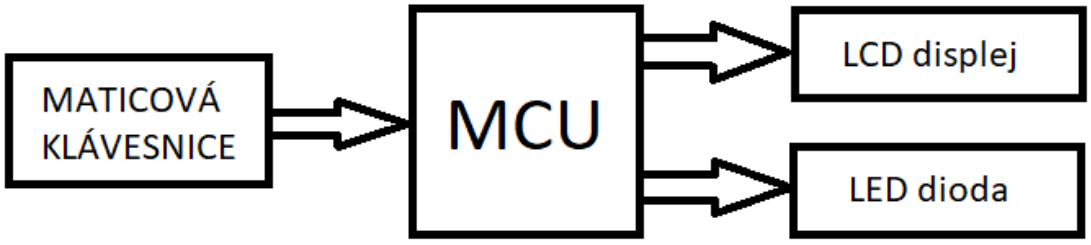
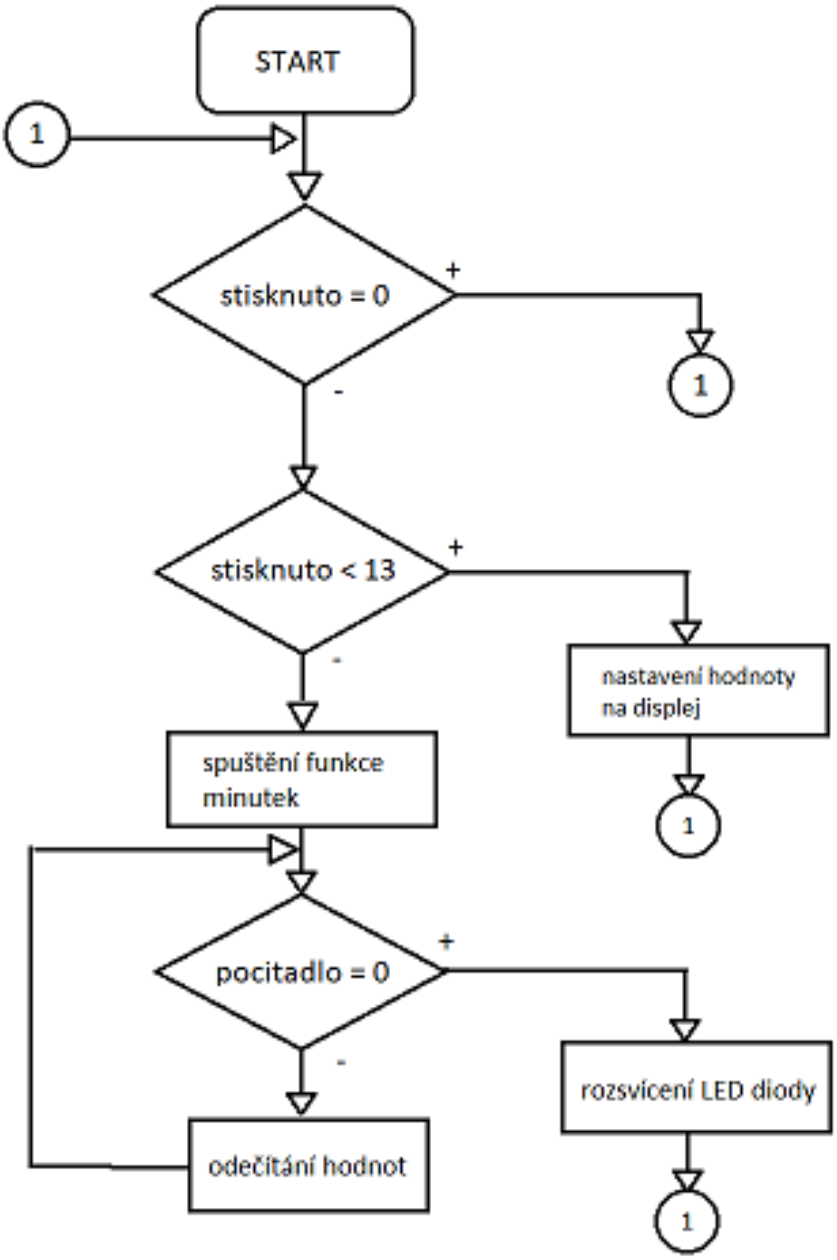
MINUTKY

BLOKOVÉ SCHÉMA:



VÝVOJOVÝ DIAGRAM:



FUNKCE:

- nastavování času a spouštění pomocí maticové klávesnice
- nastavení minut, odečítání po sekundách
- zobrazení času a stavu na LCD displej
- po ukončení signalizace LED diodou
- funkce spustit, pozastavit, vynulovat, vypnout

ZAPOJENÍ:

K STM8 kitu je připojena maticová klávesnice jako vstup. Výstupy jsou LCD displej a LED dioda, která slouží pro signalizaci ukončení minutek. LED dioda a potenciometr (pro nastavení LCD displeje) jsou zapojeny na nepájivém poli.

ZÁVĚR:

Programoval jsem mikrokontrolér STMS208RB v prostředí STVD. Minutky mají funkce spuštění, pozastavení, vynulování a vypnutí. Po vypršení času se rozsvítí LED dioda, která se vypne po stlačení tlačítka STOP. LED dioda lze jednoduše nahradit sirénkou, která může simulovat skutečné kuchyňské minutky. Hodnota se nastaví sčítáním stisknutých hodnot na maticové klávesnici.
Projekt je plně funkční a bez zjištěných chyb. S trochou fantazie lze dokonce prakticky využívat, jediný problém je však vysoká nepraktičnost do běžného života. Program je dle mého názoru velice přehledný. Tento projekt mě naučil, co v sobě skrývá program STVD a kompletně mi změnil pohled na okolní svět.

PROGRAM:

```
#include "stm8s.h"
#include "milis.h"
#include "stm8_hd44780.h"
#include "stdio.h"
#include "keypad.h"

void process_keypad(void);
void odpocet(void);
void zobraz_pocitadlo(void);
void zobraz_sekundy(void);

uint16_t pocitadlo=0;
uint16_t sekundy=60;
```

```
uint16_t lasttime=0;
uint8_t text[32];
uint8_t x=0; //proměnná sloužící k zahájení odpočtu
uint8_t y=0; //pomocná proměnná

void main(void){
CLK_HSIPrescalerConfig(CLK_PRESCALER_HSIDIV1); // 16MHz z interního RC oscilátoru
GPIO_Init(GPIOB,GPIO_PIN_7,GPIO_MODE_OUT_PP_LOW_SLOW);
init_milis();
lcd_init();
keypad_init();

  while (1){
                process_keypad(); // obsluhujeme klávesnici
                if(x == 1){             //spustí minutky
                        if(y == 0){
                                if(pocitadlo > 0){
                                        pocitadlo--;  //odpocet první minuty při začátku odpočtu
                                        y=1;
                                }
                        }
                        odpocet();
                }
        }
}

// pravidelně sleduje stav klávesnice a reaguje na stisknutí kláves
void process_keypad(void){
static uint8_t minule_stisknuto=0;      // ukládáme poslední stav klávesnice
static uint16_t last_time=0;
uint16_t stisknuto;

        if(milis()-last_time > 20){ // každých 20ms ...
                last_time = milis();
                stisknuto=keypad_scan(); // ... skenujeme klávesnici
                if(minule_stisknuto == 0 && stisknuto != 0){ // pokud byla minule uvolněná a teď je
něco stisknuto
                        minule_stisknuto = stisknuto;
                        // zachytili jsme stisk klávesy - můžeme na to zareagovat

                        if(stisknuto < 13){     //pro přidávání hodnot slouží jen hodnoty od 1 do 12,
spodní 4 klávesy jsou volné pro ovládání minutek
                                pocitadlo=pocitadlo+stisknuto;
                                lcd_gotoxy(0,0);
                                sprintf(text,"%3u",pocitadlo);
```

```
            lcd_puts(text);
            if(x == 0){
                  sekundy=60;
            }
      }
      else if(stisknuto == 15){ //nulování počítadla
            lcd_clear();
            pocitadlo=0;
            sekundy=0;
            zobraz_pocitadlo();
            zobraz_sekundy();
            x=0;
            y=0;
      }
      else if(stisknuto == 16){ //vymaže celý displej, simulace vypnutí
            pocitadlo=0;
            sekundy=0;
            x=0;
            y=0;
            lcd_clear();
      }
      else if(stisknuto == 13 && sekundy > 0){ //START tlačítko, spuštění minutek
            lcd_gotoxy(0,1);
            sprintf(text,">MINUTKY BEZI<");
            lcd_puts(text);
            x=1;
            //tímto tlačítkem se bude odpočítávání spouštět
      }
      else if(stisknuto == 14 && x == 1){ //STOP tlačítko, zastavení minutek
            lcd_clear();
            zobraz_pocitadlo();
            lcd_gotoxy(4,0);
            sprintf(text,"%2u",sekundy);
            lcd_puts(text);
            if(sekundy < 10){
                  lcd_gotoxy(4,0);
                  sprintf(text,"0");
                  lcd_puts(text);
            }
            lcd_gotoxy(0,1);
            sprintf(text,">POZASTAVENO<");
            lcd_puts(text);
            x=0;
            //tímto tlačítkem se bude odpočítávání pozastavovat
      }
```

```
                else if(stisknuto == 14 && x == 0){                    //STOP tlačítko, vypnutí
výstražného signálu pro ukončení odpočtu
                        GPIO_WriteLow(GPIOB,GPIO_PIN_7);
                }

        }
        if(stisknuto == 0){minule_stisknuto=0;}
    }
}

void odpocet(void){
    if(milis()-lasttime > 1000 && sekundy > 0){
        lasttime = milis();
        sekundy--;
        lcd_gotoxy(4,0);
        sprintf(text,"%2u",sekundy);
        lcd_puts(text);
        zobraz_pocitadlo();
        if(sekundy < 10){
            lcd_gotoxy(4,0);
            sprintf(text,"0");
            lcd_puts(text);
        }

        if(sekundy == 0 && pocitadlo > 0){
            sekundy=60;
            pocitadlo--;
        }

        else if (pocitadlo == 0 && sekundy == 0){
            lcd_clear();
            zobraz_pocitadlo();
            zobraz_sekundy();
            GPIO_WriteHigh(GPIOB,GPIO_PIN_7); //zapne výstražný signál (ledku)
            x=0;
            y=0;
        }
    }
}

void zobraz_pocitadlo(void){
    lcd_gotoxy(0,0);
    sprintf(text,"%3u:",pocitadlo);
    lcd_puts(text);
}
```

```
void zobraz_sekundy(void){
        lcd_gotoxy(4,0);
        sprintf(text,"00");
        lcd_puts(text);
}

// pod tímto komentářem nic neměňte
#ifdef USE_FULL_ASSERT

/**
  * @brief  Reports the name of the source file and the source line number
  *   where the assert_param error has occurred.
  * @param file: pointer to the source file name
  * @param line: assert_param error line source number
  * @retval : None
  */
void assert_failed(u8* file, u32 line)
{
  /* User can add his own implementation to report the file name and line number,
     ex: printf("Wrong parameters value: file %s on line %d\r\n", file, line) */

  /* Infinite loop */
  while (1)
  {
  }
}
#endif
```