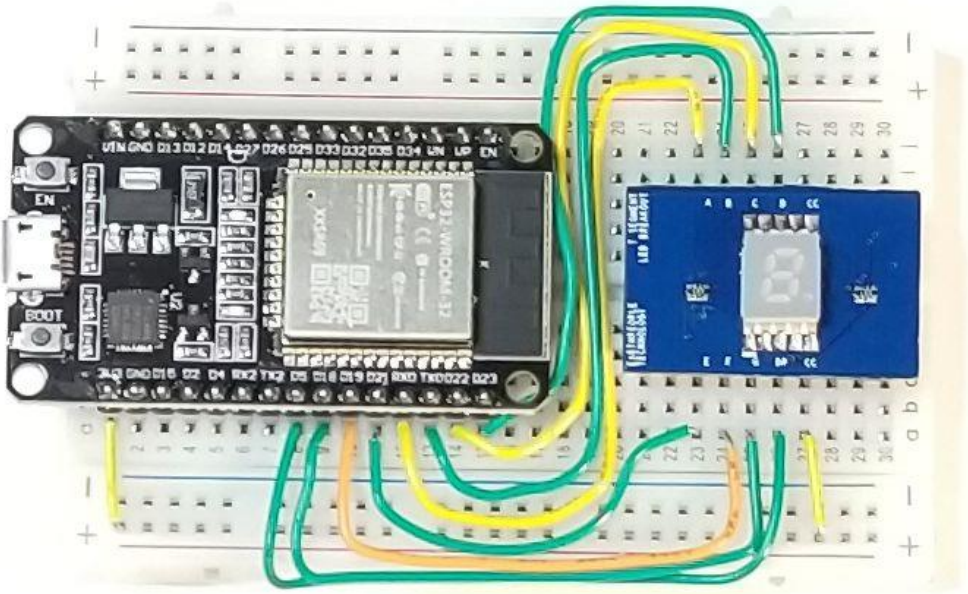
Connecting the Arduino (or any MCU) to the LED RGB is easy. In this example, we use typical bread board, a pro-mini and a VisiPort USB to Serial board.



Connector	Pin #	Signal		
J1	1	SEGMENT_A		
	2	SEGMENT_B		

	3	SEGMENT_C		
	4	SEGMENT_D		
	5	+3.3V		
	6	NC		
	7	NC		
	8	Ground		

Connector	Pin #	Signal		
J2	1	SEGMENT_E		
	2	SEGMENT_F		
	3	SEGMENT_G		
	4	SEGMENT_DP		
	5	+3.3V		
	6	NC		
	7	NC		
	8	Ground		



Arduino Code – Controlling On/Off of LEDs

When you have everything hooked up try running the below sketch. This is a simple loop that will blink each LED individually.

```
/*
  Copyright Earth People Technology Inc. 2024

  LED Segment Driver

  Platform: ESP32 WROOM Development Board

*/


  int ledSegmentA = 3;
  int ledSegmentB = 1;
  int ledSegmentC = 22;
  int ledSegmentD = 23;
  int ledSegmentE = 21;
  int ledSegmentF = 19;
  int ledSegmentG = 18;
  int ledSegmentDP = 5;

void setup() {
  pinMode(ledSegmentA, OUTPUT);
  pinMode(ledSegmentB, OUTPUT);
  pinMode(ledSegmentC, OUTPUT);
  pinMode(ledSegmentD, OUTPUT);
  pinMode(ledSegmentE, OUTPUT);
  pinMode(ledSegmentF, OUTPUT);
  pinMode(ledSegmentG, OUTPUT);
  pinMode(ledSegmentDP, OUTPUT);

}

void loop() {
  //Display  the "A" Character
  delay(500);
  characterA();
  delay(1000);
  //Display  the "b" Character
  characterB();
```

```
  characterB();
  delay(1000);
  //Display  the "C" Character
  characterC();
  delay(1000);
  //Display  the "d" Character
  characterD();
  delay(1000);
  //Display  the "E" Character
  characterE();
  delay(1000);
  //Display  the "F" Character
  characterF();
  delay(1000);
  //Display  the "G" Character
  characterG();
  delay(1000);
  //Display  the "H" Character
  characterH();
  delay(1000);
  //Display  the "I" Character
  characterI();
  delay(1000);
  //Display  the "J" Character
  characterJ();
  delay(1000);
  //Display  the "L" Character
  characterL();
  delay(1000);
  resetCharacters();

}

void characterA()
{
      //For Character "A" write LOW to A, B, C, E, F, G
      //to turn on all Green LEDs
```

```
    //Write HIGH to D
    digitalWrite(ledSegmentA, LOW);
    digitalWrite(ledSegmentB, LOW);
    digitalWrite(ledSegmentC, LOW);
    digitalWrite(ledSegmentD, HIGH);
    digitalWrite(ledSegmentE, LOW);
    digitalWrite(ledSegmentF, LOW);
    digitalWrite(ledSegmentG, LOW);
    digitalWrite(ledSegmentDP, HIGH);

}


void characterB()
{
    //For Character "b" write LOW to C, D, E, F, G
    //to turn on all Green LEDs
    //Write HIGH to A, B
    digitalWrite(ledSegmentA, HIGH);
    digitalWrite(ledSegmentB, HIGH);
    digitalWrite(ledSegmentC, LOW);
    digitalWrite(ledSegmentD, LOW);
    digitalWrite(ledSegmentE, LOW);
    digitalWrite(ledSegmentF, LOW);
    digitalWrite(ledSegmentG, LOW);
    digitalWrite(ledSegmentDP, HIGH);

}

void characterC()
{
    //For Character "C" write LOW to A, D, E, F
    //to turn on all Green LEDs
    //Write HIGH to
    digitalWrite(ledSegmentA, LOW);
    digitalWrite(ledSegmentB, HIGH);
    digitalWrite(ledSegmentC, HIGH);
    digitalWrite(ledSegmentD, LOW);
```

```
        digitalWrite(ledSegmentC, HIGH);
        digitalWrite(ledSegmentD, LOW);
        digitalWrite(ledSegmentE, LOW);
        digitalWrite(ledSegmentF, LOW);
        digitalWrite(ledSegmentG, HIGH);
        digitalWrite(ledSegmentDP, HIGH);

}

void characterD()
{
        //For Character "d" write LOW to B, C, D, E, G
        //to turn on all Green LEDs
        //Write HIGH to
        digitalWrite(ledSegmentA, HIGH);
        digitalWrite(ledSegmentB, LOW);
        digitalWrite(ledSegmentC, LOW);
        digitalWrite(ledSegmentD, LOW);
        digitalWrite(ledSegmentE, LOW);
        digitalWrite(ledSegmentF, HIGH);
        digitalWrite(ledSegmentG, LOW);
        digitalWrite(ledSegmentDP, HIGH);

}

void characterE()
{
        //For Character "E" write LOW to A, D, E, F, G
        //to turn on all Green LEDs
        //Write HIGH to
        digitalWrite(ledSegmentA, LOW);
        digitalWrite(ledSegmentB, HIGH);
        digitalWrite(ledSegmentC, HIGH);
        digitalWrite(ledSegmentD, LOW);
        digitalWrite(ledSegmentE, LOW);
        digitalWrite(ledSegmentF, LOW);
        digitalWrite(ledSegmentG, LOW);
        digitalWrite(ledSegmentDP, HIGH);
```

```
}

void characterF()
{
      //For Character "F" write LOW to A, E, F, G
      //to turn on all Green LEDs
      //Write HIGH to
      digitalWrite(ledSegmentA, LOW);
      digitalWrite(ledSegmentB, HIGH);
      digitalWrite(ledSegmentC, HIGH);
      digitalWrite(ledSegmentD, HIGH);
      digitalWrite(ledSegmentE, LOW);
      digitalWrite(ledSegmentF, LOW);
      digitalWrite(ledSegmentG, LOW);
      digitalWrite(ledSegmentDP, HIGH);

}

void characterG()
{
      //For Character "G" write LOW to A, C, D, E, F, G
      //to turn on all Green LEDs
      //Write HIGH to
      digitalWrite(ledSegmentA, LOW);
      digitalWrite(ledSegmentB, HIGH);
      digitalWrite(ledSegmentC, LOW);
      digitalWrite(ledSegmentD, LOW);
      digitalWrite(ledSegmentE, LOW);
      digitalWrite(ledSegmentF, LOW);
      digitalWrite(ledSegmentG, LOW);
      digitalWrite(ledSegmentDP, HIGH);

}
```

Note that you must set your serial monitor to a speed of 115200 baud to try out the sketch.

Code Explanation:

This sketch simply applies a low to the selected LED Segment to turn it on. Then applies a high to all other LEDs to turn them off. First, 8 signals are declared and each signal is assigned a name and a unique pin on the Pro-Mini. Initially, all 8 LED control signals are declared as outputs in the Setup function and set to high. This turns off all 8 LEDs.

During the Loop function, a delay of 500 milliseconds is applied. No output/LED is affected during the delay. Then, subsequent LED Segments are set to low which turns them all on. All other LEDs are still set to high from the initial setup. A delay of 500 milliseconds is applied. During this delay, the selected LED Segments remain on, all other LEDs remain off and no actions are taken on any outputs. The next character is selected and the appropriate LED Segments are selected and asserted. Another delay of 500 milliseconds is applied in which no output is changed. This cycle repeats for characters in the sequence. Once the code gets to the bottom of the loop, a counter variable called LoopCount is incremented by one and then transmitted to the Serial Port. The loop function starts over at the top of the loop code and repeats the cycle. This repeating cycle goes on forever until the power is removed from the Pro-Mini.