«به نام حق»

دانشکده مهندسی برق

ترم بهار 4022

درس: آزمایشگاه ریزپردازنده

استاد درس:

تهیه کننده :

رامین توکلی ۹۹۲۵۰۶۳

موضوع پروژه:

سیستم برشگر مفتول

شرح عملکرد سیستم اول:

یک شرکت تولیدکننده مفتول‌های فلزی می‌خواهد دسته مفتول‌هایی با طول مشخص شده توسط مشتری تولید نماید. برای این منظور نیاز به سیستمی مبتنی بر ریزپردازنده دارد که با دریافت طول و تعداد مفتول مورد نظر، در زمان‌های مناسب و به تعداد دفعات مورد نیاز، فرمان برش مفتول را صادر نماید، به گونه‌ای که تعداد مفتول با طول مورد نظر تولید گردد.

این سیستم دارای یک صفحه کلید ماتریسی 4*3 می‌باشد. متصدی سیستم، ابتدا یک عدد ۲ رقمی را به عنوان تعداد مفتول مورد نیاز وارد کرده و در ادامه پس از فشار دادن کلید *، با وارد کردن یک عدد ۲ رقمی، طول مفتول‌های مورد نیاز بر حسب متر را تعیین می‌کند. سپس با فشار دادن کلید #، سیستم برشگر فعال می‌گردد.

در این سیستم، در مسیر حرکت مفتول بر روی دستگاه، غلطکی قرار دارد که با عبور مفتول از روی آن، میچرخد. این غلطک به ازای هر دور چرخش خود، یک پالس الکتریکی تولید می‌کند. محیط این غلطک ۲۵ سانتی‌متر است؛ بنابراین به ازای عبور هر ۲۵ سانتی‌متر مفتول از روی آن، یک پالس تولید خواهد شد. پس از اینکه متصدی، برشگر را فعال کرد با فشردن کلید #، در هر لحظه، طول مفتول عبور کرده بر حسب متر بعد از انجام برش قبلی، بر روی نمایشگر سیستم نمایش داده خواهد شد. این نمایشگر از دو عدد 7-segment تشکیل شده است. زمانی که طول مفتول در حال عبور از دستگاه، به اندازه تعیین شده اولیه (مقدار وارد شده از طریق صفحه کلید) رسید، بایستی فرمان برش مفتول صادر گردد. برای انجام برش، کافی است که یک خروجی تک بیتی، دچار تغییر وضعیت (اگر یک است، صفر و اگر صفر است، یک شود) گردد.

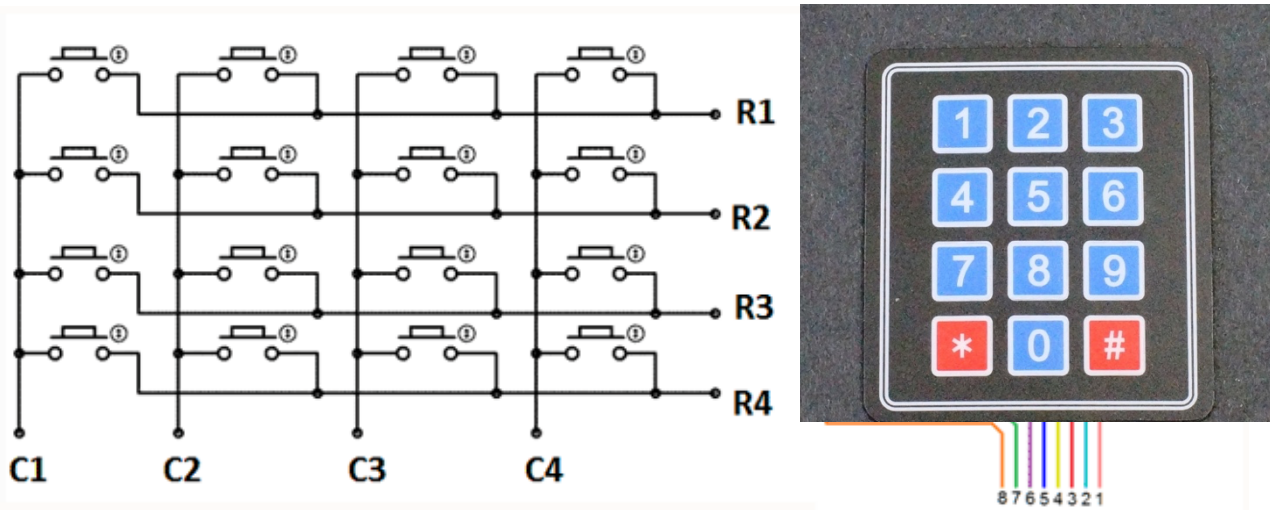
مقدمه

این پروژه شامل طراحی یک سیستم برش مفتول با استفاده از یک میکروکنترلر ATmega64 است که به اپراتور اجازه می‌دهد تعداد و طول سیم‌هایی را که باید با استفاده از صفحه کلید برش داده شود را مشخص کند. این سیستم از یک صفحه نمایش ۷ قسمتی برای نشان دادن طول فعلی سیم استفاده می کند و یک خروجی را تغییر می دهد تا در صورت رسیدن به طول مورد نظر، برش را فعال کند. کل سیستم با استفاده از پروتئوس شبیه سازی شده است.

اجزای مورد استفاده در این پروژه و شبیه سازی

۱. صفحه کلید ماتریسی[1]:

یک صفحه کلید ماتریس برای ورودی کاربر استفاده می شود. صفحه کلید به اپراتور اجازه می دهد تا تعداد سیم ها و طول هر سیم را بر حسب متر وارد کند.



صفحه کلید به میکروکنترلر متصل است که ردیف ها و ستون ها را برای تشخیص فشار دادن کلیدها اسکن می کند. فشار دادن هر کلید مربوط به یک کاراکتر خاص است که برای تنظیم پارامترهای عملیات برش مفتول استفاده می شود.

[1] Keypad

۲. نمایشگر ۷ قسمتی[۲] کاتد مشترک

دو نمایشگر ۷ قطعه ای (نوع کاتد معمولی) برای نمایش مقادیر عددی استفاده می شود. این نمایشگرها برای نشان دادن طول فعلی سیم در حال پردازش استفاده می شوند.

نمایشگرهای ۷ قسمتی توسط میکروکنترلر کنترل می شوند که بخش های مناسب را برای نشان دادن ارقام از 0 تا 9 روشن می کند. یک نمایشگر مکان ده ها و دیگری مکان واحدهای جریان را نشان می دهد.

۳. LED برای خروجی ۱ بیتی

برای نشان دادن وضعیت سیستم کاتر از LED استفاده می شود .وضعیت خود را (روشن/خاموش) تغییر می‌دهد تا زمانی که کاتر باید فعال شود سیگنال دهد.

به یک پایه خروجی دیجیتال روی میکروکنترلر متصل می شود. وقتی به طول سیم مورد نظر رسید، میکروکنترلر وضعیت این پین را تغییر می‌دهد و LED را تغییر می‌دهد و به کاتر سیگنال می‌دهد تا فعال شود.

۴. مولد پالس

یک مولد پالس سیگنال ورودی را به میکروکنترلر می دهد. هر پالس نشان دهنده ۲۵ سانتی متر سیم در حال پردازش است.

مولد پالس اندازه گیری طول سیم را شبیه سازی می کند. به ازای هر چهار پالس دریافتی (معادل ۱ متر)، میکروکنترلر شمارنده طول را افزایش می دهد و نمایشگر را به روز می کند. مولد پالس به یک پین وقفه خارجی روی میکروکنترلر متصل است.

[۲] 7_SEG

عملکرد سیستم

سیستم به صورت زیر عمل می کند:

راه اندازی اولیه

میکروکنترلر پورت های صفحه کلید، نمایشگرهای ۷ قسمتی و خروجی کاتر را مقداردهی اولیه می کند. وقفه های خارجی برای شمارش پالس های تولید کننده پالس تنظیم می شوند.

ورودی از طریق صفحه کلید

- اپراتور یک عدد ۲ رقمی را وارد می کند تا تعداد سیم ها را با استفاده از صفحه کلید مشخص کند. پس از فشار دادن کلید اپراتور یک عدد ۲ رقمی دیگر را وارد می کند تا طول هر سیم را بر حسب متر مشخص کند. با فشار دادن کلید فرآیند برش شروع می شود.

شمارش پالس:

- سیستم پالس های تولید کننده پالس را می شمارد که هر پالس ۲۵ سانتی متر است. به ازای هر چهار پالس، طول نمایش داده شده ۱ متر افزایش می یابد.

به روز رسانی صفحه نمایش:

- طول فعلی سیم در حال پردازش در نمایشگرهای ۷ قطعه نمایش داده می شود. ده ها و واحدها به طور جداگانه در دو نمایشگر نشان داده شده اند.

فعال سازی خروجی:

- هنگامی که طول جریان به طول مشخص شده رسید، میکروکنترلر پین خروجی متصل به LED را تغییر می دهد و به کاتر سیگنال می دهد تا فعال شود. این فرآیند برای تعداد سیم مشخص شده تکرار می شود.

نتیجه:

این پروژه اجرای یک سیستم برش سیم با استفاده از یک میکروکنترلر ATmega64 ، یکپارچه سازی ورودی کاربر از طریق صفحه کلید، به روز رسانی صفحه نمایش بلادرنگ با استفاده از نمایشگرهای ۷ قسمتی و مدیریت وقفه خارجی برای اندازه گیری دقیق را نشان می دهد. شبیه‌سازی پروتئوس عملکرد را تأیید می‌کند و به اصلاح بیشتر سیستم اجازه می‌دهد.

بررسی کد به زبان سی در کدویژن:

شامل کتابخانه ها و تعریف ماکروها:

```
#include <mega64.h>
#include <delay.h>
#include <interrupt.h>
```

این خطوط شامل کتابخانه های لازم برای میکروکنترلر ATmega64، توابع تاخیر و مدیریت وقفه است.

```
/* Defines macros and global variables */
// Define ports for keypad
#define ROW1 PINE .0
#define ROW2 PINE .1
#define ROW3 PINE .2
#define ROW4 PINE .3
#define COL1 PINE .4
#define COL2 PINE .5
#define COL3 PINE .6
#define keypadPort PORTE
#define keypadPin PINE
// Define the output port for cutter activation
#define CUTTER PORTC .0
```

این ماکروها اتصالات پین را برای ردیف‌ها و ستون‌های صفحه کلید و پایه خروجی را برای فعال‌سازی خروجی تعریف می‌کنند.

متغیرهای جهانی:

```
// Variables
unsigned char key;
unsigned char num_wires = 0;
unsigned char Number_Wires = 0;
unsigned char wire_length = 0;
unsigned int pulse_count = 0;
unsigned char display_value = 0;
unsigned char I1, I2 = 0;
unsigned char temp = 0;
```

این متغیرها وضعیت صفحه کلید، تعداد سیم‌ها، طول سیم، تعداد پالس‌ها، مقدار نمایش و مقادیر موقت ورودی کلید را ذخیره می‌کنند.

صفحه کلید و کدهای مربوط به نمایشگر:

```
// Define keypad matrix
const unsigned char keypad[4][3] = {
    {'1', '2', '3'},
    {'4', '5', '6'},
    {'7', '8', '9'},
    {'*', '0', '#'}};

// Bit patterns for digits 0-9 on a common-cathode 7-segment display
const unsigned char segment_map[10] = {
    0b11111100, // 0
    0b01100000, // 1
    0b11011010, // 2
    0b11110010, // 3
    0b01100110, // 4
    0b10110110, // 5
    0b10111110, // 6
    0b11100000, // 7
    0b11111110, // 8
    0b11110110  // 9
};
```

آرایه صفحه کلید کلیدهای فیزیکی را به کاراکترهای مربوطه نگاشت می کند. آرایه segment_map الگوهای بیت را برای نمایش ارقام ۰-۹ در نمایشگر ۷ قسمتی ارائه می دهد.

تعریف توابع مورد نیاز:

```
/* Function declarations */

void display_number(unsigned char number);
void setup();
unsigned char read_keypad();
void process_pulse();
void external_interrupt_setup();
```

این خطوط توابع مورد استفاده در برنامه را اعلام می کنند.

تابع شمارش طول مفتول:

```
void display_number(unsigned char number)
{
    // Map number to segments
    unsigned char tens = number / 10;
    unsigned char units = number % 10;

    // Display the tens place digit on PORTA
    PORTA = segment_map[tens];

    // Display the units place digit on PORTB
    PORTB = segment_map[units];
}
```

این تابع یک عدد را به ارقام ده ها و واحدهای آن تبدیل می کند و آنها را در دو نمایشگر ۷ قسمتی متصل به PORTA و PORTB نمایش می دهد.

تابع راه اندازی:

```
void setup()
{
    // Set ports for 7-segment display as output
    DDRA = 0xFF;
    DDRB = 0xFF;

    DDRE = 0x70;  // Columns (PE4-PE6) as output, Rows (PE0-PE3) as input
    PORTE = 0x0F; // Enable pull-ups for rows

    // Set cutter output port as output
    DDRC = 0x01;
    display_number(0);

    // Setup external interrupt
    external_interrupt_setup();
}
```

این تابع پورت های میکروکنترلر را تنظیم می کند: PORTA و PORTB برای نمایشگرهای ۷ قسمتی، PORTE برای صفحه کلید و PORTC برای خروجی کاتر. همچنین برای پیکربندی وقفه های خارجی، external_interrupt_setup را فراخوانی می کند.

تابع خوانده صفحه کلید:

```
// Function to read the keypad
unsigned char read_keypad()
{
    unsigned char col, row;

    for (col = 0; col < 3; col++)
    {
        // Set the current column to low
        PORTE = ~(0x10 << col) & 0x70 | 0x0F;
```

```
        // Check each row
        for (row = 0; row < 4; row++)
        {
            if (!(PINE & (1 << row)))
            {
                // Debounce: wait for stable press
                // delay_ms(20);
                if (!(PINE & (1 << row)))
                {
                    // Wait until the key is released
                    while (!(PINE & (1 << row)))
                        ;

                    // Return the key value
                    return keypad[row][col];
                }
            }
        }
    }

    return 0; // Return 0 if no key is pressed
}
```

این عملکرد صفحه کلید را برای تشخیص فشار دادن کلیدها اسکن می کند. هر ستون را روی کم تنظیم می کند و ردیف ها را برای سیگنال کم بررسی می کند که نشان دهنده فشار دادن کلید است. مقدار کلید برگردانده می شود.

عملکرد پالس فرآیند:

```
void process_pulse()
{
    pulse_count++;
    if (pulse_count >= 4)
    {
        pulse_count = 0;
        display_value++;
        display_number(display_value);
```

```
        if (display_value >= wire_length)
        {
            CUTTER = !CUTTER; // Toggle cutter state
        }
    }
}
```

این تابع پالس های تولید کننده پالس را پردازش می کند. هر چهار پالس (که نشان دهنده ۱ متر است)، مقدار نمایش افزایش می یابد. وقتی به طول مورد نظر رسید، حالت برش تغییر می کند.

تنظیم وقفه خارجی:

```
void external_interrupt_setup()
{
    // Configure INT0 (PD2) to trigger on rising edge
    EICRA |= (1 << ISC01) | (1 << ISC00);
    EIMSK |= (1 << INT0);
    sei(); // Enable global interrupts
}
```

این تابع وقفه خارجی را در INT۰ پیکربندی می‌کند تا در لبه‌های در حال افزایش فعال شود و وقفه‌های سراسری را فعال می‌کند.

روتین سرویس وقفه خارجی:

```
// External interrupt 0 service routine
interrupt[EXT_INT0] void ext_int0_isr(void)
{
    if (temp == 1)
    {
        process_pulse();
    }
}
```

این ISR توسط وقفه خارجی ۰ راه اندازی می شود. وقتی متغیر temp برابر با یک است، process_pulse را فراخوانی می کند تا پالس را مدیریت کند.

روند اصلی برنامه:

```
void main()
{
    setup();
    while (1)
    {
        key = read_keypad();
        if (key == '*')
        {
            // Read number of wires
            do
            {
                I1 = read_keypad();
            } while (!I1);

            do
            {
                I2 = read_keypad();
            } while (!I2);
```

```
        delay_ms(20);

        num_wires = (I1 - '0') * 10 + (I2 - '0');
        display_number(num_wires);
        I1 = 0;
        I2 = 0;
    }
    else if (key == '#')
    {
        // Read wire length
        do
        {
            I1 = read_keypad();
        } while (!I1);

        do
        {
            I2 = read_keypad();
        } while (!I2);

        delay_ms(20);

        wire_length = (I1 - '0') * 10 + (I2 - '0');
        display_number(wire_length);
        delay_ms(100);
        display_value = 0;
        display_number(display_value);

        // Start the cutter system
        display_value = 0;
        pulse_count = 0;
        Number_Wires = 0;
        while (Number_Wires < num_wires)
        {
            temp = 1;
            while (display_value < wire_length)
            {
                // Wait for the wire to reach the required length
            }
```

```
                temp = 0;
                delay_ms(150);
                display_value = 0;
                display_number(display_value);
                Number_Wires++;
                CUTTER = !CUTTER;
                pulse_count = 0;
            }
            Number_Wires = 0;
            display_number(0);
        }
    }
}
```

تابع اصلی سیستم را مقداردهی اولیه می کند و برای مدیریت ورودی کاربر وارد یک حلقه بی نهایت می شود.

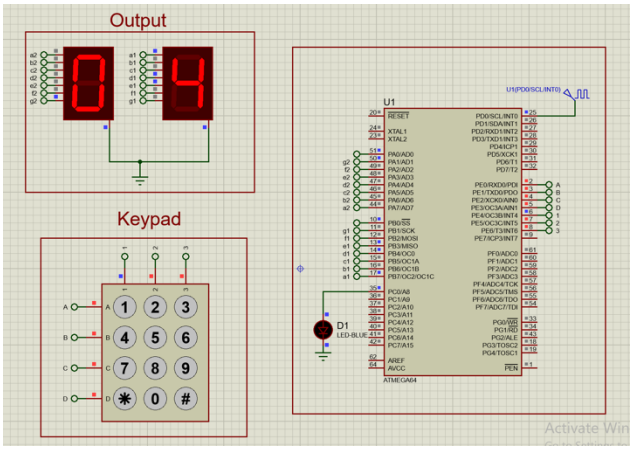
هنگامی که '*' فشار داده می شود، دو رقم را از صفحه کلید می خواند تا تعداد سیم ها را تنظیم کند.

هنگامی که "#" فشار داده می شود، دو رقم را از صفحه کلید می خواند تا طول سیم را تنظیم کند و فرآیند برش را شروع می کند.

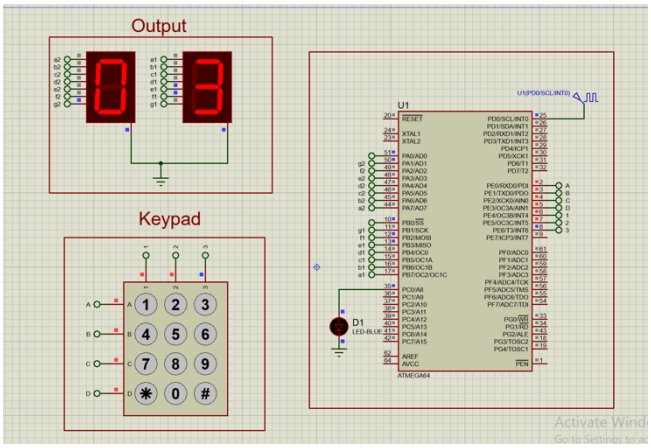
فرآیند برش شامل شمارش پالس ها و به روز رسانی صفحه نمایش تا رسیدن به طول مورد نظر است. کاتر در صورت نیاز فعال و جابجا می شود.

این توضیح دقیق هر بخش از کد و عملکرد آن را در سیستم برش سیم پوشش می دهد.

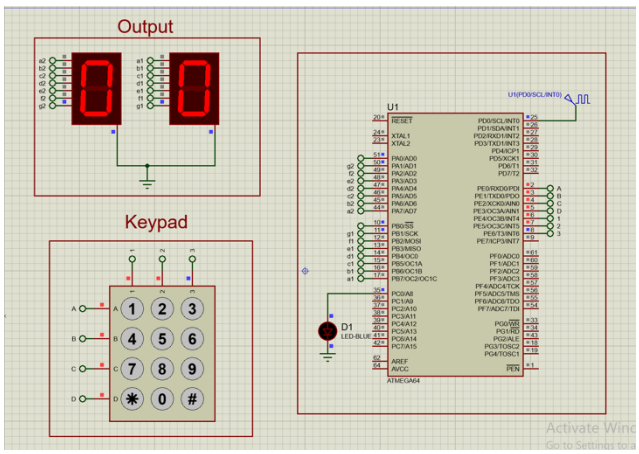
نتایج شبیه سازی برای اعمال تعداد مفتول ۳ و طول مفتول ۳ به ترتیب به شکل زیر است:



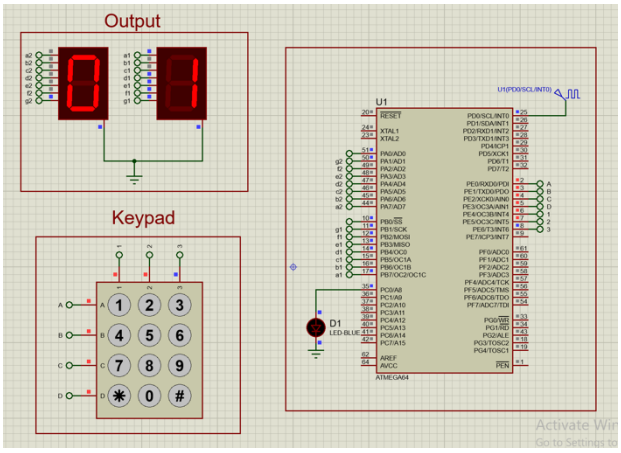
1) Start by entering the number of wiers



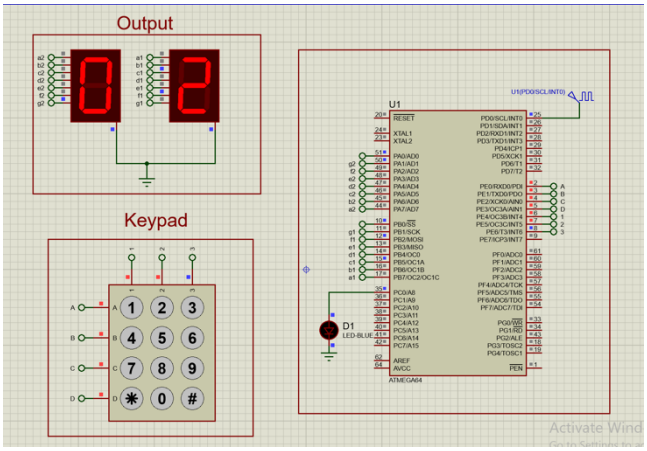
2) Enter the length of each wire



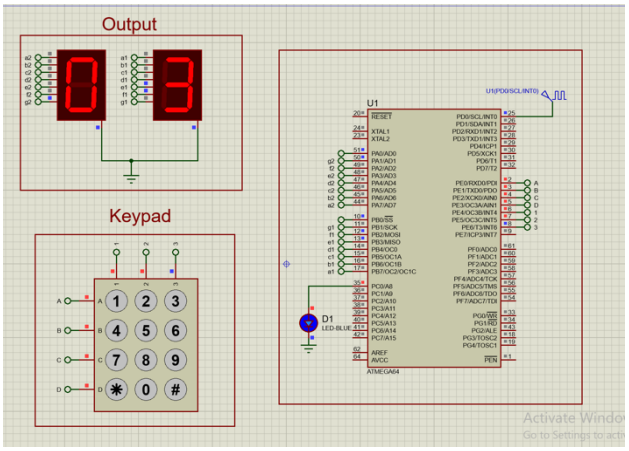
3) Start showing the length of the first passed wire



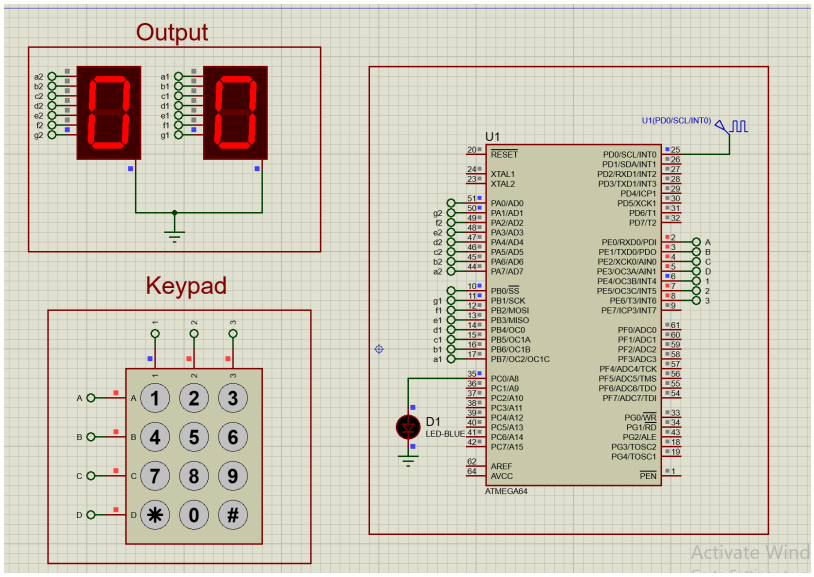
4) Display the length of the passed wire



5) Display the length of the passed wire



6) Reaching the desired length of the wire and activating the cutting command



7) After cutting all the wires, we reach the state of entering a new entry.

مشاهده می‌شود که پس از تمام شدن برش تمامی ۴ مفتول سیستم منتظر اعمال ورودی جدید برای برش‌های جدید می‌باشد.