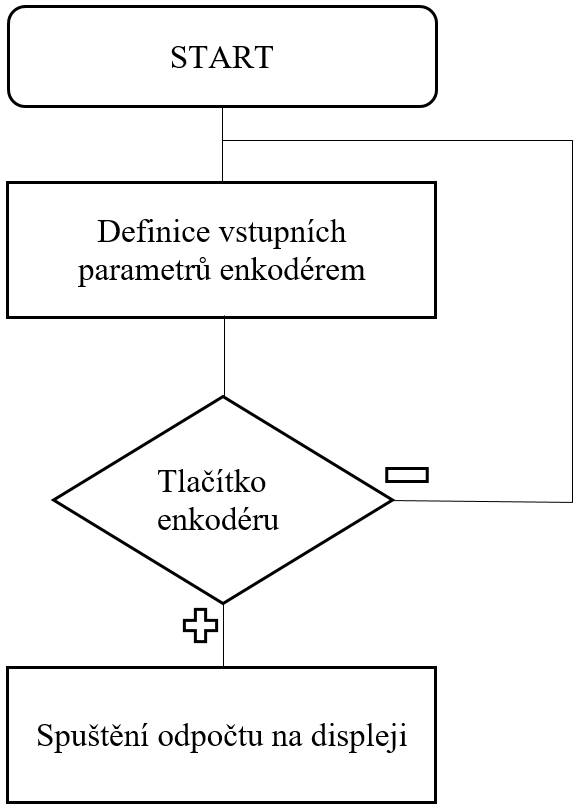
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Vyšší odborná škola a Střední průmyslová škola elektrotechnickáBožetěchova 3, OlomoucLaboratoře mikroprocesorové techniky | | | | | | | | | | | | | | |
| **SAMOSTATNÝ PROJEKT MIT** | | | | | | | | | | | | | | |
| Název úlohy | |  | | | | | | | | | | | Číslo úlohy |  |
| MINUTKA | | | | | | | | | | | | | MIT-01 | |
| Zadání   1. V levé části displeje se zobrazují minuty (0-99), v pravé části sekundy. 2. Točením enkodéru doprava se přičítá čas po 10s, doleva se čas ubírá. Otočením doleva přes nulu se čas nastaví na maximum. 3. Kliknutím enkodéru se minutky rozběhnou/zastaví 4. Podržením enkodéru na 3s se čas vyresetuje. 5. Po uběhnutí času budou minutky 10s vydávat zvuk a blikat displejem. | | | | | | | | | | | | | | |
| Poř. č. | Příjmení a jméno | | | |  | | | | | Třída | Skupina | | Školní rok |  |
| 15 | MICHALČÍK Ondřej | | | | | | | | | 4A | 2 | | 2021/22 | |
| Datum vypracování | | |  | Datum odevzdání | |  | Počet listů | |  | Klasifikace | | | | |
| 17. 2. 2022 | | | | 18. 2. 2022 | | | 5 | | |  | | | | | |
| Protokol obsahuje: | | | | | | | | Blokové schéma | | | | Program | | |
|  | | | | | | | | Vývojový diagram | | | | Závěr | | |
|  | | | | | | | | Schéma zapojení | | | |  | | |

# Zapojení:

# C:\Users\Ondra\Desktop\ŠKOLA\PRA\projekt z mit\Bez názvu.png

Ke kitu STM8 je připojen segmentový displej s driverem MAX7219, rotační enkodér, a bzučák

# Vývojový diagram:



# ZÁVĚR:

Projekt nalezne své využití v každé kuchyní. Pro praktičtější využití by bylo dobré pro něj vyrobit nějaké pouzdro (krabičku). Vypracování celého projektu zabralo 10 až 20 hodin v průběhu několika týdnů.

# Kód:

#include "stm8s.h"

#include "milis.h"

#include "swspi.h"

#define DECODEMODE (0x9<<8)

#define INTENSITY (0xa<<8)

#define SCANLIMIT (0xb<<8)

#define SHUTDOWN (0xc<<8)

#define DTEST (0xf<<8)

#define d1 (0x1<<8)

#define d2 (0x2<<8)

#define d3 (0x3<<8)

#define d4 (0x4<<8)

#define d5 (0x58<<4)

#define d5b (0x5<<8)

#define d6 (0x6<<8)

#define d7 (0x7<<8)

#define d8 (0x8<<8)

void max7219\_init(void);

void init\_enc(void);

void init\_timer(void);

//funkce

void klok(void);

void pauza (void);

void had(void);

void displej\_sender(void);

void process\_enc(void);

void end\_anim(void);

volatile int16\_t hodnota=0; //proměnné enkodéru

uint32\_t sekunda=0; //proměnné pro čas

uint32\_t l\_snake=0; //proměnné pro funkci hada na stranách displeje

uint8\_t sneaky=64;

uint16\_t l\_reset=0; //proměnné resetu

uint8\_t l\_pauza=2; //proměnné k pauzování displeje

uint8\_t end=0; //proměnné pro signalizaci oběhnutí času

uint32\_t l\_end=0;

void main(void){

CLK\_HSIPrescalerConfig(CLK\_PRESCALER\_HSIDIV1);

init\_milis();

init\_enc();

init\_timer();

swspi\_init();

max7219\_init();

enableInterrupts();

l\_snake = milis();

while(1){

klok();

had();

pauza();

displej\_sender();

}

}

void klok (void){

if (sekunda == 0){l\_pauza = 2;} //aby šel po uběhnutí času opět přičítat enkodérem

}

void had(void){

if (milis() - l\_snake >= 166){

l\_snake = milis();

sneaky = sneaky>>1;

}

if (sneaky == 1){

sneaky = 64;

}

}

void pauza (void){

if (GPIO\_ReadInputPin(GPIOF,GPIO\_PIN\_5) == RESET && l\_pauza == 0){

l\_pauza = 1;

l\_reset = milis();

}

if (GPIO\_ReadInputPin(GPIOF,GPIO\_PIN\_5) != RESET && l\_pauza == 1){

l\_pauza = 2;

}

if (GPIO\_ReadInputPin(GPIOF,GPIO\_PIN\_5) == RESET && l\_pauza == 2){

l\_pauza = 3;

l\_reset = milis();

}

if (GPIO\_ReadInputPin(GPIOF,GPIO\_PIN\_5) != RESET && l\_pauza == 3){

l\_pauza = 0;

}

if (GPIO\_ReadInputPin(GPIOF,GPIO\_PIN\_5) == RESET && l\_pauza == 1 && milis() - l\_reset >= 3000){sekunda = 0;} //reset při podržení

}

void displej\_sender(void){

swspi\_tx16(d1 | sneaky);

swspi\_tx16(d2 | sneaky);

swspi\_tx16(d3 | (sekunda%10));

swspi\_tx16(d4 | (sekunda%60/10));

swspi\_tx16(d5 | (sekunda%600/60));

swspi\_tx16(d6 | (sekunda/600));

swspi\_tx16(d7 | sneaky);

swspi\_tx16(d8 | sneaky);

}

void end\_anim(void){

l\_end = milis();

if (milis() - l\_end <=1000){

swspi\_tx16(DECODEMODE | 0);

swspi\_tx16(d1 | sneaky);

swspi\_tx16(d2 | sneaky);

swspi\_tx16(d3 | sneaky);

swspi\_tx16(d4 | sneaky);

swspi\_tx16(d5b | sneaky);

swspi\_tx16(d6 | sneaky);

swspi\_tx16(d7 | sneaky);

swspi\_tx16(d8 | sneaky);

}

if (milis() - l\_end >=1001){

displej\_sender();

}

if (milis() - l\_end >=2000){

l\_end = milis();

}

}

INTERRUPT\_HANDLER(TIM2\_UPD\_OVF\_BRK\_IRQHandler, 13){

TIM2\_ClearITPendingBit(TIM2\_IT\_UPDATE);

if((l\_pauza == 0 || l\_pauza == 3) && sekunda>0){sekunda--;} //odečítání pouze při určitých stavech stavového automatu

}

INTERRUPT\_HANDLER(TIM3\_UPD\_OVF\_BRK\_IRQHandler, 15){

TIM3\_ClearITPendingBit(TIM3\_IT\_UPDATE);

process\_enc();

}

void max7219\_init(void){

swspi\_tx16(DECODEMODE | 0x3c);

swspi\_tx16(INTENSITY | 0x07);

swspi\_tx16(SCANLIMIT | 7);

swspi\_tx16(DTEST | 0);

swspi\_tx16(SHUTDOWN | 1);

}

void init\_timer(void){

TIM3\_TimeBaseInit(TIM3\_PRESCALER\_16,1999); //timer pro enc

TIM3\_ITConfig(TIM3\_IT\_UPDATE, ENABLE);

TIM3\_Cmd(ENABLE);

TIM2\_TimeBaseInit(TIM2\_PRESCALER\_1024,15625); //timer pro čas

TIM2\_ITConfig(TIM2\_IT\_UPDATE, ENABLE);

TIM2\_Cmd(ENABLE);

}

void init\_enc(void){

GPIO\_Init(GPIOF,GPIO\_PIN\_7,GPIO\_MODE\_IN\_PU\_NO\_IT);

GPIO\_Init(GPIOF,GPIO\_PIN\_6,GPIO\_MODE\_IN\_PU\_NO\_IT);

GPIO\_Init(GPIOF,GPIO\_PIN\_5,GPIO\_MODE\_IN\_PU\_NO\_IT);

}

void process\_enc(void){

uint8\_t minule=1;

if(GPIO\_ReadInputPin(GPIOF,GPIO\_PIN\_7) == RESET && minule==1){

minule = 0;

if(GPIO\_ReadInputPin(GPIOF,GPIO\_PIN\_6) == RESET){

if(l\_pauza == 1 || l\_pauza == 2){sekunda = sekunda-10;} //zablokuje přidávání při odpočtu

if(sekunda<0){sekunda=5999;} //odečtení pod nulu nastaví na maximální čas

}else{

if(l\_pauza == 1 || l\_pauza == 2){sekunda = sekunda+10;}

if(sekunda>5999){sekunda=0;}

}

}

if(GPIO\_ReadInputPin(GPIOF,GPIO\_PIN\_7) != RESET){minule = 1;}

}

#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