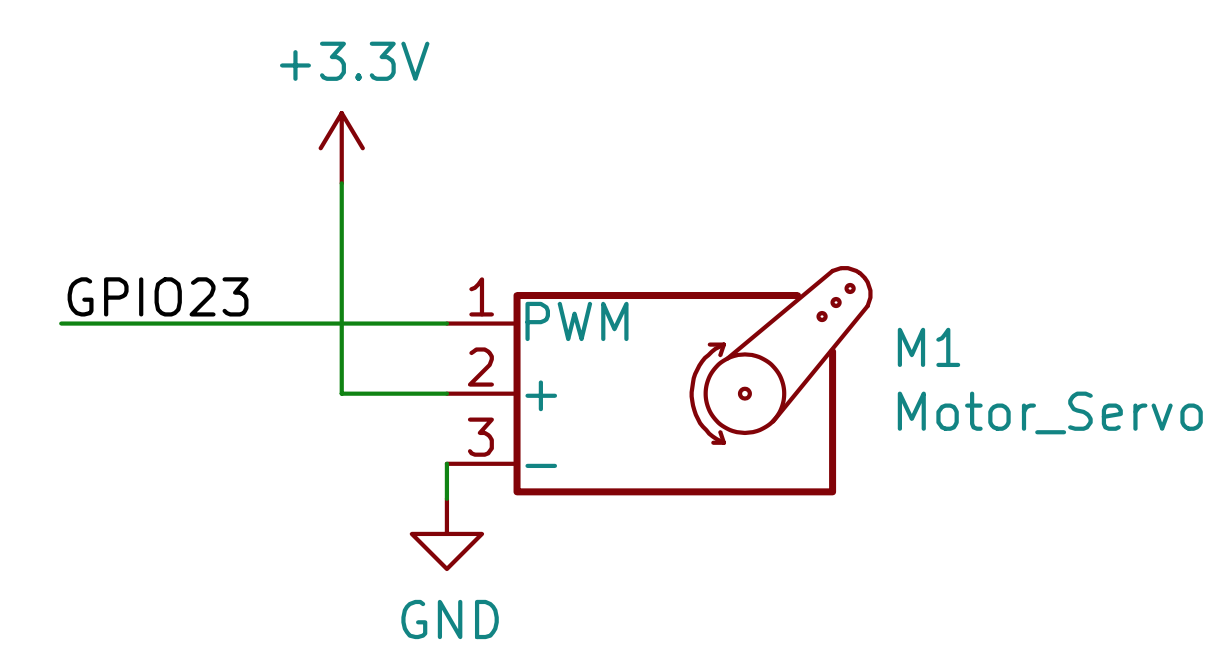
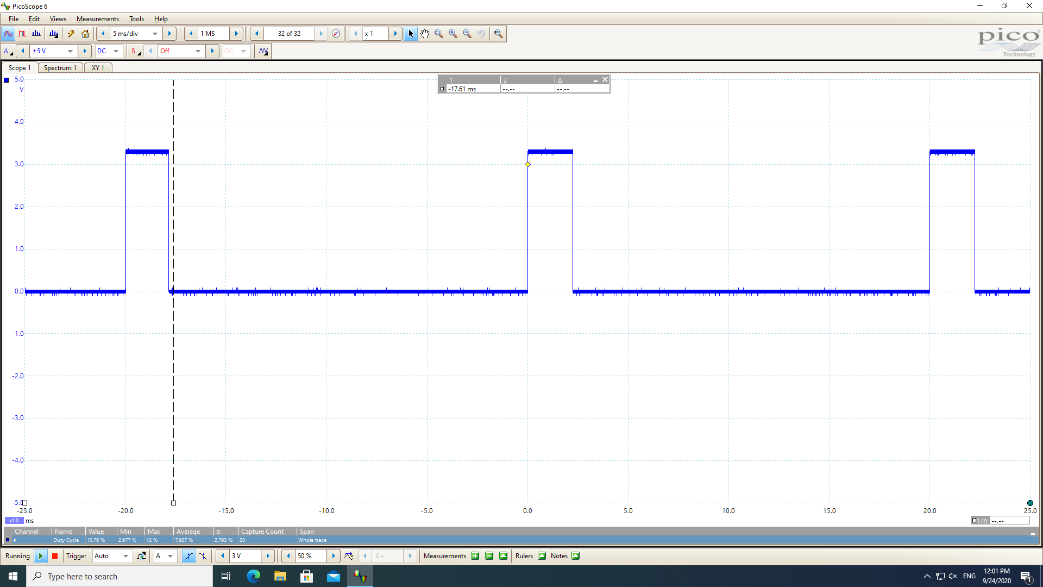
**Part A. Servo Motor**

**A1. Wire up the follow circuit and upload** ServoMotor.ino**.**

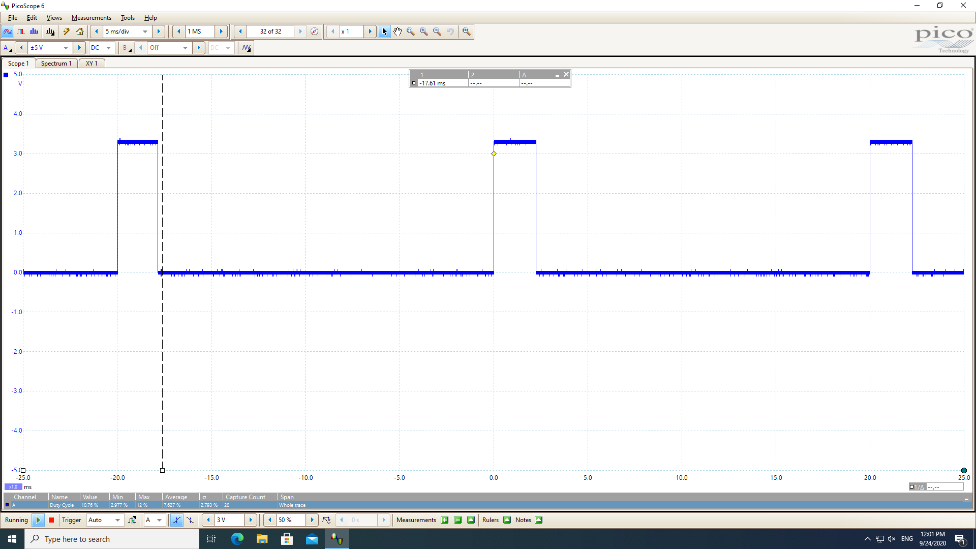


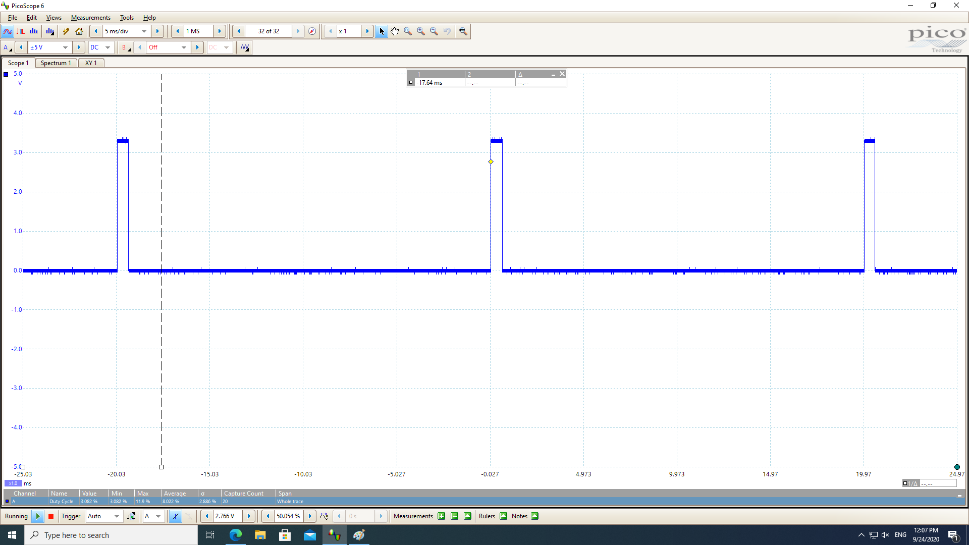
**The servo motor looks like this** .

**Use the Picoscope to measure GPIO23. Adjust the y-axis and x-axis so that you see the waveform clearly. Capture your screen and paste it below.**



**In the Picoscope while measuring GPIO23, add 2 measurements: frequency and duty cycle. To add a measurement, go to** Measurement -> Add Measurement **on the menu bar. Then select the channel and the type of measurement of interest. Capture the part of your screen focusing on the 2 measurements and paste it below.**





**Fill in the follow table.**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Frequency** | **Min. Duty Cycle** | **Max. Duty Cycle** |
| **In Souce Code** | **50 Hz** | **0** | **1** |
| **In Picoscope** | **50 Hz** | **2.822** | **11.74** |

**What is the relationship between the angle on the servo motor and the duty cycle?**

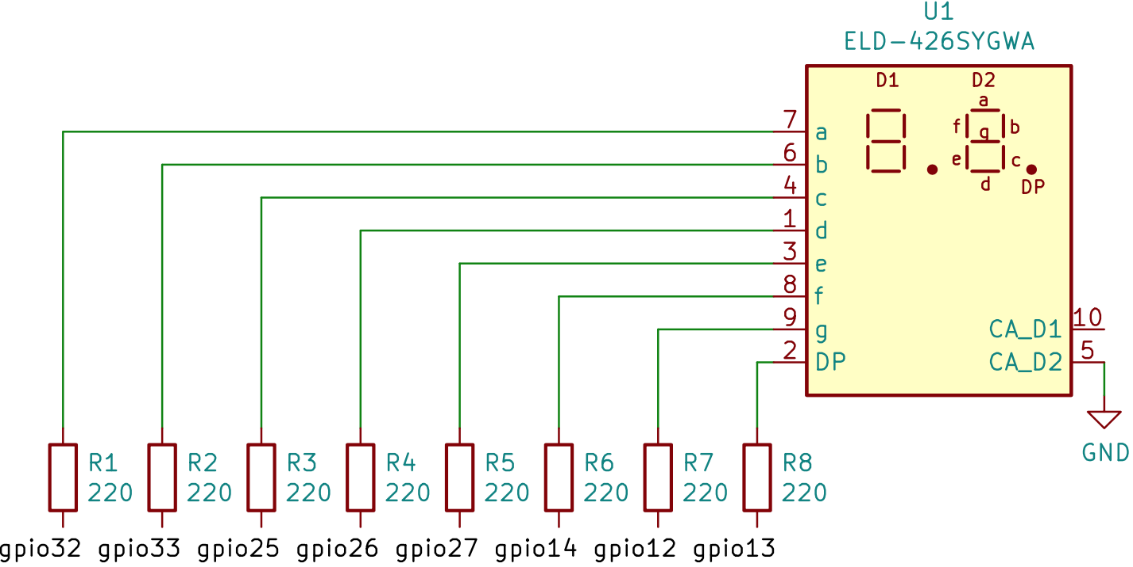
**A typical servo responds to pulse widths in the range 1000 to 2000 µs.**

**A pulse width of 1500 µs moves the servo to angle 0. Each 10 µs increase in pulse width typically moves the servo 1 degree more clockwise. Each 10 µs decrease in pulse width typically moves the servo 1 degree more anticlockwise.**

**Small 9g servos typically have an extended range and may respond to pulse widths in the range 500 to 2500 µs.**

**Part B. Seven Segment Display**

**B1. Wire up the circuit below and upload** 7Segment-1Digit.ino**. Note that your 7-Segment Display is LTD-6940HR which has different pin layout from the one in the circuit below. So stick with a, b, c, d, … segment references instead. You can consult LTD-6940HR datasheet for which pin is for which segment. CA\_D1 and CA\_D2 are common cathode for digit 1 and 2 respectively.**



**The following figure shows how to count pins on a 7-segment display.**



**In the source code, add 6 functions to display A, b, C, d, E, F. Upload your modified program to ESP32 and take 6 pictures, each of which is for each additional letter. Then put these 6 pictures below.**

**---**

int a = 32;  // define the digital interface to connect a seven segment LED

int b = 33;  // define the connection b Digital Interface 6-segment LED

int c = 25;  // define paragraph (c) Digital Interface 5 digital connection

int d = 26;  // define the digital interface 11 is connected to d-segment digital tube

int e = 27;  // define the digital interface 10 is connected to e-segment digital tube

int f = 14;  // define the digital interface 8 digital tube connection f

int g = 12;  // define the digital interface 9 g of the digital control connection

int dp = 13; // define the digital interface 4 digital tube connecting dp

int arr[8] = {a, b, c, d, e, f, g, dp};

void *digital\_0*(void) // display the number 0

{

    unsigned char j;

    for (j = 0; j < 8; j++)

*digitalWrite*(arr[j], 1);

*digitalWrite*(g, 0);

}

void *digital\_1*(void) // display the number 1

{

    unsigned char j;

    for (j = 0; j < 8; j++) // off remaining segments

*digitalWrite*(arr[j], 0);

*digitalWrite*(c, 1); // to the digital interface 5-pin 0, lit paragraph (c)

*digitalWrite*(b, 1); // lit paragraph b

*digitalWrite*(dp, 1);

}

void *digital\_2*(void) // display number 2

{

    unsigned char j;

    for (j = 0; j < 8; j++)

*digitalWrite*(arr[j], 1);

*digitalWrite*(c, 0);

*digitalWrite*(f, 0);

}

void *digital\_3*(void) // display the number 3

{

    unsigned char j;

    for (j = 0; j < 8; j++)

*digitalWrite*(arr[j], 1);

*digitalWrite*(f, 0);

*digitalWrite*(e, 0);

}

void *digital\_4*(void) // show 4

{

*digitalWrite*(c, 1);

*digitalWrite*(b, 1);

*digitalWrite*(f, 1);

*digitalWrite*(g, 1);

*digitalWrite*(a, 0);

*digitalWrite*(e, 0);

*digitalWrite*(d, 0);

}

void *digital\_5*(void) // display the number 5

{

    unsigned char j;

    for (j = 0; j < 8; j++)

*digitalWrite*(arr[j], 1);

*digitalWrite*(c, 1);

*digitalWrite*(d, 1);

*digitalWrite*(b, 0);

*digitalWrite*(e, 0);

}

void *digital\_6*(void) // display the number 6

{

    unsigned char j;

    for (j = 0; j < 8; j++)

*digitalWrite*(arr[j], 1);

*digitalWrite*(c, 1);

*digitalWrite*(b, 0);

}

void *digital\_7*(void) // display the number 7

{

    unsigned char j;

    for (j = 0; j < 8; j++)

*digitalWrite*(arr[j], 1);

    for (j = 3; j < 7; j++)

*digitalWrite*(arr[j], 0);

}

void *digital\_8*(void) // display the number 8

{

    unsigned char j;

    for (j = 0; j < 8; j++)

*digitalWrite*(arr[j], 1);

}

void *digital\_9*(void) // display the number 9

{

    unsigned char j;

    for (j = 0; j < 8; j++)

*digitalWrite*(arr[j], 1);

*digitalWrite*(e, 0);

}

void *digital\_a*(void)

{

    unsigned char j;

    for (j = 0; j < 7; j++)

*digitalWrite*(arr[j], 1);

*digitalWrite*(d, 0);

}

void *digital\_b*(void)

{

    unsigned char j;

    for (j = 0; j < 7; j++)

*digitalWrite*(arr[j], 1);

*digitalWrite*(a, 0);

*digitalWrite*(b, 0);

}

void *digital\_c*(void)

{

    unsigned char j;

    for (j = 0; j < 7; j++)

*digitalWrite*(arr[j], 1);

*digitalWrite*(g, 0);

*digitalWrite*(b, 0);

*digitalWrite*(c, 0);

}

void *digital\_d*(void)

{

    unsigned char j;

    for (j = 0; j < 7; j++)

*digitalWrite*(arr[j], 1);

*digitalWrite*(a, 0);

*digitalWrite*(f, 0);

}

void *digital\_e*(void)

{

    unsigned char j;

    for (j = 0; j < 7; j++)

*digitalWrite*(arr[j], 1);

*digitalWrite*(b, 0);

*digitalWrite*(c, 0);

}

void *digital\_f*(void)

{

    unsigned char j;

    for (j = 0; j < 7; j++)

*digitalWrite*(arr[j], 1);

*digitalWrite*(b, 0);

*digitalWrite*(c, 0);

*digitalWrite*(d, 0);

}

void *setup*()

{

    int i; // define variables

    for (i = 0; i < 8; i++)

*pinMode*(arr[i], OUTPUT); // set 4 to 11 pin to output mode

}

void *loop*()

{

*digital\_a*(); // display the number 9

*delay*(1000); // delay 1s

*digital\_b*(); // display the number 9

*delay*(1000); // delay 1s

*digital\_c*(); // display the number 9

*delay*(1000); // delay 1s

*digital\_d*(); // display the number 9

*delay*(1000); // delay 1s

*digital\_e*(); // display the number 9

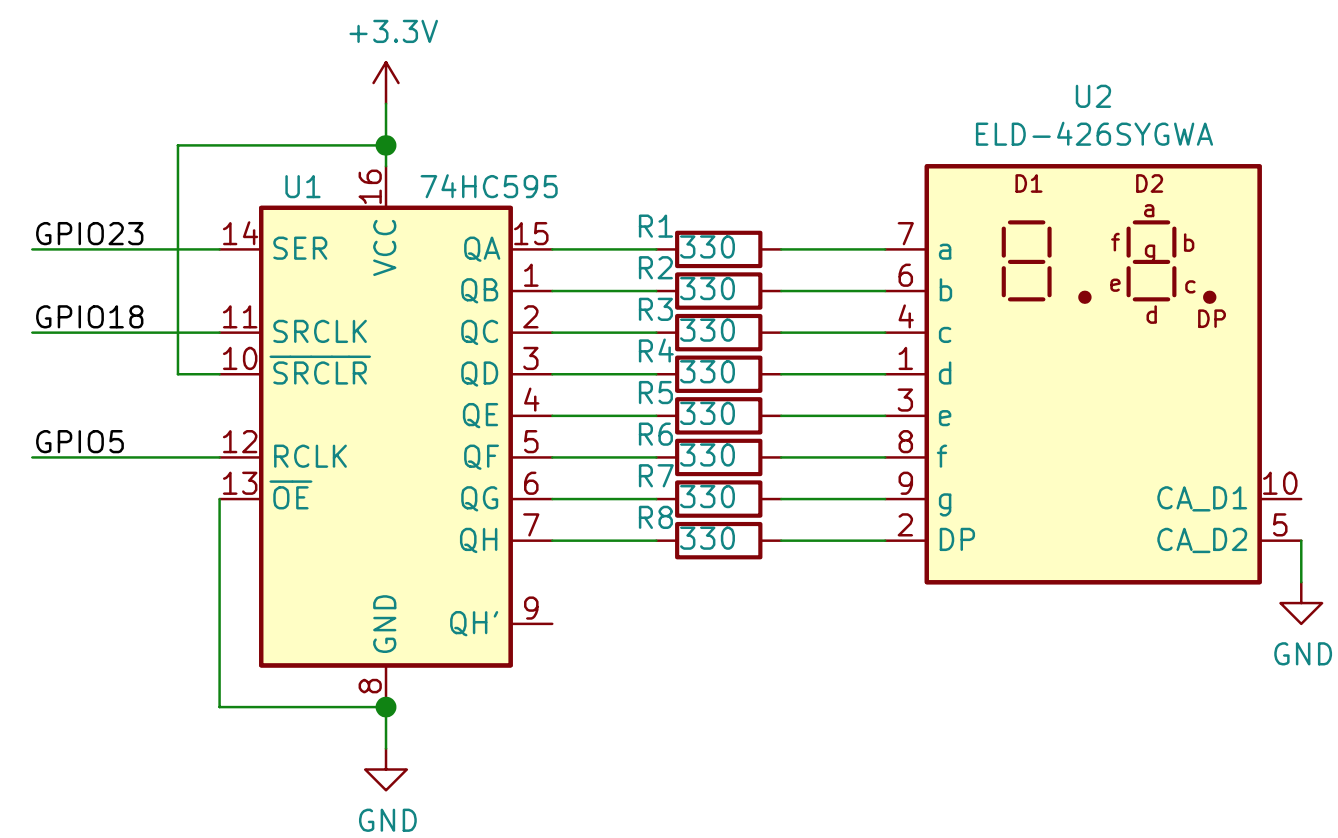
*delay*(1000); // delay 1s

*digital\_f*(); // display the number 9

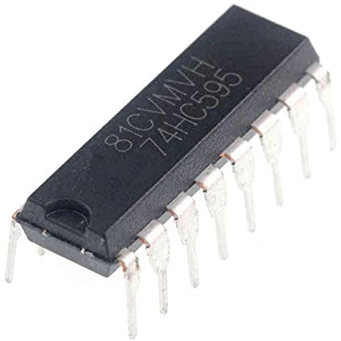
*delay*(1000); // delay 1s

}

**B2. Wire up the following circuit. Upload** 7Segment-1Digit-74HC595.ino**. Note that your 7-Segment Display is LTD-6940HR which has different pin layout from the one in the circuit below. So stick with a, b, c, d, … segment references instead. You can consult LTD-6940HR datasheet for which pin is for which segment. CA\_D1 and CA\_D2 are common cathode for digit 1 and 2 respectively.**

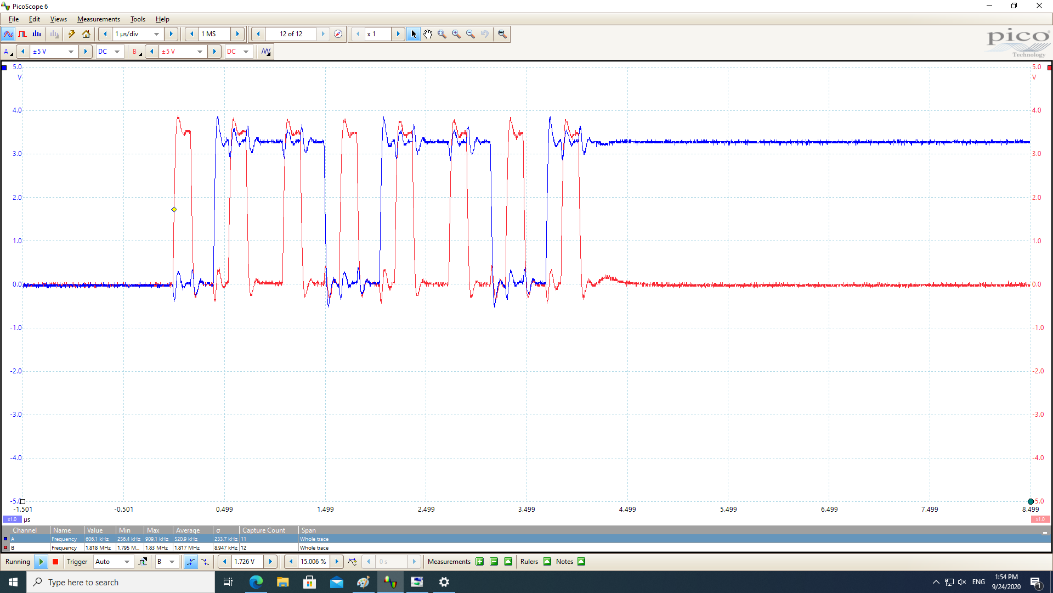


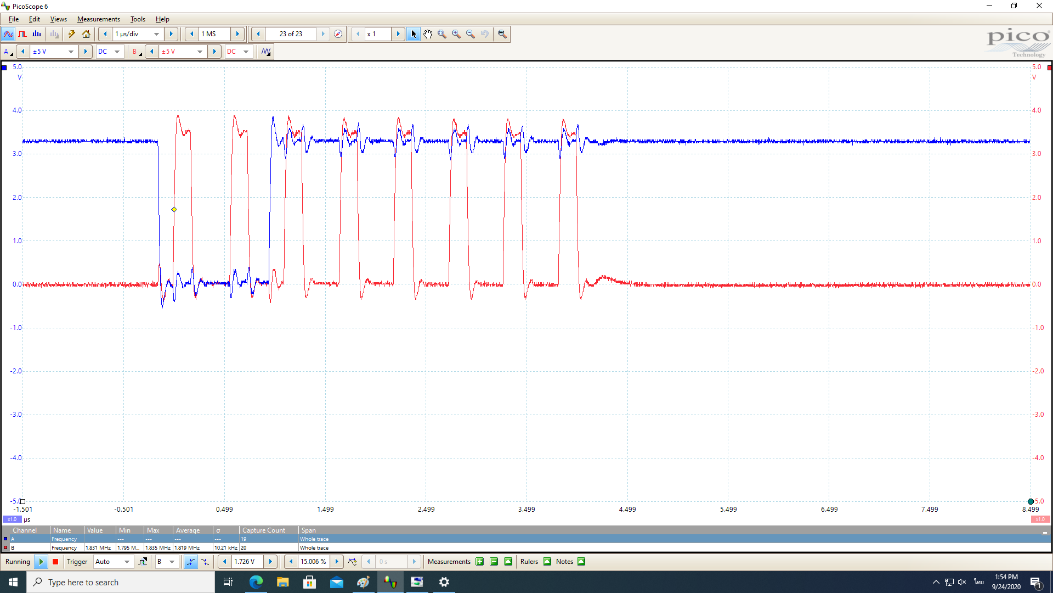
**The following picture is an example of 74HC595 which is an 8-bit shift register IC (Integrated Circuit). You can consult its datasheet for the pin layout.**



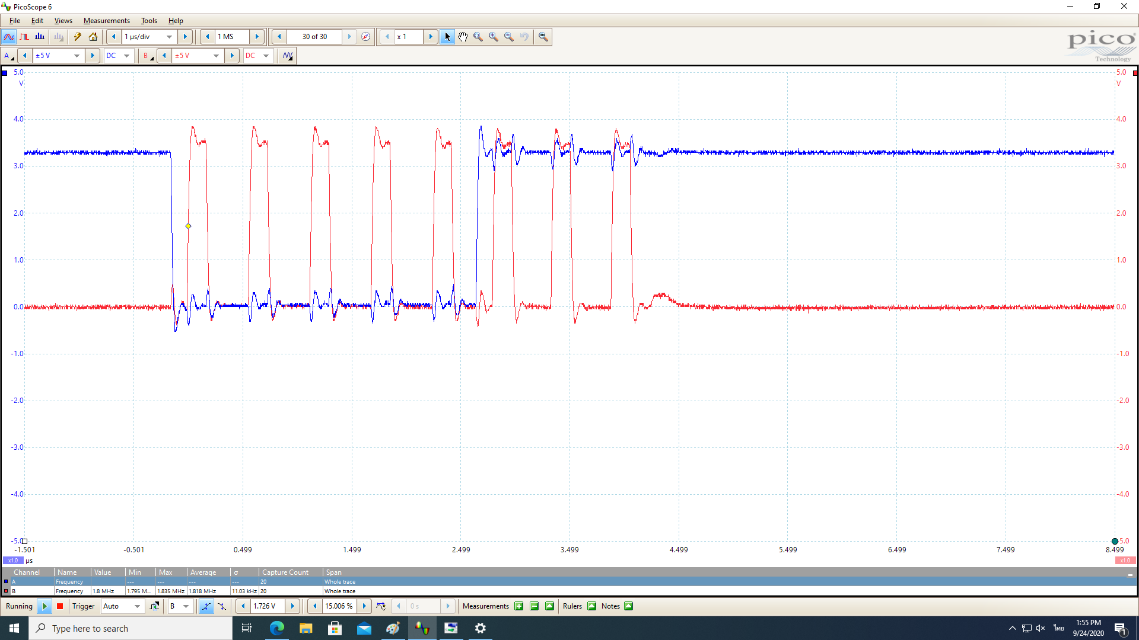
**Also note that this program is sending data to the 7-segment display by using SPI serial communication exploiting the** shiftOut() **function.**

**Use both A and B channels on Picoscope to measure GPIO23 (data) and GPIO18 (clock) resplectively. Note that you have to turn on B channel too. Adjust y-axis and x-axis so you can see the waveform clearly. Capture the screen and paste it below.**





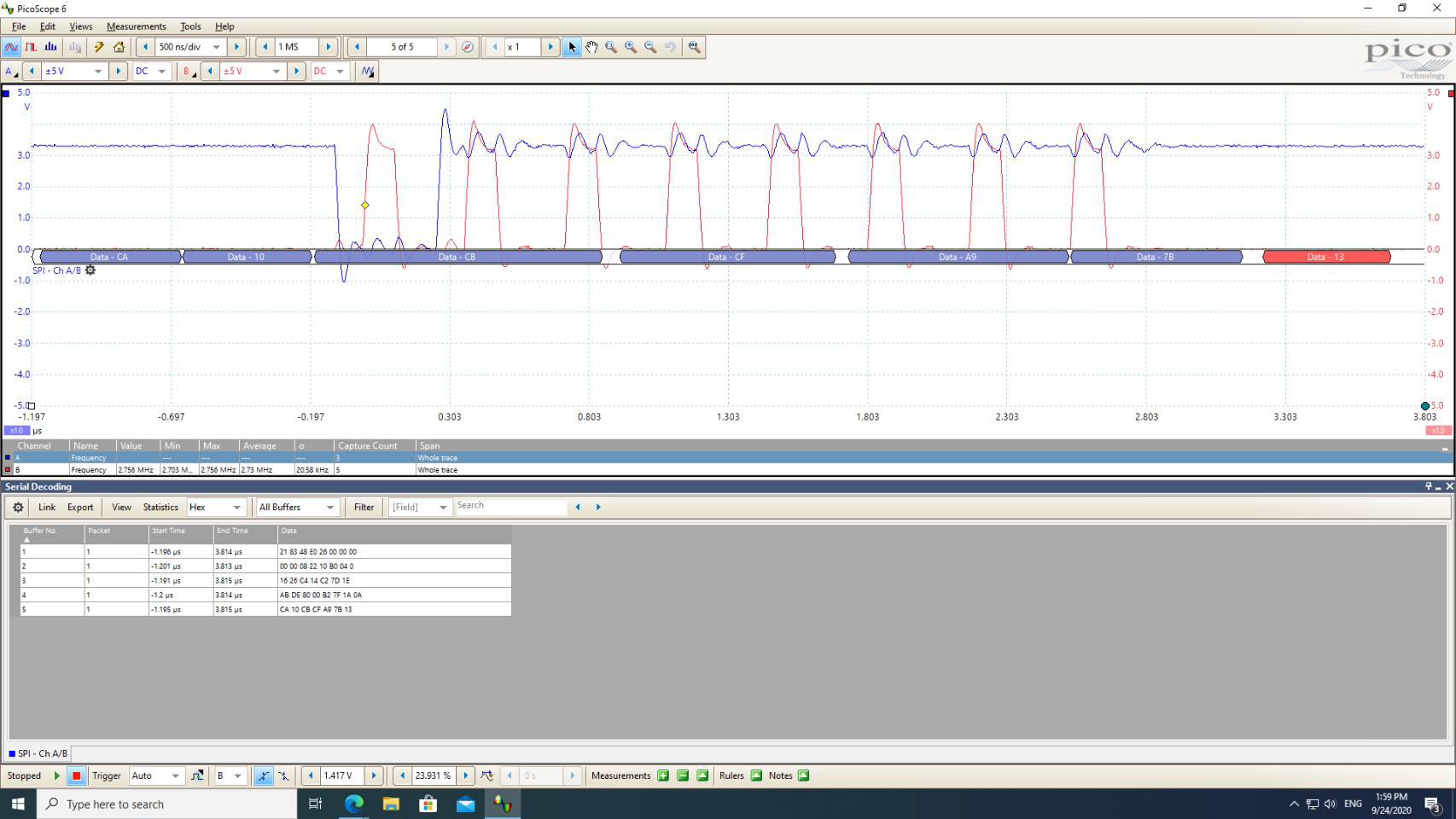
**Add frequency measurement to both channel. Capture the frequency measurements part and paste it below.**

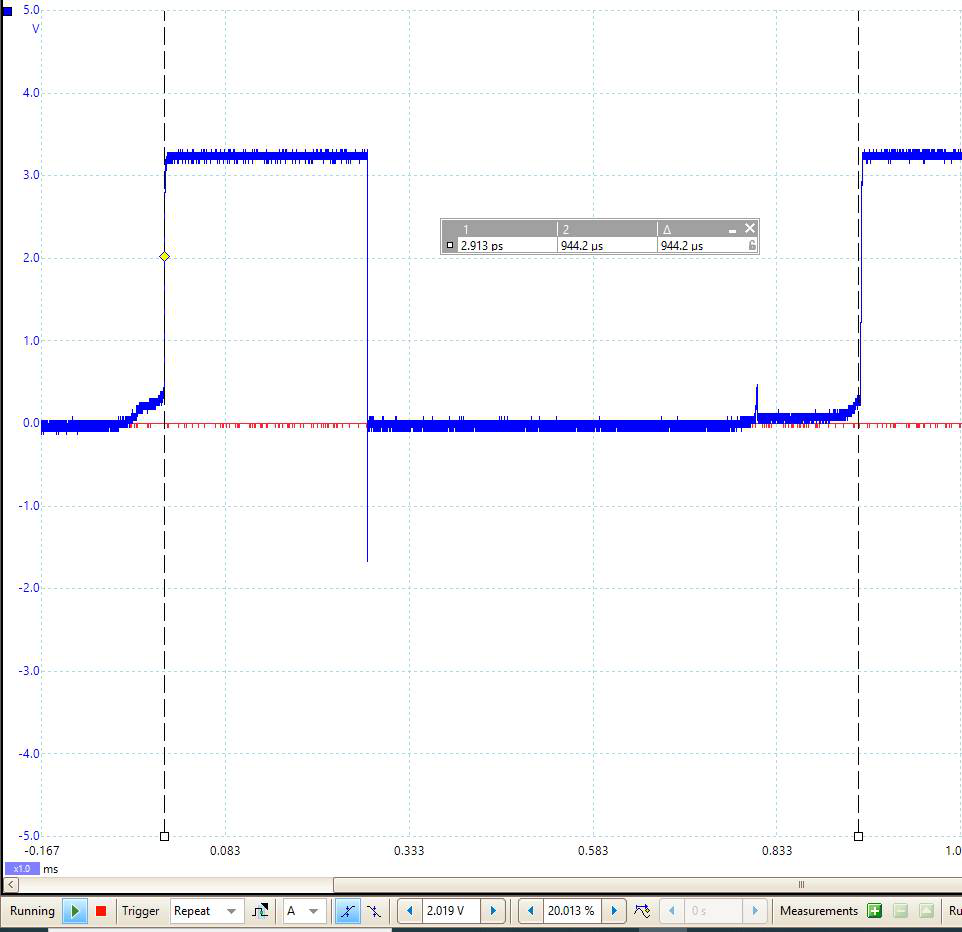


**What is the frequency of this SPI communication?**

**1.18 MHz**

**On the Picoscope, go to Tools -> Serial Decoding. Then select SPI and in the SPI window, choose data and clock as channel A and B respectively. Capture the screen including the waveforms and the decoded data when ESP32 sends number 5, 6, 7 to 74HC595. You can monitor the decoded data until you see the data for 5, 6, 7 then press the stop button and capture the screen.**



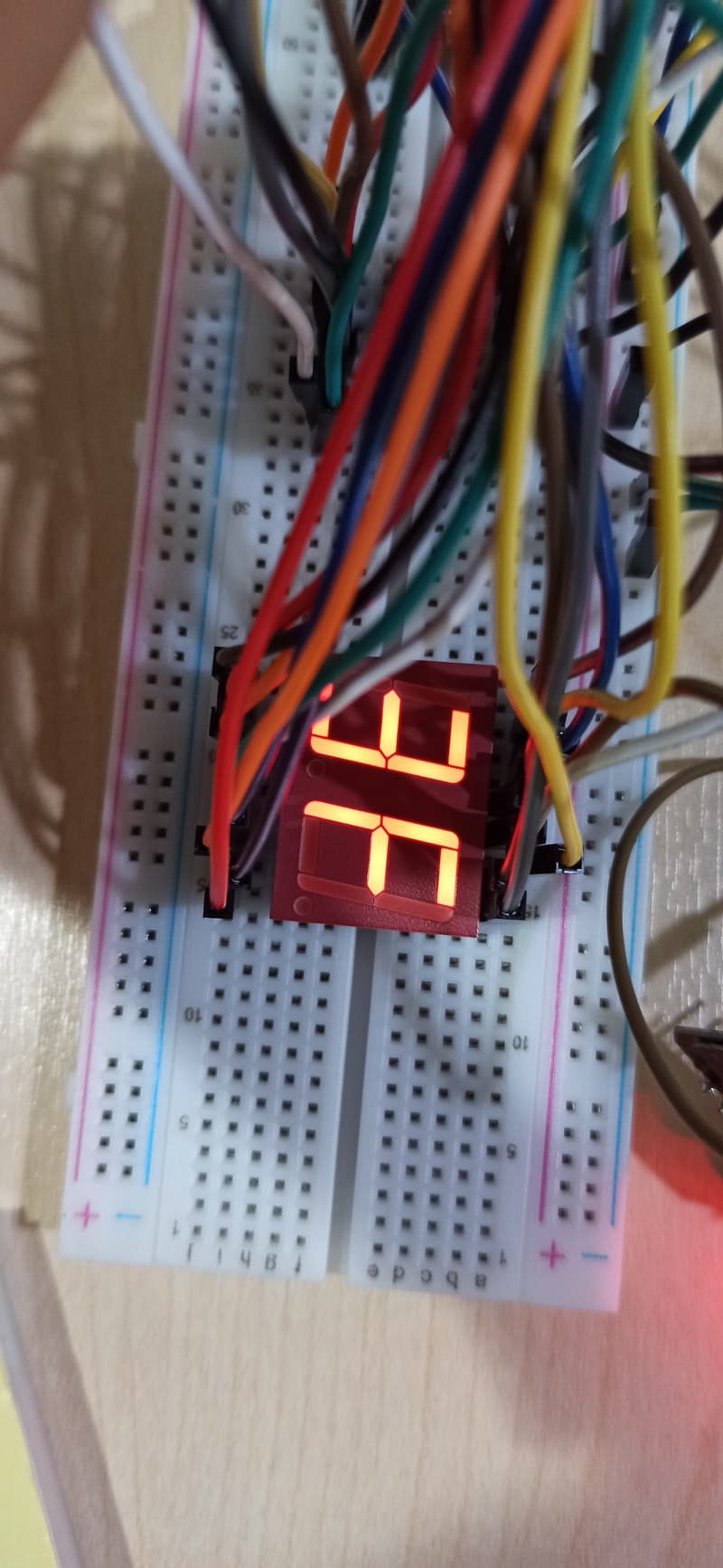


**B3. Using the same circuit and program as in B2, instead of connecting CA\_D2 to GND, connect it to an available output pin. Also connect CA\_D1 to another available output pin.**

**In your program, set the pins connecting to CA\_D2 and CA\_D1 to OUTPUT and always write LOW or 0 to those pins. The results should be that both digits display the same number. Take a picture of your 7-segment display and put it below.**

**---**

**Now always write HIGH or 1 to pins connecting to CA\_D2 while CA\_D1 is still LOW or 0. The results should be digit 2 is off while digit 1 is on. Take a picture of your 7-segment display and put it below.**



**If you want to display two digits differently at the same time, you have to send the data for digit 1 and turn it on while digit 2 is off. Then quickly turn digit 1 off, send the data for digit 2 and turn digit 2 on. This process has to be done so quick that human eyes see both digit as though they were displayed simultaneously. To turn on or off each digit, you have to control the common cathode.**

**Write a program so that it can display 2 different digits on your 7-segment display. The program displays numbers counting from 00 to FF in every 500ms. Name your program as** 7Segment-2Digits-74HC595.ino**. Put it in the same folder as this document and submit both as a ZIP file as described in the assignment description.**