آزمایش 2

طاها موسوی 98243058

نیلوفر مرادی جم 97243063

گروه 2

سوالات تحلیلی:

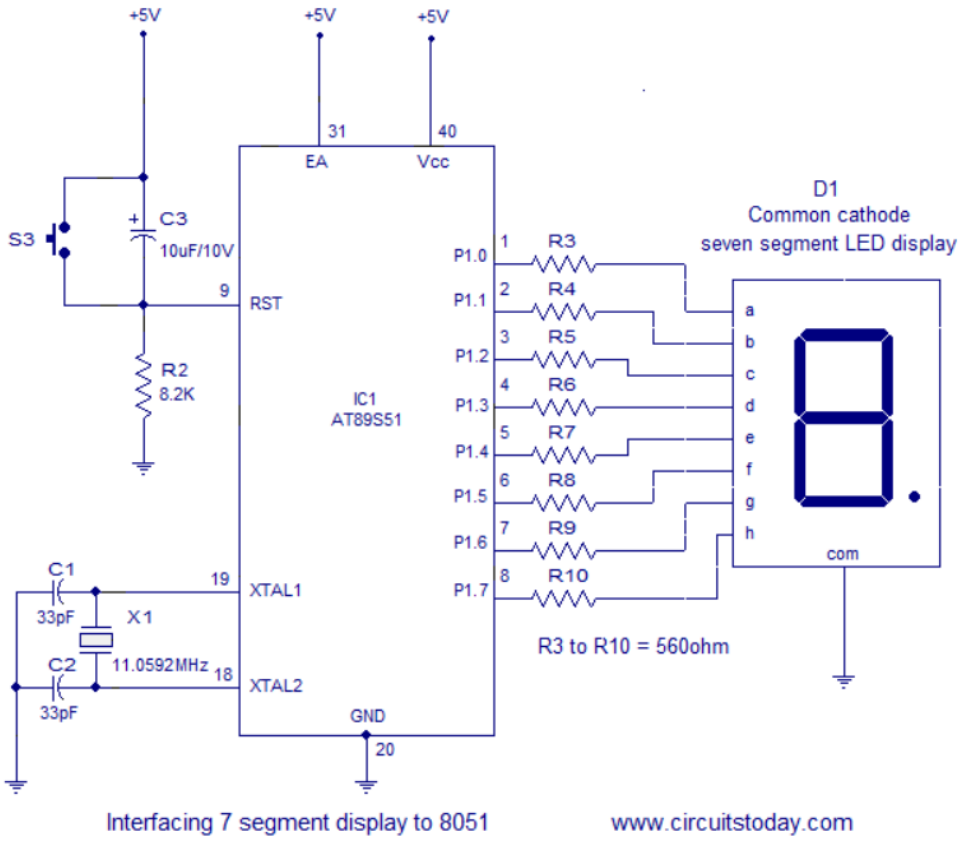
1 – مزایای به کارگیری کتابخانه های استاندارد CMSIS را شرح دهید. کاربرد CMSIS-DSP چیست؟

CMSIS یک رابط پیکربندی بسیار ساده بین هسته میکروکنترلر و واحدهای جانبی ارائه می دهد. این فناوری به شما امکان می دهد تا هم فرآیند پیکربندی لوازم جانبی و هم زمان رسیدن به بازار را برای موفقیت تجاری افزایش دهید. همچنین از خطاهای پیکربندی جلوگیری می کند. علاوه بر این، این فناوری جهانی را می توان در هر میکروکنترلر ARM Cortex-M و Cortex-A استفاده کرد. و از دیگر مواردی که در کلاس درس به آن اشاره شد: برنامه نویسی برای ARM را راحت تر می‌کند. این استاندارد سازی باعث می‌شود که کد ها قابلیت استفاده مجدد داشته باشند. و یک زیر ساختی برای تولید و فهم کد ایجاد می‌کند. و یاد گیری را آسانتر می‌کند. یکی دیگر از مزایاش این است که وابسته به toolchain خاصی نیست و مبتنی بر استاندارد c است. همینطور ریپوزیتوری آن open source است.

کاربرد CMSIS-DSP: مجموعه کتابخانه DSP با بیش از 60 عملکرد برای انواع داده های مختلف: نقطه ثابت و تک نقطه شناور دقیق (32 بیت). پیاده سازی های بهینه سازی شده برای مجموعه دستورات SIMD برای Cortex-M4/M7/M33/M35P موجود است. CMSIS-DSP امکان توسعه یک سیستم پردازش سیگنال دیجیتال DSP در زمان واقعی را فراهم می کند که مانند الگوریتم های DSP بی اهمیت نیست. کتابخانه CMSIS-DSP مجموعه ای غنی از توابع DSP است که برای هسته های مختلف پردازنده Cortex-M بهینه شده است. CMSIS-DSP به طور گسترده در صنعت استفاده می شود و همچنین تولید کد C بهینه شده را از MATLAB امکان پذیر می کند.

2 – آیا درایو مستقیم سون سگمنت (اتصال مستقیم آن به پایه های میکروکنترلر) کار صحیحی است؟ چنانچه پاسخ شما به این پرسش «مثبت »است، دلایل خود را بیان کنید و در صورت پاسخ «منفی » راهکار جایگزین را شرح دهید.

خیر. کار صحیحی نیست. چون ممکن است برخی جریان ها به آن آسیب بزنند. برای همین از مقاومت ها برای جلوگیری از این اتفاق استفاده می‌کنیم. همانطور که در تصویر زیر مشهود است:



Interfacing 7 segment display to 8051

3 – چه تفاوتی در ساختارLED ها با رنگهای متفاوت وجود دارد؟ آیا می‌توان در هر مداری آنها را جایگزین یکدیگر نمود؟ شرح دهید.

رنگ و کارایی نوری LED ها به مواد و فرآیندهای ساخت ال ای دی مربوط می شود. در حال حاضر، قرمز، سبز و آبی به طور گسترده ای استفاده می شود. مواد مورد استفاده در ساخت ال ای دی ها می توانند فوتون هایی با انرژی های مختلف تولید کنند و از این طریق طول موج نور ساطع شده از LED، یعنی طیف یا رنگ را کنترل کنند. طول موج LED را می توان با افت ولتاژ اتصال PN تعیین کرد.

ماده به کار رفته در عنصر نیمه هادی LED رنگ آن را تعیین می کند. دو نوع اصلی LED که در حال حاضر برای سیستم های روشنایی استفاده می شود آلیاژهای آلومینیم گالیوم ایندیم فسفید برای LED های قرمز، نارنجی و زرد استفاده می شود. و آلیاژهای نیترید گالیم برای LED های سبز، آبی و سفید. تغییرات جزئی در ترکیب این آلیاژها باعث تغییر رنگ نور ساطع شده می شود.

و به همین دلایل، LED ها با رنگ های متفاوت دارای ولتاژ های متفاوتی نیز هستند و در صورتی که نادرست به جای هم به کار روند باعث سوختن آن ها می‌شود.

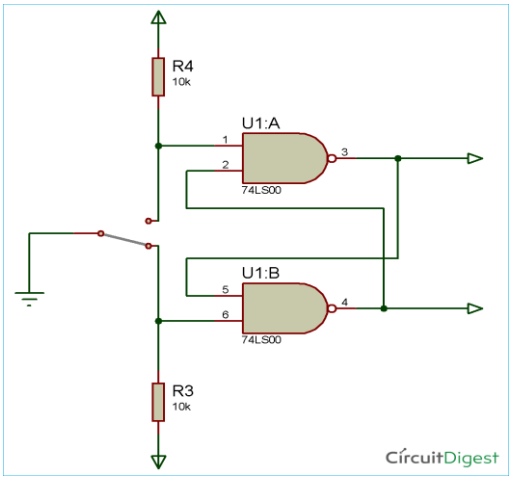
4 – مفهوم Switch Bouncing را توضیح دهید. راهکار های نرم افزاری و سخت افزاری قابل به کارگیری برای حل این مشکل را مختصرا توضیح دهید.

طبق مواردی که در کلاس درس تدریس شده، در سوییچ های مکانیکی یکی از اشکالاتی که میتواند رخ بدهد این است که در هنگام زدن دکمه سوییچ، یک سری نوساناتی رخ بدهد یا به اصطلاح Bounce کند. و این اتفاق ممکن است در فرکانس های بالا باعث شود هنگام خواندن چندبار اعداد 1 و 0 را ببینیم، به گونه ای که انگار چندبار دکمه سوییچ کلیک شده. و این باعث ایجاد اشتباه در نتیجه می‌شود.

راه حل سخت افزاری:

Hardwar Debouncing:

یکی از راه های سخت افزاری استفاده از فلیپ فالپ از نوع Reset & Set است که از چند گیت nand و دو مقاومت up-Pull بهره می برد. که از پرس سوییچ مدار جلوگیری می‌کند



R-C Debouncing : استفاده از یک خازن، که به کمک مقاومتی که در مدار است، یکی مدار RC، با یک ثابت زمانی درست می‌کند. که باعث می‌شود تغییر ولتاژی که از 0 به 1 یا برعکس رخ می‌دهد، با یک ثابت زمانی و آهسته رخ دهد، نه بسیار سریع. و باعث می‌شود نوساناتی که در یک لحظه رخ می‌دهد، حذف شوند.

Software Debouncing.: یکی از راه ها ایجاد Delay در کد می باشد البته فزودن تاخیر بصورت دستی مناسب نیست و راهکار بهتر نرم افزاری استفاده ازInterrupt ها می باشد. ابتدا مقدار را بخوانیم. و بعد از مدتی(ثابت زمانی یا یک دیلی ای که در نظر گرفته می‌شود) چک کنیم که آیا هنوز مقدار برابر قبلی است یا خیر. به این صورت می‌توانیم سوییچ را تشخیص دهیم. اگر تغییر کرده بود که این تغییر را در نظر می‌گیریم و اگر نه آن را نوسان در نظر می‌گیریم.

رفرنس های سوالات تحلیلی:

- کلاس درس و اسلاید های درسی

https://developer.arm.com/tools-and-software/embedded/cmsis

https://www.google.com/search?q=google+translate&rlz=1C1GCEA_enIR867IR867&oq=googl&aqs=chrome.0.69i59j69i57j69i59l2j46i67i199i465j0i67l5.1316j0j15&sourceid=chrome&ie=UTF-8

https://www.lrc.rpi.edu/programs/nlpip/lightinganswers/led/color.asp

https://www.quora.com/How-do-different-LEDs-produce-different-colors

https://www.circuitstoday.com/interfacing-seven-segment-display-to-8051

https://circuitdigest.com/electronic-circuits/what-is-switch-bouncing-and-how-to-prevent-it-using-debounce-circuit

دستور کار:

در این آزمایش ما از اجزای زیر برای شبیه سازی در پروتئوس استفاده کردیم:

- Stm32f401re microcontroller
- LED های سبز و قرمز
- 7-segment دوتایی
- مقاومت
- کلید

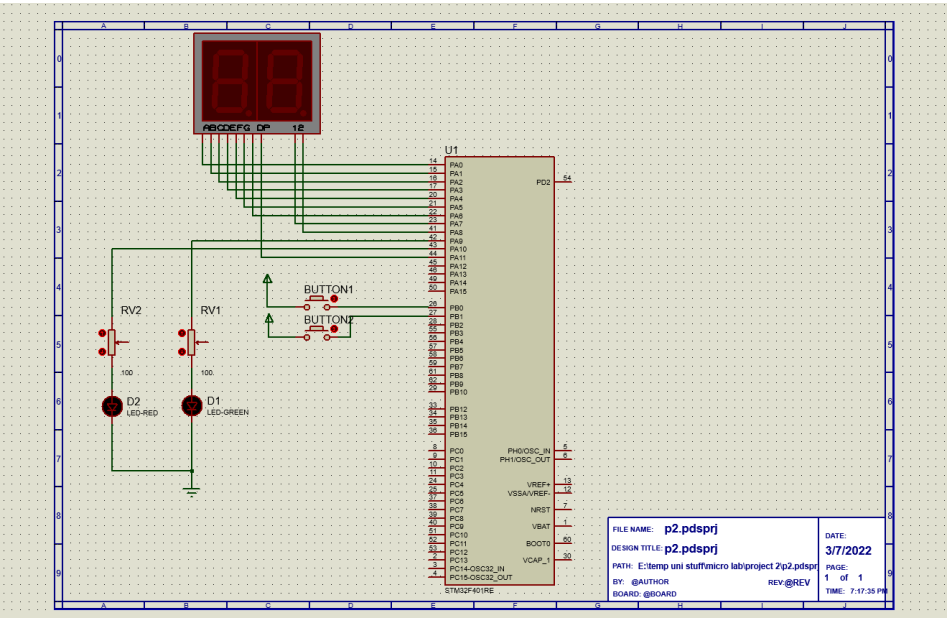
سپس انواع پروت ها و نوع خروجی و ورودی آن ها را مشخص می‌کنیم.

Port	PA0	PA1	PA2	PA3	PA4	PA5	PA6	PA7	PA8
نوع	Out	Out	Out	Out	Out	Out	Out	Out	Out
اجزا	ورودی a 7-seg	ورودی b 7-seg	ورودی c 7-seg	ورودی d 7-seg	ورودی e 7-seg	ورودی f 7-seg	ورودی g 7-seg	فعال کننده عدد سمت چپ	فعال کننده عدد سمت راست

Port	PA9	PA10	PA11	PB0	PB1
نوع	Out	Out	Out	In	In
اجزا	Green LED	Red LED	چراغ کوچک هر عدد	کلید1	کلید2

طبق این جدول اجزا را به هم متصل می‌کنیم. و نکته ای که مهم است این است که قبل از هر LED یک مقاومت می‌گداریم تا از سوختن آن جلوگیری کنیم.

و در نهایت شکل مدار به این صورت می‌شود:



از پورت های GPIOA برای خرویج و از پورت های GPIOB برای ورودی ها استفاده میکنیم.

در ادامه برای دیدن کارکرد مدار نیازمند نوشتن کد برای میکروکنترلر داریم. کد به زبان c نوشته میشود و فایل hex تولید شده از خروجی آن به میکروکنترلر داده خواهد شد.

دیفاین مقادیر پر استفاده و تعریف اینام برای استیت مدار:

```c
#include <stm32f4xx.h>

typedef enum {stop, resume, redTurn, greenTurn} state;


#define portA 0  // 7 Segment ports
#define portB 1
#define portC 2
#define portD 3
#define portE 4
#define portF 5
#define portG 6
#define LD 7
#define RD 8
#define dp 11

#define Green 9 //Colors
#define Red 10



#define MASK(x) (1UL <<(x))  // Shift 1, x time to the left.
#define NMASK(x) (~(1UL <<(x)))  //  ~ mask
```

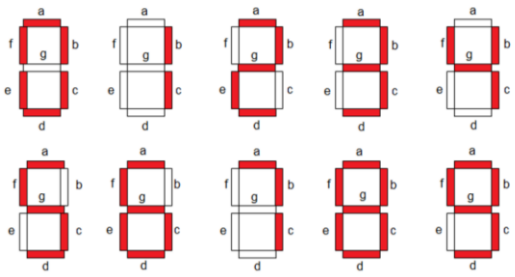
به دو عملیات پرکاربرد MASK شیفت دادن یک به اندازه ورودی داده شده بیت به سمت چپ و NMASK نقیض MASK نیاز داریم. در صورتی که عملیات MASK با OR به کار گرفت شود، بیت ورودی مقدار یک خواهد گرفت و در صورتی که عملیات NMASK با AND به کار گرفته شود، بیت ورودی مقدار صفر خواهد گرفت. از این دو عملیات به جهت خاموش و روشن کردن خروجی ها و تغییر مقادیر پورت های ریزپردازنده استفاده میکنیم.

```
void setGPIOConfig(void){
  RCC -> AHB1ENR |= RCC_AHB1ENR_GPIOAEN; // Setting GPIO Configuration
  RCC -> AHB1ENR |= RCC_AHB1ENR_GPIOBEN;
  GPIOA -> MODER = 0x555555;
  GPIOB -> MODER &= 0x0;
  GPIOB -> PUPDR &= NMASK(portA);
  GPIOB -> PUPDR |= MASK(portB);
  GPIOB -> PUPDR &= NMASK(portC);
  GPIOB -> PUPDR |= MASK(portD);
  RCC -> APB2ENR |= RCC_APB2ENR_SYSCFGEN;
}
```

بعد از تعریف مقادیر پیش فرض و متغیرهای مورد نیاز, در ابتدای تابع main نیاز است تنظیمات و پیکربندی های اولیه ریزپردازنده یعنی فعال کردن کالک های دو بخش مورد استفاده، تعیین پورت های ورودی و خروجی و مقداردهی مقادیر مربوط به مود Down-Pull ریز پردازنده و پیکربندی کالک خود سیستم انجام شود.

به جهت پیکربندی پورت های ورودی و خروجی, پورت ورودی با 01 و پورت خروجی با 00 نشان داده میشود که از عدد هگز آن ها استفاده شده است. همچنین برای فعال کردن Down-Pull بودن ورودی ها نیز نیاز است بیت های متناظر آنها در بخش مربوطه برابر با 10 قرار بگیرد که از عملیات های MASK و NMASK استفاده کرده ایم.

نمایش سون سگمنت:




```
void UpdateSevenSeg(void){  //Changing the value of seven segment
  switch(counter){
    case 0 : GPIOA -> ODR |= 0x3F; break;
    case 1 : GPIOA -> ODR |= 0x06; break;
    case 2 : GPIOA -> ODR |= 0x5B; break;
    case 3 : GPIOA -> ODR |= 0x4F; break;
    case 4 : GPIOA -> ODR |= 0x66; break;
    case 5 : GPIOA -> ODR |= 0x6D; break;
    case 6 : GPIOA -> ODR |= 0x7D; break;
    case 7 : GPIOA -> ODR |= 0x07; break;
    case 8 : GPIOA -> ODR |= 0x7F; break;
    case 9 : GPIOA -> ODR |= 0x6F; break;
  }
}
```

تابع , UpdateSevenSeg با توجه به مقدار Counter که نشان دهنده عدد موردنظر برای نمایش در LED میباشد, عدد هگز متناظر با آن مقدار را بر روی پورت های خروجی متناظر با آن در ریز پردازنده قرار میدهد. از این تابع هربار برای نمایش مقدار Counter بر روی LED استفاده میکنیم.

با توجه به حالت اولیه سیستم LED ,های خارجی و نشانگر های داخلی سون سگمنت مقدار دهی شوند. عملیات های MASK و, NMASK مقداردهی اولیه به گونه ای که LED قرمز و LED سمت چپ سون سگمنت روشن باشد, انجام شده است و مقادیر آن ها بر روی بیت های متناظر با آنها ر وی خروجی ریزپردازنده قرار گرفته است.

```
85 
86 void ChangeLEDsToRed(void){  //Turning on the red LED and turning off the green LED
87        GPIOA -> ODR |= MASK(Red);
88        GPIOA -> ODR &= NMASK(Green);
89        GPIOA -> ODR &= NMASK(LD);
90        GPIOA -> ODR |= MASK(RD);
91  }
92 
93 void ChangeLEDsToGreen(void){  //Turning on the green LED and turning off the red LED
94        GPIOA -> ODR &= NMASK(Red);
95        GPIOA -> ODR |= MASK(Green);
96        GPIOA -> ODR |= MASK(LD);
97        GPIOA -> ODR &= NMASK(RD);
98  }
```

تابع کنترل دیلی زمانی:

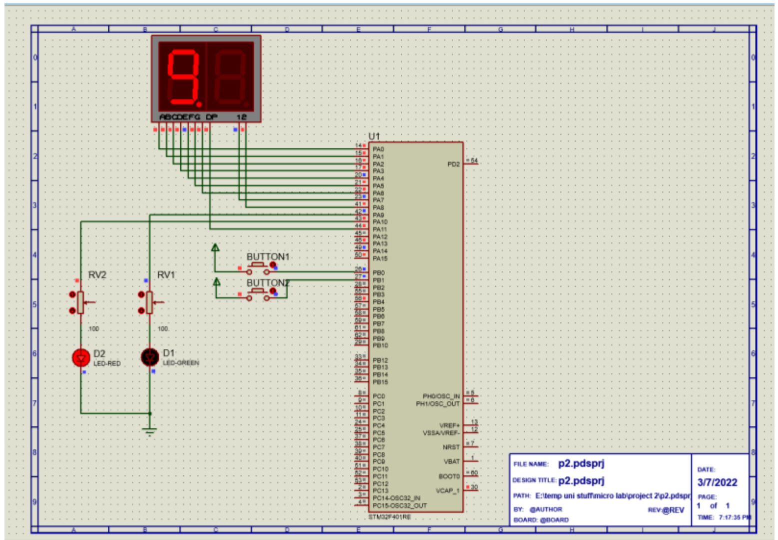
```
123 void Delay(volatile int time){
124   for(int i = 0; i< time; i++){
125     for ( int j = 0 ; j < time ; j++){ // mask(0) is used for button 1, and mask(1) for button 2
126         if(GPIOB -> IDR & MASK(0) ){
127           btn1Counter++;
128           do{ //Nothing to do, just is used for handling the clicks
129           }while(GPIOB -> IDR & MASK(0));
130           if(btn1Counter == 2){
131             isOp = 1; // Changing the color event
132             break;
133           }
134         }else if (GPIOB -> IDR & MASK(1)){
135             btn2Counter++;
136             do{ //Nothing to do, just is used for handling the clicks
137             }while(GPIOB -> IDR & MASK(1));
138             if(btn2Counter == 3){
139             tmpState = stateOfProgram;
140             stateOfProgram = stop;
141             isOp = 0; // stop event
142             }
143             if(btn2Counter == 4){
144               stateOfProgram = resume;
145               isOp = 1;  // Resume event
146               break;
147             }
148         }
149     }
150   }
151 }
```

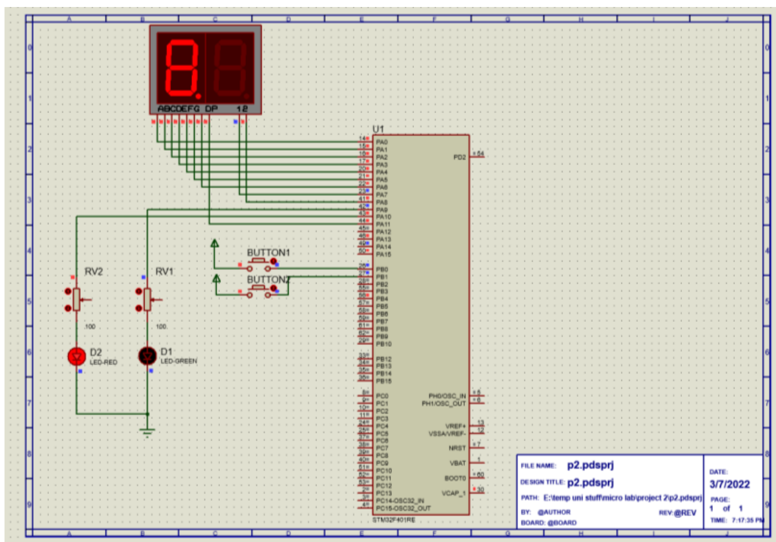
در یک حلقه بی نهایت به صورت مداوم و تکرار شونده، برنامه را اجرا کنیم. در ابتدا هر بار نیاز است وقفه ای به صورت داخلی با استفاده از حلقه های تو در تو ایجاد کنیم و در هر پیمایش، فشرده شدن دکمه ها را بررسی کنیم و متناسب با آن ها عملیات های مورد نیاز را انجام دهیم. سعی شده است مقدار پیمایش این حلقه ها به گونه ای باشد که اجرای اولیه تا پایان حلقه ها در حدود یک ثانیه طول بکشد. همچنین از آنجا که بعد از فشرده شدن هر دکمه تا صفر شدنEvent مربوط به این موضوع، ممکن است حلقه بارها پیمایش کند به همین جهت سبب افزایش بی مورد شمارشگر ها تنها با یک بار فشردن دکمه ها میشود، نیاز است حلقه های بی نهایتی بعد از فشرده شدن هر دکمه قرار دهیم تا وقتی که Event مربوط به فشرده شدن دکمه ها صفر شود. آن گاه به طور کامل مطمئن هستیم که دکمه تنها یک بار فشرده شده است.

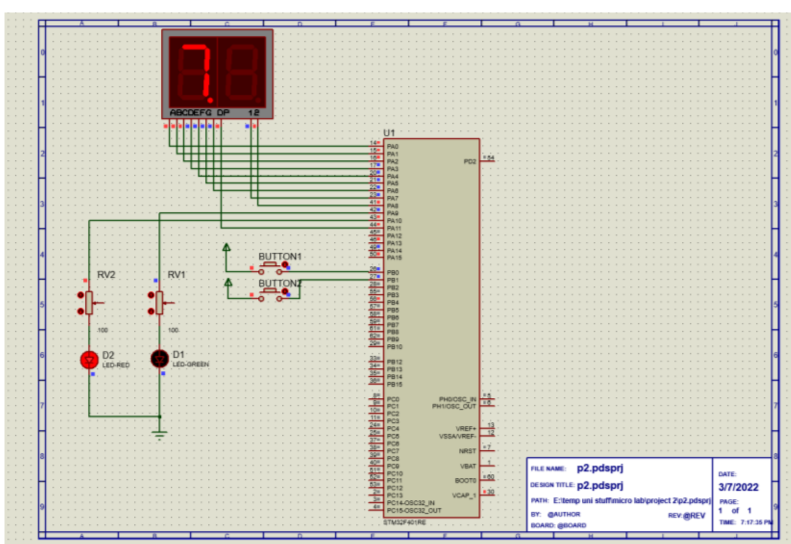
```
151 }
152
153 int main(){
154   setGPIOConfig();
155   UpdateSevenSeg();
156   init();
157   while(1){
158   Delay(1000);
159   if(stateOfProgram == stop){
160     if(isOp == 1){
161       ResetLED();
162       ChangeLEDs();
163       UpdateSevenSeg();
164       isOp = 0;
165     }
166     GPIOA -> ODR &= NMASK(dp); // Turnning Off 7 Segment Indicators
167   }
168   else{
169       if(stateOfProgram == resume){
170         stateOfProgram = tmpState;
171         counter --;
172         btn1Counter = btn2Counter = isOp =  0;
173       }else{
174         if(isOp == 0){
175         counter -- ;
176       }
177         // Push Button Event
178       else{
179         btn1Counter = btn2Counter = isOp = 0;
180         ChangeLEDs();
181       }
182     }
183     if(counter == 0){
184       ChangeLEDs();
185     }
186     ResetLED();
187     UpdateSevenSeg();
188     GPIOA -> ODR |= MASK(dp); // Turnning On 7 Segment Indicators
189   }
190   }
191 }
```

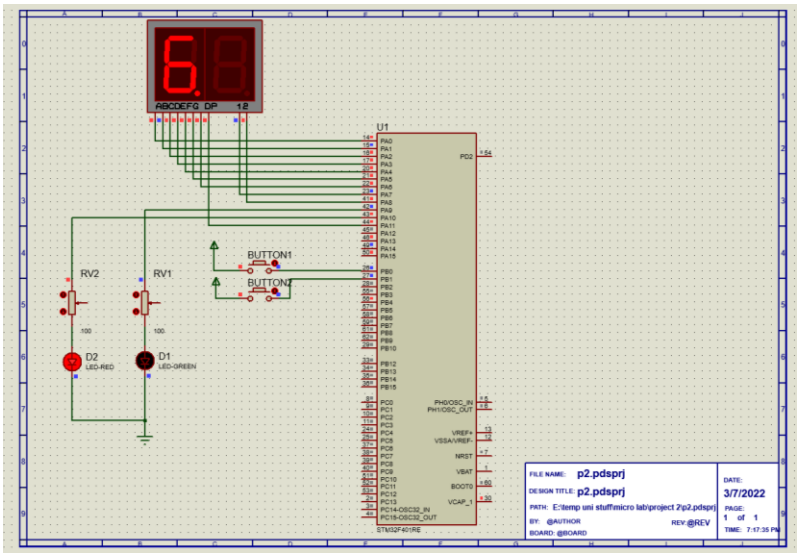
بعد از پایان تابع وقفه, نیاز است با توجه به متغیرهای کنترلی، تصمیماتی متناسب با دستورات مشخص شده در گزارش کار گرفته شود. تصویر بالا، دستورات اجرایی با توجه به مقادیر کنترلی را نشان میدهد که به صورت کلی عملکرد سیستم را مطابق با خواسته دستور کار تشکیل می دهند.

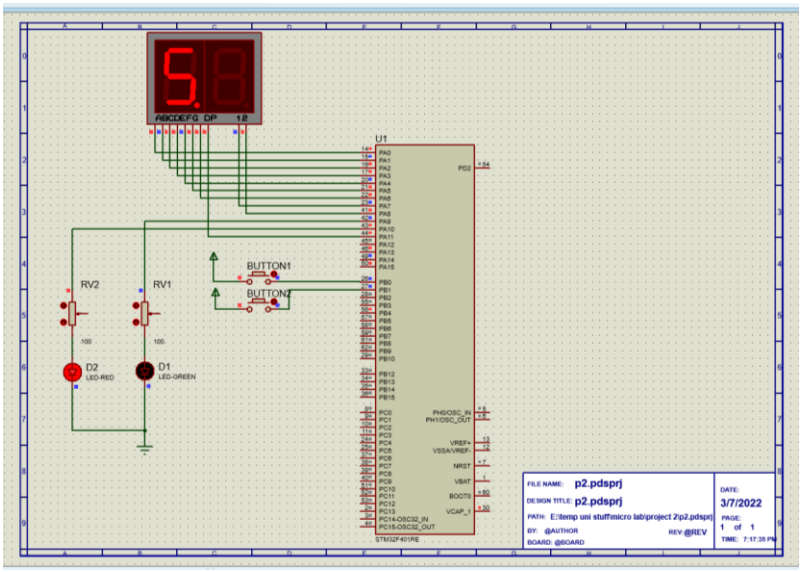
مشاهده خروجی پروژه در پروتئوس:

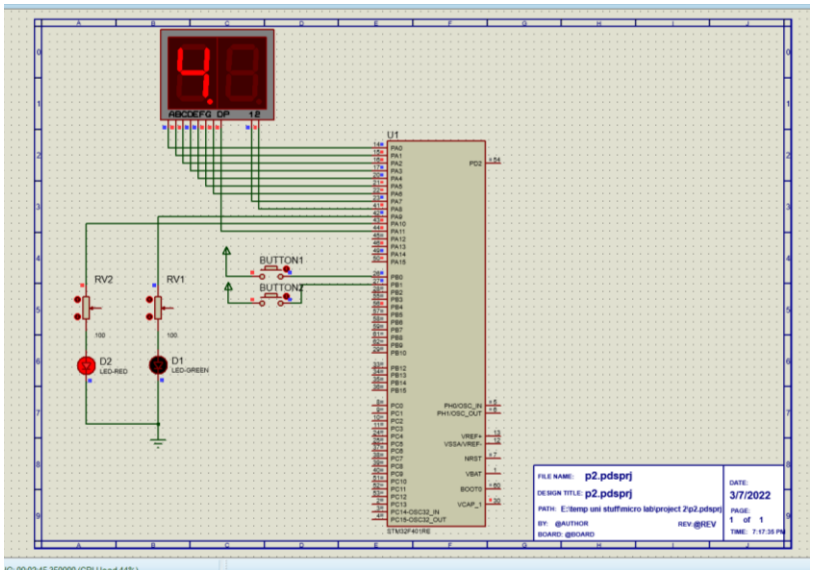


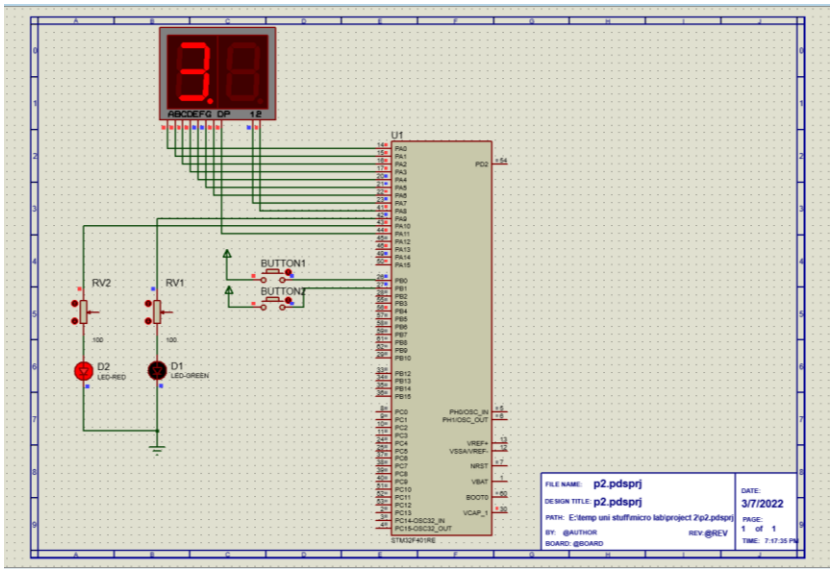


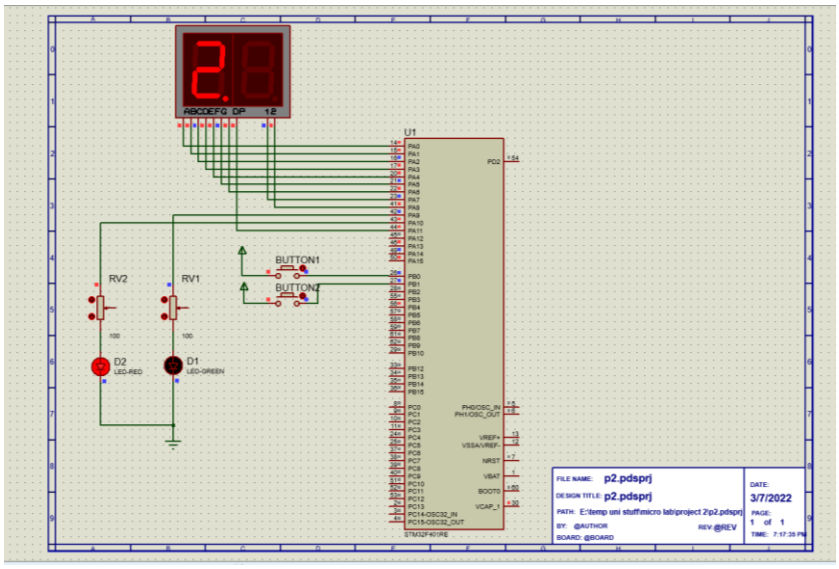


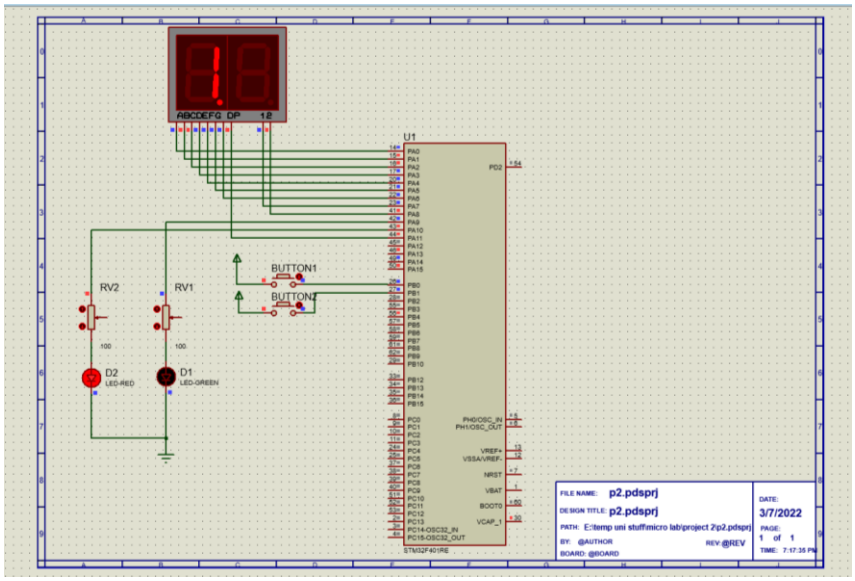

ABCDEFG DP 1 2
U1
BUTTON1
BUTTON2
RV2
RV1
D2
LED-RED
D1
LED-GREEN
STM32F401RE
FILE NAME: p2.pdsprj
DESIGN TITLE: p2.pdsprj
PATH: E:\temp uni stuff\micro lab\project 2\p2.pdsprj
BY: @AUTHOR
REV:@REV
BOARD: @BOARD
DATE: 3/7/2022
PAGE: 1 of 1
TIME: 7:17:35 PM

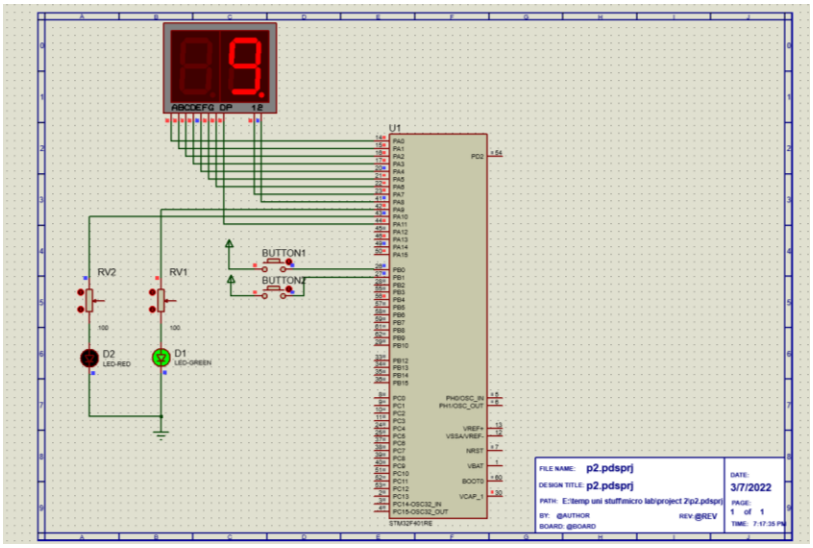

ABCDEFG DP 1 2
U1
BUTTON1
BUTTON2
RV2
RV1
D2
LED-RED
D1
LED-GREEN
STM32F401RE
FILE NAME: p2.pdsprj
DESIGN TITLE: p2.pdsprj
PATH: E:\temp uni stuff\micro lab\project 2\p2.pdsprj
BY: @AUTHOR
REV:@REV
BOARD: @BOARD
DATE: 3/7/2022
PAGE: 1 of 1
TIME: 7:17:35 PM

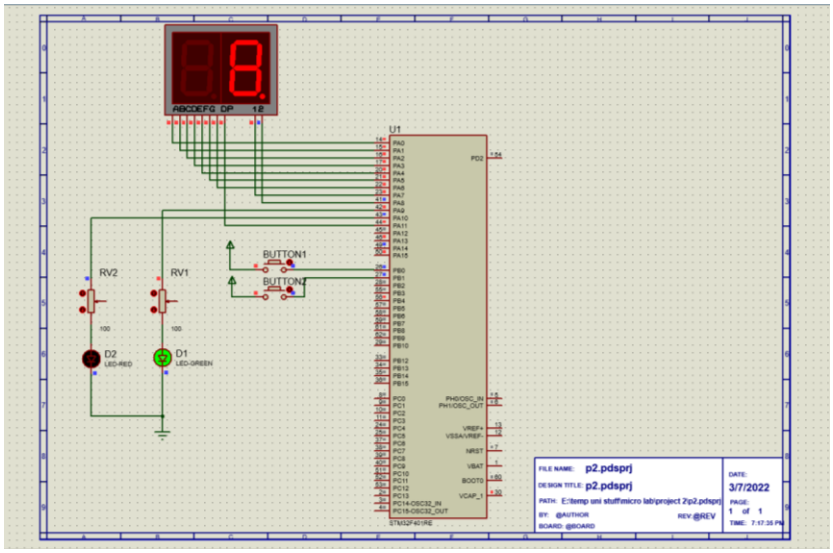

ABCDEFG DP 1 2
U1
BUTTON1
BUTTON2
RV2
RV1
D2
LED-RED
D1
LED-GREEN
STM32F401RE
FILE NAME: p2.pdsprj
DESIGN TITLE: p2.pdsprj
PATH: E:\temp uni stuff\micro lab\project 2\p2.pdsprj
BY: @AUTHOR
REV:@REV
BOARD: @BOARD
DATE: 3/7/2022
PAGE: 1 of 1
TIME: 7:17:35 PM

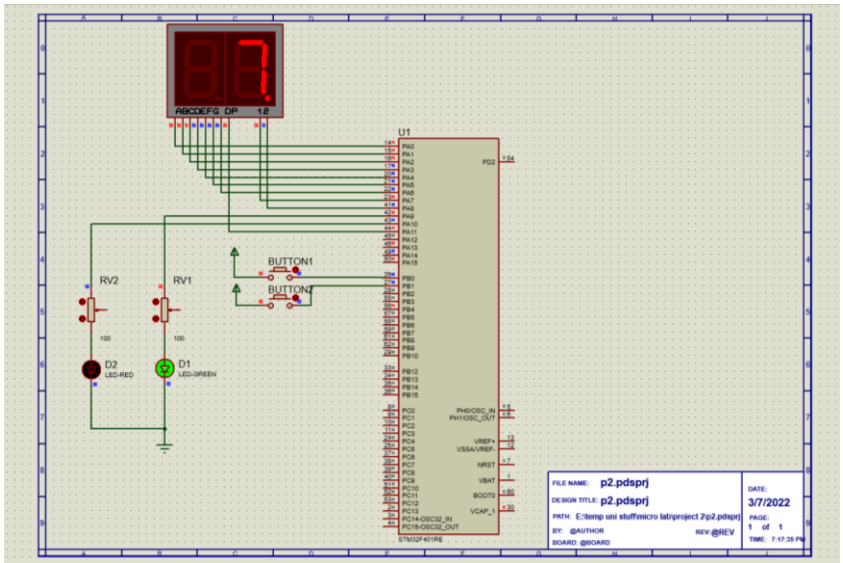

3
ABCDEFG DP 12
U1
BUTTON1
BUTTON2
RV2
RV1
100
100
D2
LED-RED
D1
LED-GREEN
STM32F401RE
FILE NAME: p2.pdsprj
DESIGN TITLE: p2.pdsprj
PATH: E:\temp uni stuff\micro lab\project 2\p2.pdsprj
BY: @AUTHOR
REV:@REV
BOARD: @BOARD
DATE: 3/7/2022
PAGE: 1 of 1
TIME: 7:17:35 PM

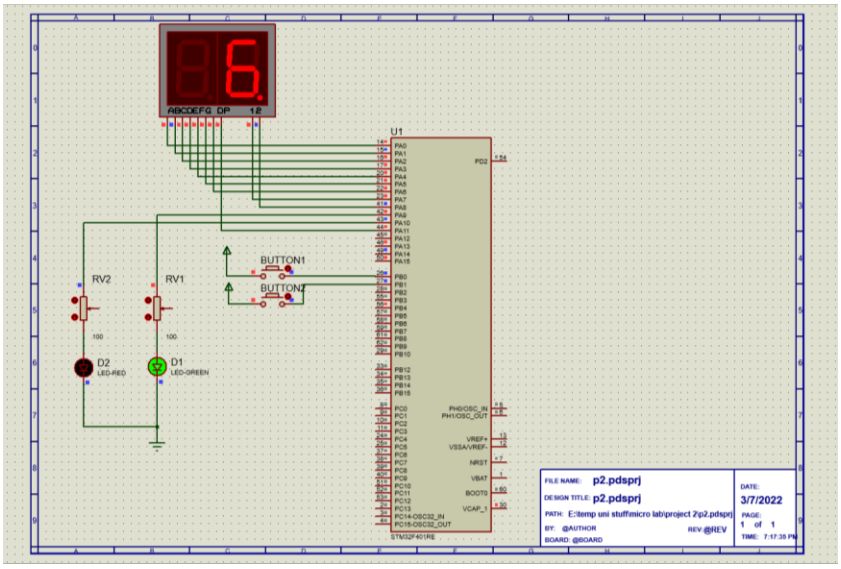

2
ABCDEFG DP 12
U1
BUTTON1
BUTTON2
RV2
RV1
100
100
D2
LED-RED
D1
LED-GREEN
STM32F401RE
FILE NAME: p2.pdsprj
DESIGN TITLE: p2.pdsprj
PATH: E:\temp uni stuff\micro lab\project 2\p2.pdsprj
BY: @AUTHOR
REV:@REV
BOARD: @BOARD
DATE: 3/7/2022
PAGE: 1 of 1
TIME: 7:17:35 PM

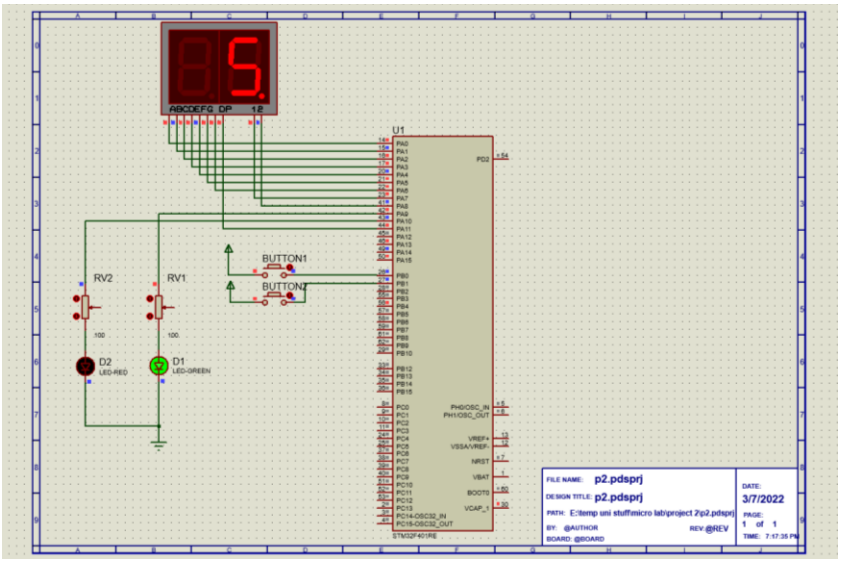

1
ABCDEFG DP 12
U1
BUTTON1
BUTTON2
RV2
RV1
100
100
D2
LED-RED
D1
LED-GREEN
STM32F401RE
FILE NAME: p2.pdsprj
DESIGN TITLE: p2.pdsprj
PATH: E:\temp uni stuff\micro lab\project 2\p2.pdsprj
BY: @AUTHOR
REV:@REV
BOARD: @BOARD
DATE: 3/7/2022
PAGE: 1 of 1
TIME: 7:17:35 PM

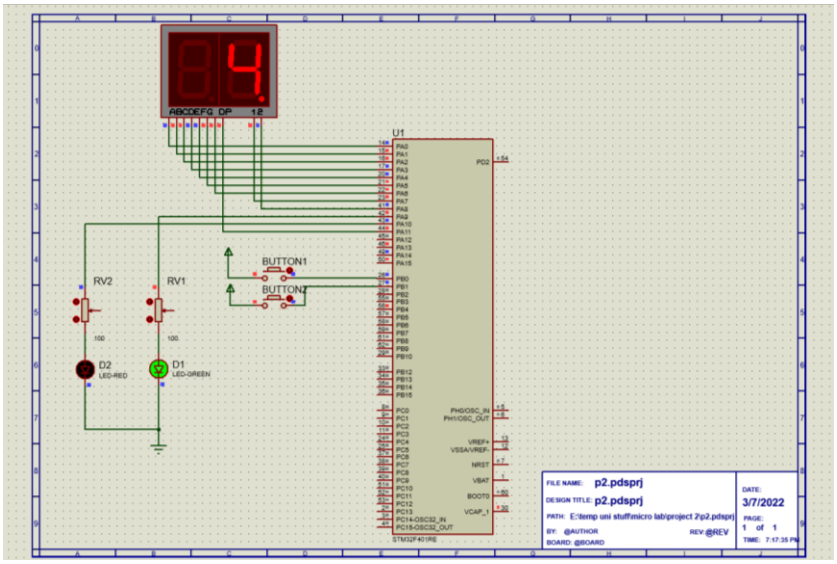

ABCDEFG DP 12
U1
RV2
RV1
BUTTON1
BUTTON2
D2
LED-RED
D1
LED-GREEN
STM32F401RE
FILE NAME: p2.pdsprj
DESIGN TITLE: p2.pdsprj
PATH: E:\temp uni stuff\micro lab\project 2\p2.pdsprj
BY: @AUTHOR
BOARD: @BOARD
REV: @REV
DATE: 3/7/2022
PAGE: 1 of 1
TIME: 7:17:35 PM

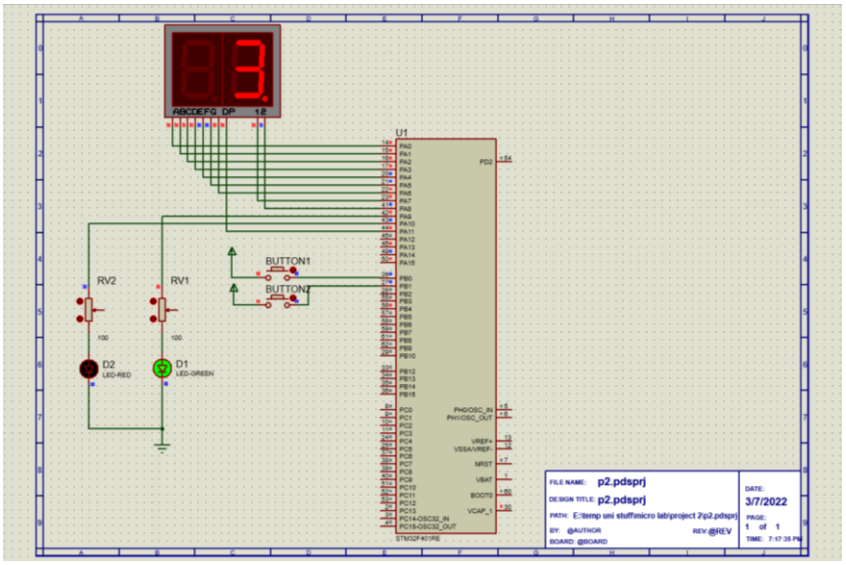

ABCDEFG DP 12
U1
RV2
RV1
BUTTON1
BUTTON2
D2
LED-RED
D1
LED-GREEN
STM32F401RE
FILE NAME: p2.pdsprj
DESIGN TITLE: p2.pdsprj
PATH: E:\temp uni stuff\micro lab\project 2\p2.pdsprj
BY: @AUTHOR
BOARD: @BOARD
REV: @REV
DATE: 3/7/2022
PAGE: 1 of 1
TIME: 7:17:35 PM

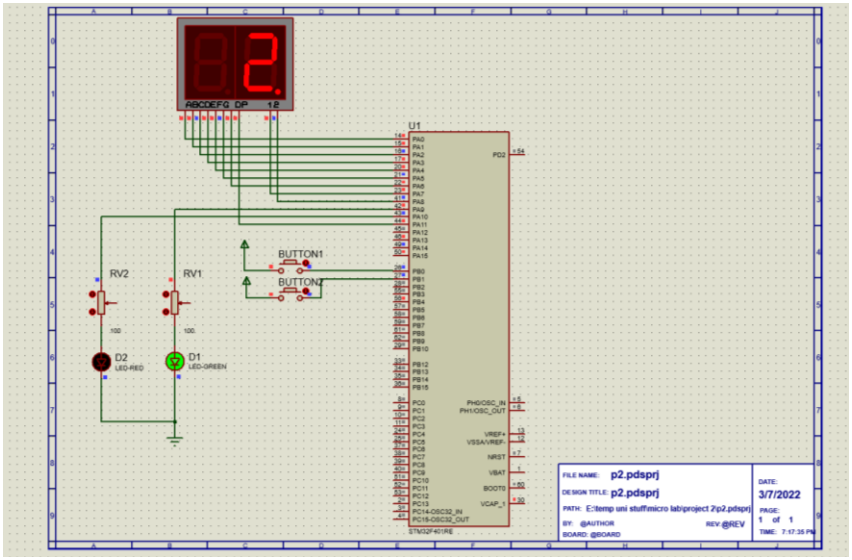

ABCDEFG DP 12
U1
RV2
RV1
BUTTON1
BUTTON2
D2
LED-RED
D1
LED-GREEN
STM32F401RE
FILE NAME: p2.pdsprj
DESIGN TITLE: p2.pdsprj
PATH: E:\temp uni stuff\micro lab\project 2\p2.pdsprj
BY: @AUTHOR
BOARD: @BOARD
REV: @REV
DATE: 3/7/2022
PAGE: 1 of 1
TIME: 7:17:35 PM

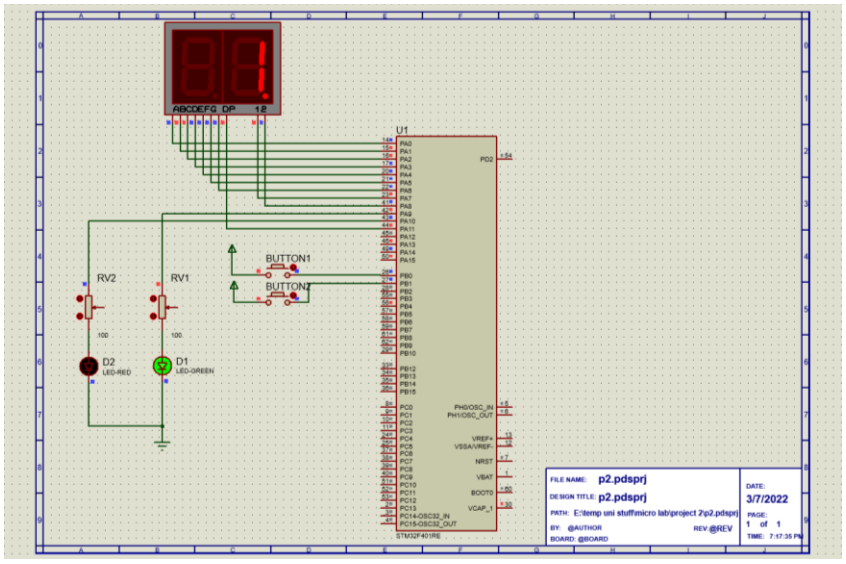

ABCDEFG DP 12
U1
RV2
RV1
BUTTON1
BUTTON2
D2
LED-RED
D1
LED-GREEN
FILE NAME: p2.pdsprj
DESIGN TITLE: p2.pdsprj
PATH: E:\temp uni stuff\micro lab\project 2\p2.pdsprj
BY: @AUTHOR
REV:@REV
BOARD: @BOARD
DATE: 3/7/2022
PAGE: 1 of 1
TIME: 7:17:35 PM


ABCDEFG DP 12
U1
RV2
RV1
BUTTON1
BUTTON2
D2
LED-RED
D1
LED-GREEN
FILE NAME: p2.pdsprj
DESIGN TITLE: p2.pdsprj
PATH: E:\temp uni stuff\micro lab\project 2\p2.pdsprj
BY: @AUTHOR
REV:@REV
BOARD: @BOARD
DATE: 3/7/2022
PAGE: 1 of 1
TIME: 7:17:35 PM


ABCDEFG DP 12
U1
RV2
RV1
BUTTON1
BUTTON2
D2
LED-RED
D1
LED-GREEN
FILE NAME: p2.pdsprj
DESIGN TITLE: p2.pdsprj
PATH: E:\temp uni stuff\micro lab\project 2\p2.pdsprj
BY: @AUTHOR
REV:@REV
BOARD: @BOARD
DATE: 3/7/2022
PAGE: 1 of 1
TIME: 7:17:35 PM


ABCDEFG DP 12
U1
BUTTON1
BUTTON2
RV2
RV1
100
100
D2
LED-RED
D1
LED-GREEN
STM32F401RE
FILE NAME: p2.pdsprj
DESIGN TITLE: p2.pdsprj
PATH: E:\temp uni stuff\micro lab\project 2\p2.pdsprj
BY: @AUTHOR
REV:@REV
BOARD: @BOARD
DATE: 3/7/2022
PAGE: 1 of 1
TIME: 7:17:35 PM


ABCDEFG DP 12
U1
BUTTON1
BUTTON2
RV2
RV1
100
100
D2
LED-RED
D1
LED-GREEN
STM32F401RE
FILE NAME: p2.pdsprj
DESIGN TITLE: p2.pdsprj
PATH: E:\temp uni stuff\micro lab\project 2\p2.pdsprj
BY: @AUTHOR
REV:@REV
BOARD: @BOARD
DATE: 3/7/2022
PAGE: 1 of 1
TIME: 7:17:35 PM


ABCDEFG DP 12
U1
BUTTON1
BUTTON2
RV2
RV1
100
100
D2
LED-RED
D1
LED-GREEN
STM32F401RE
FILE NAME: p2.pdsprj
DESIGN TITLE: p2.pdsprj
PATH: E:\temp uni stuff\micro lab\project 2\p2.pdsprj
BY: @AUTHOR
REV:@REV
BOARD: @BOARD
DATE: 3/7/2022
PAGE: 1 of 1
TIME: 7:17:35 PM

رفرنس دستور کار:

کلاس درس و اسلاید های درسی

دیتا شیت و رفرنس منوآل

https://www.keil.com/pack/doc/cmsis/DSP/html/index.html

https://www.st.com/en/embedded-software/stsw-stm32065.html

https://www.keil.com/dd/docs/arm/st/stm32f4xx/stm32f4xx.h

https://www.keil.com/dd/docs/arm/st/stm32f4xx/stm32f4xx.h

https://stm32f4-discovery.net/2014/04/stm32f429-discovery-gpio-tutorial-with-onboard-leds-and-button/

https://microcontrollerslab.com/7-segment-display-interfacing-with-pic-microcontroller/