GEBZE TECHNICAL UNIVERSITY

ELECTRONICS ENGINEERING

ELM 335

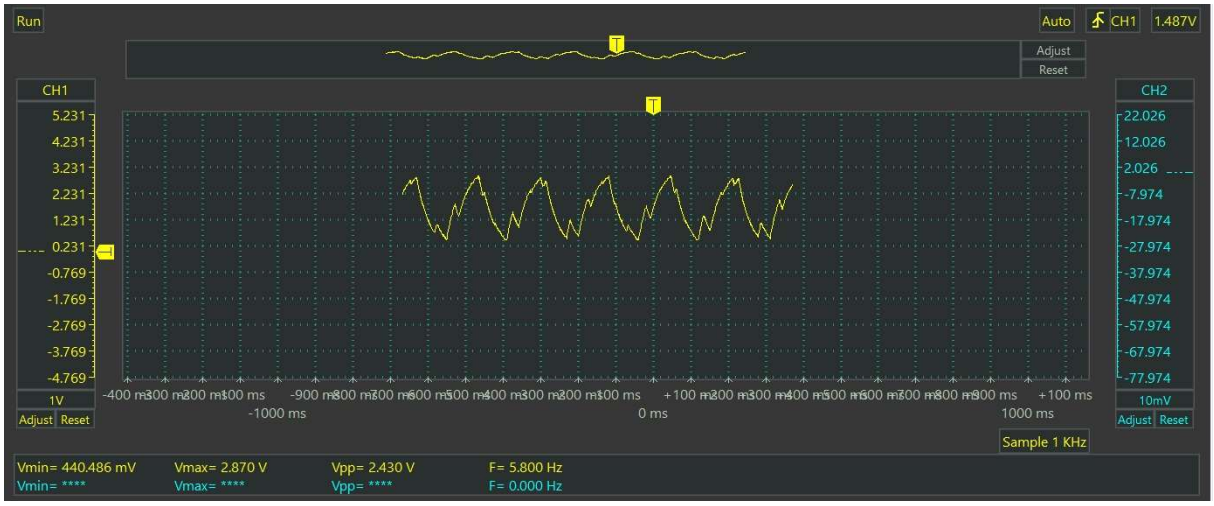
Microprocessors Laboratory

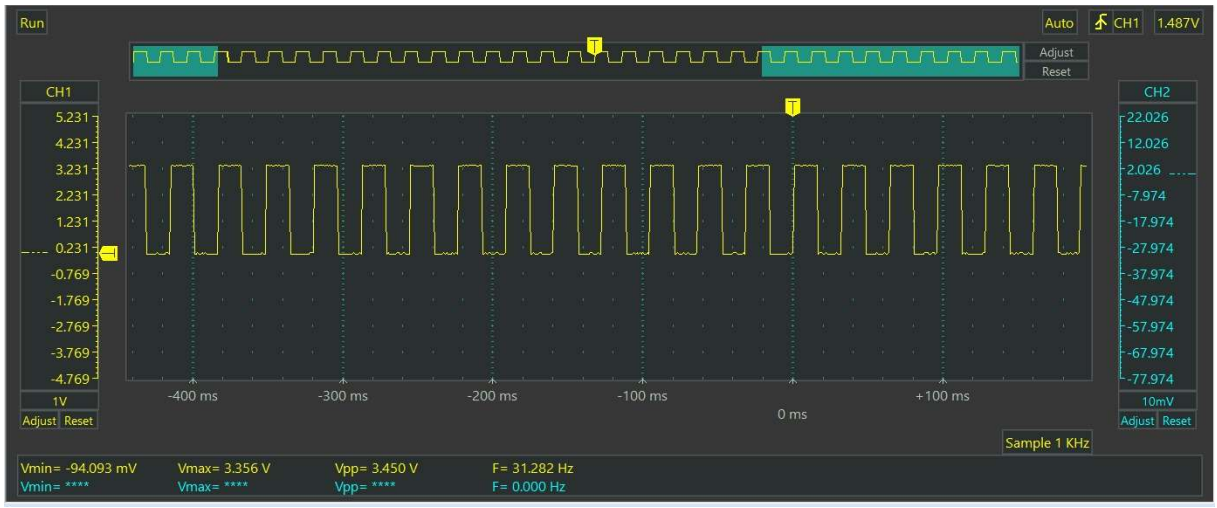
Proje 1

Yasin Özbek-171024008

Abdurrahim Deniz Kumbaracı-151024008

Altough we were able to generate signals using look-up tables,due to design of our filter, only the harmonics of sine wave was isolated so we were only able to capture distorted sine waves. And due to comlexity of PWM mode of the GPIOs we were not able to control the frequency and amplitude of the signals. Using DMA would have been a much more efficent way in terms of workload to duty cycles, and also using DAC would have been more eficent to generate triangle and noise signals.





CODE

```
#include "stm32g0xx.h"
#include "nucleo.h"

volatile  uint16_t digits[4]={0x02,0x08,0x40,0x80};  //PB2-6-7-8 PB1-3-6-7
{0x04,0x40,0x80,0x100}

enum Letters{t=0,r=1,i=2,s=3,n=4,e=5,c=6,o=7,a=8,u=9};

volatile int loops[4] = {3,3,3,3};


enum TimeStates{
      State0,
      State1,
      State2,
      State3,
      State4,
      State5
```

```
};
enum TimeStates Scale = State0 ;

void clearSSD(void);
void setSSD(int);

void clearRowsKeypad(void);
void setRowsKeypad(void);

void init_SSD_Pins();
void init_Clocks();
void init_interrupt();


uint32_t
sinus[]={4000,4251,4501,4750,4995,5236,5472,5703,5927,6143,6351,6550,6738,6916,708
2,
7236,7377,7505,7619,7719,7804,7874,7929,7968,7992,8000,7992,7968,7929,7874,
7804,7719,7619,7505,7377,7236,7082,6916,6738,6550,6351,6143,5927,5703,5472,
5236,4995,4750,4501,4251,4000,3749,3499,3250,3005,2764,2528,2297,2073,1857,
1649,1450,1262,1084,918,764,623,495,381,281,196,126,71,32,8,
0,8,32,71,126,196,281,381,495,623,764,918,1084,1262,1450,
1649,1857,2073,2297,2528,2764,3005,3250,3499,3749,4000,
};
uint32_t
tri[]={160,320,480,640,800,960,1120,1280,1440,1600,1760,1920,2080,2240,2400,
2560,2720,2880,3040,3200,3360,3520,3680,3840,4000,4160,4320,4480,4640,4800,
4960,5120,5280,5440,5600,5760,5920,6080,6240,6400,6560,6720,6880,7040,7200,
7360,7520,7680,7840,8000,7840,7680,7520,7360,7200,7040,6880,6720,6560,6400,
6240,6080,5920,5760,5600,5440,5280,5120,4960,4800,4640,4480,4320,4160,4000,
3840,3680,3520,3360,3200,3040,2880,2720,2560,2400,2240,2080,1920,1760,1600,
1440,1280,1120,960,800,640,480,320,160,0,160,
};
uint32_t
saw[]={160,320,480,640,800,960,1120,1280,1440,1600,1760,1920,2080,2240,2400,
2560,2720,2880,3040,3200,3360,3520,3680,3840,4000,4160,4320,4480,4640,4800,
4960,5120,5280,5440,5600,5760,5920,6080,6240,6400,6560,6720,6880,7040,7200,
7360,7520,7680,7840,8000,160,320,480,640,800,960,1120,1280,1440,1600,1760,1920,208
0,2240,2400,
2560,2720,2880,3040,3200,3360,3520,3680,3840,4000,4160,4320,4480,4640,4800,
4960,5120,5280,5440,5600,5760,5920,6080,6240,6400,6560,6720,6880,7040,7200,
7360,7520,7680,7840,8000,
};

uint32_t
ar[]={22500,23913,25320,26716,28096,29453,30783,32080,33339,34556,35725,36842,3790
2,38902,39837,
40703,41497,42217,42859,43420,43899,44293,44601,44823,44956,45000,44956,44823,4460
1,44293,
43899,43420,42859,42217,41497,40703,39837,38902,37902,36842,35725,34556,33339,3208
0,30783,
29453,28096,26716,25320,23913,22500,21087,19680,18284,16904,15547,14217,12920,1166
1,10444,
9275,8158,7098,6098,5163,4297,3503,2783,2141,1580,1101,707,399,177,44,
0,44,177,399,707,1101,1580,2141,2783,3503,4297,5163,6098,7098,8158,
9275,10444,11661,12920,14217,15547,16904,18284,19680,21087,22500,};

void TIM1_BRK_UP_TRG_COM_IRQHandler(void){
```

```
	for(int k=0;k<=100;++k){
			TIM1->ARR = 8000;
			if(Scale==State0){
				TIM1->CCR3 = (sinus[k]);
			}
			if(Scale==State1){
				TIM1->CCR3 = (saw[k]);
			}
			if(Scale==State2){
				TIM1->CCR3 = (tri[k]);
			}
			if(Scale==State3){
				TIM1->CCR3 = 4000;
			}
			if(k>=100){
				k=0;



			}
			if(Scale>State3){
			Scale=State0;
			}


			TIM1 -> SR &= ~ (1U << 0);
	}

}


void clearSSD(){
    GPIOA->ODR |= (1U << 0);  //PA0  A
    GPIOA->ODR |= (1U << 1); //PA1 B
    GPIOA->ODR |= (1U << 4); //PA4 C
    GPIOA->ODR |= (1U << 5); // PA5 D
    GPIOA->ODR |= (1U << 12); // PA12 E
    GPIOA->ODR |= (1U << 11); // PA11 F
    GPIOA->ODR |= (1U << 6); //PA6 G

    GPIOB->ODR &= ~(1U << 1);
    GPIOB->ODR &= ~(1U << 6);
    GPIOB->ODR &= ~(1U << 7);
    GPIOB->ODR &= ~(1U << 3);
}

void disp_num(int digit){
	clearSSD();
	 GPIOB->ODR |=  digits[digit];
	 setSSD(loops[digit]);
}

void display(){

	volatile  uint32_t i=40;
for(;i>0;--i){
	disp_num(1);
	disp_num(3);
	disp_num(2);
```

```
        disp_num(0);
        }

}

void Tri(){
        loops[0] = t;
        loops[1] = r;
        loops[2] = i;
        loops[3] = 10;
}

void Sin(){
        loops[0] = s;
        loops[1] = i;
        loops[2] = n;
        loops[3] = 10;
}

void Rect(){
        loops[0] = r;
        loops[1] = e;
        loops[2] = c;
        loops[3] = t;
}
void Nois(){
        loops[0] = n;
        loops[1] = o;
        loops[2] = i;
        loops[3] = s;
}
void Saw(){
        loops[0] = s;
        loops[1] = a;
        loops[2] = u;
        loops[3] = u;
}

void EXTI0_1_IRQHandler(void){


 Scale = Scale + 1;
 EXTI->RPR1 |= (1U << 0);
}

void setSSD(int x){


        switch(x){
              case 0: // t
                GPIOA->ODR &= ~(1U << 5); //PA5 D
                GPIOA->ODR &= ~(1U << 12); //PA12 E
                GPIOA->ODR &= ~(1U << 11); //PA11 F
                GPIOA->ODR &= ~(1U << 6); //PA6 G
                break;
              case 1: //r
                GPIOA->ODR &= ~(1U << 12); //PA12 E
                GPIOA->ODR &= ~(1U << 6); //PA6 G
                break;
```

```
  case 2: //i
    GPIOA->ODR &= ~(1U << 12); //PA12 E
    break;
  case 3: //s
    GPIOA->ODR &= ~(1U << 0); //PA0 A
    GPIOA->ODR &= ~(1U << 4); //PA4 C
    GPIOA->ODR &= ~(1U << 5); //PA5 D
    GPIOA->ODR &= ~(1U << 11); //PA11 F
    GPIOA->ODR &= ~(1U << 6); //PA6 G
    break;
  case 4: //n
    GPIOA->ODR &= ~(1U << 4); //PA4 C
    GPIOA->ODR &= ~(1U << 12); //PA12 E
    GPIOA->ODR &= ~(1U << 6); //PA6 G
    break;
  case 5://e
    GPIOA->ODR &= ~(1U << 0); //PA0 A
    GPIOA->ODR &= ~(1U << 5); //PA5 D
    GPIOA->ODR &= ~(1U << 12); //PA12 E
    GPIOA->ODR &= ~(1U << 11); //PA11 F
    GPIOA->ODR &= ~(1U << 6); //PA6 G
    break;
  case 6://c
    GPIOA->ODR &= ~(1U << 5); //PA5 D
    GPIOA->ODR &= ~(1U << 12); //PA12 E
    GPIOA->ODR &= ~(1U << 6); //PA6 G
    break;
  case 7://o
    GPIOA->ODR &= ~(1U << 4); //PA4 C
    GPIOA->ODR &= ~(1U << 5); //PA5 D
    GPIOA->ODR &= ~(1U << 12); //PA12 E
    GPIOA->ODR &= ~(1U << 6); //PA6 G

    break;
  case 8://a
    GPIOA->ODR &= ~(1U << 0); //PA0 A
    GPIOA->ODR &= ~(1U << 1); //PA1 B
    GPIOA->ODR &= ~(1U << 4); //PA4 C
    GPIOA->ODR &= ~(1U << 12); //PA12 E
    GPIOA->ODR &= ~(1U << 11); //PA11 F
    GPIOA->ODR &= ~(1U << 6); //PA6 G


    break;
  case 9://u
    GPIOA->ODR &= ~(1U << 4); //PA4 C
    GPIOA->ODR &= ~(1U << 5); //PA5 D
    GPIOA->ODR &= ~(1U << 12); //PA12 E
    break;
  case 10:
GPIOA->ODR |= (1U << 0);  //PA0  A
      GPIOA->ODR |= (1U << 1); //PA1 B
      GPIOA->ODR |= (1U << 4); //PA4 C
      GPIOA->ODR |= (1U << 5); // PA5 D
      GPIOA->ODR |= (1U << 12); // PA12 E
      GPIOA->ODR |= (1U << 11); // PA11 F
      GPIOA->ODR |= (1U << 6); //PA6 G
      break;
```

```
      }
}

void init_SSD_Pins(){
       GPIOA->MODER &= ~(3U << 2*0);  // PA0 A0
          GPIOA->MODER |= (1U << 2*0);

          GPIOA->MODER &= ~(3U << 2*1);  // PA1 A1
          GPIOA->MODER |= (1U << 2*1);

          GPIOA->MODER &= ~(3U << 2*4);  // PA4 A2
          GPIOA->MODER |= (1U << 2*4);

          GPIOA->MODER &= ~(3U << 2*5);  // PA5 A3
          GPIOA->MODER |= (1U << 2*5);

          GPIOA->MODER &= ~(3U << 2*12);  // PA12 A4
          GPIOA->MODER |= (1U << 2*12);

          GPIOA->MODER &= ~(3U << 2*11);  // PA11 A5
          GPIOA->MODER |= (1U << 2*11);

          GPIOA->MODER &= ~(3U << 2*6);  // PA6 A6
          GPIOA->MODER |= (1U << 2*6);

}

void init_DigitOutput(){
    GPIOB->MODER &= ~(3U << 2*1);  // PB2 D7 digit1
    GPIOB->MODER |= (1U << 2*1);

    GPIOB->ODR |= (1U << 1);

    GPIOB->MODER &= ~(3U << 2*6);  // PB6 D1 digit2
    GPIOB->MODER |= (1U << 2*6);

    GPIOB->ODR |= (1U << 6);

    GPIOB->MODER &= ~(3U << 2*7);  // PB7 D0 digit3
    GPIOB->MODER |= (1U << 2*7);

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

    GPIOB->MODER &= ~(3U << 2*3);  // PB8 D8  digit4
    GPIOB->MODER |= (1U << 2*3);

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


}

void init_Clocks(){
    RCC->IOPENR |= (1U << 0);
    RCC->IOPENR |= (1U << 1);
}

void init_interrupt(){
    GPIOB->MODER &= ~(3U << 2*0); // PB0 D6
    GPIOB->PUPDR |= (2U << 2*0);
```

```
	EXTI->EXTICR[0] |= (1U << 0);
	EXTI->RTSR1 |= (1U << 0);
	EXTI->IMR1  |= (1U << 0);
    NVIC_SetPriority(EXTI0_1_IRQn , 0);
    NVIC_EnableIRQ(EXTI0_1_IRQn);
}

void clearRowsKeypad(void){

    GPIOA->ODR &= ~(1U << 8);
    GPIOB->ODR &= ~(1U << 9);
    GPIOB->ODR &= ~(1U << 5);
    GPIOB->ODR &= ~(1U << 4);

}

void setRowsKeypad(void){

    GPIOA->ODR |= (1U << 8);
    GPIOB->ODR |= (1U << 9);
    GPIOB->ODR |= (1U << 5);
    GPIOB->ODR |= (1U << 4);
}


int main(void){
	init_Clocks();
	init_interrupt();
	init_SSD_Pins();
	init_DigitOutput();

	RCC->APBENR2 |= RCC_APBENR2_TIM1EN;
	RCC->IOPENR |= RCC_IOPENR_GPIOAEN;
	GPIOA->MODER &= ~ GPIO_MODER_MODE10_0;
	GPIOA->MODER |= GPIO_MODER_MODE10_1;



	GPIOA->AFR[1] = (2U<<8);
	timer1_init();
	TIM1 ->PSC =63;
	TIM1->ARR = 8000;
	TIM1->CCR3 = 0;
	TIM1->CCMR2 |= TIM_CCMR2_OC3M_1 | TIM_CCMR2_OC3M_2 |TIM_CCMR2_OC3PE;
	TIM1->CCER |= TIM_CCER_CC3E;
	TIM1->BDTR |= TIM_BDTR_MOE;
	TIM1->CR1 |= TIM_CR1_CEN;
	TIM1 -> EGR |= TIM_EGR_UG;


while(1){

	display();
	if (Scale == State0){
//Tri yaz

		Tri();
```

```
	}
	if (Scale == State1){
//Sin yaz
		Sin();

	}
	if (Scale == State2){
//rect yaz
		Rect();


	}
	if (Scale == State3){
//nois yaz


		Nois();

	}
	if (Scale == State4){
//saw yaz
		Saw();

	}
	if (Scale > State4){
		Scale = State0;
	}

	}
return 0;
	}
```

Nucleo.c

```
/*
 * nucleo.c
 *
 *  Created on: Nov 29, 2021
 *      Author: Deniz
 */

#include "nucleo.h"
#include "stm32g0xx.h"

#define KILO    1000
#define MEGA    1000000

void nucleo_PA0_button_init(){
	RCC -> IOPENR |= (1U <<0 );
	GPIOA -> MODER &= ~ (3U << 0);
	GPIOA -> PUPDR &= ~ (3U << 0);
	GPIOA -> PUPDR |=  (2U << 0);
}
```

```c
int nucleo_PA0_button_read(void) {

	int a = ((GPIOA -> IDR >> 2 ) & 0x01);
		if (a) return 0;
		else return 1;

}

void nucleo_PA0_button_INT(){
	EXTI -> RTSR1 |= (1U <<0 );
	EXTI -> EXTICR[0] |= (0U <<0 );

	EXTI -> IMR1 |= (1U <<0 );
	NVIC_SetPriority(EXTI0_1_IRQn , 0);
	NVIC_EnableIRQ(EXTI0_1_IRQn);
}

void nucleo_PA0_button_statclear(){
	EXTI -> RPR1 &= ~ (1U << 0);
}

void nucleo_led_init(void){
	RCC -> IOPENR |= (1U <<2 );
	GPIOC -> MODER &= ~ (3U << 2*6);
	GPIOC -> MODER |= (1U << 2*6);
	GPIOC -> BRR |= (1U << 6);

}

void nucleo_led_set(void){
	GPIOC -> ODR |= (1U << 6) ;
}

void nucleo_led_clear(void){
	GPIOC -> BRR |= (1U << 6);
}

void nucleo_led_toggle(void){
	GPIOC -> ODR ^= (1U << 6) ;
}

void nucleo_button_init(void){
	RCC -> IOPENR |= (1U << 5 );
	GPIOF -> MODER &= ~ (3U << 2*2);

}

int nucleo_button_read(void) {

	int a = ((GPIOF -> IDR >> 2 ) & 0x01);
		if (a) return 0;
		else return 1;

}

void nucleo_ext_led_init(void){

	RCC -> IOPENR |= (1U <<0 );
	GPIOB -> MODER &= ~ (3U << 2*4);
```

```
	GPIOB -> MODER |= (1U << 2*4);
	GPIOB -> BRR |= (1U << 4);

}
void nucleo_ext_led_set(void){
	GPIOB -> ODR |= (1U <<4 );
}
void nucleo_ext_led_clear(void){
	GPIOB -> BRR |= (1U <<4 );
}
void nucleo_ext_led_toggle(void){
	GPIOB -> ODR ^= (1U <<4 );
}
void timer1_init(void) {

	RCC -> APBENR2 |= (1U << 11 );
	TIM1 -> CR1 = 0;
	TIM1 -> CR1 |= (1 << 7 );
	TIM1 -> CNT = 0;
	TIM1 -> PSC = 999;
	TIM1 -> ARR = 16000;
	//TIM1 -> DIER |= (1 << 0 );
	TIM1 -> CR1 |= (1 << 0 );
	NVIC_SetPriority(TIM1_BRK_UP_TRG_COM_IRQn , 1);
	NVIC_EnableIRQ(TIM1_BRK_UP_TRG_COM_IRQn);
}
void timer1_s(void){
	TIM1 -> PSC = 999;
	TIM1 -> ARR = 16000;
}
void timer2_s(void){
	TIM2 -> PSC = 999;
	TIM2 -> ARR = 16000;
}

void timer1_s2(void){
	TIM1 -> PSC = 999;
	TIM1 -> ARR = 8000;
}
void timer2_s2(void){
	TIM2 -> PSC = 999;
	TIM2 -> ARR = 8000;
}

void timer1_s3(void){
	TIM1 -> PSC = 999;
	TIM1 -> ARR = 1600;
}
void timer2_s3(void){
	TIM2 -> PSC = 999;
	TIM2 -> ARR = 1600;
}

void timer1_s4(void){
	TIM1 -> PSC = 999;
	TIM1 -> ARR = 160;
}
void timer2_s4(void){
	TIM2 -> PSC = 999;
```

```
	TIM2 -> ARR = 160;
}

void timer1_s5(void){
	TIM1 -> PSC = 999;
	TIM1 -> ARR = 16;
}
void timer2_s5(void){
	TIM2 -> PSC = 999;
	TIM2 -> ARR = 16;
}

void timer2_init(void) {
	SystemCoreClockUpdate();
	RCC -> APBENR1 |= (1U << 0 );
	TIM2 -> CR1 = 0;
	TIM2 -> CR1 |= (1 << 7 );
	TIM2 -> CNT = 0;
	TIM2 -> DIER |= (1 << 0 );
	TIM2 -> CR1 |= (1 << 0 );
	NVIC_SetPriority(TIM2_IRQn  , 0);
	NVIC_EnableIRQ(TIM2_IRQn );
}


void systic_init(void){
	SysTick->CTRL |= SysTick_CTRL_ENABLE_Msk;
	SysTick->VAL=0;
	SysTick->CTRL |= SysTick_CTRL_TICKINT_Msk;
	NVIC_EnableIRQ(SysTick_IRQn);
	NVIC_SetPriority (SysTick_IRQn,0);
}


void timer1_statclear(void){
	TIM1 -> SR &= ~ (1U << 0);
}
void timer2_statclear(void){
	TIM2 -> SR &= ~ (1U << 0);
}

void systick_delay_ms() {
	SystemCoreClockUpdate();
	SysTick_Config((SystemCoreClock / KILO));
}
void systick_delay_s(){
	SystemCoreClockUpdate();
	SysTick_Config((SystemCoreClock / MEGA));
}

// Project functions

// GPIO Functions
// Input Init
void Init_PA0_Input(){
    RCC->IOPENR |= (1U << 0);
    GPIOA->MODER &= ~(3U << 2*0);

}
```

```c
void Init_PA1_Input(){
    RCC->IOPENR |= (1U << 0);
    GPIOA->MODER &= ~(3U << 2*1);

}
void Init_PA2_Input(){
    RCC->IOPENR |= (1U << 0);
    GPIOA->MODER &= ~(3U << 2*2);

}
void Init_PA3_Input(){
    RCC->IOPENR |= (1U << 0);
    GPIOA->MODER &= ~(3U << 2*3);

}
void Init_PA4_Input(){
    RCC->IOPENR |= (1U << 0);
    GPIOA->MODER &= ~(3U << 2*4);

}
void Init_PA5_Input(){
    RCC->IOPENR |= (1U << 0);
    GPIOA->MODER &= ~(3U << 2*5);

}
void Init_PA6_Input(){
    RCC->IOPENR |= (1U << 0);
    GPIOA->MODER &= ~(3U << 2*6);

}
void Init_PA7_Input(){
    RCC->IOPENR |= (1U << 0);
    GPIOA->MODER &= ~(3U << 2*7);

}
void Init_PA8_Input(){
    RCC->IOPENR |= (1U << 0);
    GPIOA->MODER &= ~(3U << 2*8);

}
void Init_PA9_Input(){
    RCC->IOPENR |= (1U << 0);
    GPIOA->MODER &= ~(3U << 2*9);

}
void Init_PA10_Input(){
    RCC->IOPENR |= (1U << 0);
    GPIOA->MODER &= ~(3U << 2*10);

}
void Init_PA11_Input(){
    RCC->IOPENR |= (1U << 0);
    GPIOA->MODER &= ~(3U << 2*11);

}
void Init_PA12_Input(){
    RCC->IOPENR |= (1U << 0);
    GPIOA->MODER &= ~(3U << 2*12);
```

```
}
void Init_PA13_Input(){
    RCC->IOPENR |= (1U << 0);
    GPIOA->MODER &= ~(3U << 2*13);

}
void Init_PA14_Input(){
    RCC->IOPENR |= (1U << 0);
    GPIOA->MODER &= ~(3U << 2*14);

}
void Init_PA15_Input(){
    RCC->IOPENR |= (1U << 0);
    GPIOA->MODER &= ~(3U << 2*15);

}
void Init_PB0_Input(){
    RCC->IOPENR |= (1U << 1);
    GPIOA->MODER &= ~(3U << 2*0);

}
void Init_PB1_Input(){
    RCC->IOPENR |= (1U << 1);
    GPIOA->MODER &= ~(3U << 2*1);

}
void Init_PB2_Input(){
    RCC->IOPENR |= (1U << 1);
    GPIOA->MODER &= ~(3U << 2*2);

}
void Init_PB3_Input(){
    RCC->IOPENR |= (1U << 1);
    GPIOA->MODER &= ~(3U << 2*3);

}
void Init_PB4_Input(){
    RCC->IOPENR |= (1U << 1);
    GPIOA->MODER &= ~(3U << 2*4);

}
void Init_PB5_Input(){
    RCC->IOPENR |= (1U << 1);
    GPIOA->MODER &= ~(3U << 2*5);

}
void Init_PB6_Input(){
    RCC->IOPENR |= (1U << 1);
    GPIOA->MODER &= ~(3U << 2*6);

}
void Init_PB7_Input(){
    RCC->IOPENR |= (1U << 1);
    GPIOA->MODER &= ~(3U << 2*7);

}
void Init_PB8_Input(){
    RCC->IOPENR |= (1U << 1);
    GPIOA->MODER &= ~(3U << 2*8);
```

```
}
void Init_PB9_Input(){
    RCC->IOPENR |= (1U << 1);
    GPIOA->MODER &= ~(3U << 2*9);

}
void Init_PB10_Input(){
    RCC->IOPENR |= (1U << 1);
    GPIOA->MODER &= ~(3U << 2*10);

}
void Init_PB11_Input(){
    RCC->IOPENR |= (1U << 1);
    GPIOA->MODER &= ~(3U << 2*11);

}
void Init_PB12_Input(){
    RCC->IOPENR |= (1U << 1);
    GPIOA->MODER &= ~(3U << 2*12);

}
void Init_PB13_Input(){
    RCC->IOPENR |= (1U << 1);
    GPIOA->MODER &= ~(3U << 2*13);

}
void Init_PB14_Input(){
    RCC->IOPENR |= (1U << 1);
    GPIOA->MODER &= ~(3U << 2*14);

}
void Init_PB15_Input(){
    RCC->IOPENR |= (1U << 1);
    GPIOA->MODER &= ~(3U << 2*15);

}
//Input Functions

//Output Init
void Init_PB0_Output(){
    RCC->IOPENR |= (1U << 1);
    GPIOB->MODER &= ~(3U << 2*0);
    GPIOB->MODER |= (1U << 2*0);
}
void Init_PB1_Output(){
    RCC->IOPENR |= (1U << 1);
    GPIOB->MODER &= ~(3U << 2*1);
    GPIOB->MODER |= (1U << 2*1);
}
void Init_PB2_Output(){
    RCC->IOPENR |= (1U << 1);
    GPIOB->MODER &= ~(3U << 2*2);
    GPIOB->MODER |= (1U << 2*2);
}
void Init_PB3_Output(){
    RCC->IOPENR |= (1U << 1);
    GPIOB->MODER &= ~(3U << 2*3);
    GPIOB->MODER |= (1U << 2*3);
```

```
}
void Init_PB4_Output(){
    RCC->IOPENR |= (1U << 1);
    GPIOB->MODER &= ~(3U << 2*4);
    GPIOB->MODER |= (1U << 2*4);
}
void Init_PB5_Output(){
    RCC->IOPENR |= (1U << 1);
    GPIOB->MODER &= ~(3U << 2*5);
    GPIOB->MODER |= (1U << 2*5);
}
void Init_PB6_Output(){
    RCC->IOPENR |= (1U << 1);
    GPIOB->MODER &= ~(3U << 2*6);
    GPIOB->MODER |= (1U << 2*6);
}
void Init_PB7_Output(){
    RCC->IOPENR |= (1U << 1);
    GPIOB->MODER &= ~(3U << 2*7);
    GPIOB->MODER |= (1U << 2*7);
}
void Init_PB8_Output(){
    RCC->IOPENR |= (1U << 1);
    GPIOB->MODER &= ~(3U << 2*8);
    GPIOB->MODER |= (1U << 2*8);
}
void Init_PB9_Output(){
    RCC->IOPENR |= (1U << 1);
    GPIOB->MODER &= ~(3U << 2*9);
    GPIOB->MODER |= (1U << 2*9);
}
void Init_PB10_Output(){
    RCC->IOPENR |= (1U << 1);
    GPIOB->MODER &= ~(3U << 2*10);
    GPIOB->MODER |= (1U << 2*10);
}
void Init_PB11_Output(){
    RCC->IOPENR |= (1U << 1);
    GPIOB->MODER &= ~(3U << 2*11);
    GPIOB->MODER |= (1U << 2*11);
}
void Init_PB12_Output(){
    RCC->IOPENR |= (1U << 1);
    GPIOB->MODER &= ~(3U << 2*12);
    GPIOB->MODER |= (1U << 2*12);
}
void Init_PB13_Output(){
    RCC->IOPENR |= (1U << 1);
    GPIOB->MODER &= ~(3U << 2*13);
    GPIOB->MODER |= (1U << 2*13);
}
void Init_PB14_Output(){
    RCC->IOPENR |= (1U << 1);
    GPIOB->MODER &= ~(3U << 2*14);
    GPIOB->MODER |= (1U << 2*14);
}
void Init_PB15_Output(){
    RCC->IOPENR |= (1U << 1);
    GPIOB->MODER &= ~(3U << 2*15);
```

```
    GPIOB->MODER |= (1U << 2*15);
}
void Init_PA0_Output(){
    RCC->IOPENR |= (1U << 0);
    GPIOB->MODER &= ~(3U << 2*0);
    GPIOB->MODER |= (1U << 2*0);
}
void Init_PA1_Output(){
    RCC->IOPENR |= (1U << 0);
    GPIOB->MODER &= ~(3U << 2*1);
    GPIOB->MODER |= (1U << 2*1);
}
void Init_PA2_Output(){
    RCC->IOPENR |= (1U << 0);
    GPIOB->MODER &= ~(3U << 2*2);
    GPIOB->MODER |= (1U << 2*2);
}
void Init_PA3_Output(){
    RCC->IOPENR |= (1U << 0);
    GPIOB->MODER &= ~(3U << 2*3);
    GPIOB->MODER |= (1U << 2*3);
}
void Init_PA4_Output(){
    RCC->IOPENR |= (1U << 0);
    GPIOB->MODER &= ~(3U << 2*4);
    GPIOB->MODER |= (1U << 2*4);
}
void Init_PA5_Output(){
    RCC->IOPENR |= (1U << 0);
    GPIOB->MODER &= ~(3U << 2*5);
    GPIOB->MODER |= (1U << 2*5);
}
void Init_PA6_Output(){
    RCC->IOPENR |= (1U << 0);
    GPIOB->MODER &= ~(3U << 2*6);
    GPIOB->MODER |= (1U << 2*6);
}
void Init_PA7_Output(){
    RCC->IOPENR |= (1U << 0);
    GPIOB->MODER &= ~(3U << 2*7);
    GPIOB->MODER |= (1U << 2*7);
}
void Init_PA8_Output(){
    RCC->IOPENR |= (1U << 0);
    GPIOB->MODER &= ~(3U << 2*8);
    GPIOB->MODER |= (1U << 2*8);
}
void Init_PA9_Output(){
    RCC->IOPENR |= (1U << 0);
    GPIOB->MODER &= ~(3U << 2*9);
    GPIOB->MODER |= (1U << 2*9);
}
void Init_PA10_Output(){
    RCC->IOPENR |= (1U << 0);
    GPIOA->MODER &= ~(3U << 2*10);
    GPIOA->MODER |= (1U << 2*10);
}
void Init_PA11_Output(){
    RCC->IOPENR |= (1U << 0);
```

```
    GPIOB->MODER &= ~(3U << 2*11);
    GPIOB->MODER |= (1U << 2*11);
}
void Init_PA12_Output(){
    RCC->IOPENR |= (1U << 0);
    GPIOB->MODER &= ~(3U << 2*12);
    GPIOB->MODER |= (1U << 2*12);
}
void Init_PA13_Output(){
    RCC->IOPENR |= (1U << 0);
    GPIOB->MODER &= ~(3U << 2*13);
    GPIOB->MODER |= (1U << 2*13);
}
void Init_PA14_Output(){
    RCC->IOPENR |= (1U << 0);
    GPIOB->MODER &= ~(3U << 2*14);
    GPIOB->MODER |= (1U << 2*14);
}
void Init_PA15_Output(){
    RCC->IOPENR |= (1U << 0);
    GPIOB->MODER &= ~(3U << 2*15);
    GPIOB->MODER |= (1U << 2*15);
}
//output Functions
void Set_PB0(){
    GPIOB->ODR |= (1U<<0) ;
}
void Set_PB1(){
    GPIOB->ODR |= (1U<<1) ;
}
void Set_PB2(){
    GPIOB->ODR |= (1U<<2) ;
}
void Set_PB3(){
    GPIOB->ODR |= (1U<<3) ;
}
void Set_PB4(){
    GPIOB->ODR |= (1U<<4) ;
}
void Set_PB5(){
    GPIOB->ODR |= (1U<<5) ;
}
void Set_PB6(){
    GPIOB->ODR |= (1U<<6) ;
}
void Set_PB7(){
    GPIOB->ODR |= (1U<<7) ;
}
void Set_PB8(){
    GPIOB->ODR |= (1U<<8) ;
}
void Set_PB9(){
    GPIOB->ODR |= (1U<<9) ;
}
void Set_PB10(){
    GPIOB->ODR |= (1U<<10) ;
}
void Set_PB11(){
    GPIOB->ODR |= (1U<<11) ;
```

```
}
void Set_PB12(){
    GPIOB->ODR |= (1U<<12) ;
}
void Set_PB13(){
    GPIOB->ODR |= (1U<<13) ;
}
void Set_PB14(){
    GPIOB->ODR |= (1U<<14) ;
}
void Set_PB15(){
    GPIOB->ODR |= (1U<<15) ;
}
void Set_PA0(){
    GPIOA->ODR |= (1U<<0) ;
}
void Set_PA1(){
    GPIOA->ODR |= (1U<<1) ;
}
void Set_PA2(){
    GPIOA->ODR |= (1U<<2) ;
}
void Set_PA3(){
    GPIOA->ODR |= (1U<<3) ;
}
void Set_PA4(){
    GPIOA->ODR |= (1U<<4) ;
}
void Set_PA5(){
    GPIOA->ODR |= (1U<<5) ;
}
void Set_PA6(){
    GPIOA->ODR |= (1U<<6) ;
}
void Set_PA7(){
    GPIOA->ODR |= (1U<<7) ;
}
void Set_PA8(){
    GPIOA->ODR |= (1U<<8) ;
}
void Set_PA9(){
    GPIOA->ODR |= (1U<<9) ;
}
void Set_PA10(){
    GPIOA->ODR |= (1U<<10) ;
}
void Set_PA11(){
    GPIOA->ODR |= (1U<<11) ;
}
void Set_PA12(){
    GPIOA->ODR |= (1U<<12) ;
}
void Set_PA13(){
    GPIOA->ODR |= (1U<<13) ;
}
void Set_PA14(){
    GPIOA->ODR |= (1U<<14) ;
}
void Set_PA15(){
```

```
    GPIOA->ODR |= (1U<<15) ;
}


void Clear_PB0(){
    GPIOB->ODR &= ~(1U<<0) ;
}
void Clear_PB1(){
    GPIOB->ODR &= ~(1U<<1) ;
}
void Clear_PB2(){
    GPIOB->ODR &= ~(1U<<2) ;
}
void Clear_PB3(){
    GPIOB->ODR &= ~(1U<<3) ;
}
void Clear_PB4(){
    GPIOB->ODR &= ~(1U<<4) ;
}
void Clear_PB5(){
    GPIOB->ODR &= ~(1U<<5) ;
}
void Clear_PB6(){
    GPIOB->ODR &= ~(1U<<6) ;
}
void Clear_PB7(){
    GPIOB->ODR &= ~(1U<<7) ;
}
void Clear_PB8(){
    GPIOB->ODR &= ~(1U<<8) ;
}
void Clear_PB9(){
    GPIOB->ODR &= ~(1U<<9) ;
}
void Clear_PB10(){
    GPIOB->ODR &= ~(1U<<10) ;
}
void Clear_PB11(){
    GPIOB->ODR &= ~(1U<<11) ;
}
void Clear_PB12(){
    GPIOB->ODR &= ~(1U<<12) ;
}
void Clear_PB13(){
    GPIOB->ODR &= ~(1U<<13) ;
}
void Clear_PB14(){
    GPIOB->ODR &= ~(1U<<14) ;
}
void Clear_PB15(){
    GPIOB->ODR &= ~(1U<<15) ;
}
void Clear_PA0(){
    GPIOA->ODR &= ~(1U<<0) ;
}
void Clear_PA1(){
    GPIOA->ODR &= ~(1U<<1) ;
}
void Clear_PA2(){
```

```
    GPIOA->ODR &= ~(1U<<2) ;
}
void Clear_PA3(){
    GPIOA->ODR &= ~(1U<<3) ;
}
void Clear_PA4(){
    GPIOA->ODR &= ~(1U<<4) ;
}
void Clear_PA5(){
    GPIOA->ODR &= ~(1U<<5) ;
}
void Clear_PA6(){
    GPIOA->ODR &= ~(1U<<6) ;
}
void Clear_PA7(){
    GPIOA->ODR &= ~(1U<<7) ;
}
void Clear_PA8(){
    GPIOA->ODR &= ~(1U<<8) ;
}
void Clear_PA9(){
    GPIOA->ODR &= ~(1U<<9) ;
}
void Clear_PA10(){
    GPIOA->ODR &= ~(1U<<10) ;
}
void Clear_PA11(){
    GPIOA->ODR &= ~(1U<<11) ;
}
void Clear_PA12(){
    GPIOA->ODR &= ~(1U<<12) ;
}
void Clear_PA13(){
    GPIOA->ODR &= ~(1U<<13) ;
}
void Clear_PA14(){
    GPIOA->ODR &= ~(1U<<14) ;
}
void Clear_PA15(){
    GPIOA->ODR &= ~(1U<<15) ;
}



void Toggle_PB0(){

      GPIOB->ODR ^= (1U<<0) ;
}
void Toggle_PB1(){

      GPIOB->ODR ^= (1U<<1) ;
}
void Toggle_PB2(){

      GPIOB->ODR ^= (1U<<2) ;
}
void Toggle_PB3(){

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

```
}
void Toggle_PB4(){

	GPIOB->ODR ^= (1U<<4) ;
}
void Toggle_PB5(){

	GPIOB->ODR ^= (1U<<5) ;
}
void Toggle_PB6(){

	GPIOB->ODR ^= (1U<<6) ;
}
void Toggle_PB7(){

	GPIOB->ODR ^= (1U<<7) ;
}
void Toggle_PB8(){

	GPIOB->ODR ^= (1U<<8) ;
}
void Toggle_PB9(){

	GPIOB->ODR ^= (1U<<9) ;
}
void Toggle_PB10(){

	GPIOB->ODR ^= (1U<<10) ;
}
void Toggle_PB11(){

	GPIOB->ODR ^= (1U<<11) ;
}
void Toggle_PB12(){

	GPIOB->ODR ^= (1U<<12) ;
}
void Toggle_PB13(){

	GPIOB->ODR ^= (1U<<13) ;
}
void Toggle_PB14(){

	GPIOB->ODR ^= (1U<<14) ;
}
void Toggle_PB15(){

	GPIOB->ODR ^= (1U<<15) ;
}
void Toggle_PA0(){

	GPIOA->ODR ^= (1U<<0) ;
}
void Toggle_PA1(){

	GPIOA->ODR ^= (1U<<1) ;
}
void Toggle_PA2(){
```

```
	GPIOA->ODR ^= (1U<<2) ;
}
void Toggle_PA3(){

	GPIOA->ODR ^= (1U<<3) ;
}
void Toggle_PA4(){

	GPIOA->ODR ^= (1U<<4) ;
}
void Toggle_PA5(){

	GPIOA->ODR ^= (1U<<5) ;
}
void Toggle_PA6(){

	GPIOA->ODR ^= (1U<<6) ;
}
void Toggle_PA7(){

	GPIOA->ODR ^= (1U<<7) ;
}
void Toggle_PA8(){

	GPIOA->ODR ^= (1U<<8) ;
}
void Toggle_PA9(){

	GPIOA->ODR ^= (1U<<9) ;
}
void Toggle_PA10(){

	GPIOA->ODR ^= (1U<<10) ;
}
void Toggle_PA11(){

	GPIOA->ODR ^= (1U<<11) ;
}
void Toggle_PA12(){

	GPIOA->ODR ^= (1U<<12) ;
}
void Toggle_PA13(){

	GPIOA->ODR ^= (1U<<13) ;
}
void Toggle_PA14(){

	GPIOA->ODR ^= (1U<<14) ;
}
void Toggle_PA15(){

	GPIOA->ODR ^= (1U<<15) ;
}
```

Nucleo.h

```
/*
 * nucleo.h
 *
 *  Created on: Nov 29, 2021
 *      Author: Deniz
 */

#ifndef NUCLEO_H_
#define NUCLEO_H_

// On-Board LED //

void nucleo_led_init();
void nucleo_led_set();
void nucleo_led_clear();
void nucleo_led_toggle();
void nucleo_ext_led_init();
void nucleo_ext_led_set();
void nucleo_ext_led_clear();
void nucleo_ext_led_toggle();

// Button Functions//

void nucleo_button_init();
int nucleo_button_read();
void nucleo_PA0_button_init();
int nucleo_PA0_button_read();
void nucleo_PA0_button_INT();
void nucleo_PA0_button_statclear();
//
// Timer interrupts

void timer1_init();
void timer2_init();
void timer2_s();
void systic_init();
void systick_delay_ms();
void systick_delay_s();
void timer1_interrupt();
void timer1_statclear();
void timer2_statclear(void);
void timer1_s();
void timer1_s2();
void timer1_s3();
void timer1_s4();
void timer1_s5();


// Project functions

// GPIO Functions
// Input Initialise
void Init_PA0_Input();
void Init_PA1_Input();
void Init_PA2_Input();
```

```
void Init_PA3_Input();
void Init_PA4_Input();
void Init_PA5_Input();
void Init_PA6_Input();
void Init_PA7_Input();
void Init_PA8_Input();
void Init_PA9_Input();
void Init_PA10_Input();
void Init_PA11_Input();
void Init_PA12_Input();
void Init_PA13_Input();
void Init_PA14_Input();
void Init_PA15_Input();
void Init_PB0_Input();
void Init_PB1_Input();
void Init_PB2_Input();
void Init_PB3_Input();
void Init_PB4_Input();
void Init_PB5_Input();
void Init_PB6_Input();
void Init_PB7_Input();
void Init_PB8_Input();
void Init_PB9_Input();
void Init_PB10_Input();
void Init_PB11_Input();
void Init_PB12_Input();
void Init_PB13_Input();
void Init_PB14_Input();
void Init_PB15_Input();
// Output Initialise
void Init_PB0_Output();
void Init_PB1_Output();
void Init_PB2_Output();
void Init_PB3_Output();
void Init_PB4_Output();
void Init_PB5_Output();
void Init_PB6_Output();
void Init_PB7_Output();
void Init_PB8_Output();
void Init_PB9_Output();
void Init_PB10_Output();
void Init_PB11_Output();
void Init_PB12_Output();
void Init_PB13_Output();
void Init_PB14_Output();
void Init_PB15_Output();
void Init_PA0_Output();
void Init_PA1_Output();
void Init_PA2_Output();
void Init_PA3_Output();
void Init_PA4_Output();
void Init_PA5_Output();
void Init_PA6_Output();
void Init_PA7_Output();
void Init_PA8_Output();
void Init_PA9_Output();
void Init_PA10_Output();
void Init_PA11_Output();
void Init_PA12_Output();
```

```
void Init_PA13_Output();
void Init_PA14_Output();
void Init_PA15_Output();
//


// Set Output
void Set_PB0();
void Set_PB1();
void Set_PB2();
void Set_PB3();
void Set_PB4();
void Set_PB5();
void Set_PB6();
void Set_PB7();
void Set_PB8();
void Set_PB9();
void Set_PB10();
void Set_PB11();
void Set_PB12();
void Set_PB13();
void Set_PB14();
void Set_PB15();
void Set_PA0();
void Set_PA1();
void Set_PA2();
void Set_PA3();
void Set_PA4();
void Set_PA5();
void Set_PA6();
void Set_PA7();
void Set_PA8();
void Set_PA9();
void Set_PA10();
void Set_PA11();
void Set_PA12();
void Set_PA13();
void Set_PA14();
void Set_PA15();


void Clear_PA0();
void Clear_PA1();
void Clear_PA2();
void Clear_PA3();
void Clear_PA4();
void Clear_PA5();
void Clear_PA6();
void Clear_PA7();
void Clear_PA8();
void Clear_PA9();
void Clear_PA10();
void Clear_PA11();
void Clear_PA12();
void Clear_PA13();
void Clear_PA14();
void Clear_PA15();
void Clear_PB4();
void Clear_PB0();
```

```
void Clear_PB1();
void Clear_PB2();
void Clear_PB3();
void Clear_PB4();
void Clear_PB5();
void Clear_PB6();
void Clear_PB7();
void Clear_PB8();
void Clear_PB9();
void Clear_PB10();
void Clear_PB11();
void Clear_PB12();
void Clear_PB13();
void Clear_PB14();
void Clear_PB15();


void Toggle_PB0();
void Toggle_PB1();
void Toggle_PB2();
void Toggle_PB3();
void Toggle_PB4();
void Toggle_PB5();
void Toggle_PB6();
void Toggle_PB7();
void Toggle_PB8();
void Toggle_PB9();
void Toggle_PB10();
void Toggle_PB11();
void Toggle_PB12();
void Toggle_PB13();
void Toggle_PB14();
void Toggle_PB15();
void Toggle_PA0();
void Toggle_PA1();
void Toggle_PA2();
void Toggle_PA3();
void Toggle_PA4();
void Toggle_PA5();
void Toggle_PA6();
void Toggle_PA7();
void Toggle_PA8();
void Toggle_PA9();
void Toggle_PA10();
void Toggle_PA11();
void Toggle_PA12();
void Toggle_PA13();
void Toggle_PA14();
void Toggle_PA15();
#endif /* NUCLEO_H_ */
```