GEBZE TECHNİCAL UNIVERTİSY

ELEC 335

PROJECT - 02

REPORT

CALCULATOR

BERKAY TÜRK

171024024

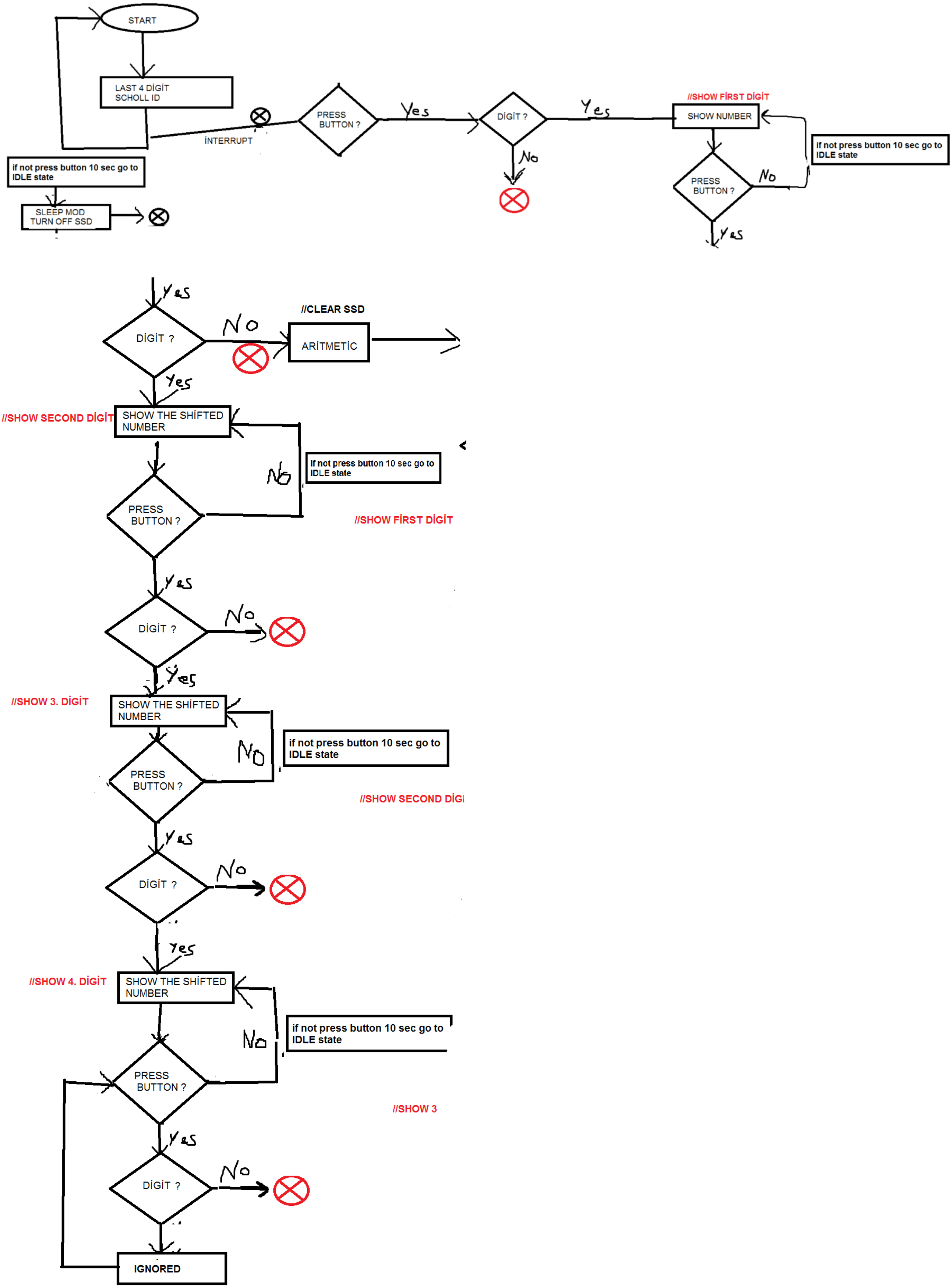
INTRODUCTION:

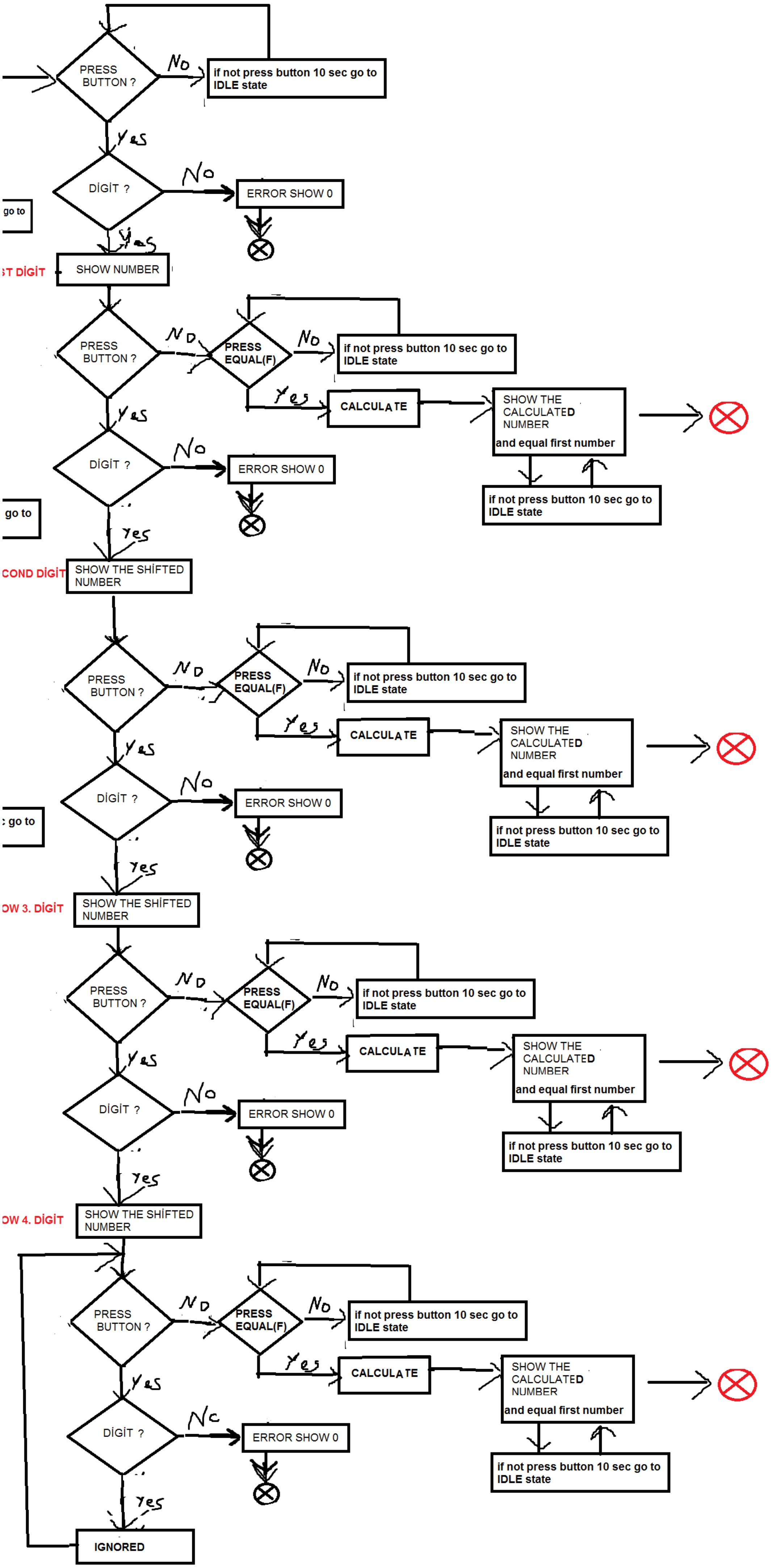
Our aim in this Project is to create basicly one calculator.

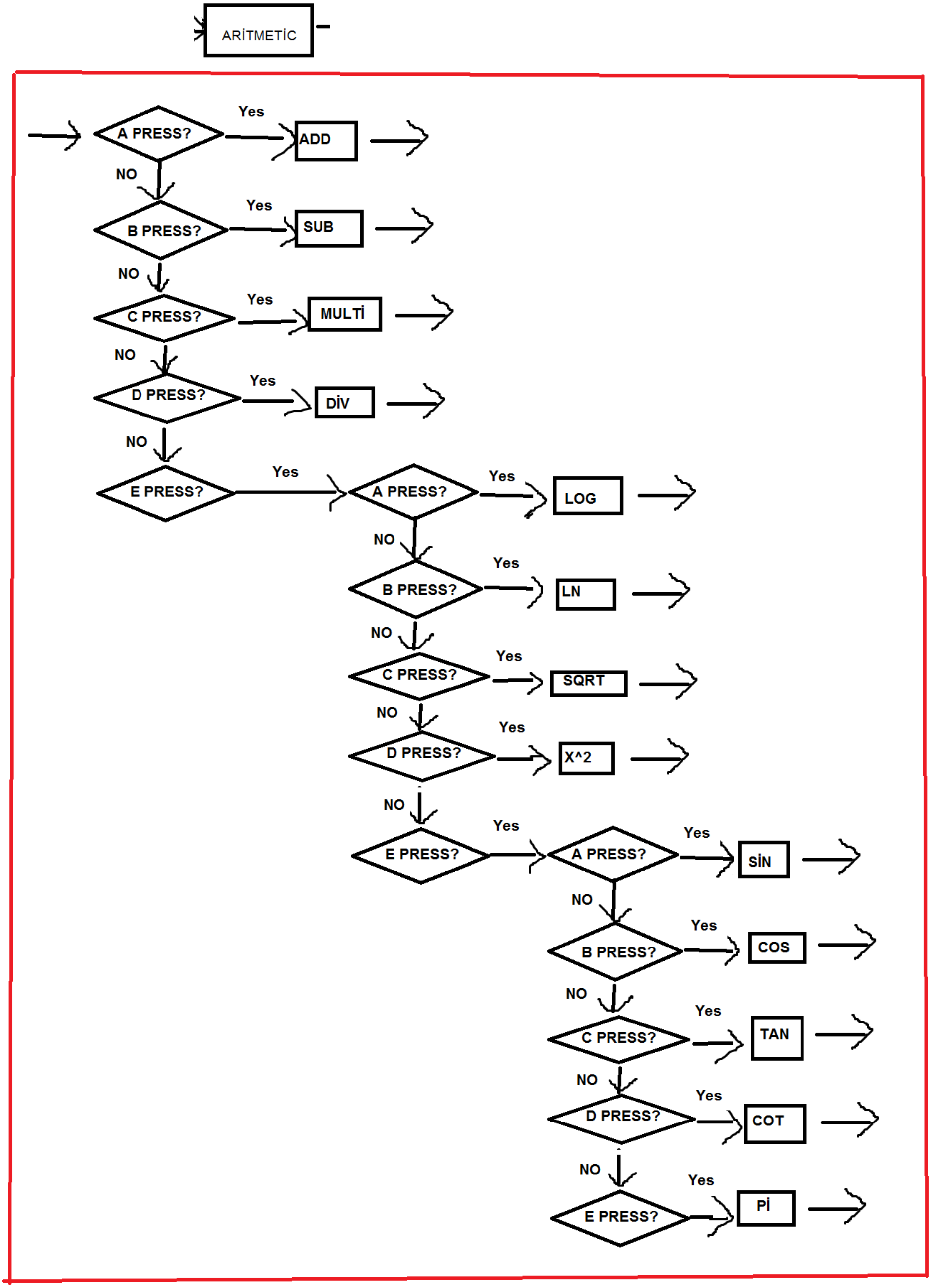
Detailed Requirements :

- Written in C
- A keypad and a seven segment display should be attached
- At beginig SSD should show ID
  -As soon as a number is pressed, everything should be cleared and only your number should be displayed.
  -If no buton is pressed for 10 seconds ,the SSD should turn off and go back to IDLE state.
- When keys are entered,the SSD should shift the numbers to left, while not displaying anything for empty digits.
- If the digits are already full ,new number key presses should be ignored.
- Then ABCDEF keys should be used as:
  -A is for addition
  -B is for subtraction
  -C is for multiplication
  -D is for division
  -F is for enter/equal
  -E key is scientific mode and will expect another keypress.
  *EA is for log
  *EB is for ln
  *EC is for sqrt
  *ED is for x^2
  -EE is for trigonometric mode and will expect another keypress.
  *EEA is for sin
  *EEB is for cos
  *EEC is for tan
  *EED is for cot
  *EEE is for pi
- Scientific and trigonometric modes will require floating point number system.
- Should Negative number should have a negative sign i.e -124 on the SSD.
- If the numbers overflows 9999 or -999 , it should display overflow.
- If the operation is invalid it should display invalid.
- If no keys are pressed for 10 seconds the SSD should turn off and go back IDLE state.
- If directly a function is invoked the current value should be used .For example if the last answer is 4 and – 4 is pressed it should do 4 – 4 operation and dispaly 0 .If in the beginning the number sould be assumed 0.

I first created a flowchar in this direction and I dive into small task in flowchart. Finally I combine the tasks.




PRESS BUTTON ?
No
if not press button 10 sec go to IDLE state
Yes
DİGİT ?
No
ERROR SHOW 0
go to
Yes
T DİGİT
SHOW NUMBER
PRESS BUTTON ?
No
PRESS EQUAL(F)
No
if not press button 10 sec go to IDLE state
Yes
CALCULATE
SHOW THE CALCULATED NUMBER and equal first number
if not press button 10 sec go to IDLE state
Yes
DİGİT ?
No
ERROR SHOW 0
go to
Yes
COND DİGİT
SHOW THE SHİFTED NUMBER
PRESS BUTTON ?
No
PRESS EQUAL(F)
No
if not press button 10 sec go to IDLE state
Yes
CALCULATE
SHOW THE CALCULATED NUMBER and equal first number
if not press button 10 sec go to IDLE state
Yes
DİGİT ?
No
ERROR SHOW 0
go to
Yes
OW 3. DİGİT
SHOW THE SHİFTED NUMBER
PRESS BUTTON ?
No
PRESS EQUAL(F)
No
if not press button 10 sec go to IDLE state
Yes
CALCULATE
SHOW THE CALCULATED NUMBER and equal first number
if not press button 10 sec go to IDLE state
Yes
DİGİT ?
No
ERROR SHOW 0
Yes
OW 4. DİGİT
SHOW THE SHİFTED NUMBER
PRESS BUTTON ?
No
PRESS EQUAL(F)
No
if not press button 10 sec go to IDLE state
Yes
CALCULATE
SHOW THE CALCULATED NUMBER and equal first number
if not press button 10 sec go to IDLE state
Yes
DİGİT ?
Nc
ERROR SHOW 0
Yes
IGNORED


ARİTMETİC
A PRESS?
Yes
ADD
NO
B PRESS?
Yes
SUB
NO
C PRESS?
Yes
MULTİ
NO
D PRESS?
Yes
DİV
NO
E PRESS?
Yes
A PRESS?
Yes
LOG
NO
B PRESS?
Yes
LN
NO
C PRESS?
Yes
SQRT
NO
D PRESS?
Yes
X^2
NO
E PRESS?
Yes
A PRESS?
Yes
SİN
NO
B PRESS?
Yes
COS
NO
C PRESS?
Yes
TAN
NO
D PRESS?
Yes
COT
NO
E PRESS?
Yes
Pİ

Flowchart

TASK 1: (+)

Connect one 4xSSD to the board and turn on one part of a segment and I knew how it all turned on and off .My SSD is common katot .I make figure 1.

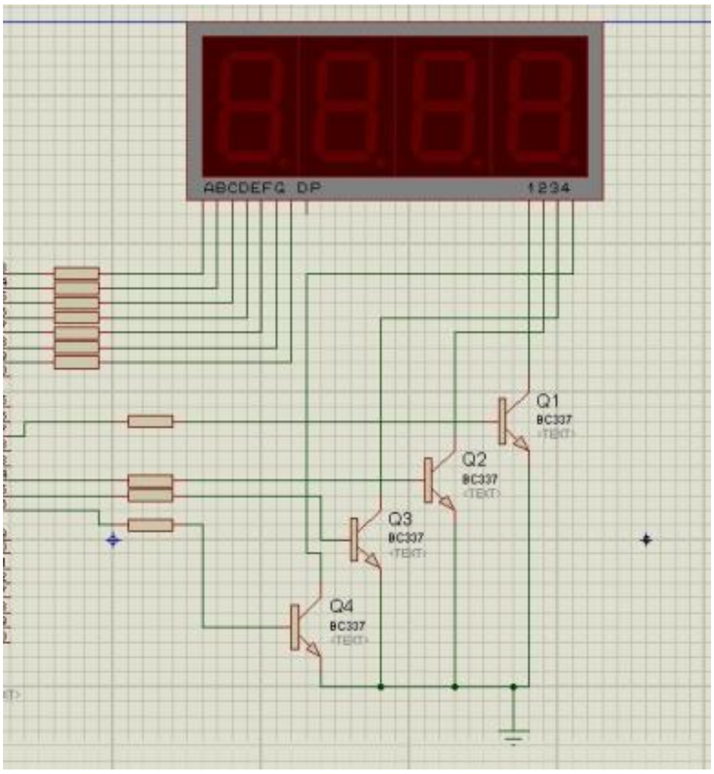


Figure 1.

TASK 2: (+)

I connect to Keypad the way I learned from the applications lesson and I know connect leds and button and I make figure 2.

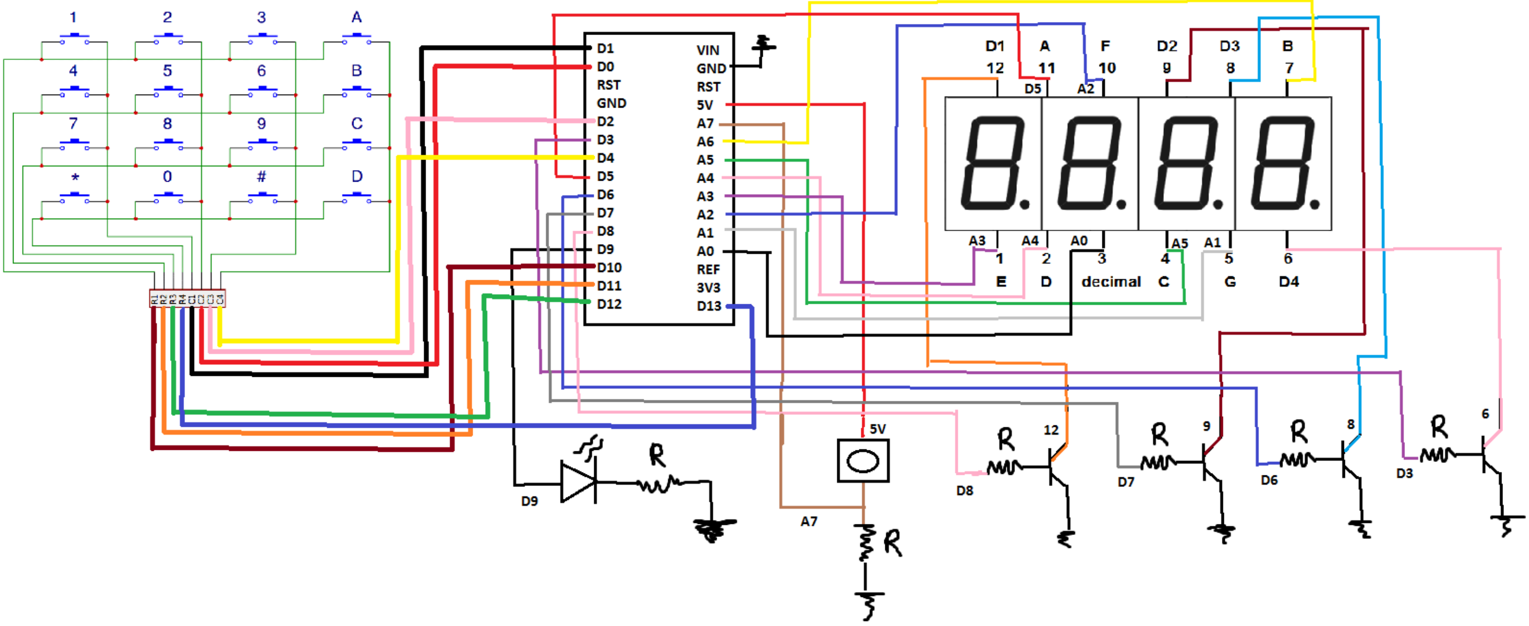


Figure 2.Connection Diagram

TASK 3: (+)

My flowchart is too long and I divide small piece. I knew how it's done to show our school number and i learned to interrupt to keypad and write it.

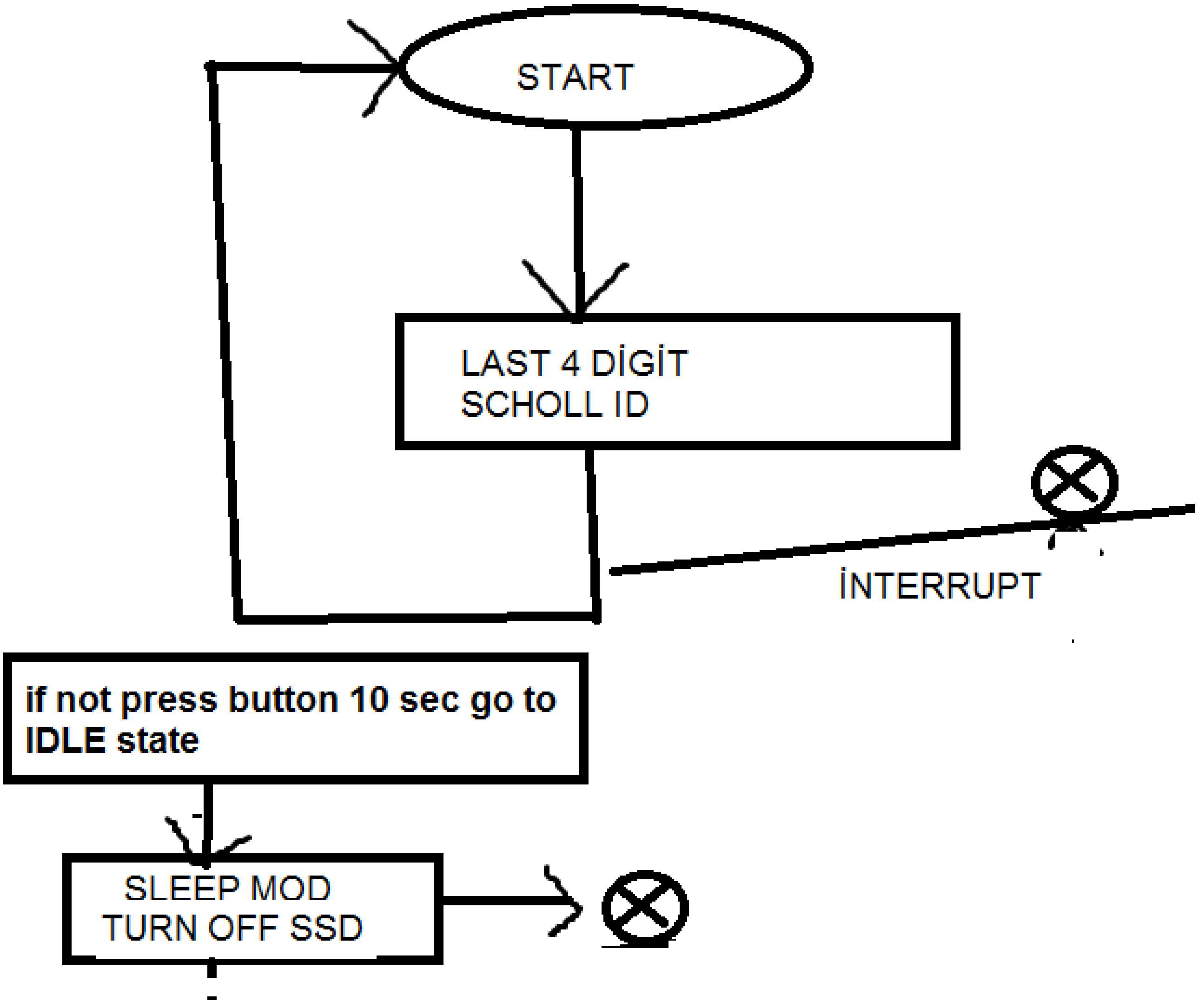


Figure 3.

TASK 4: (+)

After I make figure 4 .I created function when pressed buton wait 10 second and back to IDLE state and turn off SSD. I wrote this section on all of them after the button was pressed.

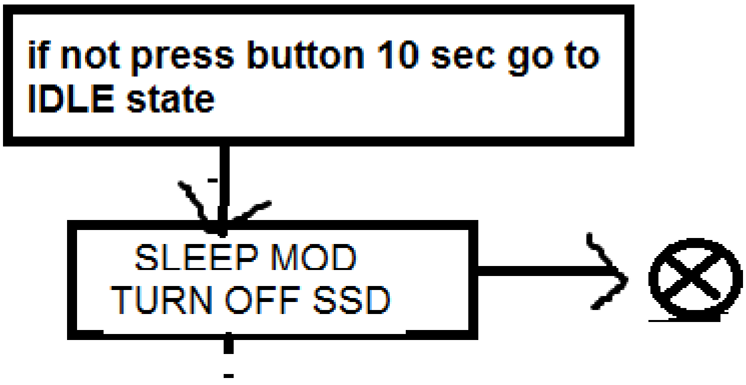


Figure 4.

TASK 5: (+)

And I work figure 5. I thought how we can print the numbers by swiping to the left. I thought that as the interrupt comes, variables are assigned to each other and I wrote the this section.

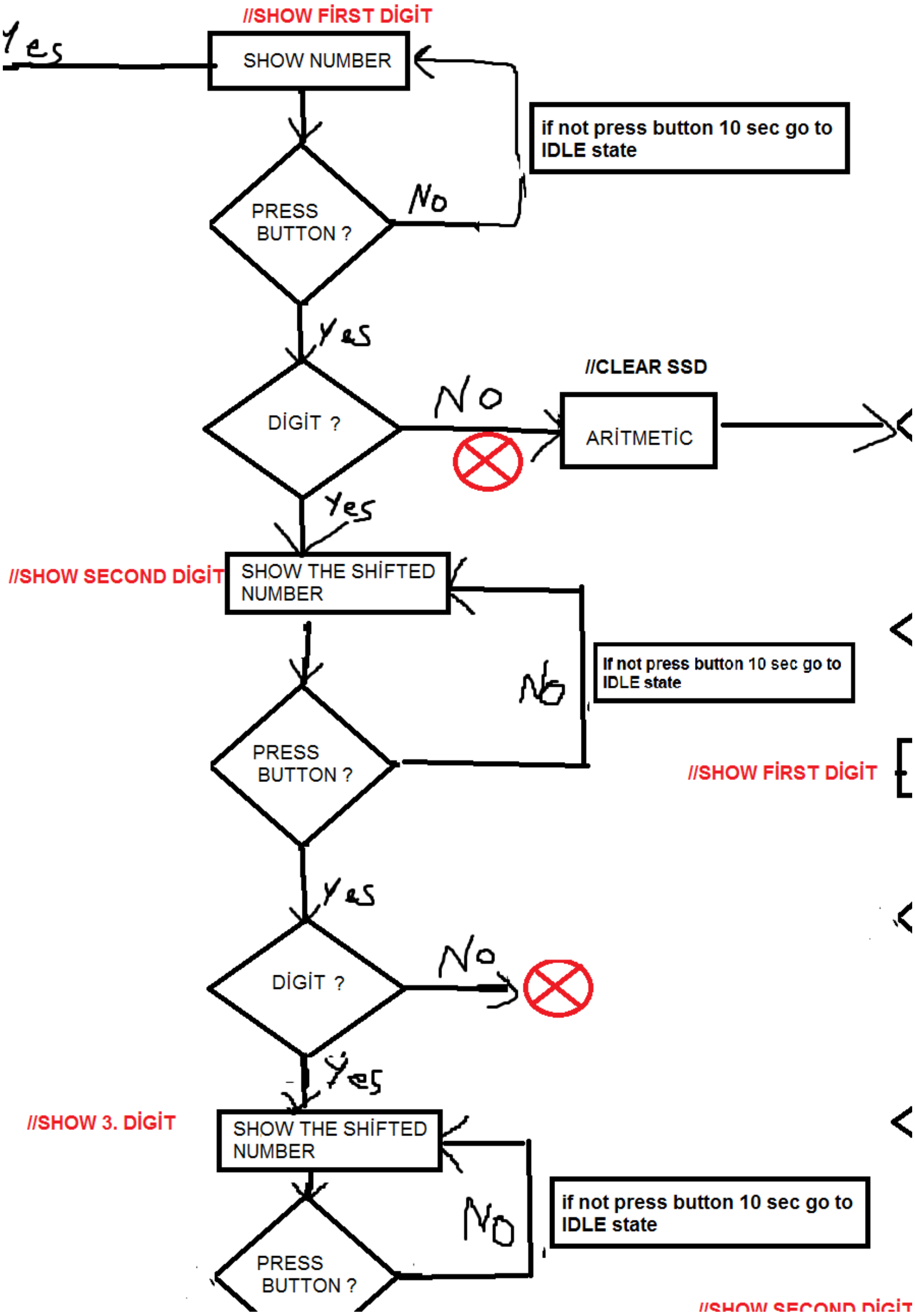


Figure 5.

TASK 6: (+)

And same way I work figure 6 and when the pressed aritmetic Keep in memory a global variable. then I will use this variable in arithmetic operations.

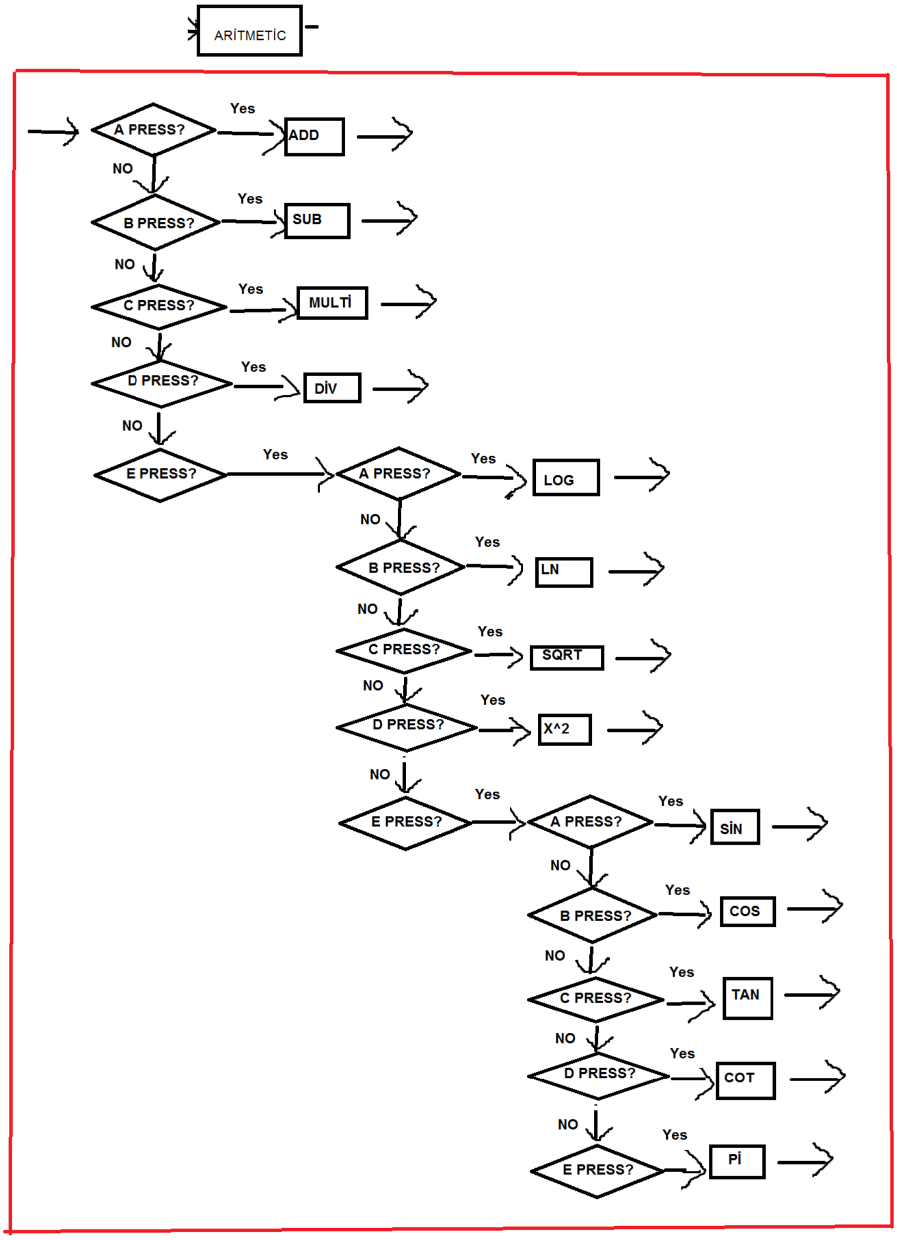


Figure 6.

TASK 7: (+)

And same way I work figure 7. I repeated the operations I did for the first number to the second number. I assign it to a different variable so that the variables don't mix.

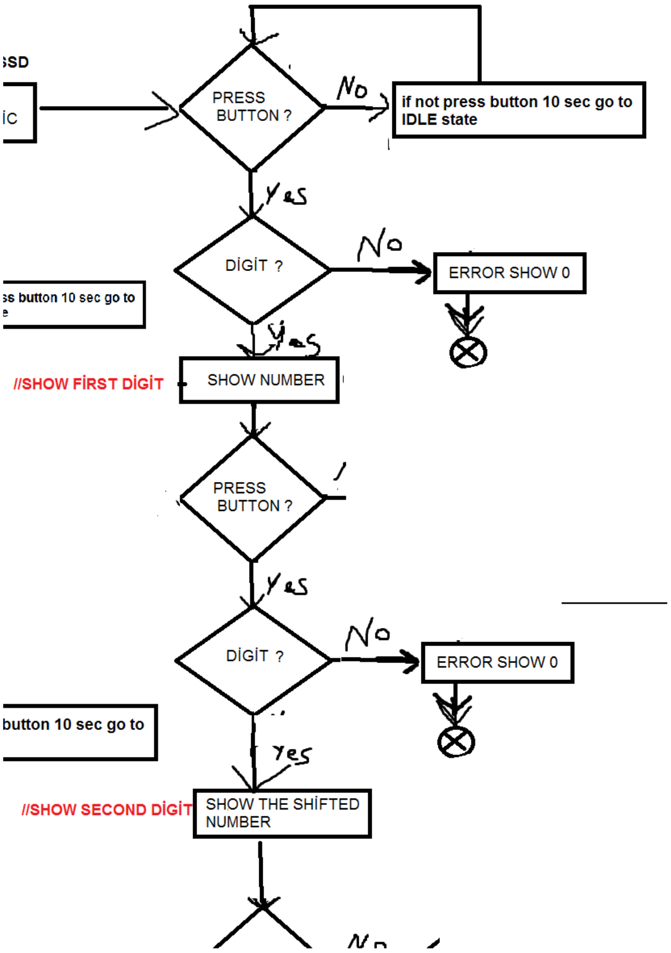


Figure 7.

TASK 8: (+)

And same way I work figure 8. I designed a function to do the calculations after the equals interrupt. This function detects the variables coming from the first and second numbers and then works with the number coming from the arithmetic.

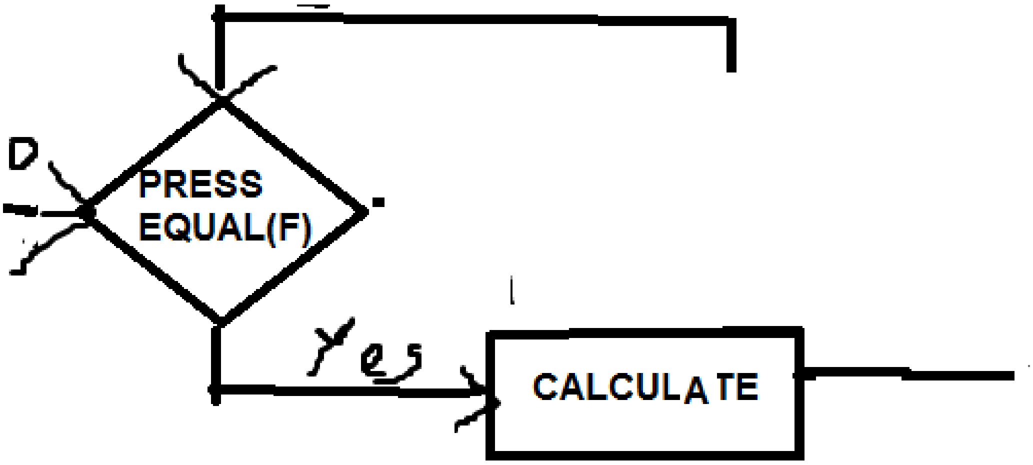


Figure 8.

TASK 9: (+)

And same way I work figure 9. After the I assigned this number to the first number so that I could use the calculated number again.

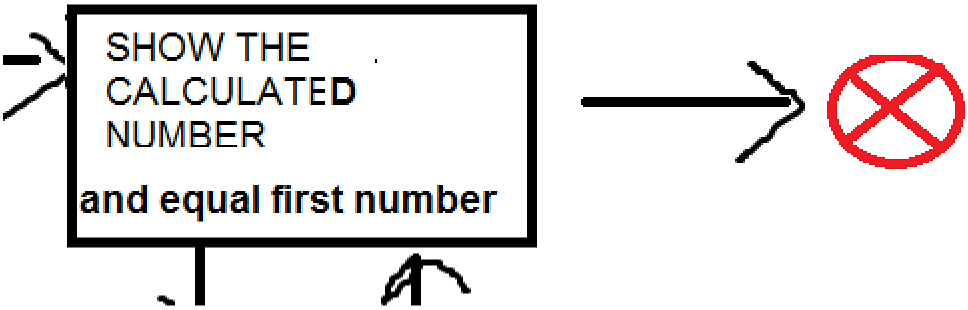


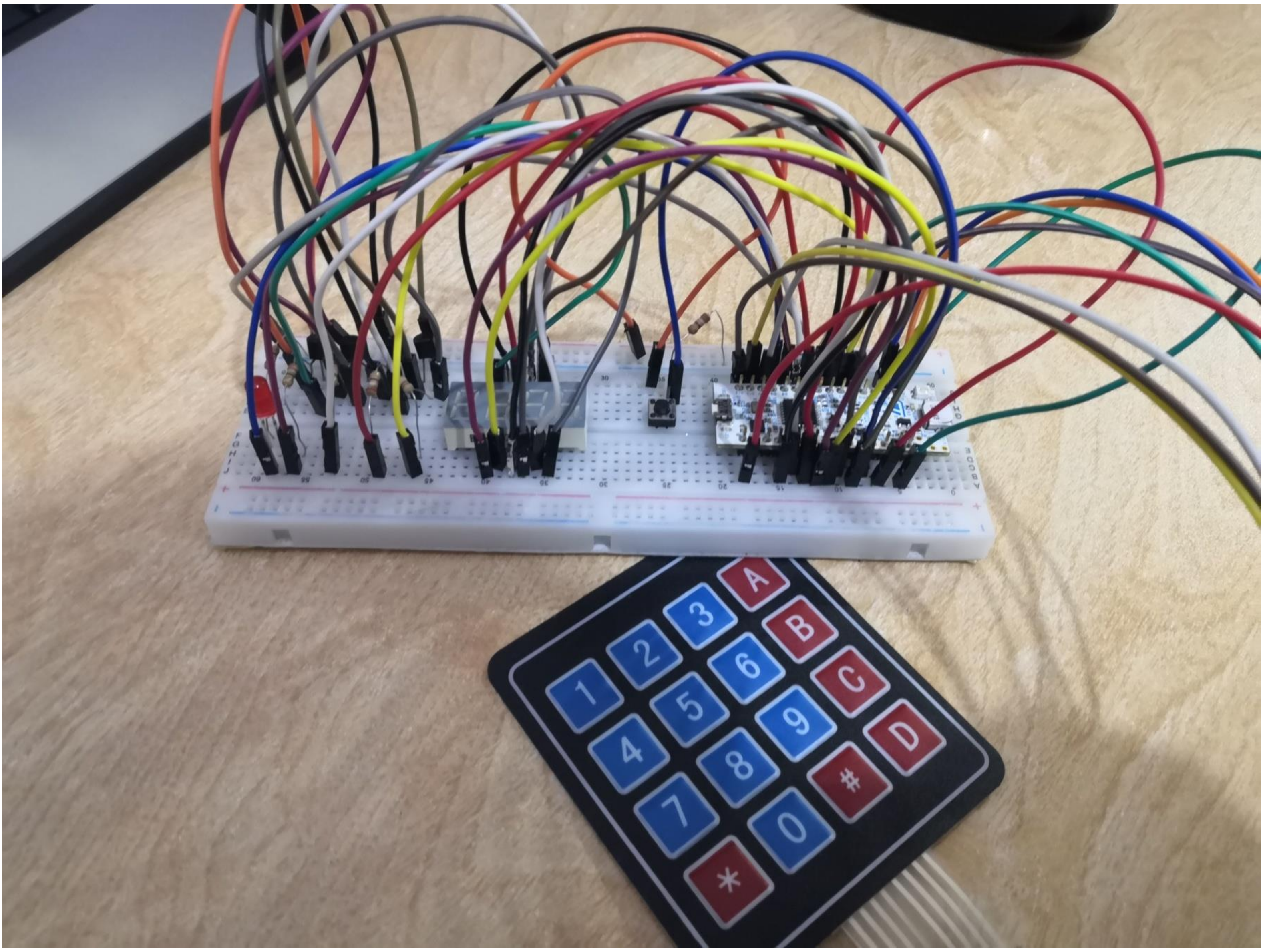
Figure 9.

TASK 10: (-)

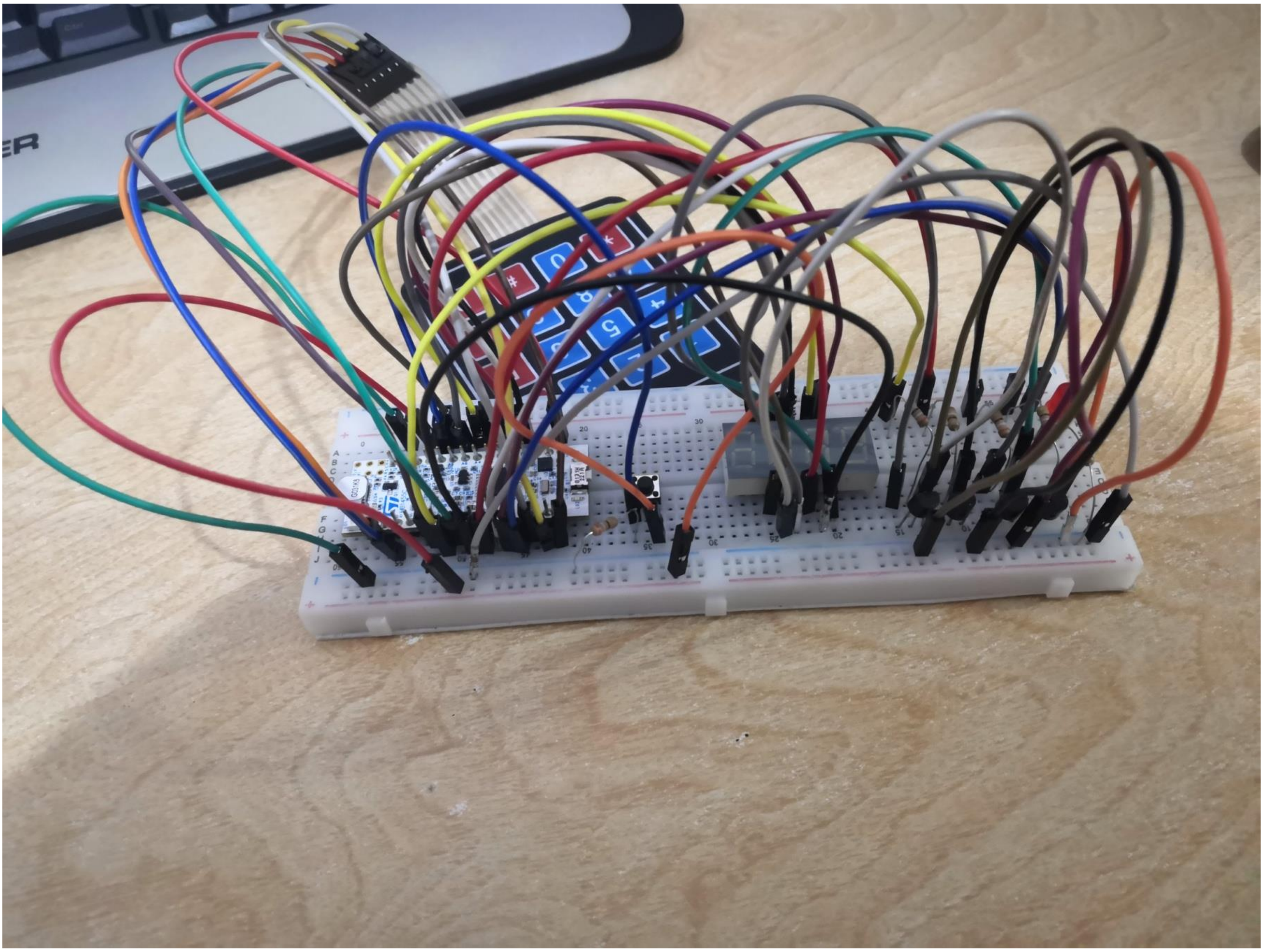
I've thought of something for minus numbers, but what I was thinking didn't make much sense. In the subtraction process, I took the absolute value and designed a system according to which number is larger. this logic crashed while doing subsequent operations

TASK 11: (-)

I had trouble showing numbers with commas because the float process got me confused. I set up a build for testing but it didn't work



Front



Back

PART LİST:

NUCLEO-G031K8	X1	110TL
JUMPER CABLE	X25	5TL
RESİSTANCE 470Ω	X6	1TL
4XSEVEN SEGMENT	X1	7TL
4x4 KEYPAD	X1	10TL
TRANSİSTOR	X4	1TL
BUTTON	X1	0.25TL
LED	X1	0.25TL
SUM		134.5TL

CONCLUSİON:

As a result, I leard to how keypad is connected the board and how is interrupt . I made a simple calculator.

This project is open to improve because other operations can be made. It seems unlikely to do it, especially when a difficult operation is entered For example (log10+log10=) We can use other functions to do these operations.

The biggest challenge I encountered was the numbers with commas. I couldn't find how to separate them and how do I keep the negative number in memory.

VİDEO LİNK:

Code explanation:

https://youtu.be/3SOBrzTywiQ

Some examples:

1)https://youtu.be/SphHBhATDjs

2)https://youtu.be/HReYlQHZODU

REFERANCES:

The_Definitive_Guide_to_ARM_CortexM0_M0+ *Second Edition* Joseph Yiu

RM0444 Reference manual

https://elektrokod.wordpress.com/2013/12/09/7-segment-display-sayici-uygulamasi/

https://components101.com/misc/4x4-keypad-module-pinout-configuration-features-datasheet

CODE:

```
////////////////////////////////main.c////////////////
/*
 * proje2.c
 * main.c
 * author: Berkay Türk 171024024
 *
 * description: calculator
 *   G031K8 Nucleo board.
 */


#include "stm32g0xx.h"

#include "bsp.h"


#define LEDDELAY    1600000U




int main(void) {
      BSP_led_init();
      clearSSD();

      Keypad_enable();

       return 0;
}

////////////////////////////////bsp.h////////////////
#ifndef BSP_H_
#define BSP_H_

#include "stm32g0xx.h"


/* Common API functions for nucleo */

void delay_ms(uint32_t);
void delay(volatile unsigned int);
void BSP_system_init();

void clearSSD(void);
void SwitchSSD(int );
void setSSD(int , int );
void Keypad_enable();
void EXTI4_15_IRQHandler(void );
void clearRowsKeypad(void);
void setRowsKeypad(void);
void SysTick_Handler(void);
// LED related functions
void BSP_led_init();
void BSP_led_set();
void BSP_led_clear();
void BSP_led_toggle();



// Button related functions
void BSP_button_init();
int BSP_button_read();

#endif
```

```
////////////////////////////////bsp.c///////////////
#include "bsp.h"
#include "time.h"
#include "math.h"
#include "stm32g0xx.h"


static volatile  uint32_t tick = 0;
int a=0;//START VALUE 1. number LAST DİGİT (SSD 1. left digit)
int b=0;//START VALUE 1. number (SSD 2. digit)
int c=0;//START VALUE 1. number (SSD 3. digit)
int d=0;//START VALUE 1. number FİRST DİGİT (SSD 4. digit)
int t=0;
int k=0;
int l=0;
int m=0;
int n=0;
int a1=0;//START VALUE 2. number LAST DİGİT (SSD left digit)
int b1=0;//START VALUE 2. number (SSD 2. digit)
int c1=0;//START VALUE 2. number (SSD 2. digit)
int d1=0;//START VALUE 2. number FİRST DİGİT (SSD 4. digit)

void BSP_led_init(void) {

       /* Enable GPIOA clock */  /* Enable GPIOB clock */
           RCC->IOPENR |= (3U << 0);

       /* setup PA(0,1,4,5,6,8,9,11,12) for seven segment A,B,C,D,E,F,G,DH for bits in MODER */
           GPIOA->MODER &= ~(0x3CF3F0F);
           GPIOA->MODER |= (0x1451505);
       /* setup PB(0,1,2,8) for seven segment D4,D3,D2,D1 for in MODER */
           GPIOB->MODER &= ~(0x3003F);
           GPIOB->MODER |= (0x10015);

}

// initialize on board connected to A6
void BSP_button_init() {
       RCC->IOPENR |= (3U << 0);
       GPIOA->MODER &= ~(3U << 2*6);
}

int BSP_button_read(){

      int b = ((GPIOA->IDR >> 6) & 0x0001);

      if (b) return 0;
      else return 1;
}

void delay(volatile unsigned int s) {
    for(; s>0; s--);
}

void delay_ms(uint32_t s) {
    tick = 0;
    while(tick);
}
void BSP_led_set() {

      GPIOA->ODR |= (1U << 7);
}

void BSP_led_clear() {

      GPIOA->BRR |= (1U << 7);
}
```

```
void BSP_button_board() {

      GPIOA->BRR |= (1U << 6);
}


void BSP_led_toggle() {

      GPIOA->ODR ^= (1U << 7);
}

void SysTick_Handler(void) {

      if(tick > 0){

             --tick;
      }

}



void EXTI4_15_IRQHandler(void) {   //INTERRUPT function
      clearSSD();

      if((EXTI->RPR1 >>6) & 1  ){/* Interrupt from PB6 */

      clearRowsKeypad();

      GPIOB->ODR ^= (1U << 9);  // PB9
      if((GPIOB->IDR >> 6) & 1 ){//'1'
             if(t==0){//First interrupt
             d=1;
             k=1;
            }
             if(t==1){//second interrupt
             c=d;//transfer values  from other values
             d=1;
             k=2;
                 }
             if(t==2){//third interrupt
             b=c;//transfer values  from other values
             c=d;
             d=1;
             k=3;
                 }
             if(t==3){//fourth interrupt
             a=b;//transfer values  from other values
             b=c;
             c=d;
             d=1;
             k=4;
                 }
             if(t==4){// 2. number 1. interrupt
             d1=1;
             k=6;
            }
             if(t==5){// 2. number 2. interrupt
             c1=d1;//transfer values  from other values
             d1=1;
             k=7;
                 }
             if(t==6){// 2. number 3. interrupt
             b1=c1;//transfer values  from other values
             c1=d1;
             d1=1;
             k=8;
                 }
```

```
               if(t==7){// 2. number 4. interrupt
               a1=b1;//transfer values  from other values
               b1=c1;
               c1=d1;
               d1=1;
               k=9;
                   }
               }

        GPIOB->ODR ^= (1U << 9);

        GPIOB->ODR ^= (1U << 5);  // PB5
        if((GPIOB->IDR >> 6) & 1 ){//'4'
               if(t==0){//First interrupt
               d=4;
               k=1;
              }
               if(t==1){//second interrupt
               c=d;//transfer values  from other values
               d=4;
               k=2;
                   }
               if(t==2){//third interrupt
               b=c;//transfer values  from other values
               c=d;
               d=4;
               k=3;
                   }
               if(t==3){//fourth interrupt
               a=b;//transfer values  from other values
               b=c;
               c=d;
               d=4;
               k=4;
                   }
               if(t==4){// 2. number 1. interrupt
               d1=4;
               k=6;
              }
               if(t==5){// 2. number 2. interrupt
               c1=d1;//transfer values  from other values
               d1=4;
               k=7;
                   }
               if(t==6){// 2. number 3. interrupt
               b1=c1;//transfer values  from other values
               c1=d1;
               d1=4;
               k=8;
                   }
               if(t==7){// 2. number 4. interrupt
               a1=b1;//transfer values  from other values
               b1=c1;
               c1=d1;
               d1=4;
               k=9;
                   }
        }

         GPIOB->ODR ^= (1U << 5);
         GPIOB->ODR ^= (1U << 4);  // PB4
         if((GPIOB->IDR >> 6) & 1 ){//'7'
                     if(t==0){//First interrupt
                     d=7;
                     k=1;
                    }
                     if(t==1){//second interrupt
                     c=d;//transfer values  from other values
                     d=7;
                     k=2;
                         }
                   if(t==2){//third interrupt
                   b=c;//transfer values  from other values
                   c=d;
                   d=7;
                   k=3;
                       }
                   if(t==3){//fourth interrupt
                   a=b;//transfer values  from other values
                   b=c;
                   c=d;
                   d=7;
                   k=4;
                       }
                   if(t==4){// 2. number 1. interrupt
                   d1=7;
                   k=6;
                   }
                   if(t==5){// 2. number 2. interrupt
                   c1=d1;//transfer values  from other values
                   d1=7;
                   k=7;
                       }
                   if(t==6){// 2. number 3. interrupt
                   b1=c1;//transfer values  from other values
                   c1=d1;
                   d1=7;
                   k=8;
                       }
                   if(t==7){// 2. number 4. interrupt
                   a1=b1;//transfer values  from other values
                   b1=c1;
                   c1=d1;
                   d1=7;
                   k=9;
                       }
      }

       GPIOB->ODR ^= (1U << 4);

       GPIOB->ODR ^= (1U << 3);  // PB3
       if((GPIOB->IDR >> 6) & 1 ){//* 'E'
        if(l==0){
                   m=5;
                   k=5;
         }
             if(l==1){
            m=6;
            k=5;
             }
             if(l==2){
                          k=5;
                          n=9;//pi values
                          m=5;
             }
      }
       GPIOB->ODR ^= (1U << 3);

        EXTI->RPR1 |= (1U << 6);//Clear interrupt flag
        setRowsKeypad();

}
      if((EXTI->RPR1 >>7) & 1 ){/* Interrupt from PB7 */

      clearRowsKeypad();

      GPIOB->ODR ^= (1U << 9);  // PB9
      if((GPIOB->IDR >> 7) & 1 ){//'2'
            if(t==0){//First interrupt
            d=2;
            k=1;
            }
        if(t==1){//second interrupt
        c=d;//transfer values  from other values
        d=2;
        k=2;
            }
        if(t==2){//third interrupt
        b=c;//transfer values  from other values
        c=d;
        d=2;
        k=3;
            }
        if(t==3){//fourth interrupt
        a=b;//transfer values  from other values
        b=c;
        c=d;
        d=2;
        k=4;
            }
        if(t==4){// 2. number 1. interrupt
        d1=2;
        k=6;
        }
        if(t==5){// 2. number 2. interrupt
        c1=d1;//transfer values  from other values
        d1=2;
        k=7;
            }
        if(t==6){// 2. number 3. interrupt
        b1=c1;//transfer values  from other values
        c1=d1;
        d1=2;
        k=8;
            }
        if(t==7){// 2. number 4. interrupt
        a1=b1;//transfer values  from other values
        b1=c1;
        c1=d1;
        d1=2;
        k=9;
            }
    }

    GPIOB->ODR ^= (1U << 9);
    GPIOB->ODR ^= (1U << 5);  // PB5
    if((GPIOB->IDR >> 7) & 1 ){//'5'
        if(t==0){//First interrupt
        d=5;
        k=1;
        }
        if(t==1){//second interrupt
        c=d;//transfer values  from other values
        d=5;
        k=2;
            }
        if(t==2){//third interrupt
        b=c;//transfer values  from other values
        c=d;
        d=5;
        k=3;
            }
        if(t==3){//fourth interrupt
        a=b;//transfer values  from other values
        b=c;
        c=d;
        d=5;
        k=4;
            }
        if(t==4){// 2. number 1. interrupt
        d1=5;
        k=6;
        }
```

```
              if(t==5){// 2. number 2. interrupt
              c1=d1;//transfer values  from other values
              d1=5;
              k=7;
                  }
              if(t==6){// 2. number 3. interrupt
              b1=c1;//transfer values  from other values
              c1=d1;
              d1=5;
              k=8;
                  }
              if(t==7){// 2. number 4. interrupt
              a1=b1;//transfer values  from other values
              b1=c1;
              c1=d1;
              d1=5;
              k=9;
                  }
        }

         GPIOB->ODR ^= (1U << 5);

         GPIOB->ODR ^= (1U << 4);  // PB4
         if((GPIOB->IDR >> 7) & 1 ){//'8'
                     if(t==0){//First interrupt
                     d=8;
                     k=1;
                    }
                     if(t==1){//second interrupt
                     c=d;//transfer values  from other values
                     d=8;
                     k=2;
                         }
                     if(t==2){//third interrupt
                     b=c;//transfer values  from other values
                     c=d;
                     d=8;
                     k=3;
                         }
                     if(t==3){//fourth interrupt
                     a=b;//transfer values  from other values
                     b=c;
                     c=d;
                     d=8;
                     k=4;
                         }
                     if(t==4){// 2. number 1. interrupt
                     d1=8;
                     k=6;
                    }
                     if(t==5){// 2. number 2. interrupt
                     c1=d1;//transfer values  from other values
                     d1=8;
                     k=7;
                         }
                     if(t==6){// 2. number 3. interrupt
                     b1=c1;//transfer values  from other values
                     c1=d1;
                     d1=8;
                     k=8;
                         }
                     if(t==7){// 2. number 4. interrupt
                     a1=b1;//transfer values  from other values
                     b1=c1;
                     c1=d1;
                     d1=8;
                     k=9;
                         }
        }
```

```
 GPIOB->ODR ^= (1U << 4);

 GPIOB->ODR ^= (1U << 3);  // PB3
 if((GPIOB->IDR >> 7) & 1 ){//'0'
             if(t==0){//First interrupt
             d=0;
             k=1;
            }
             if(t==1){//second interrupt
             c=d;//transfer values  from other values
             d=0;
             k=2;
                 }
             if(t==2){//third interrupt
             b=c;//transfer values  from other values
             c=d;
             d=0;
             k=3;
                 }
             if(t==3){//fourth interrupt
             a=b;//transfer values  from other values
             b=c;
             c=d;
             d=0;
             k=4;
                 }
             if(t==4){// 2. number 1. interrupt
             d1=0;
             k=6;
            }
             if(t==5){// 2. number 2. interrupt
             c1=d1;//transfer values  from other values
             d1=0;
             k=7;
                 }
             if(t==6){// 2. number 3. interrupt
             b1=c1;//transfer values  from other values
             c1=d1;
             d1=0;
             k=8;
                 }
             if(t==7){// 2. number 4. interrupt
             a1=b1;//transfer values  from other values
             b1=c1;
             c1=d1;
             d1=0;
             k=9;
                 }
}
 GPIOB->ODR ^= (1U << 3);

 EXTI->RPR1 |= (1U << 7); //Clear interrupt flag
        setRowsKeypad();
 }

if((EXTI->RPR1 >> 15) & 1 ){/* Interrupt from PA15 */

clearRowsKeypad();

GPIOB->ODR ^= (1U << 9);  // PB9
if((GPIOA->IDR >> 15) & 1 ){//'3'
      if(t==0){//First interrupt
      d=3;
      k=1;
      }
      if(t==1){//second interrupt
      c=d;//transfer values  from other values
      d=3;
      k=2;
          }
```

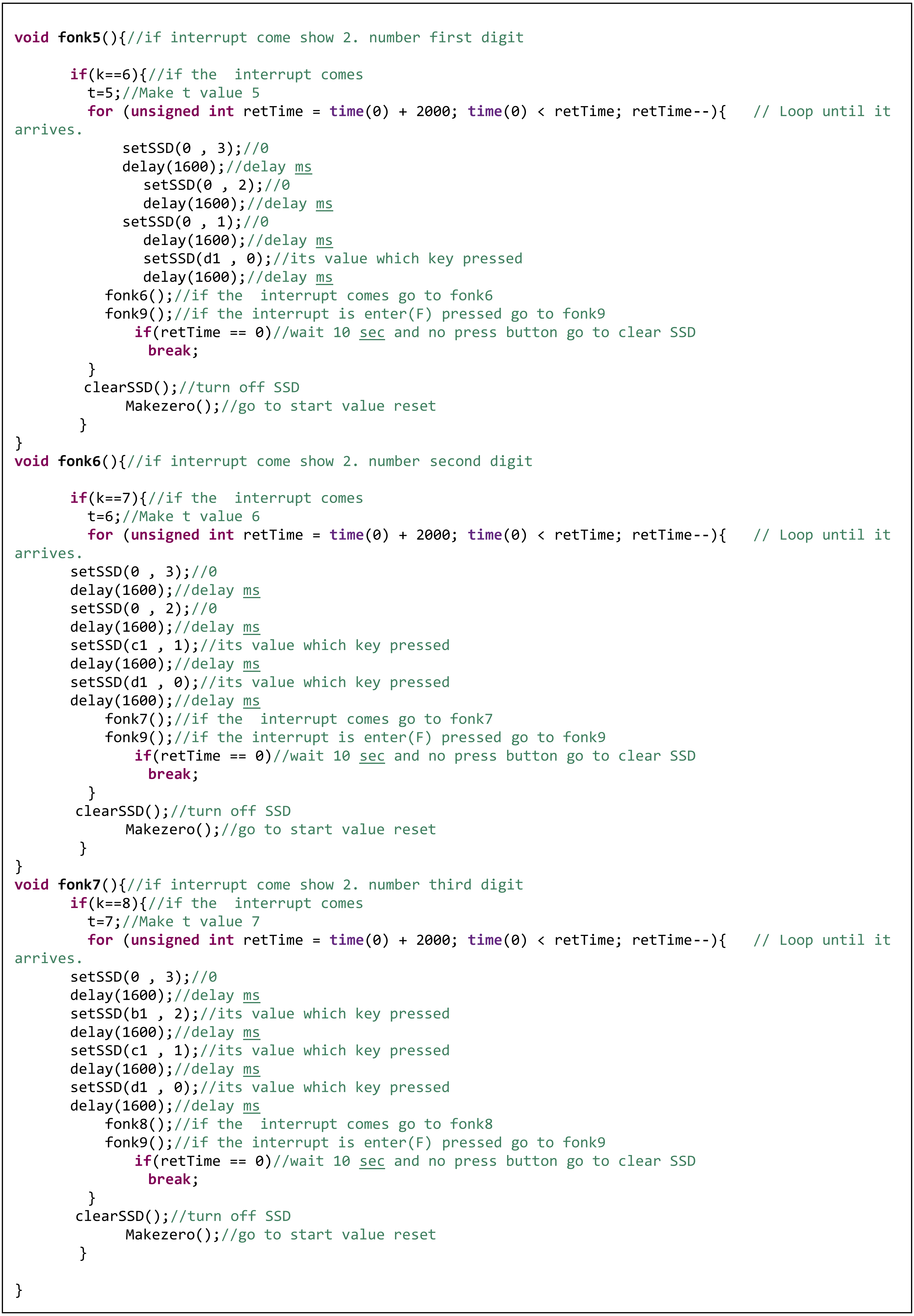
```
              if(t==2){//third interrupt
              b=c;//transfer values  from other values
              c=d;
              d=3;
              k=3;
                  }
              if(t==3){//fourth interrupt
              a=b;//transfer values  from other values
              b=c;
              c=d;
              d=3;
              k=4;
                  }
              if(t==4){// 2. number 1. interrupt
              d1=3;
              k=6;
              }
              if(t==5){// 2. number 2. interrupt
              c1=d1;//transfer values  from other values
              d1=3;
              k=7;
                  }
              if(t==6){// 2. number 3. interrupt
              b1=c1;//transfer values  from other values
              c1=d1;
              d1=3;
              k=8;
                  }
              if(t==7){// 2. number 4. interrupt
              a1=b1;//transfer values  from other values
              b1=c1;
              c1=d1;
              d1=3;
              k=9;
                  }
        }

        GPIOB->ODR ^= (1U << 9);
        GPIOB->ODR ^= (1U << 5);  // PB5
        if((GPIOA->IDR >> 15) & 1 ){//'6'
              if(t==0){//First interrupt
              d=6;
              k=1;
              }
              if(t==1){//second interrupt
              c=d;//transfer values  from other values
              d=6;
              k=2;
                  }
              if(t==2){//third interrupt
              b=c;//transfer values  from other values
              c=d;
              d=6;
              k=3;
                  }
              if(t==3){//fourth interrupt
              a=b;//transfer values  from other values
              b=c;
              c=d;
              d=6;
              k=4;
                  }
              if(t==4){// 2. number 1. interrupt
              d1=6;
              k=6;
              }
              if(t==5){// 2. number 2. interrupt
              c1=d1;//transfer values  from other values
              d1=6;
              k=7;
                  }
```

```
      if(t==6){// 2. number 3. interrupt
      b1=c1;//transfer values  from other values
      c1=d1;
      d1=6;
      k=8;
          }
      if(t==7){// 2. number 4. interrupt
      a1=b1;//transfer values  from other values
      b1=c1;
      c1=d1;
      d1=6;
      k=9;
          }
}

 GPIOB->ODR ^= (1U << 5);
 GPIOB->ODR ^= (1U << 4);  // PB4
 if((GPIOA->IDR >> 15) & 1 ){//'9'
             if(t==0){//First interrupt
             d=9;
             k=1;
             }
             if(t==1){//second interrupt
             c=d;//transfer values  from other values
             d=9;
             k=2;
                 }
             if(t==2){//third interrupt
             b=c;//transfer values  from other values
             c=d;
             d=9;
             k=3;
                 }
             if(t==3){//fourth interrupt
             a=b;//transfer values  from other values
             b=c;
             c=d;
             d=9;
             k=4;
                 }
             if(t==4){// 2. number 1. interrupt
             d1=9;
             k=6;
             }
             if(t==5){// 2. number 2. interrupt
             c1=d1;//transfer values  from other values
             d1=9;
             k=7;
                 }
             if(t==6){// 2. number 3. interrupt
             b1=c1;//transfer values  from other values
             c1=d1;
             d1=9;
             k=8;
                 }
             if(t==7){// 2. number 4. interrupt
             a1=b1;//transfer values  from other values
             b1=c1;
             c1=d1;
             d1=9;
             k=9;
                 }
}
 GPIOB->ODR ^= (1U << 4);
 GPIOB->ODR ^= (1U << 3);  // PB3
 if((GPIOA->IDR >> 15) & 1 ){//# '='

             k=10;//go to fonk9

}
```

```
         GPIOB->ODR ^= (1U << 3);

         EXTI->RPR1 |= (1U << 15);//Clear interrupt flag
                setRowsKeypad();
         }


        if((EXTI->RPR1 >> 10) & 1 ){/* Interrupt from PA10 */

        clearRowsKeypad();

        GPIOB->ODR ^= (1U << 9);  // PB9
        if((GPIOA->IDR >> 10) & 1 ){//A
              if(m==0){//interrupt aritmetick
              k=5;
              m=1;//go to ADD
              }
              if(m==5){
              k=5;
              n=1;// go to log

              }
              if(m==6){
                     k=5;
                     n=5;//go to sin
                     m=5;
              }
        }

        GPIOB->ODR ^= (1U << 9);

        GPIOB->ODR ^= (1U << 5);  // PB5
        if((GPIOA->IDR >> 10) & 1 ){//B
              if(m==0){//interrupt aritmetick
              k=5;
              m=2;//go to SUB
              }
              if(m==5){
              k=5;
              n=2;//go to ln
              }
              if(m==6){
                     k=5;
                     n=6;//go to cos
                     m=5;
              }
        }

         GPIOB->ODR ^= (1U << 5);

         GPIOB->ODR ^= (1U << 4);  // PB4
         if((GPIOA->IDR >> 10) & 1 ){//C
               if(m==0){//interrupt aritmetick
              k=5;
            m=3;//go to MULTI
               }
              if(m==5){
              k=5;
              n=3;//go to sqrt

              }
              if(m==6){
                     k=5;
                     n=7;//go to tan
                     m=5;
              }

        }

         GPIOB->ODR ^= (1U << 4);
```

```
       GPIOB->ODR ^= (1U << 3);  // PB3
       if((GPIOA->IDR >> 10) & 1 ){//D
             if(m==0){//interrupt aritmetick
             k=5;
             m=4;//go to DIV
             }
            if(m==5){
            k=5;
            n=4;//go to x^2

            }
            if(m==6){
                  k=5;
                  n=8;//go to cot
                  m=5;
            }
      }
       GPIOB->ODR ^= (1U << 3);

       EXTI->RPR1 |= (1U << 10);//Clear interrupt flag
              setRowsKeypad();
       }

 delay(800000);//wait 1 sec because interrups go same


  }


void showNumber() {

  for (unsigned int retTime = time(0) + 2000; time(0) < retTime; retTime--){   // Loop until it
arrives.

      showID();    //My school ID show and loop
      arithmetic();// if press the aritmetic (interupt) go to aritmetic
      if(k==1){    // if press the number go to fonk1
            fonk1();
      }
       if(retTime == 0)//wait 10 sec and no press button go to clear SSD
         break;

  }
       clearSSD();//off SSD
       while(1){
       fonk1();//wait here until the press button
       }
}



void Keypad_enable(){

/*   Setup Output pins (rows) */

        GPIOB->MODER &= ~(3U << 2*9);  /// PB9 is output
        GPIOB->MODER |= (1U << 2*9);


        GPIOB->MODER &= ~(3U << 2*5);  /// PB5 is output
        GPIOB->MODER |= (1U << 2*5);


        GPIOB->MODER &= ~(3U << 2*4);  /// PB4 is output
        GPIOB->MODER |= (1U << 2*4);


        GPIOB->MODER &= ~(3U << 2*3);  /// PB3 is output
        GPIOB->MODER |= (1U << 2*3);
        /*   Setup Input pins (Columns)   */

        GPIOB->MODER &= ~(3U << 2*6);  /// PB6 is input
        GPIOB->PUPDR |= (2U << 2*6);   /// Pull-Down mode


        GPIOB->MODER &= ~(3U << 2*7);  /// PB7 is input
        GPIOB->PUPDR |= (2U << 2*7);    /// Pull-Down mode


        GPIOA->MODER &= ~(3U << 2*15);  /// PA15 is input
        GPIOA->PUPDR |= (2U << 2*15);    /// Pull-Down mode


        GPIOA->MODER &= ~(3U << 2*10);  /// PA10 is input
        GPIOA->PUPDR |= (2U << 2*10);    /// Pull-Down mode


           /* Setup interrupts for inputs */
        EXTI->EXTICR[1] |= (1U << 8*2);   // PB6
        EXTI->EXTICR[1] |= (1U << 8*3);   // PB7
        EXTI->EXTICR[3] |= (0U << 8*3);   // PA15
        EXTI->EXTICR[2] |= (0U << 8*2);   // PA10


        /* RISING Edge*/
        EXTI->RTSR1 |= (1U << 6);      // 6th pin
        EXTI->RTSR1 |= (1U << 7);      // 7th pin
        EXTI->RTSR1 |= (1U << 15);      // 15th pin
        EXTI->RTSR1 |= (1U << 10);      // 10th pin

        /* MASK*/
        EXTI->IMR1 |= (1U << 6);
        EXTI->IMR1 |= (1U << 7);
        EXTI->IMR1 |= (1U << 15);
        EXTI->IMR1 |= (1U << 10);

        /*NVIC */

        NVIC_SetPriority(EXTI4_15_IRQn , 0);
        NVIC_EnableIRQ(EXTI4_15_IRQn);


           /* Setup all rows*/
       GPIOB->ODR |= (1U << 9);   /// PB9
       GPIOB->ODR |= (1U << 5);   /// PB5
       GPIOB->ODR |= (1U << 4);   /// PB4
       GPIOB->ODR |= (1U << 3);   /// PB3

       clearSSD();//turn off SSD
       while(1){
            if(t==0){ // start value t=0 must be in
           showNumber();    //  show School number  wait here
             }

    }
}

void showID(){ //My school ID show
    setSSD(1 , 3);//1
    delay(1600);//delay ms
    setSSD(7 , 2);//7
    delay(1600);//delay ms
    setSSD(2 , 1);//2
    delay(1600);//delay ms
    setSSD(4 , 0);//4
    delay(1600);//delay ms
}
```

```
void fonk1(){//Shows first interrupt the first digit

    if(k==1){//if first interrupt comes
      t=1; //Make t value 1

  for (unsigned int retTime = time(0) + 2000; time(0) < retTime; retTime--){   // Loop until it 
arrives.
      setSSD(0 , 3);//0
      delay(1600);//delay ms
        setSSD(0 , 2);//0
        delay(1600);//delay ms
      setSSD(0 , 1);//0
        delay(1600);//delay ms
        setSSD(d , 0);//its value which key pressed
        delay(1600);//delay ms
             fonk2();//if the second interrupt comes go to fonk2
             arithmetic();//if the second interrupt comes to arithmetic
              if(retTime == 0)//wait 10 sec and no press button go to clear SSD
               break;
        }
        clearSSD();//turn off SSD
             Makezero();//go to start value reset
    }
    arithmetic();//if first interrupt comes to arithmetic

}
void fonk2(){//second interrupt Shows the second digit

      if(k==2){//if the second interrupt comes
        t=2;   //Make t value 2
        for (unsigned int retTime = time(0) + 2000; time(0) < retTime; retTime--){   // Loop until it 
arrives.
      setSSD(0 , 3);//0
      delay(1600);//delay ms
      setSSD(0 , 2);//0
      delay(1600);//delay ms
      setSSD(c , 1);//its value which key pressed
      delay(1600);//delay ms
      setSSD(d , 0);//its value which key pressed
      delay(1600);//delay ms
          fonk3();//if the third interrupt comes go to fonk3
         arithmetic();//if the third interrupt comes go to arithmetic
              if(retTime == 0)//wait 10 sec and no press button go to clear SSD
               break;
        }
        clearSSD();//turn off SSD
             Makezero();//go to start value reset
       }

}
void fonk3(){//third interrupt Shows the third digit

      if(k==3){//if the third interrupt comes
        t=3;    //Make t value 3

        for (unsigned int retTime = time(0) + 2000; time(0) < retTime; retTime--){   // Loop until it 
arrives.
      setSSD(0 , 3);//0
      delay(1600);//delay ms
      setSSD(b , 2);//its value which key pressed
      delay(1600);//delay ms
      setSSD(c , 1);//its value which key pressed
      delay(1600);//delay ms
      setSSD(d , 0);//its value which key pressed
      delay(1600);//delay ms
          fonk4();//if the fourth interrupt comes go to fonk4
         arithmetic();//if the fourth interrupt comes go to arithmetic
              if(retTime == 0)//wait 10 sec and no press button go to clear SSD
               break;
        }
        clearSSD();//turn off SSD
             Makezero();//go to start value reset
       }
}

void fonk4(){//fourth interrupt Shows the fourth digit

      if(k==4){//if the fourth interrupt comes
      //  t=4;//not need to do it here

      for (unsigned int retTime = time(0) + 2000; time(0) < retTime; retTime--){   // Loop until it
arrives.
      setSSD(a , 3);//its value which key pressed
      delay(1600);//delay ms
      setSSD(b , 2);//its value which key pressed
      delay(1600);//delay ms
      setSSD(c , 1);//its value which key pressed
      delay(1600);//delay ms
      setSSD(d , 0);//its value which key pressed
      delay(1600);//delay ms
          arithmetic();//if the fifth interrupt comes go to arithmetic
              if(retTime == 0)//wait 10 sec and no press button go to clear SSD
               break;
        }
        clearSSD();//turn off SSD
             Makezero();//go to start value reset
       }
  }

void arithmetic(){//shows last number
      if(k==5){//if the  interrupt comes
         t=4;//Make t value 4
         if(m==5){//fifth interrupt if press button E
             l=1;  //make l value 1
         }

for (unsigned int retTime = time(0) + 2000; time(0) < retTime; retTime--){   // Loop until it arrives.
       clearSSD();//show last number
         if(m==6){//sixth interrupt if press button E
             l=2;  //make l value 2
         }
             setSSD(a , 3);//its value which key pressed
             delay(1600);//delay ms
             setSSD(b , 2);//its value which key pressed
             delay(1600);//delay ms
             setSSD(c , 1);//its value which key pressed
             delay(1600);//delay ms
             setSSD(d , 0);//its value which key pressed
             delay(1600);//delay ms
          fonk5();//if the  interrupt comes go to fonk5
          fonk9();//if the interrupt is enter(F) pressed go to fonk9
       if(retTime == 0)//wait 10 sec and no press button go to clear SSD
         break;
  }
    clearSSD();//turn off SSD
      Makezero();//go to start value reset

      }
}
```



```
void fonk5(){//if interrupt come show 2. number first digit

      if(k==6){//if the  interrupt comes
        t=5;//Make t value 5
        for (unsigned int retTime = time(0) + 2000; time(0) < retTime; retTime--){   // Loop until it
arrives.
            setSSD(0 , 3);//0
            delay(1600);//delay ms
               setSSD(0 , 2);//0
               delay(1600);//delay ms
            setSSD(0 , 1);//0
               delay(1600);//delay ms
               setSSD(d1 , 0);//its value which key pressed
               delay(1600);//delay ms
          fonk6();//if the  interrupt comes go to fonk6
          fonk9();//if the interrupt is enter(F) pressed go to fonk9
              if(retTime == 0)//wait 10 sec and no press button go to clear SSD
               break;
        }
        clearSSD();//turn off SSD
             Makezero();//go to start value reset
       }
}
void fonk6(){//if interrupt come show 2. number second digit

      if(k==7){//if the  interrupt comes
        t=6;//Make t value 6
        for (unsigned int retTime = time(0) + 2000; time(0) < retTime; retTime--){   // Loop until it
arrives.
      setSSD(0 , 3);//0
      delay(1600);//delay ms
      setSSD(0 , 2);//0
      delay(1600);//delay ms
      setSSD(c1 , 1);//its value which key pressed
      delay(1600);//delay ms
      setSSD(d1 , 0);//its value which key pressed
      delay(1600);//delay ms
          fonk7();//if the  interrupt comes go to fonk7
          fonk9();//if the interrupt is enter(F) pressed go to fonk9
              if(retTime == 0)//wait 10 sec and no press button go to clear SSD
               break;
        }
       clearSSD();//turn off SSD
             Makezero();//go to start value reset
       }
}
void fonk7(){//if interrupt come show 2. number third digit
      if(k==8){//if the  interrupt comes
        t=7;//Make t value 7
        for (unsigned int retTime = time(0) + 2000; time(0) < retTime; retTime--){   // Loop until it
arrives.
      setSSD(0 , 3);//0
      delay(1600);//delay ms
      setSSD(b1 , 2);//its value which key pressed
      delay(1600);//delay ms
      setSSD(c1 , 1);//its value which key pressed
      delay(1600);//delay ms
      setSSD(d1 , 0);//its value which key pressed
      delay(1600);//delay ms
          fonk8();//if the  interrupt comes go to fonk8
          fonk9();//if the interrupt is enter(F) pressed go to fonk9
              if(retTime == 0)//wait 10 sec and no press button go to clear SSD
               break;
        }
       clearSSD();//turn off SSD
             Makezero();//go to start value reset
       }

}
```

```
void fonk8(){//if interrupt come show 2. number fourth digit
      if(k==9){//if the  interrupt comes
        t=8;//Make t value 8
        for (unsigned int retTime = time(0) + 2000; time(0) < retTime; retTime--){   // Loop until it 
arrives.
      setSSD(a1 , 3);//its value which key pressed
      delay(1600);//delay ms
      setSSD(b1 , 2);//its value which key pressed
      delay(1600);//delay ms
      setSSD(c1 , 1);//its value which key pressed
      delay(1600);//delay ms
      setSSD(d1 , 0);//its value which key pressed
      delay(1600);//delay ms
          fonk9();//if the interrupt is enter(F) pressed go to fonk9
             if(retTime == 0)//wait 10 sec and no press button go to clear SSD
              break;
        }
       clearSSD();//turn off SSD
            Makezero();//go to start value reset
       }

}
void fonk9(){//if interrupt come press 'F' ENTER show CALCULTE NUMBER
      if(k==10){
            t=4;//for the second operation value t=4
            l=0;//for the second operation value l=0
            k=0;//for the second operation value k=0
      int   x=1000*a+100*b+10*c+1*d;//combine first number values
      int y=1000*a1+100*b1+10*c1+1*d1;//combine second number values
      int z=0;//first value z=0

            if(m==1){//ADD
                  m=0;//for the second operation value m=0
                  z=x+y;
            int   z4=z/1000;//separate number
            int   z3=((z-(z4*1000))/100);
            int   z2=((z-(z3*100+z4*1000))/10);
            int   z1=((z-(z2*10+z3*100+z4*1000))/1);
            a=z4;//assign the result to the first number
            b=z3;
            c=z2;
            d=z1;
            a1=0;//reset to second number
            b1=0;
            c1=0;
            d1=0;
                    for (unsigned int retTime = time(0) + 2000; time(0) < retTime; retTime--){   // 
Loop until it arrives.
                  setSSD(z4 , 3);//calculated number
                  delay(1600);//delay ms
                  setSSD(z3 , 2);//calculated number
                  delay(1600);//delay ms
                  setSSD(z2 , 1);//calculated number
                  delay(1600);//delay ms
                  setSSD(z1 , 0);//calculated number
                  delay(1600);//delay ms
                  arithmetic();//if the  interrupt comes go to arithmetic
                         if(retTime == 0)//wait 10 sec and no press button go to clear SSD
                          break;
                    }
                  clearSSD();//turn off SSD
                       Makezero();//go to start value reset
            }
            if(m==2){//SUB
                  m=0;//for the second operation value m=0
                  z=abs(x-y);//get absolute value
            int   z4=z/1000;//separate number
            int   z3=((z-(z4*1000))/100);
            int   z2=((z-(z3*100+z4*1000))/10);
            int   z1=((z-(z2*10+z3*100+z4*1000))/1);
            a=z4;//assign the result to the first number
            b=z3;
            c=z2;
            d=z1;
            a1=0;//reset to second number
            b1=0;
            c1=0;
            d1=0;

                      for (unsigned int retTime = time(0) + 2000; time(0) < retTime; retTime--){   //
Loop until it arrives.
                                 if(x>=y){//show positive number
                    setSSD(z4 , 3);//calculated number
                    delay(1600);//delay ms
                    setSSD(z3 , 2);//calculated number
                    delay(1600);//delay ms
                    setSSD(z2 , 1);//calculated number
                    delay(1600);//delay ms
                    setSSD(z1 , 0);//calculated number
                    delay(1600);//delay ms
                          }
                    if(x<y){//show negative number
                          setSSD(10 , 3);//'-'
                          delay(1600);//delay ms
                          setSSD(z3 , 2);//calculated number
                          delay(1600);//delay ms
                          setSSD(z2 , 1);//calculated number
                          delay(1600);//delay ms
                          setSSD(z1 , 0);//calculated number
                          delay(1600);//delay ms
            }
                    arithmetic();//if the  interrupt comes go to arithmetic
                           if(retTime == 0)//wait 10 sec and no press button go to clear SSD
                             break;
                      }
                     clearSSD();//turn off SSD
                          Makezero();//go to start value reset
            }
            if(m==3){//MULTİ
                    m=0;//for the second operation value m=0
                    z=x*y;
            int     z4=z/1000;//separate number
            int     z3=((z-(z4*1000))/100);
            int     z2=((z-(z3*100+z4*1000))/10);
            int     z1=((z-(z2*10+z3*100+z4*1000))/1);
            a=z4;//assign the result to the first number
            b=z3;
            c=z2;
            d=z1;
            a1=0;//reset to second number
            b1=0;
            c1=0;
            d1=0;
                      for (unsigned int retTime = time(0) + 2000; time(0) < retTime; retTime--){   //
Loop until it arrives.
                    setSSD(z4 , 3);//calculated number
                    delay(1600);//delay ms
                    setSSD(z3 , 2);//calculated number
                    delay(1600);//delay ms
                    setSSD(z2 , 1);//calculated number
                    delay(1600);//delay ms
                    setSSD(z1 , 0);//calculated number
                    delay(1600);//delay ms
                    arithmetic();//if the  interrupt comes go to arithmetic
                           if(retTime == 0)//wait 10 sec and no press button go to clear SSD
                             break;
                      }
                     clearSSD();//turn off SSD
                          Makezero();//go to start value reset
            }
```

```
             if(m==4){//DİV
                   m=0;//for the second operation value m=0
             /*float      z=x/y;//not working here
             float  z7=z/1000;
             float  z6=((z-(z7*1000))/100);
             float  z5=((z-(z6*100+z7*1000))/10);
             float  z4=((z-(z5*10+z6*100+z7*1000))/1);
             float  z3=((z-(z4*1+z5*10+z6*100+z7*1000))*10);
             float  z2=((z-((z3/10)+z4*1+z5*10+z6*100+z7*1000))*100);
             float  z1=((z-((z2/100)+(z3/10)+z4*1+z5*10+z6*100+z7*1000))*1000);*/
                   z=x/y;
             int   z4=z/1000;//separate number
             int   z3=((z-(z4*1000))/100);
             int   z2=((z-(z3*100+z4*1000))/10);
             int   z1=((z-(z2*10+z3*100+z4*1000))/1);
             a=z4;//assign the result to the first number
             b=z3;
             c=z2;
             d=z1;
             a1=0;//reset to second number
             b1=0;
             c1=0;
             d1=0;
                     for (unsigned int retTime = time(0) + 2000; time(0) < retTime; retTime--){   //
Loop until it arrives.
/*                 if(z7!=0){ //not working here
                   setSSD(z7 , 3);
                   delay(1600);//delay ms
                   setSSD(z6 , 2);
                   delay(1600);//delay ms
                   setSSD(z5 , 1);
                   delay(1600);//delay ms
                   setSSD(z4 , 0);
                   delay(1600);//delay ms
                   if(z7==0){
                         setSSD(z6 , 3);
                         delay(1600);//delay ms
                         setSSD(z5 , 2);
                         delay(1600);//delay ms
                         setSSD(z4 , 1);
                         delay(1600);//delay ms
                         setSSD(12 , 1);'.'
                         delay(1600);//delay ms
                         setSSD(z3 , 0);
                         delay(1600);//delay ms
                         if(z6==0){
                               setSSD(z5 , 3);
                               delay(1600);//delay ms
                               setSSD(z4 , 2);
                               delay(1600);//delay ms
                               setSSD(z3 , 1);
                               delay(1600);//delay ms
                               setSSD(12 , 2);'.'
                               delay(1600);//delay ms
                               setSSD(z2 , 0);
                               delay(1600);//delay ms
                         }
                         if(z5==0){
                               setSSD(z4 , 3);
                               delay(1600);//delay ms
                               setSSD(z3 , 2);
                               delay(1600);//delay ms
                               setSSD(z2 , 1);
                               delay(1600);//delay ms
                               setSSD(12 , 3);'.'
                               delay(1600);//delay ms
                               setSSD(z1 , 0);
                               delay(1600);//delay ms
                         }
```

```
                          if(z4==0){
                                setSSD(z4 , 3);
                                delay(1600);//delay ms
                                setSSD(z3 , 2);
                                delay(1600);//delay ms
                                setSSD(z2 , 1);
                                delay(1600);//delay ms
                                setSSD(12 , 3);'.'
                                delay(1600);//delay ms
                                setSSD(z1 , 0);
                                delay(1600);//delay ms
                          }

                    }*/
                                setSSD(z4 , 3);//calculated number
                                delay(1600);//delay ms
                                setSSD(z3 , 2);//calculated number
                                delay(1600);//delay ms
                                setSSD(z2 , 1);//calculated number
                                delay(1600);//delay ms
                                setSSD(z1 , 0);//calculated number
                                delay(1600);//delay ms
                           arithmetic();//if the  interrupt comes go to arithmetic

                           if(retTime == 0)//wait 10 sec and no press button go to clear SSD
                            break;
                      }
                    clearSSD();//turn off SSD
                          Makezero();//go to start value reset
}
             if(m==5){
                   m=0;//for the second operation value m=0
                   if(n==1){//log
                         z=log10(x);
                   int   z4=z/1000;//separate number
                   int   z3=((z-(z4*1000))/100);
                   int   z2=((z-(z3*100+z4*1000))/10);
                   int   z1=((z-(z2*10+z3*100+z4*1000))/1);
                   a=z4;//assign the result to the first number
                   b=z3;
                   c=z2;
                   d=z1;
                   a1=0;//reset to second number
                   b1=0;
                   c1=0;
                   d1=0;
                           for (unsigned int retTime = time(0) + 2000; time(0) < retTime; retTime--){
// Loop until it arrives.
                         setSSD(z4 , 3);//calculated number
                         delay(1600);//delay ms
                         setSSD(z3 , 2);//calculated number
                         delay(1600);//delay ms
                         setSSD(z2 , 1);//calculated number
                         delay(1600);//delay ms
                         setSSD(z1 , 0);//calculated number
                         delay(1600);//delay ms
                         arithmetic();//if the  interrupt comes go to arithmetic
                                if(retTime == 0)//wait 10 sec and no press button go to clear SSD
                                 break;
                           }
                          clearSSD();//turn off SSD
                               Makezero();//go to start value reset
                   }
                   if(n==2){//ln
                         z=log(x);
                   int   z4=z/1000;//separate number
                   int   z3=((z-(z4*1000))/100);
                   int   z2=((z-(z3*100+z4*1000))/10);
                   int   z1=((z-(z2*10+z3*100+z4*1000))/1);
                   a=z4;//assign the result to the first number
                   b=z3;
                   c=z2;
                   d=z1;
                   a1=0;//reset to second number
                   b1=0;
                   c1=0;
                   d1=0;
                           for (unsigned int retTime = time(0) + 2000; time(0) < retTime; retTime--){
// Loop until it arrives.
                         setSSD(z4 , 3);//calculated number
                         delay(1600);//delay ms
                         setSSD(z3 , 2);//calculated number
                         delay(1600);//delay ms
                         setSSD(z2 , 1);//calculated number
                         delay(1600);//delay ms
                         setSSD(z1 , 0);//calculated number
                         delay(1600);//delay ms
                         arithmetic();//if the  interrupt comes go to arithmetic
                                if(retTime == 0)//wait 10 sec and no press button go to clear SSD
                                 break;
                           }
                          clearSSD();//turn off SSD
                               Makezero();//go to start value reset
                   }
                   if(n==3){//sqrt
                         z=sqrt(x);
                   int   z4=z/1000;//separate number
                   int   z3=((z-(z4*1000))/100);
                   int   z2=((z-(z3*100+z4*1000))/10);
                   int   z1=((z-(z2*10+z3*100+z4*1000))/1);
                   a=z4;//assign the result to the first number
                   b=z3;
                   c=z2;
                   d=z1;
                   a1=0;//reset to second number
                   b1=0;
                   c1=0;
                   d1=0;
                           for (unsigned int retTime = time(0) + 2000; time(0) < retTime; retTime--){
// Loop until it arrives.
                         setSSD(z4 , 3);//calculated number
                         delay(1600);//delay ms
                         setSSD(z3 , 2);//calculated number
                         delay(1600);//delay ms
                         setSSD(z2 , 1);//calculated number
                         delay(1600);//delay ms
                         setSSD(z1 , 0);//calculated number
                         delay(1600);//delay ms
                         arithmetic();//if the  interrupt comes go to arithmetic
                                if(retTime == 0)//wait 10 sec and no press button go to clear SSD
                                 break;
                           }
                          clearSSD();//turn off SSD
                               Makezero();//go to start value reset

                   }
                   if(n==4){//x^2
                         z=x*x;
                   int   z4=z/1000;//separate number
                   int   z3=((z-(z4*1000))/100);
                   int   z2=((z-(z3*100+z4*1000))/10);
                   int   z1=((z-(z2*10+z3*100+z4*1000))/1);
                   a=z4;//assign the result to the first number
                   b=z3;
                   c=z2;
                   d=z1;
                   a1=0;//reset to second number
                   b1=0;
                   c1=0;
                   d1=0;
```

```
                            for (unsigned int retTime = time(0) + 2000; time(0) < retTime; retTime--){
// Loop until it arrives.
                          setSSD(z4 , 3);//calculated number
                          delay(1600);//delay ms
                          setSSD(z3 , 2);//calculated number
                          delay(1600);//delay ms
                          setSSD(z2 , 1);//calculated number
                          delay(1600);//delay ms
                          setSSD(z1 , 0);//calculated number
                          delay(1600);//delay ms
                          arithmetic();//if the  interrupt comes go to arithmetic
                                 if(retTime == 0)//wait 10 sec and no press button go to clear SSD
                                   break;
                            }
                          clearSSD();//turn off SSD
                                Makezero();//go to start value reset

                   }
                   if(n==5){//sin
                          z=sin(x);
                   int    z4=z/1000;//separate number
                   int    z3=((z-(z4*1000))/100);
                   int    z2=((z-(z3*100+z4*1000))/10);
                   int    z1=((z-(z2*10+z3*100+z4*1000))/1);
                   a=z4;//assign the result to the first number
                   b=z3;
                   c=z2;
                   d=z1;
                   a1=0;//reset to second number
                   b1=0;
                   c1=0;
                   d1=0;
                            for (unsigned int retTime = time(0) + 2000; time(0) < retTime; retTime--){
// Loop until it arrives.
                          setSSD(z4 , 3);//calculated number
                          delay(1600);//delay ms
                          setSSD(z3 , 2);//calculated number
                          delay(1600);//delay ms
                          setSSD(z2 , 1);//calculated number
                          delay(1600);//delay ms
                          setSSD(z1 , 0);//calculated number
                          delay(1600);//delay ms
                          arithmetic();//if the  interrupt comes go to arithmetic
                                 if(retTime == 0)//wait 10 sec and no press button go to clear SSD
                                   break;
                            }
                          clearSSD();//turn off SSD
                                Makezero();//go to start value reset

                   }
                   if(n==6){//cos
                          z=cos(x);
                   int    z4=z/1000;//separate number
                   int    z3=((z-(z4*1000))/100);
                   int    z2=((z-(z3*100+z4*1000))/10);
                   int    z1=((z-(z2*10+z3*100+z4*1000))/1);
                   a=z4;//assign the result to the first number
                   b=z3;
                   c=z2;
                   d=z1;
                   a1=0;//reset to second number
                   b1=0;
                   c1=0;
                   d1=0;
```

```
                            for (unsigned int retTime = time(0) + 2000; time(0) < retTime; retTime--){
// Loop until it arrives.
                          setSSD(z4 , 3);//calculated number
                          delay(1600);//delay ms
                          setSSD(z3 , 2);//calculated number
                          delay(1600);//delay ms
                          setSSD(z2 , 1);//calculated number
                          delay(1600);//delay ms
                          setSSD(z1 , 0);//calculated number
                          delay(1600);//delay ms
                          arithmetic();//if the  interrupt comes go to arithmetic
                                 if(retTime == 0)//wait 10 sec and no press button go to clear SSD
                                   break;
                            }
                           clearSSD();//turn off SSD
                                Makezero();//go to start value reset

                   }
                   if(n==7){//tan
                          z=tan(x);
                   int    z4=z/1000;//separate number
                   int    z3=((z-(z4*1000))/100);
                   int    z2=((z-(z3*100+z4*1000))/10);
                   int    z1=((z-(z2*10+z3*100+z4*1000))/1);
                   a=z4;//assign the result to the first number
                   b=z3;
                   c=z2;
                   d=z1;
                   a1=0;//reset to second number
                   b1=0;
                   c1=0;
                   d1=0;
                            for (unsigned int retTime = time(0) + 2000; time(0) < retTime; retTime--){
// Loop until it arrives.
                          setSSD(z4 , 3);//calculated number
                          delay(1600);//delay ms
                          setSSD(z3 , 2);//calculated number
                          delay(1600);//delay ms
                          setSSD(z2 , 1);//calculated number
                          delay(1600);//delay ms
                          setSSD(z1 , 0);//calculated number
                          delay(1600);//delay ms
                          arithmetic();//if the  interrupt comes go to arithmetic
                                 if(retTime == 0)//wait 10 sec and no press button go to clear SSD
                                   break;
                            }
                           clearSSD();//turn off SSD
                                Makezero();//go to start value reset

                   }

                   if(n==8){//cot
                          z=1/tan(x);
                   int    z4=z/1000;//separate number
                   int    z3=((z-(z4*1000))/100);
                   int    z2=((z-(z3*100+z4*1000))/10);
                   int    z1=((z-(z2*10+z3*100+z4*1000))/1);
                   a=z4;//assign the result to the first number
                   b=z3;
                   c=z2;
                   d=z1;
                   a1=0;//reset to second number
                   b1=0;
                   c1=0;
                   d1=0;
```

```
                            for (unsigned int retTime = time(0) + 2000; time(0) < retTime; retTime--){
// Loop until it arrives.
                          setSSD(z4 , 3);//calculated number
                          delay(1600);//delay ms
                          setSSD(z3 , 2);//calculated number
                          delay(1600);//delay ms
                          setSSD(z2 , 1);//calculated number
                          delay(1600);//delay ms
                          setSSD(z1 , 0);//calculated number
                          delay(1600);//delay ms
                          arithmetic();//if the  interrupt comes go to arithmetic
                                  if(retTime == 0)//wait 10 sec and no press button go to clear SSD
                                   break;
                            }
                           clearSSD();//turn off SSD
                                 Makezero();//go to start value reset

                    }
                    if(n==9){//PI
                          z=3;
                    int   z4=z/1000;//separate number
                    int   z3=((z-(z4*1000))/100);
                    int   z2=((z-(z3*100+z4*1000))/10);
                    int   z1=((z-(z2*10+z3*100+z4*1000))/1);
                    a=z4;//assign the result to the first number
                    b=z3;
                    c=z2;
                    d=z1;
                    a1=0;//reset to second number
                    b1=0;
                    c1=0;
                    d1=0;
                            for (unsigned int retTime = time(0) + 2000; time(0) < retTime; retTime--){
// Loop until it arrives.
                          setSSD(z4 , 3);//calculated number
                          delay(1600);//delay ms
                          setSSD(z3 , 2);//calculated number
                          delay(1600);//delay ms
                          setSSD(z2 , 1);//calculated number
                          delay(1600);//delay ms
                          setSSD(z1 , 0);//calculated number
                          delay(1600);//delay ms
                          arithmetic();//if the  interrupt comes go to arithmetic
                                  if(retTime == 0)//wait 10 sec and no press button go to clear SSD
                                   break;
                            }
                           clearSSD();//turn off SSD
                                 Makezero();//go to start value reset
                    }
             }
       }
}
void Makezero(){//set to start values zero
       a=0;
       b=0;
       c=0;
       d=0;
       t=0;
       k=0;
       l=0;
       m=0;
       n=0;
       a1=0;
       b1=0;
       c1=0;
       d1=0;
       while(1){//go to fonk1 and wait here
             fonk1();
       }
}
```

```
void clearSSD(void) {//turn off SSD

       /* Set all output connected to SSD (clear SSD)*/
       GPIOA->BRR |= (0x1A73);

}

void SwitchSSD(int x) {


       switch (x)
             {

             case 0:

                    /* turn on led connected to A,B,C,D,E,F in ODR*/
                   GPIOA->ODR |= (0x1A70);
                   /* turn off led connected to G in ODR*/
                   GPIOA->BRR |= (0x2);
                   break;

             case 1:
                    /* turn on led connected to B,C in ODR*/
                   GPIOA->ODR |= (0x840);
                   /* turn off led connected to A,D,E,F,G in ODR*/
                   GPIOA->BRR |= (0x1232);
                   break;
             case 2:
                    /* turn on led connected to A,B,D,E,G in ODR*/
                   GPIOA->ODR |= (0x1262);
                   /* turn off led connected to C,F in ODR*/
                   GPIOA->BRR |= (0x810);
                   break;
             case 3:
                    /* turn on led connected to A,B,C,D,G in ODR*/
                   GPIOA->ODR |= (0x1A42);
                   /* turn off led connected to E,F in ODR*/
                   GPIOA->BRR |= (0x30);
                   break;
             case 4:
                    /* turn on led connected to B,C,G,F in ODR*/
                   GPIOA->ODR |= (0x852);
                   /* turn off led connected to A,D,E in ODR*/
                   GPIOA->BRR |= (0x1220);
                   break;

             case 5:
                    /* turn on led connected to A,C,D,F,G in ODR*/
                   GPIOA->ODR |= (0x1A12);
                   /* turn off led connected to B,E in ODR*/
                   GPIOA->BRR |= (0x60);
                   break;

             case 6:
                    /* turn on led connected to A,B,C,D,E,F,G in ODR*/
                   GPIOA->ODR |= (0x1A32);
                   /* turn off led connected to B in ODR*/
                   GPIOA->BRR |= (0x40);
                   break;
             case 7:
                    /* turn on led connected to A,B,C in ODR*/
                   GPIOA->ODR |= (0xA40);
                   /* turn off led connected to D,E,F,G in ODR*/
                   GPIOA->BRR |= (0x1032);
                   break;
             case 8:
                    /* turn on led connected to all in ODR*/
                   GPIOA->ODR |= (0x1A72);
                   break;
             case 9:
                    /* turn on led connected to A,B,C,D,F,G in ODR*/
                   GPIOA->ODR |= (0x1A52);
                   /* turn off led connected to E in ODR*/
                   GPIOA->BRR |= (0x20);
                   break;
             case 10://'-'
                    /* turn on led connected to G in ODR*/
                   GPIOA->ODR |= (0x2);
                   /* turn off led connected to A,B,C,D,E,F in ODR*/
                   GPIOA->BRR |= (0x1A70);
            break;
             case 11://'H'
                   /* turn on led connected to B,E,F,G in ODR*/
                   GPIOA->ODR |= (0x872);
                   /* turn off led connected to A,D in ODR*/
                   GPIOA->BRR |= (0x1200);
            break;

             case 12://'.'
                   GPIOA->ODR |= (0x1);
            break;
     /*
                   GPIOA->BRR |= (0x1);//turn off '.'
      */


             }
}

void setSSD(int x , int y) { // x is the number led(0 , 1)  Y is digit (SSD1 , SSD2)
      //clearSSD();


      if(y == 3){

                   /* turn on SSD 1(LEFT).*/
                    /* turn on ODR*/
                   GPIOB->ODR |= (0x100);

                   /* turn off SSD 2.*/
                    /* turn off ODR*/
                   GPIOB->BRR |= (0x4);

                   /* turn off SSD 3.*/
                    /* turn off ODR*/
                   GPIOB->BRR |= (0x1);

                   /* turn off SSD 4.*/
                    /* turn off ODR*/
                   GPIOB->BRR |= (0x2);

                    SwitchSSD(x);
  }


      if(y == 2){

                   /* turn off SSD 1(LEFT).*/
                    /* turn off ODR*/
                   GPIOB->BRR |= (0x100);

                   /* turn on SSD 2.*/
                    /* turn on ODR*/
                   GPIOB->ODR |= (0x4);

                   /* turn off SSD 3.*/
                    /* turn off ODR*/
                   GPIOB->BRR |= (0x1);
                   /* turn off SSD 4.*/
                    /* turn off ODR*/
                   GPIOB->BRR |= (0x2);

                    SwitchSSD(x);
  }


      if(y == 1){

                   /* turn off SSD 1(LEFT).*/
                    /* turn off ODR*/
                   GPIOB->BRR |= (0x100);

                   /* turn off SSD 2.*/
                    /* turn off ODR*/
                   GPIOB->BRR |= (0x4);

                   /* turn on SSD 3.*/
                    /* turn on ODR*/
                   GPIOB->ODR |= (0x1);

                   /* turn off SSD 4.*/
                    /* turn off ODR*/
                   GPIOB->BRR |= (0x2);

                    SwitchSSD(x);
  }

      if(y == 0){

                   /* turn off SSD 1(LEFT).*/
                    /* turn off ODR*/
                   GPIOB->BRR |= (0x100);

                   /* turn off SSD 2.*/
                    /* turn off ODR*/
                   GPIOB->BRR |= (0x4);

                   /* turn off SSD 3.*/
                    /* turn off ODR*/
                   GPIOB->BRR |= (0x1);

                   /* turn on SSD 4.*/
                    /* turn on ODR*/
                   GPIOB->ODR |= (0x2);

                    SwitchSSD(x);
  }

}

void clearRowsKeypad(void){
           /* Clearing the rows here */
            GPIOB->ODR &= ~(1U << 9);   /// PB9
               GPIOB->ODR &= ~(1U << 5);   /// PB5
               GPIOB->ODR &= ~(1U << 4);   /// PB4
               GPIOB->ODR &= ~(1U << 3);   /// PB3
}


void setRowsKeypad(void){
         /* Setting the rows here   */
               GPIOB->ODR |= (1U << 9);   /// PB9
            GPIOB->ODR |= (1U << 5);   /// PB5
               GPIOB->ODR |= (1U << 4);   /// PB4
               GPIOB->ODR |= (1U << 3);   /// PB3

}
```