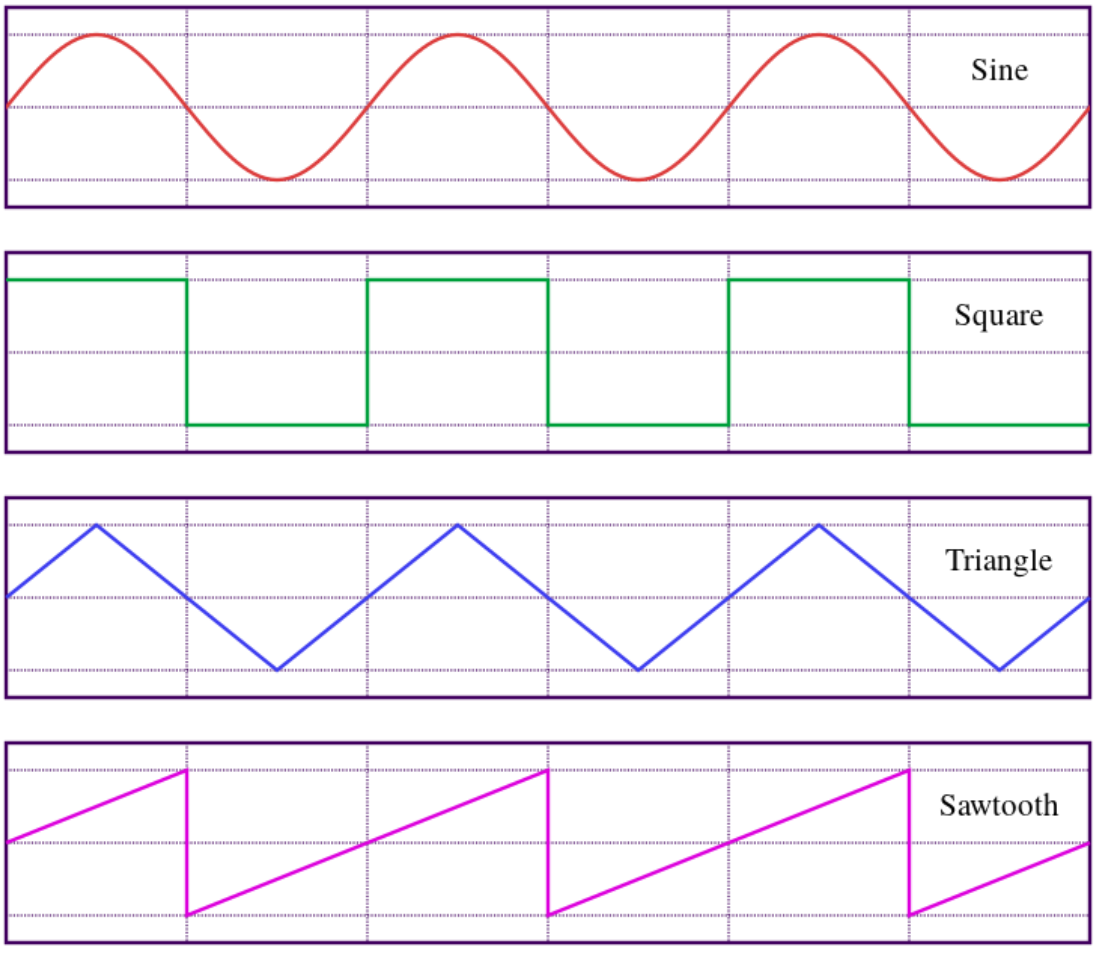
Function Generator

Francisco Irazaba

CPE 316 | Section 1 | Fall Quarter

Professor Hummel

10/28/24



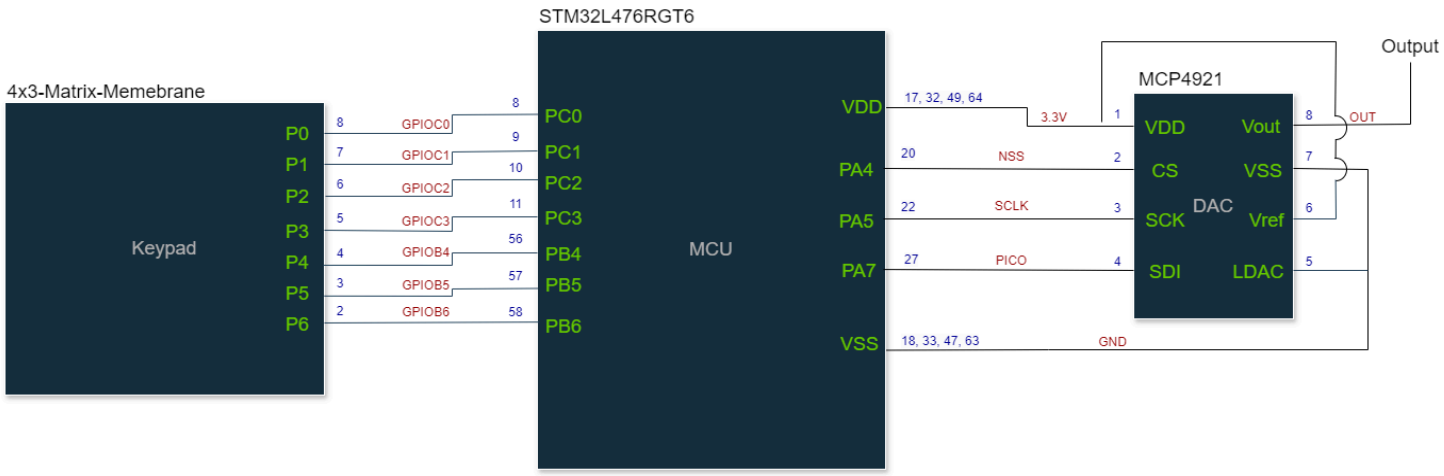
Behavior Description

A function generator is a device used in electronics to send signals in waves. In this project, the purpose was to design an operational function generator using a microcontroller. The function generator is designed to output four waveforms: sine, triangle, sawtooth, and square. Users can select and adjust the frequency of these signals between 100 Hz and 500 Hz. Additionally, the duty cycle can be customized for the square wave from 10% to 90% to create varied pulse widths. All waveform selection and adjustments are controlled via a simple keypad interface, allowing quick and easy manipulation of signal type, frequency, and duty cycle.

System Specification

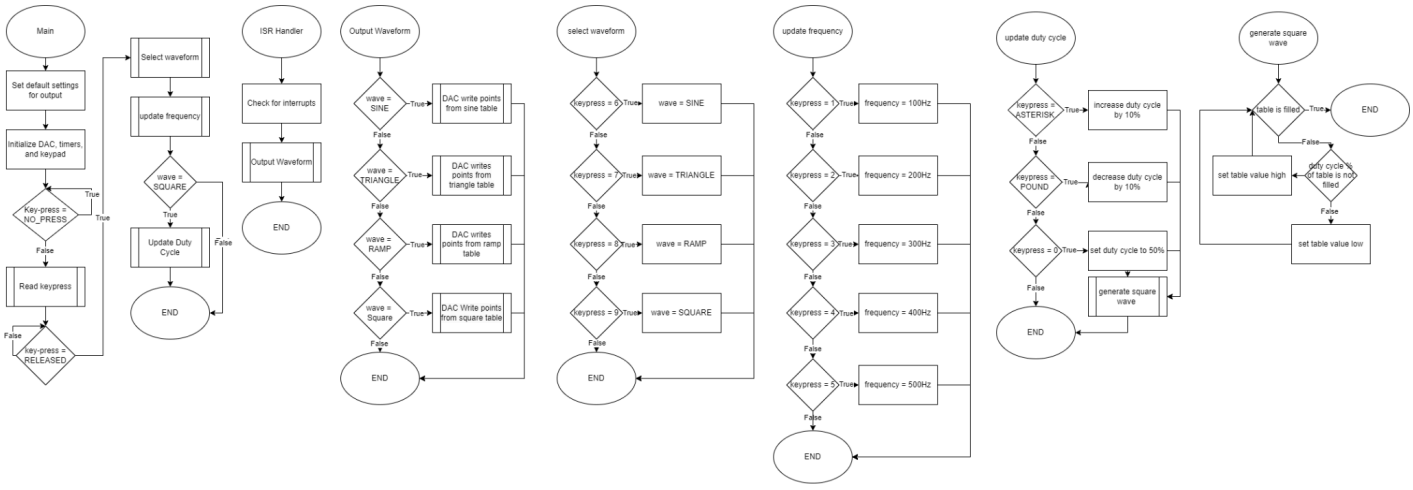
Specification	Details
Microcontroller	STM32L476RGT6
DAC	MCP4921
Power Supply Voltage	3.3V DC (USB or external adapter)
Power Consumption	11.72 mW
Clock Frequency (MCU)	40MHz
Output Frequency Range	100Hz to 500Hz
Amplitude	3V peak-to-peak (non-adjustable)
Output Waveforms	Sine, Triangle, Sawtooth, and Square,
DAC Resolution	12-bit
Resolution/Sampling Rate	521,739 samples/sec
Waveform Frequency Accuracy	±2.5 Hz
User Interface	4x3 Keypad
Connectors	BNC for signal output, USB Type-C for power and data connection
Supported Protocols	USB for remote control

System Schematic



Software Architecture

The program logic for the function generator begins by setting default values for variables used to output waves. The program then initializes the DAC, timer, and keypad. After initialization, the program waits for a key press. After a key is pressed, the keypad is read and the value is stored. Based on the key press, the frequency, waveform, or duty cycle (if square wave) is updated or remains the same. Every 1.92 microseconds, a timer interrupt is triggered to write data points from the DAC to the oscilloscope. The points selected and how they are written depend on the current settings, determined by the key press value or the default preset settings. If the key press changes the duty cycle, a new square wave lookup table is generated with adjusted high and low values. If the key press changes the frequency, the lookup table index is incremented by an amount proportional to the new frequency to control the output rate. If the key press changes the waveform, the lookup table corresponding to the selected waveform (sine, triangle, sawtooth, or square) is chosen for output.



Maximum Resolution Calculation

Variable	Calculation	Data/Result
ISR Time	Measured	1.14 µs
MCO clock pulses to execute ISR	Measured	44 MCO Clock Cycles
Minimal CCR1 value	Measured	57 MCO Clock Cycles
Overhead (in clock cycles)	57−44 =	13 MCO clock cycles
Overhead (in time)	$\frac{13}{40MHz}$ =	0.325 µs
Minimum ISR time	1.14µs + 0.325µs =	1.465µs
Max resolution	$\frac{1}{1.465\mu s}$ =	684,932 samples/sec

Appendices

References:

1. Dr LUT Wave Table Generator Ppelikan. *Dr Lut*, ppelikan.github.io/drlut/. Accessed 28 Oct. 2024.
2. *MCP4921/4922 12-Bit DAC with SPI Interface*, ww1.microchip.com/downloads/en/devicedoc/21897b.pdf. Accessed 28 Oct. 2024.
3. *4x4 Matrix Membrane Keypad (#27899)*, cdn.sparkfun.com/assets/f/f/a/5/0/DS-16038.pdf. Accessed 28 Oct. 2024.
4. *RM0351 Reference Manual*, www.st.com/resource/en/reference_manual/rm0351-stm32l47xxx-stm32l48xxx-stm32l49xxx-and-stm32l4axxx-advanced-armbased-32bit-mcus-stmicroelectronics.pdf. Accessed 28 Oct. 2024.

Source Code:

main.c

```
/* USER CODE BEGIN Header */
/**
 ******************************************************************************
 * @file           : main.c
```

```
 * @brief          : Main program body
 ******************************************************************************
 * @attention
 *
 * Copyright (c) 2024 STMicroelectronics.
 * All rights reserved.
 *
 * This software is licensed under terms that can be found in the LICENSE file
 * in the root directory of this software component.
 * If no LICENSE file comes with this software, it is provided AS-IS.
 *
 ******************************************************************************
 */
/* USER CODE END Header */
/* Includes ------------------------------------------------------------------*/
#include "main.h"
#include "DAC.h"
#include "timer.h"
#include "keypad.h"
#include "LUTs.h"
#define SINE 0
#define TRIANGLE 1
#define RAMP 2
#define SQUARE 3
#define ONE_HUNDRED 1
#define TWO_HUNDRED 2
#define THREE_HUNDRED 3
#define FOUR_HUNDRED 4
#define FIVE_HUNDRED 5
#define LUT_SIZE 1720 //size of lookup arrays
#define MAX_DUTY_CYCLE 9 //max duty cycle
#define MIN_DUTY_CYCLE 1 //min duty cycle
//prototype for interrupt function
void TIM2_IRQHandler(void);
//prototype for select waveform function
void select_waveform(int8_t keypress);
//prototype for output waveform function
void output_waveform();
//prototype for update frequency function
void update_freq(int8_t keypress);
//prototype for update duty cycle function
void update_duty_cycle(int8_t keypress);
// Global variable to store keypress value
volatile int8_t keypress_val = 9; //initialized to 9 for square wave
//Global variable to store frequency
volatile uint16_t freq = ONE_HUNDRED; //initialized to 1 for 100Hz
//Global variable to store duty cycle
volatile uint8_t duty_cycle = 5; //initialize to 50%
//Global variable for wave sel SINE = 0, TRIANGLE = 1, RAMP = 2, SQUARE = 3
volatile uint8_t wave_sel = SQUARE; //initialize to square
//Global variable for LUT index
volatile uint16_t lut_index = 0;
//Square wave array
uint16_t square[1720];
/* Private function prototypes -----------------------------------------------*/
void SystemClock_Config(void);
```

```c
/**
 * @brief  The application entry point.
 * @retval int
 */
int main(void)
{
 /* MCU Configuration--------------------------------------------------------*/
 /* Reset of all peripherals, Initializes the Flash interface and the Systick. */
 HAL_Init();
 /* Configure the system clock */
 SystemClock_Config();
 /* Initialize all configured peripherals */
 //initialize DAC, TIM2, and keypad
 DAC_init();
 TIM2_init();
 keypad_init();
 //configure square wave
 gen_square_wave(square, duty_cycle);//fill in square array
 /* Infinite loop */
 while (1)
 {
        int8_t keypad = NO_PRESS;
        while(keypad == NO_PRESS){ //wait for key-press
              keypad = keypad_func(); //read key-press
        }
        while(keypad_func() != NO_PRESS); //wait for key release
        keypress_val = keypad; //update global key-press to keypad value
        select_waveform(keypress_val); //select waveform to output
        update_freq(keypress_val); //update frequency if needed
        //check if waveform is square wave
        if (wave_sel == SQUARE){
              update_duty_cycle(keypress_val); //update duty cycle if needed
        }
 }
}
void TIM2_IRQHandler(void){
      //check for ARR flag
      if (TIM2->SR & TIM_SR_UIF){
            output_waveform(); //output according waveform
            TIM2->SR &= ~(TIM_SR_UIF);// clear update event interrupt flag
      }
}
/**
 * @brief Output Waveform
 * @retval None
 */
void output_waveform(){
      if(wave_sel == SINE){
            DAC_write(DAC_volt_conv(sine[lut_index])); //output sine wave
      }
      else if(wave_sel == TRIANGLE){
            DAC_write(DAC_volt_conv(triangle[lut_index])); //output triangle wave
      }
      else if(wave_sel == RAMP){
            DAC_write(DAC_volt_conv(ramp[lut_index])); //output ramp wave
      }
```

```
	else if(wave_sel == SQUARE){
		DAC_write(DAC_volt_conv(square[lut_index])); //output square wave
	}
	lut_index+=freq; //index by frequency
	if (lut_index >= LUT_SIZE) {
		lut_index = 0; // Loop back to the start
	}
	return;
}
/**
 * @brief Select Waveform
 * @retval None
 */
void select_waveform(int8_t keypress){
	if(keypress == 6){
		wave_sel = SINE; //sel sine wave
	}
	else if(keypress == 7){
		wave_sel = TRIANGLE; //sel triangle wave
	}
	else if(keypress == 8){
		wave_sel = RAMP; //sel ramp wave
	}
	else if(keypress == 9){
		wave_sel = SQUARE; //sel square wave
	}
	return;
}
/**
 * @brief Update Frequency
 * @retval None
 */
void update_freq(int8_t keypress){
	if(keypress == 1){
		freq = ONE_HUNDRED; //update frequency to 100Hz
	}
	else if(keypress == 2){
		freq = TWO_HUNDRED; //update frequency to 200Hz
	}
	else if(keypress == 3){
		freq = THREE_HUNDRED; //update frequency to 300Hz
		}
	else if(keypress == 4){
		freq = FOUR_HUNDRED; //update frequency to 400Hz
	}
	else if(keypress == 5){
		freq = FIVE_HUNDRED; //update frequency to 500Hz
	}
	else{
		return; //frequency stays the same
	}
}
/**
 * @brief Update Duty Cycle
 * @retval None
 */
```

```
void update_duty_cycle(int8_t keypress){
	//check if duty cycle needs to be increased by 10%
	if(keypress == POUND && duty_cycle != MAX_DUTY_CYCLE){
		duty_cycle += 1; //increase duty cycle by 10%
		gen_square_wave(square, duty_cycle); //change square wave array
	}
	//check if duty cycle needs to be decreased by 10%
	else if(keypress == ASTERISK && duty_cycle != MIN_DUTY_CYCLE){
		duty_cycle -= 1; //decrease duty cycle by 10%
		gen_square_wave(square, duty_cycle); // change square wave array
	}
	//check if duty cycle needs to be reset to 50%
	else if(keypress == 0){
		duty_cycle = 5; //set duty cycle to 50%
		gen_square_wave(square, duty_cycle); //change square wave array
	}
	//otherwise do nothing
	else{
		return;
	}
}
/**
 * @brief System Clock Configuration
 * @retval None
 */
void SystemClock_Config(void)
{
 RCC_OscInitTypeDef RCC_OscInitStruct = {0};
 RCC_ClkInitTypeDef RCC_ClkInitStruct = {0};
 /** Configure the main internal regulator output voltage
 */
 if (HAL_PWREx_ControlVoltageScaling(PWR_REGULATOR_VOLTAGE_SCALE1) != HAL_OK)
 {
   Error_Handler();
 }
 /** Initializes the RCC Oscillators according to the specified parameters
 * in the RCC_OscInitTypeDef structure.
 */
 RCC_OscInitStruct.OscillatorType = RCC_OSCILLATORTYPE_MSI;
 RCC_OscInitStruct.MSIState = RCC_MSI_ON;
 RCC_OscInitStruct.MSICalibrationValue = 0;
 RCC_OscInitStruct.MSIClockRange = RCC_MSIRANGE_6;
 RCC_OscInitStruct.PLL.PLLState = RCC_PLL_ON;
 RCC_OscInitStruct.PLL.PLLSource = RCC_PLLSOURCE_MSI;
 RCC_OscInitStruct.PLL.PLLM = 1;
 RCC_OscInitStruct.PLL.PLLN = 20;
 RCC_OscInitStruct.PLL.PLLP = RCC_PLLP_DIV7;
 RCC_OscInitStruct.PLL.PLLQ = RCC_PLLQ_DIV2;
 RCC_OscInitStruct.PLL.PLLR = RCC_PLLR_DIV2;
 if (HAL_RCC_OscConfig(&RCC_OscInitStruct) != HAL_OK)
 {
   Error_Handler();
 }
 /** Initializes the CPU, AHB and APB buses clocks
 */
 RCC_ClkInitStruct.ClockType = RCC_CLOCKTYPE_HCLK|RCC_CLOCKTYPE_SYSCLK
```

```
                                |RCC_CLOCKTYPE_PCLK1|RCC_CLOCKTYPE_PCLK2;
 RCC_ClkInitStruct.SYSCLKSource = RCC_SYSCLKSOURCE_PLLCLK;
 RCC_ClkInitStruct.AHBCLKDivider = RCC_SYSCLK_DIV1;
 RCC_ClkInitStruct.APB1CLKDivider = RCC_HCLK_DIV1;
 RCC_ClkInitStruct.APB2CLKDivider = RCC_HCLK_DIV1;
 if (HAL_RCC_ClockConfig(&RCC_ClkInitStruct, FLASH_LATENCY_2) != HAL_OK)
 {
   Error_Handler();
 }
}
/* USER CODE BEGIN 4 */
/* USER CODE END 4 */
/**
 * @brief  This function is executed in case of error occurrence.
 * @retval None
 */
void Error_Handler(void)
{
 /* USER CODE BEGIN Error_Handler_Debug */
 /* User can add his own implementation to report the HAL error return state */
 __disable_irq();
 while (1)
 {
 }
 /* USER CODE END Error_Handler_Debug */
}
#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(uint8_t *file, uint32_t line)
{
 /* USER CODE BEGIN 6 */
 /* 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) */
 /* USER CODE END 6 */
}
#endif /* USE_FULL_ASSERT */
```

timer.h

```
/*
* timer.h
*
*  Created on: Oct 23, 2024
*      Author: firaz
*/
#ifndef INC_TIMER_H_
#define INC_TIMER_H_
#define ARR_VAL 685 //value of ARR
#define CCR_VAL 0 //value of CCR
```

```
void TIM2_init(void);
#endif /* INC_TIMER_H_ */
```

timer.c

```
/*
* timer.c
*
*  Created on: Oct 23, 2024
*      Author: firaz
*/
#include "main.h"
#define ARR_VAL 77 //value of ARR
void TIM2_init(void){
	/*-------------------- Configure PA0 for GPIOA output --------------------*/
	//configure GPIOA clock
	RCC->AHB2ENR   |=  (RCC_AHB2ENR_GPIOAEN);
	//setup MODER for row output
	GPIOA->MODER &= ~(GPIO_MODER_MODE0);
	GPIOA->MODER |= (GPIO_MODER_MODE0_0);
	//set push-pull output type
	GPIOA->OTYPER &= ~(GPIO_OTYPER_OT0);
	//no PUPD
	GPIOA->PUPDR |= (GPIO_PUPDR_PUPD0_1);
	//set to high speed
	GPIOA->OSPEEDR |= (GPIO_OSPEEDR_OSPEED0_Msk);
	/*-------------------- Configure TIM2 and interrupt --------------------*/
	//configure TIM2 clock
	RCC->APB1ENR1 |= (RCC_APB1ENR1_TIM2EN);
	//set TIM2 to count up
	TIM2->CR1 &= ~(TIM_CR1_DIR);
	//set ARR clock
	TIM2->ARR = ARR_VAL;
	//enable update event interrupt in TIM2
	TIM2->DIER |= (TIM_DIER_UIE);
	//clear the flag before starting
	TIM2->SR &= ~(TIM_SR_UIF);
	//start timer
	TIM2->CR1 |= TIM_CR1_CEN;
	//enable interrupts globally
	__enable_irq();
	//enable TIM2 in NVIC
	NVIC->ISER[0] = (1 << TIM2_IRQn);
}
```

DAC.h

```
/*
* DAC.h
*
*  Created on: Oct 23, 2024
*      Author: firaz
*/
#ifndef INC_DAC_H_
```

```
#define INC_DAC_H_
void DAC_init(void);
void DAC_write(uint16_t transmission_data);
uint16_t DAC_volt_conv (uint16_t voltage_value);
#endif /* INC_DAC_H_ */
```

DAC.c

```
/*
* DAC.c
*
*  Created on: Oct 23, 2024
*      Author: firaz
*/
/**
 * @brief initialize the SPI peripheral to communicate with the DAC
 * @retval None
 */
#include "main.h"
#define AF5 5 //Alternate Function Register
#define MAX_12BIT_VAL 0xFFF //max 12-bit value
#define MAX_VOLT 3300 //max mV value
#define DEFAULT_DAC 0x3000 //default settings for first four bits of DAC
#define MASK 0xF000//mask for DAC initialization
void SystemClock_Config(void);
void DAC_init(void){
      //set clock for GPIOA and SPI
      RCC->AHB2ENR  |= (RCC_AHB2ENR_GPIOAEN);
      RCC->APB2ENR |= (RCC_APB2ENR_SPI1EN);
      /*---- Configure GPIOA PA4, PA5, PA7 for NSS, MOSI, and SCK from SPI ----*/
      //setup MODER in Alternate Function Mode
      GPIOA->MODER &= ~(GPIO_MODER_MODE4 | GPIO_MODER_MODE5 | GPIO_MODER_MODE7);
      GPIOA->MODER |= (GPIO_MODER_MODE4_1 | GPIO_MODER_MODE5_1 |
GPIO_MODER_MODE7_1);
      //set alternate function I/O to map to AF5
      GPIOA-> AFR[0] &= ~(GPIO_AFRL_AFSEL4 | GPIO_AFRL_AFSEL5 | GPIO_AFRL_AFSEL7);
      GPIOA-> AFR[0] |= (AF5 << GPIO_AFRL_AFSEL4_Pos | AF5 << GPIO_AFRL_AFSEL5_Pos
| AF5 << GPIO_AFRL_AFSEL7_Pos);
      //set push pull output type
      GPIOA->OTYPER  &=  ~(GPIO_OTYPER_OT4 | GPIO_OTYPER_OT5 | GPIO_OTYPER_OT7);
      //no PUPD
      GPIOA->PUPDR &= ~(GPIO_PUPDR_PUPD4 | GPIO_PUPDR_PUPD5 |GPIO_PUPDR_PUPD7);
      //set to high speed
      GPIOA->OSPEEDR |= (GPIO_OSPEEDR_OSPEED4 | GPIO_OSPEEDR_OSPEED5 |
GPIO_OSPEEDR_OSPEED7);
      /*---------------- Configure SPI to communicate with DAC --------------*/
      //set baud rate to 2MHz (4Mhz clock/2)
      SPI1->CR1 &= ~(SPI_CR1_BR);
      //set clock polarity to 0 when idle
      SPI1->CR1 &= ~(SPI_CR1_CPOL);
      //set clock phase to 0 to capture data on first transition
      SPI1->CR1 &= ~(SPI_CR1_CPHA);
```

```
	//set to FULLPLEX
	SPI1->CR1 &= ~(SPI_CR1_RXONLY);
	//set frame format to MSB first
	SPI1->CR1 &= ~(SPI_CR1_LSBFIRST);
	//disable CRC calculation
	SPI1->CR1 &= ~(SPI_CR1_CRCEN);
	//Disable SSM
	SPI1->CR1 &= ~(SPI_CR1_SSM);
	//set MCU as master
	SPI1->CR1 |= (SPI_CR1_MSTR);
	//set data size to 16-bit
	SPI1->CR2 &= ~(SPI_CR2_DS);
	SPI1->CR2 |= (SPI_CR2_DS);
	//enable SS output
	SPI1->CR2 |= (SPI_CR2_SSOE);
	//set frame format to Motorola mode
	SPI1->CR2 &= ~(SPI_CR2_FRF);
	//enable NSS pulse management
	SPI1->CR2 |= (SPI_CR2_NSSP);
	//set FIFO reception threshold to 16-bit
	SPI1->CR2 &= ~(SPI_CR2_FRXTH);
	//enable SPI
	SPI1->CR1 |= SPI_CR1_SPE;
}
/**
 * @brief write a 12-bit value to the DAC
 * @retval None
 */
void DAC_write(uint16_t transmission_data){
	uint16_t data_frame = transmission_data & ~(MASK); //Mask first 4 bits
	data_frame |= (DEFAULT_DAC);
	while(!(SPI1->SR & SPI_SR_TXE)){}; // Wait while transmission buffer is
full, check other SR flags(?)
	SPI1->DR = data_frame;// load data into data register
}
/**
 * @brief convert a voltage value into a 12-bit value to control the DAC
 * @retval uint16_t
 */
uint16_t DAC_volt_conv (uint16_t voltage_value){
	if (voltage_value > MAX_VOLT){
		return MAX_12BIT_VAL; //return max 12-bit value
	}
	else{
		return	(voltage_value*MAX_12BIT_VAL)/MAX_VOLT; //formula for
voltage conversion
	}
}
```

keypad.h

```
/*
* keypad.h
```

```
*
*  Created on: Oct 23, 2024
*      Author: firaz
*/
#ifndef INC_KEYPAD_H_
#define INC_KEYPAD_H_
#define SRC_KEYPAD_H_
#define NUM_OF_ROWS 4  // 4-row keypad
#define NUM_OF_COLS 3  // 3-column keypad
#define NO_PRESS (int8_t) -1//signfies no button was pressed
#define ASTERISK (int8_t) 10 //signifies asterisk keypress
#define POUND (int8_t) 11 //signifies pound keypress
void keypad_init(void);
int8_t keypad_func(void);
int8_t calculate_key(int8_t row,int8_t col);
#endif /* INC_KEYPAD_H_ */
```

keypad.c

```
/*
* keypad.c
*
*  Created on: Oct 23, 2024
*      Author: firaz
*/
#include "main.h"
#define SRC_KEYPAD_H_
#define NUM_OF_ROWS 4  // 4-row keypad
#define NUM_OF_COLS 3  // 3-column keypad
#define NO_PRESS (int8_t) -1//signfies no button was pressed
#define ASTERISK (int8_t) 10 //signifies asterisk keypress
#define POUND (int8_t) 11 //signifies pound keypress
/**
 * @brief: function to initialize keypad ports and set columns to 1
 * @retval: None
 */
void keypad_init(void){
        //set clock for GPIOA, GPIOB, and GPIOC
        RCC->AHB2ENR   |=  (RCC_AHB2ENR_GPIOBEN | RCC_AHB2ENR_GPIOCEN);
        /*--------------- Configure PB4-PB6 for GPIOB column input -----------*/
        //setup MODER for columns input
        GPIOB->MODER &= ~(GPIO_MODER_MODE4 | GPIO_MODER_MODE5 | GPIO_MODER_MODE6);
        //setup pull down resistor to avoid floating
        GPIOB->PUPDR &= ~(GPIO_PUPDR_PUPD4 |GPIO_PUPDR_PUPD5 | GPIO_PUPDR_PUPD6);
        GPIOB->PUPDR |= (GPIO_PUPDR_PUPD4_1 |GPIO_PUPDR_PUPD5_1 |
GPIO_PUPDR_PUPD6_1);
        /*---------------- Configure PC0-PC3 for GPIOC row output -------------*/
        //setup MODER for row output
        GPIOC->MODER &= ~(GPIO_MODER_MODE0 | GPIO_MODER_MODE1 | GPIO_MODER_MODE2 |
GPIO_MODER_MODE3);
        GPIOC->MODER |= (GPIO_MODER_MODE0_0 | GPIO_MODER_MODE1_0 |
GPIO_MODER_MODE2_0 | GPIO_MODER_MODE3_0);
        //set push pull output type
        GPIOC->OTYPER  &=  ~(GPIO_OTYPER_OT0 | GPIO_OTYPER_OT1 | GPIO_OTYPER_OT2 |
GPIO_OTYPER_OT3);
```

```
        //no PUPD
        GPIOC->PUPDR |= (GPIO_PUPDR_PUPD0_1 |GPIO_PUPDR_PUPD1_1 |
GPIO_PUPDR_PUPD2_1 | GPIO_PUPDR_PUPD3_1);
        //set to high speed
        GPIOC->OSPEEDR |= (GPIO_OSPEEDR_OSPEED0_Msk | GPIO_OSPEEDR_OSPEED1_Msk
                    | GPIO_OSPEEDR_OSPEED2_Msk | GPIO_OSPEEDR_OSPEED3_Msk);
        /*------------------- Initialize rows-------------------*/
        //set rows to 1
        GPIOC->ODR |= (GPIO_ODR_OD0 | GPIO_ODR_OD1 | GPIO_ODR_OD2 | GPIO_ODR_OD3);
}
/**
 * @brief Helper function to calculate the key for keypad_func
 * @retval int8_t
 */
int8_t calculate_key(int8_t row,int8_t col){
      //2D array to represent keypad
      int8_t keypad[NUM_OF_ROWS][NUM_OF_COLS] = {
          {1, 2, 3},
          {4, 5, 6},
          {7, 8, 9},
          {ASTERISK, 0, POUND}
      };
            return keypad[row][col];  // Return the character for the pressed key
}
/**
 * @brief: function to check for key-press and return key value if pressed
 * @retval: int8_t
 */
int8_t keypad_func(void){
      int8_t pressed_row = NO_PRESS;
   int8_t pressed_col = NO_PRESS;
   int16_t idr_value = GPIOB->IDR;
   //read cols
   idr_value &= (GPIO_IDR_ID4  | GPIO_IDR_ID5 | GPIO_IDR_ID6);
   if(!((idr_value == 0b10000) || (idr_value == 0b100000) || (idr_value ==
0b1000000))){
      //set rows back to zero for next press
      GPIOC->ODR |= (GPIO_ODR_OD0 | GPIO_ODR_OD1 | GPIO_ODR_OD2 | GPIO_ODR_OD3);
      return NO_PRESS;
   }
   //cycle through rows to determine key-press
   for(int row = 0; row < NUM_OF_ROWS; row++){
       GPIOC->ODR &= ~(GPIO_ODR_OD0 | GPIO_ODR_OD1 | GPIO_ODR_OD2 | GPIO_ODR_OD3);
       GPIOC->ODR |= (1<<row);
       idr_value = GPIOB->IDR;
       //checks to see if we get correct idr, stores row, and associated col based
on col idr
       if(idr_value & GPIO_IDR_ID4 ){ //check column 0
           pressed_col = 0;
           pressed_row = row;
           break;
       }
       if(idr_value & GPIO_IDR_ID5 ){ //check column 1
           pressed_col = 1;
           pressed_row = row;
           break;
```

```
        }
        if(idr_value & GPIO_IDR_ID6){ //check column 3
            pressed_col = 2;
            pressed_row = row;
            break;
        }
    }
    if(pressed_row == NO_PRESS || pressed_col == NO_PRESS){
       //set rows back to zero for next press
       GPIOC->ODR |= (GPIO_ODR_OD0 | GPIO_ODR_OD1 | GPIO_ODR_OD2 | GPIO_ODR_OD3);
       return NO_PRESS;
    }
    else{
       //set rows back to zero for next press
       GPIOC->ODR |= (GPIO_ODR_OD0 | GPIO_ODR_OD1 | GPIO_ODR_OD2 | GPIO_ODR_OD3);
       return calculate_key(pressed_row, pressed_col);
    }
}
```

LUTs.h

```
/*
* LULtables.h
*
*  Created on: Oct 23, 2024
*      Author: firaz
*/
#ifndef INC_LUTS_H_
#define INC_LUTS_H_
extern const uint16_t sine[1720];
extern const uint16_t ramp[1720];
extern const uint16_t triangle[1720];
void gen_square_wave(uint16_t *square_array, uint8_t dc);
#endif /* INC_LUTS_H_ */
```

LUTs.c

```
/*
* LULtables.c
*
*  Created on: Oct 23, 2024
*      Author: firaz
*/
#include "main.h"
#define VOLT_HIGH (uint16_t)2850 //3V high voltage for duty cycle
```

```
#define VOLT_LOW (uint16_t)0 //low voltage for duty cycle

/**
 * @brief Generate Square Wave
 * @retval None
 */
void gen_square_wave(uint16_t *square_array, uint8_t dc){
      for (uint16_t i = 0; i < 1720; i++) {
            if (i < (1720*dc)/10) {
                  square_array[i] = VOLT_HIGH; // Set to high voltage
            } else {
                  square_array[i] = VOLT_LOW; // Set to low voltage
            }
      }
}

/** Generated using Dr LUT - Free Lookup Table Generator
 * https://github.com/ppelikan/drlut
 **/
// Formula: sin(2*pi*t/T)
const uint16_t sine[1720] = {
1470, 1475, 1481, 1486, 1491, 1497, 1502, 1508,
1513, 1518, 1524, 1529, 1534, 1540, 1545, 1551,
1556, 1561, 1567, 1572, 1577, 1583, 1588, 1593,
1599, 1604, 1609, 1615, 1620, 1625, 1631, 1636,
1641, 1647, 1652, 1657, 1663, 1668, 1673, 1679,
1684, 1689, 1695, 1700, 1705, 1711, 1716, 1721,
1726, 1732, 1737, 1742, 1748, 1753, 1758, 1763,
1769, 1774, 1779, 1784, 1790, 1795, 1800, 1805,
1811, 1816, 1821, 1826, 1831, 1837, 1842, 1847,
1852, 1857, 1863, 1868, 1873, 1878, 1883, 1888,
1894, 1899, 1904, 1909, 1914, 1919, 1924, 1929,
1934, 1940, 1945, 1950, 1955, 1960, 1965, 1970,
1975, 1980, 1985, 1990, 1995, 2000, 2005, 2010,
2015, 2020, 2025, 2030, 2035, 2040, 2045, 2050,
2055, 2060, 2065, 2070, 2074, 2079, 2084, 2089,
2094, 2099, 2104, 2109, 2113, 2118, 2123, 2128,
2133, 2137, 2142, 2147, 2152, 2156, 2161, 2166,
2171, 2175, 2180, 2185, 2189, 2194, 2199, 2203,
2208, 2213, 2217, 2222, 2227, 2231, 2236, 2240,
2245, 2249, 2254, 2259, 2263, 2268, 2272, 2277,
2281, 2286, 2290, 2294, 2299, 2303, 2308, 2312,
2317, 2321, 2325, 2330, 2334, 2338, 2343, 2347,
2351, 2356, 2360, 2364, 2368, 2373, 2377, 2381,
2385, 2390, 2394, 2398, 2402, 2406, 2410, 2414,
2419, 2423, 2427, 2431, 2435, 2439, 2443, 2447,
2451, 2455, 2459, 2463, 2467, 2471, 2475, 2479,
2483, 2486, 2490, 2494, 2498, 2502, 2506, 2509,
2513, 2517, 2521, 2525, 2528, 2532, 2536, 2539,
2543, 2547, 2550, 2554, 2558, 2561, 2565, 2568,
2572, 2575, 2579, 2583, 2586, 2590, 2593, 2596,
2600, 2603, 2607, 2610, 2614, 2617, 2620, 2624,
2627, 2630, 2633, 2637, 2640, 2643, 2647, 2650,
2653, 2656, 2659, 2662, 2666, 2669, 2672, 2675,
2678, 2681, 2684, 2687, 2690, 2693, 2696, 2699,
2702, 2705, 2708, 2711, 2713, 2716, 2719, 2722,
```

2725, 2728, 2730, 2733, 2736, 2739, 2741, 2744,
2747, 2749, 2752, 2755, 2757, 2760, 2762, 2765,
2767, 2770, 2772, 2775, 2777, 2780, 2782, 2785,
2787, 2789, 2792, 2794, 2796, 2799, 2801, 2803,
2806, 2808, 2810, 2812, 2814, 2817, 2819, 2821,
2823, 2825, 2827, 2829, 2831, 2833, 2835, 2837,
2839, 2841, 2843, 2845, 2847, 2849, 2851, 2852,
2854, 2856, 2858, 2860, 2861, 2863, 2865, 2866,
2868, 2870, 2871, 2873, 2875, 2876, 2878, 2879,
2881, 2882, 2884, 2885, 2887, 2888, 2889, 2891,
2892, 2894, 2895, 2896, 2897, 2899, 2900, 2901,
2902, 2904, 2905, 2906, 2907, 2908, 2909, 2910,
2911, 2913, 2914, 2915, 2916, 2917, 2917, 2918,
2919, 2920, 2921, 2922, 2923, 2924, 2924, 2925,
2926, 2927, 2927, 2928, 2929, 2929, 2930, 2931,
2931, 2932, 2932, 2933, 2933, 2934, 2934, 2935,
2935, 2936, 2936, 2936, 2937, 2937, 2937, 2938,
2938, 2938, 2939, 2939, 2939, 2939, 2939, 2940,
2940, 2940, 2940, 2940, 2940, 2940, 2940, 2940,
2940, 2940, 2940, 2940, 2940, 2940, 2939, 2939,
2939, 2939, 2939, 2938, 2938, 2938, 2937, 2937,
2937, 2936, 2936, 2936, 2935, 2935, 2934, 2934,
2933, 2933, 2932, 2932, 2931, 2931, 2930, 2929,
2929, 2928, 2927, 2927, 2926, 2925, 2924, 2924,
2923, 2922, 2921, 2920, 2919, 2918, 2917, 2917,
2916, 2915, 2914, 2913, 2911, 2910, 2909, 2908,
2907, 2906, 2905, 2904, 2902, 2901, 2900, 2899,
2897, 2896, 2895, 2894, 2892, 2891, 2889, 2888,
2887, 2885, 2884, 2882, 2881, 2879, 2878, 2876,
2875, 2873, 2871, 2870, 2868, 2866, 2865, 2863,
2861, 2860, 2858, 2856, 2854, 2852, 2851, 2849,
2847, 2845, 2843, 2841, 2839, 2837, 2835, 2833,
2831, 2829, 2827, 2825, 2823, 2821, 2819, 2817,
2814, 2812, 2810, 2808, 2806, 2803, 2801, 2799,
2796, 2794, 2792, 2789, 2787, 2785, 2782, 2780,
2777, 2775, 2772, 2770, 2767, 2765, 2762, 2760,
2757, 2755, 2752, 2749, 2747, 2744, 2741, 2739,
2736, 2733, 2730, 2728, 2725, 2722, 2719, 2716,
2713, 2711, 2708, 2705, 2702, 2699, 2696, 2693,
2690, 2687, 2684, 2681, 2678, 2675, 2672, 2669,
2666, 2662, 2659, 2656, 2653, 2650, 2647, 2643,
2640, 2637, 2633, 2630, 2627, 2624, 2620, 2617,
2614, 2610, 2607, 2603, 2600, 2596, 2593, 2590,
2586, 2583, 2579, 2575, 2572, 2568, 2565, 2561,
2558, 2554, 2550, 2547, 2543, 2539, 2536, 2532,
2528, 2525, 2521, 2517, 2513, 2509, 2506, 2502,
2498, 2494, 2490, 2486, 2483, 2479, 2475, 2471,
2467, 2463, 2459, 2455, 2451, 2447, 2443, 2439,
2435, 2431, 2427, 2423, 2419, 2414, 2410, 2406,
2402, 2398, 2394, 2390, 2385, 2381, 2377, 2373,
2368, 2364, 2360, 2356, 2351, 2347, 2343, 2338,
2334, 2330, 2325, 2321, 2317, 2312, 2308, 2303,
2299, 2294, 2290, 2286, 2281, 2277, 2272, 2268,
2263, 2259, 2254, 2249, 2245, 2240, 2236, 2231,
2227, 2222, 2217, 2213, 2208, 2203, 2199, 2194,
2189, 2185, 2180, 2175, 2171, 2166, 2161, 2156,

```
2152, 2147, 2142, 2137, 2133, 2128, 2123, 2118,
2113, 2109, 2104, 2099, 2094, 2089, 2084, 2079,
2074, 2070, 2065, 2060, 2055, 2050, 2045, 2040,
2035, 2030, 2025, 2020, 2015, 2010, 2005, 2000,
1995, 1990, 1985, 1980, 1975, 1970, 1965, 1960,
1955, 1950, 1945, 1940, 1934, 1929, 1924, 1919,
1914, 1909, 1904, 1899, 1894, 1888, 1883, 1878,
1873, 1868, 1863, 1857, 1852, 1847, 1842, 1837,
1831, 1826, 1821, 1816, 1811, 1805, 1800, 1795,
1790, 1784, 1779, 1774, 1769, 1763, 1758, 1753,
1748, 1742, 1737, 1732, 1726, 1721, 1716, 1711,
1705, 1700, 1695, 1689, 1684, 1679, 1673, 1668,
1663, 1657, 1652, 1647, 1641, 1636, 1631, 1625,
1620, 1615, 1609, 1604, 1599, 1593, 1588, 1583,
1577, 1572, 1567, 1561, 1556, 1551, 1545, 1540,
1534, 1529, 1524, 1518, 1513, 1508, 1502, 1497,
1491, 1486, 1481, 1475, 1470, 1465, 1459, 1454,
1449, 1443, 1438, 1432, 1427, 1422, 1416, 1411,
1406, 1400, 1395, 1389, 1384, 1379, 1373, 1368,
1363, 1357, 1352, 1347, 1341, 1336, 1331, 1325,
1320, 1315, 1309, 1304, 1299, 1293, 1288, 1283,
1277, 1272, 1267, 1261, 1256, 1251, 1245, 1240,
1235, 1229, 1224, 1219, 1214, 1208, 1203, 1198,
1192, 1187, 1182, 1177, 1171, 1166, 1161, 1156,
1150, 1145, 1140, 1135, 1129, 1124, 1119, 1114,
1109, 1103, 1098, 1093, 1088, 1083, 1077, 1072,
1067, 1062, 1057, 1052, 1046, 1041, 1036, 1031,
1026, 1021, 1016, 1011, 1006, 1000,  995,  990,
 985,  980,  975,  970,  965,  960,  955,  950,
 945,  940,  935,  930,  925,  920,  915,  910,
 905,  900,  895,  890,  885,  880,  875,  870,
 866,  861,  856,  851,  846,  841,  836,  831,
 827,  822,  817,  812,  807,  803,  798,  793,
 788,  784,  779,  774,  769,  765,  760,  755,
 751,  746,  741,  737,  732,  727,  723,  718,
 713,  709,  704,  700,  695,  691,  686,  681,
 677,  672,  668,  663,  659,  654,  650,  646,
 641,  637,  632,  628,  623,  619,  615,  610,
 606,  602,  597,  593,  589,  584,  580,  576,
 572,  567,  563,  559,  555,  550,  546,  542,
 538,  534,  530,  526,  521,  517,  513,  509,
 505,  501,  497,  493,  489,  485,  481,  477,
 473,  469,  465,  461,  457,  454,  450,  446,
 442,  438,  434,  431,  427,  423,  419,  415,
 412,  408,  404,  401,  397,  393,  390,  386,
 382,  379,  375,  372,  368,  365,  361,  357,
 354,  350,  347,  344,  340,  337,  333,  330,
 326,  323,  320,  316,  313,  310,  307,  303,
 300,  297,  293,  290,  287,  284,  281,  278,
 274,  271,  268,  265,  262,  259,  256,  253,
 250,  247,  244,  241,  238,  235,  232,  229,
 227,  224,  221,  218,  215,  212,  210,  207,
 204,  201,  199,  196,  193,  191,  188,  185,
 183,  180,  178,  175,  173,  170,  168,  165,
 163,  160,  158,  155,  153,  151,  148,  146,
 144,  141,  139,  137,  134,  132,  130,  128,
```

```
126,  123,  121,  119,  117,  115,  113,  111,
109,  107,  105,  103,  101,   99,   97,   95,
 93,   91,   89,   88,   86,   84,   82,   80,
 79,   77,   75,   74,   72,   70,   69,   67,
 65,   64,   62,   61,   59,   58,   56,   55,
 53,   52,   51,   49,   48,   46,   45,   44,
 43,   41,   40,   39,   38,   36,   35,   34,
 33,   32,   31,   30,   29,   27,   26,   25,
 24,   23,   23,   22,   21,   20,   19,   18,
 17,   16,   16,   15,   14,   13,   13,   12,
 11,   11,   10,    9,    9,    8,    8,    7,
  7,    6,    6,    5,    5,    4,    4,    4,
  3,    3,    3,    2,    2,    2,    1,    1,
  1,    1,    1,    0,    0,    0,    0,    0,
  0,    0,    0,    0,    0,    0,    0,    0,
  0,    0,    1,    1,    1,    1,    1,    2,
  2,    2,    3,    3,    3,    4,    4,    4,
  5,    5,    6,    6,    7,    7,    8,    8,
  9,    9,   10,   11,   11,   12,   13,   13,
 14,   15,   16,   16,   17,   18,   19,   20,
 21,   22,   23,   23,   24,   25,   26,   27,
 29,   30,   31,   32,   33,   34,   35,   36,
 38,   39,   40,   41,   43,   44,   45,   46,
 48,   49,   51,   52,   53,   55,   56,   58,
 59,   61,   62,   64,   65,   67,   69,   70,
 72,   74,   75,   77,   79,   80,   82,   84,
 86,   88,   89,   91,   93,   95,   97,   99,
101,  103,  105,  107,  109,  111,  113,  115,
117,  119,  121,  123,  126,  128,  130,  132,
134,  137,  139,  141,  144,  146,  148,  151,
153,  155,  158,  160,  163,  165,  168,  170,
173,  175,  178,  180,  183,  185,  188,  191,
193,  196,  199,  201,  204,  207,  210,  212,
215,  218,  221,  224,  227,  229,  232,  235,
238,  241,  244,  247,  250,  253,  256,  259,
262,  265,  268,  271,  274,  278,  281,  284,
287,  290,  293,  297,  300,  303,  307,  310,
313,  316,  320,  323,  326,  330,  333,  337,
340,  344,  347,  350,  354,  357,  361,  365,
368,  372,  375,  379,  382,  386,  390,  393,
397,  401,  404,  408,  412,  415,  419,  423,
427,  431,  434,  438,  442,  446,  450,  454,
457,  461,  465,  469,  473,  477,  481,  485,
489,  493,  497,  501,  505,  509,  513,  517,
521,  526,  530,  534,  538,  542,  546,  550,
555,  559,  563,  567,  572,  576,  580,  584,
589,  593,  597,  602,  606,  610,  615,  619,
623,  628,  632,  637,  641,  646,  650,  654,
659,  663,  668,  672,  677,  681,  686,  691,
695,  700,  704,  709,  713,  718,  723,  727,
732,  737,  741,  746,  751,  755,  760,  765,
769,  774,  779,  784,  788,  793,  798,  803,
807,  812,  817,  822,  827,  831,  836,  841,
846,  851,  856,  861,  866,  870,  875,  880,
885,  890,  895,  900,  905,  910,  915,  920,
925,  930,  935,  940,  945,  950,  955,  960,
```

```
 965,  970,  975,  980,  985,  990,  995, 1000,
1006, 1011, 1016, 1021, 1026, 1031, 1036, 1041,
1046, 1052, 1057, 1062, 1067, 1072, 1077, 1083,
1088, 1093, 1098, 1103, 1109, 1114, 1119, 1124,
1129, 1135, 1140, 1145, 1150, 1156, 1161, 1166,
1171, 1177, 1182, 1187, 1192, 1198, 1203, 1208,
1214, 1219, 1224, 1229, 1235, 1240, 1245, 1251,
1256, 1261, 1267, 1272, 1277, 1283, 1288, 1293,
1299, 1304, 1309, 1315, 1320, 1325, 1331, 1336,
1341, 1347, 1352, 1357, 1363, 1368, 1373, 1379,
1384, 1389, 1395, 1400, 1406, 1411, 1416, 1422,
1427, 1432, 1438, 1443, 1449, 1454, 1459, 1465 };

/** Generated using Dr LUT - Free Lookup Table Generator
 * https://github.com/ppelikan/drlut
 **/
// Formula: 2*(t%T)/T-1
const uint16_t ramp[1720] = {
   0,    2,    3,    5,    7,    9,   10,   12,
  14,   15,   17,   19,   21,   22,   24,   26,
  27,   29,   31,   32,   34,   36,   38,   39,
  41,   43,   44,   46,   48,   50,   51,   53,
  55,   56,   58,   60,   62,   63,   65,   67,
  68,   70,   72,   74,   75,   77,   79,   80,
  82,   84,   85,   87,   89,   91,   92,   94,
  96,   97,   99,  101,  103,  104,  106,  108,
 109,  111,  113,  115,  116,  118,  120,  121,
 123,  125,  126,  128,  130,  132,  133,  135,
 137,  138,  140,  142,  144,  145,  147,  149,
 150,  152,  154,  156,  157,  159,  161,  162,
 164,  166,  168,  169,  171,  173,  174,  176,
 178,  179,  181,  183,  185,  186,  188,  190,
 191,  193,  195,  197,  198,  200,  202,  203,
 205,  207,  209,  210,  212,  214,  215,  217,
 219,  221,  222,  224,  226,  227,  229,  231,
 232,  234,  236,  238,  239,  241,  243,  244,
 246,  248,  250,  251,  253,  255,  256,  258,
 260,  262,  263,  265,  267,  268,  270,  272,
 273,  275,  277,  279,  280,  282,  284,  285,
 287,  289,  291,  292,  294,  296,  297,  299,
 301,  303,  304,  306,  308,  309,  311,  313,
 315,  316,  318,  320,  321,  323,  325,  326,
 328,  330,  332,  333,  335,  337,  338,  340,
 342,  344,  345,  347,  349,  350,  352,  354,
 356,  357,  359,  361,  362,  364,  366,  368,
 369,  371,  373,  374,  376,  378,  379,  381,
 383,  385,  386,  388,  390,  391,  393,  395,
 397,  398,  400,  402,  403,  405,  407,  409,
 410,  412,  414,  415,  417,  419,  420,  422,
 424,  426,  427,  429,  431,  432,  434,  436,
 438,  439,  441,  443,  444,  446,  448,  450,
 451,  453,  455,  456,  458,  460,  462,  463,
 465,  467,  468,  470,  472,  473,  475,  477,
 479,  480,  482,  484,  485,  487,  489,  491,
 492,  494,  496,  497,  499,  501,  503,  504,
 506,  508,  509,  511,  513,  515,  516,  518,
```

 520, 521, 523, 525, 526, 528, 530, 532,
 533, 535, 537, 538, 540, 542, 544, 545,
 547, 549, 550, 552, 554, 556, 557, 559,
 561, 562, 564, 566, 567, 569, 571, 573,
 574, 576, 578, 579, 581, 583, 585, 586,
 588, 590, 591, 593, 595, 597, 598, 600,
 602, 603, 605, 607, 609, 610, 612, 614,
 615, 617, 619, 620, 622, 624, 626, 627,
 629, 631, 632, 634, 636, 638, 639, 641,
 643, 644, 646, 648, 650, 651, 653, 655,
 656, 658, 660, 662, 663, 665, 667, 668,
 670, 672, 673, 675, 677, 679, 680, 682,
 684, 685, 687, 689, 691, 692, 694, 696,
 697, 699, 701, 703, 704, 706, 708, 709,
 711, 713, 714, 716, 718, 720, 721, 723,
 725, 726, 728, 730, 732, 733, 735, 737,
 738, 740, 742, 744, 745, 747, 749, 750,
 752, 754, 756, 757, 759, 761, 762, 764,
 766, 767, 769, 771, 773, 774, 776, 778,
 779, 781, 783, 785, 786, 788, 790, 791,
 793, 795, 797, 798, 800, 802, 803, 805,
 807, 809, 810, 812, 814, 815, 817, 819,
 820, 822, 824, 826, 827, 829, 831, 832,
 834, 836, 838, 839, 841, 843, 844, 846,
 848, 850, 851, 853, 855, 856, 858, 860,
 861, 863, 865, 867, 868, 870, 872, 873,
 875, 877, 879, 880, 882, 884, 885, 887,
 889, 891, 892, 894, 896, 897, 899, 901,
 903, 904, 906, 908, 909, 911, 913, 914,
 916, 918, 920, 921, 923, 925, 926, 928,
 930, 932, 933, 935, 937, 938, 940, 942,
 944, 945, 947, 949, 950, 952, 954, 956,
 957, 959, 961, 962, 964, 966, 967, 969,
 971, 973, 974, 976, 978, 979, 981, 983,
 985, 986, 988, 990, 991, 993, 995, 997,
 998, 1000, 1002, 1003, 1005, 1007, 1008, 1010,
1012, 1014, 1015, 1017, 1019, 1020, 1022, 1024,
1026, 1027, 1029, 1031, 1032, 1034, 1036, 1038,
1039, 1041, 1043, 1044, 1046, 1048, 1050, 1051,
1053, 1055, 1056, 1058, 1060, 1061, 1063, 1065,
1067, 1068, 1070, 1072, 1073, 1075, 1077, 1079,
1080, 1082, 1084, 1085, 1087, 1089, 1091, 1092,
1094, 1096, 1097, 1099, 1101, 1103, 1104, 1106,
1108, 1109, 1111, 1113, 1114, 1116, 1118, 1120,
1121, 1123, 1125, 1126, 1128, 1130, 1132, 1133,
1135, 1137, 1138, 1140, 1142, 1144, 1145, 1147,
1149, 1150, 1152, 1154, 1155, 1157, 1159, 1161,
1162, 1164, 1166, 1167, 1169, 1171, 1173, 1174,
1176, 1178, 1179, 1181, 1183, 1185, 1186, 1188,
1190, 1191, 1193, 1195, 1197, 1198, 1200, 1202,
1203, 1205, 1207, 1208, 1210, 1212, 1214, 1215,
1217, 1219, 1220, 1222, 1224, 1226, 1227, 1229,
1231, 1232, 1234, 1236, 1238, 1239, 1241, 1243,
1244, 1246, 1248, 1250, 1251, 1253, 1255, 1256,
1258, 1260, 1261, 1263, 1265, 1267, 1268, 1270,
1272, 1273, 1275, 1277, 1279, 1280, 1282, 1284,

1285, 1287, 1289, 1291, 1292, 1294, 1296, 1297,
1299, 1301, 1302, 1304, 1306, 1308, 1309, 1311,
1313, 1314, 1316, 1318, 1320, 1321, 1323, 1325,
1326, 1328, 1330, 1332, 1333, 1335, 1337, 1338,
1340, 1342, 1344, 1345, 1347, 1349, 1350, 1352,
1354, 1355, 1357, 1359, 1361, 1362, 1364, 1366,
1367, 1369, 1371, 1373, 1374, 1376, 1378, 1379,
1381, 1383, 1385, 1386, 1388, 1390, 1391, 1393,
1395, 1397, 1398, 1400, 1402, 1403, 1405, 1407,
1408, 1410, 1412, 1414, 1415, 1417, 1419, 1420,
1422, 1424, 1426, 1427, 1429, 1431, 1432, 1434,
1436, 1438, 1439, 1441, 1443, 1444, 1446, 1448,
1449, 1451, 1453, 1455, 1456, 1458, 1460, 1461,
1463, 1465, 1467, 1468, 1470, 1472, 1473, 1475,
1477, 1479, 1480, 1482, 1484, 1485, 1487, 1489,
1491, 1492, 1494, 1496, 1497, 1499, 1501, 1502,
1504, 1506, 1508, 1509, 1511, 1513, 1514, 1516,
1518, 1520, 1521, 1523, 1525, 1526, 1528, 1530,
1532, 1533, 1535, 1537, 1538, 1540, 1542, 1543,
1545, 1547, 1549, 1550, 1552, 1554, 1555, 1557,
1559, 1561, 1562, 1564, 1566, 1567, 1569, 1571,
1573, 1574, 1576, 1578, 1579, 1581, 1583, 1585,
1586, 1588, 1590, 1591, 1593, 1595, 1596, 1598,
1600, 1602, 1603, 1605, 1607, 1608, 1610, 1612,
1614, 1615, 1617, 1619, 1620, 1622, 1624, 1626,
1627, 1629, 1631, 1632, 1634, 1636, 1638, 1639,
1641, 1643, 1644, 1646, 1648, 1649, 1651, 1653,
1655, 1656, 1658, 1660, 1661, 1663, 1665, 1667,
1668, 1670, 1672, 1673, 1675, 1677, 1679, 1680,
1682, 1684, 1685, 1687, 1689, 1690, 1692, 1694,
1696, 1697, 1699, 1701, 1702, 1704, 1706, 1708,
1709, 1711, 1713, 1714, 1716, 1718, 1720, 1721,
1723, 1725, 1726, 1728, 1730, 1732, 1733, 1735,
1737, 1738, 1740, 1742, 1743, 1745, 1747, 1749,
1750, 1752, 1754, 1755, 1757, 1759, 1761, 1762,
1764, 1766, 1767, 1769, 1771, 1773, 1774, 1776,
1778, 1779, 1781, 1783, 1785, 1786, 1788, 1790,
1791, 1793, 1795, 1796, 1798, 1800, 1802, 1803,
1805, 1807, 1808, 1810, 1812, 1814, 1815, 1817,
1819, 1820, 1822, 1824, 1826, 1827, 1829, 1831,
1832, 1834, 1836, 1837, 1839, 1841, 1843, 1844,
1846, 1848, 1849, 1851, 1853, 1855, 1856, 1858,
1860, 1861, 1863, 1865, 1867, 1868, 1870, 1872,
1873, 1875, 1877, 1879, 1880, 1882, 1884, 1885,
1887, 1889, 1890, 1892, 1894, 1896, 1897, 1899,
1901, 1902, 1904, 1906, 1908, 1909, 1911, 1913,
1914, 1916, 1918, 1920, 1921, 1923, 1925, 1926,
1928, 1930, 1932, 1933, 1935, 1937, 1938, 1940,
1942, 1943, 1945, 1947, 1949, 1950, 1952, 1954,
1955, 1957, 1959, 1961, 1962, 1964, 1966, 1967,
1969, 1971, 1973, 1974, 1976, 1978, 1979, 1981,
1983, 1984, 1986, 1988, 1990, 1991, 1993, 1995,
1996, 1998, 2000, 2002, 2003, 2005, 2007, 2008,
2010, 2012, 2014, 2015, 2017, 2019, 2020, 2022,
2024, 2026, 2027, 2029, 2031, 2032, 2034, 2036,
2037, 2039, 2041, 2043, 2044, 2046, 2048, 2049,

2051, 2053, 2055, 2056, 2058, 2060, 2061, 2063,
2065, 2067, 2068, 2070, 2072, 2073, 2075, 2077,
2079, 2080, 2082, 2084, 2085, 2087, 2089, 2090,
2092, 2094, 2096, 2097, 2099, 2101, 2102, 2104,
2106, 2108, 2109, 2111, 2113, 2114, 2116, 2118,
2120, 2121, 2123, 2125, 2126, 2128, 2130, 2131,
2133, 2135, 2137, 2138, 2140, 2142, 2143, 2145,
2147, 2149, 2150, 2152, 2154, 2155, 2157, 2159,
2161, 2162, 2164, 2166, 2167, 2169, 2171, 2173,
2174, 2176, 2178, 2179, 2181, 2183, 2184, 2186,
2188, 2190, 2191, 2193, 2195, 2196, 2198, 2200,
2202, 2203, 2205, 2207, 2208, 2210, 2212, 2214,
2215, 2217, 2219, 2220, 2222, 2224, 2226, 2227,
2229, 2231, 2232, 2234, 2236, 2237, 2239, 2241,
2243, 2244, 2246, 2248, 2249, 2251, 2253, 2255,
2256, 2258, 2260, 2261, 2263, 2265, 2267, 2268,
2270, 2272, 2273, 2275, 2277, 2278, 2280, 2282,
2284, 2285, 2287, 2289, 2290, 2292, 2294, 2296,
2297, 2299, 2301, 2302, 2304, 2306, 2308, 2309,
2311, 2313, 2314, 2316, 2318, 2320, 2321, 2323,
2325, 2326, 2328, 2330, 2331, 2333, 2335, 2337,
2338, 2340, 2342, 2343, 2345, 2347, 2349, 2350,
2352, 2354, 2355, 2357, 2359, 2361, 2362, 2364,
2366, 2367, 2369, 2371, 2373, 2374, 2376, 2378,
2379, 2381, 2383, 2384, 2386, 2388, 2390, 2391,
2393, 2395, 2396, 2398, 2400, 2402, 2403, 2405,
2407, 2408, 2410, 2412, 2414, 2415, 2417, 2419,
2420, 2422, 2424, 2425, 2427, 2429, 2431, 2432,
2434, 2436, 2437, 2439, 2441, 2443, 2444, 2446,
2448, 2449, 2451, 2453, 2455, 2456, 2458, 2460,
2461, 2463, 2465, 2467, 2468, 2470, 2472, 2473,
2475, 2477, 2478, 2480, 2482, 2484, 2485, 2487,
2489, 2490, 2492, 2494, 2496, 2497, 2499, 2501,
2502, 2504, 2506, 2508, 2509, 2511, 2513, 2514,
2516, 2518, 2520, 2521, 2523, 2525, 2526, 2528,
2530, 2531, 2533, 2535, 2537, 2538, 2540, 2542,
2543, 2545, 2547, 2549, 2550, 2552, 2554, 2555,
2557, 2559, 2561, 2562, 2564, 2566, 2567, 2569,
2571, 2572, 2574, 2576, 2578, 2579, 2581, 2583,
2584, 2586, 2588, 2590, 2591, 2593, 2595, 2596,
2598, 2600, 2602, 2603, 2605, 2607, 2608, 2610,
2612, 2614, 2615, 2617, 2619, 2620, 2622, 2624,
2625, 2627, 2629, 2631, 2632, 2634, 2636, 2637,
2639, 2641, 2643, 2644, 2646, 2648, 2649, 2651,
2653, 2655, 2656, 2658, 2660, 2661, 2663, 2665,
2667, 2668, 2670, 2672, 2673, 2675, 2677, 2678,
2680, 2682, 2684, 2685, 2687, 2689, 2690, 2692,
2694, 2696, 2697, 2699, 2701, 2702, 2704, 2706,
2708, 2709, 2711, 2713, 2714, 2716, 2718, 2719,
2721, 2723, 2725, 2726, 2728, 2730, 2731, 2733,
2735, 2737, 2738, 2740, 2742, 2743, 2745, 2747,
2749, 2750, 2752, 2754, 2755, 2757, 2759, 2761,
2762, 2764, 2766, 2767, 2769, 2771, 2772, 2774,
2776, 2778, 2779, 2781, 2783, 2784, 2786, 2788,
2790, 2791, 2793, 2795, 2796, 2798, 2800, 2802,
2803, 2805, 2807, 2808, 2810, 2812, 2814, 2815,

```
2817, 2819, 2820, 2822, 2824, 2825, 2827, 2829,
2831, 2832, 2834, 2836, 2837, 2839, 2841, 2843,
2844, 2846, 2848, 2849, 2851, 2853, 2855, 2856,
2858, 2860, 2861, 2863, 2865, 2866, 2868, 2870,
2872, 2873, 2875, 2877, 2878, 2880, 2882, 2884,
2885, 2887, 2889, 2890, 2892, 2894, 2896, 2897,
2899, 2901, 2902, 2904, 2906, 2908, 2909, 2911,
2913, 2914, 2916, 2918, 2919, 2921, 2923, 2925,
2926, 2928, 2930, 2931, 2933, 2935, 2937, 2938 };

/** Generated using Dr LUT - Free Lookup Table Generator
 * https://github.com/ppelikan/drlut
 **/
// Formula: 4*abs(t/T-floor(t/T+1/2))-1
const uint16_t triangle[1720] = {
   0,    3,    7,   10,   14,   17,   21,   24,
  27,   31,   34,   38,   41,   44,   48,   51,
  55,   58,   62,   65,   68,   72,   75,   79,
  82,   85,   89,   92,   96,   99,  103,  106,
 109,  113,  116,  120,  123,  126,  130,  133,
 137,  140,  144,  147,  150,  154,  157,  161,
 164,  168,  171,  174,  178,  181,  185,  188,
 191,  195,  198,  202,  205,  209,  212,  215,
 219,  222,  226,  229,  232,  236,  239,  243,
 246,  250,  253,  256,  260,  263,  267,  270,
 273,  277,  280,  284,  287,  291,  294,  297,
 301,  304,  308,  311,  315,  318,  321,  325,
 328,  332,  335,  338,  342,  345,  349,  352,
 356,  359,  362,  366,  369,  373,  376,  379,
 383,  386,  390,  393,  397,  400,  403,  407,
 410,  414,  417,  420,  424,  427,  431,  434,
 438,  441,  444,  448,  451,  455,  458,  462,
 465,  468,  472,  475,  479,  482,  485,  489,
 492,  496,  499,  503,  506,  509,  513,  516,
 520,  523,  526,  530,  533,  537,  540,  544,
 547,  550,  554,  557,  561,  564,  567,  571,
 574,  578,  581,  585,  588,  591,  595,  598,
 602,  605,  609,  612,  615,  619,  622,  626,
 629,  632,  636,  639,  643,  646,  650,  653,
 656,  660,  663,  667,  670,  673,  677,  680,
 684,  687,  691,  694,  697,  701,  704,  708,
 711,  714,  718,  721,  725,  728,  732,  735,
 738,  742,  745,  749,  752,  756,  759,  762,
 766,  769,  773,  776,  779,  783,  786,  790,
 793,  797,  800,  803,  807,  810,  814,  817,
 820,  824,  827,  831,  834,  838,  841,  844,
 848,  851,  855,  858,  861,  865,  868,  872,
 875,  879,  882,  885,  889,  892,  896,  899,
 903,  906,  909,  913,  916,  920,  923,  926,
 930,  933,  937,  940,  944,  947,  950,  954,
 957,  961,  964,  967,  971,  974,  978,  981,
 985,  988,  991,  995,  998, 1002, 1005, 1008,
1012, 1015, 1019, 1022, 1026, 1029, 1032, 1036,
1039, 1043, 1046, 1050, 1053, 1056, 1060, 1063,
1067, 1070, 1073, 1077, 1080, 1084, 1087, 1091,
1094, 1097, 1101, 1104, 1108, 1111, 1114, 1118,
```

1121, 1125, 1128, 1132, 1135, 1138, 1142, 1145,
1149, 1152, 1155, 1159, 1162, 1166, 1169, 1173,
1176, 1179, 1183, 1186, 1190, 1193, 1197, 1200,
1203, 1207, 1210, 1214, 1217, 1220, 1224, 1227,
1231, 1234, 1238, 1241, 1244, 1248, 1251, 1255,
1258, 1261, 1265, 1268, 1272, 1275, 1279, 1282,
1285, 1289, 1292, 1296, 1299, 1302, 1306, 1309,
1313, 1316, 1320, 1323, 1326, 1330, 1333, 1337,
1340, 1344, 1347, 1350, 1354, 1357, 1361, 1364,
1367, 1371, 1374, 1378, 1381, 1385, 1388, 1391,
1395, 1398, 1402, 1405, 1408, 1412, 1415, 1419,
1422, 1426, 1429, 1432, 1436, 1439, 1443, 1446,
1449, 1453, 1456, 1460, 1463, 1467, 1470, 1473,
1477, 1480, 1484, 1487, 1491, 1494, 1497, 1501,
1504, 1508, 1511, 1514, 1518, 1521, 1525, 1528,
1532, 1535, 1538, 1542, 1545, 1549, 1552, 1555,
1559, 1562, 1566, 1569, 1573, 1576, 1579, 1583,
1586, 1590, 1593, 1596, 1600, 1603, 1607, 1610,
1614, 1617, 1620, 1624, 1627, 1631, 1634, 1638,
1641, 1644, 1648, 1651, 1655, 1658, 1661, 1665,
1668, 1672, 1675, 1679, 1682, 1685, 1689, 1692,
1696, 1699, 1702, 1706, 1709, 1713, 1716, 1720,
1723, 1726, 1730, 1733, 1737, 1740, 1743, 1747,
1750, 1754, 1757, 1761, 1764, 1767, 1771, 1774,
1778, 1781, 1785, 1788, 1791, 1795, 1798, 1802,
1805, 1808, 1812, 1815, 1819, 1822, 1826, 1829,
1832, 1836, 1839, 1843, 1846, 1849, 1853, 1856,
1860, 1863, 1867, 1870, 1873, 1877, 1880, 1884,
1887, 1890, 1894, 1897, 1901, 1904, 1908, 1911,
1914, 1918, 1921, 1925, 1928, 1932, 1935, 1938,
1942, 1945, 1949, 1952, 1955, 1959, 1962, 1966,
1969, 1973, 1976, 1979, 1983, 1986, 1990, 1993,
1996, 2000, 2003, 2007, 2010, 2014, 2017, 2020,
2024, 2027, 2031, 2034, 2037, 2041, 2044, 2048,
2051, 2055, 2058, 2061, 2065, 2068, 2072, 2075,
2079, 2082, 2085, 2089, 2092, 2096, 2099, 2102,
2106, 2109, 2113, 2116, 2120, 2123, 2126, 2130,
2133, 2137, 2140, 2143, 2147, 2150, 2154, 2157,
2161, 2164, 2167, 2171, 2174, 2178, 2181, 2184,
2188, 2191, 2195, 2198, 2202, 2205, 2208, 2212,
2215, 2219, 2222, 2226, 2229, 2232, 2236, 2239,
2243, 2246, 2249, 2253, 2256, 2260, 2263, 2267,
2270, 2273, 2277, 2280, 2284, 2287, 2290, 2294,
2297, 2301, 2304, 2308, 2311, 2314, 2318, 2321,
2325, 2328, 2331, 2335, 2338, 2342, 2345, 2349,
2352, 2355, 2359, 2362, 2366, 2369, 2373, 2376,
2379, 2383, 2386, 2390, 2393, 2396, 2400, 2403,
2407, 2410, 2414, 2417, 2420, 2424, 2427, 2431,
2434, 2437, 2441, 2444, 2448, 2451, 2455, 2458,
2461, 2465, 2468, 2472, 2475, 2478, 2482, 2485,
2489, 2492, 2496, 2499, 2502, 2506, 2509, 2513,
2516, 2520, 2523, 2526, 2530, 2533, 2537, 2540,
2543, 2547, 2550, 2554, 2557, 2561, 2564, 2567,
2571, 2574, 2578, 2581, 2584, 2588, 2591, 2595,
2598, 2602, 2605, 2608, 2612, 2615, 2619, 2622,
2625, 2629, 2632, 2636, 2639, 2643, 2646, 2649,

2653, 2656, 2660, 2663, 2667, 2670, 2673, 2677,
2680, 2684, 2687, 2690, 2694, 2697, 2701, 2704,
2708, 2711, 2714, 2718, 2721, 2725, 2728, 2731,
2735, 2738, 2742, 2745, 2749, 2752, 2755, 2759,
2762, 2766, 2769, 2772, 2776, 2779, 2783, 2786,
2790, 2793, 2796, 2800, 2803, 2807, 2810, 2814,
2817, 2820, 2824, 2827, 2831, 2834, 2837, 2841,
2844, 2848, 2851, 2855, 2858, 2861, 2865, 2868,
2872, 2875, 2878, 2882, 2885, 2889, 2892, 2896,
2899, 2902, 2906, 2909, 2913, 2916, 2919, 2923,
2926, 2930, 2933, 2937, 2940, 2937, 2933, 2930,
2926, 2923, 2919, 2916, 2913, 2909, 2906, 2902,
2899, 2896, 2892, 2889, 2885, 2882, 2878, 2875,
2872, 2868, 2865, 2861, 2858, 2855, 2851, 2848,
2844, 2841, 2837, 2834, 2831, 2827, 2824, 2820,
2817, 2814, 2810, 2807, 2803, 2800, 2796, 2793,
2790, 2786, 2783, 2779, 2776, 2772, 2769, 2766,
2762, 2759, 2755, 2752, 2749, 2745, 2742, 2738,
2735, 2731, 2728, 2725, 2721, 2718, 2714, 2711,
2708, 2704, 2701, 2697, 2694, 2690, 2687, 2684,
2680, 2677, 2673, 2670, 2667, 2663, 2660, 2656,
2653, 2649, 2646, 2643, 2639, 2636, 2632, 2629,
2625, 2622, 2619, 2615, 2612, 2608, 2605, 2602,
2598, 2595, 2591, 2588, 2584, 2581, 2578, 2574,
2571, 2567, 2564, 2561, 2557, 2554, 2550, 2547,
2543, 2540, 2537, 2533, 2530, 2526, 2523, 2520,
2516, 2513, 2509, 2506, 2502, 2499, 2496, 2492,
2489, 2485, 2482, 2478, 2475, 2472, 2468, 2465,
2461, 2458, 2455, 2451, 2448, 2444, 2441, 2437,
2434, 2431, 2427, 2424, 2420, 2417, 2414, 2410,
2407, 2403, 2400, 2396, 2393, 2390, 2386, 2383,
2379, 2376, 2373, 2369, 2366, 2362, 2359, 2355,
2352, 2349, 2345, 2342, 2338, 2335, 2331, 2328,
2325, 2321, 2318, 2314, 2311, 2308, 2304, 2301,
2297, 2294, 2290, 2287, 2284, 2280, 2277, 2273,
2270, 2267, 2263, 2260, 2256, 2253, 2249, 2246,
2243, 2239, 2236, 2232, 2229, 2226, 2222, 2219,
2215, 2212, 2208, 2205, 2202, 2198, 2195, 2191,
2188, 2184, 2181, 2178, 2174, 2171, 2167, 2164,
2161, 2157, 2154, 2150, 2147, 2143, 2140, 2137,
2133, 2130, 2126, 2123, 2120, 2116, 2113, 2109,
2106, 2102, 2099, 2096, 2092, 2089, 2085, 2082,
2079, 2075, 2072, 2068, 2065, 2061, 2058, 2055,
2051, 2048, 2044, 2041, 2037, 2034, 2031, 2027,
2024, 2020, 2017, 2014, 2010, 2007, 2003, 2000,
1996, 1993, 1990, 1986, 1983, 1979, 1976, 1973,
1969, 1966, 1962, 1959, 1955, 1952, 1949, 1945,
1942, 1938, 1935, 1932, 1928, 1925, 1921, 1918,
1914, 1911, 1908, 1904, 1901, 1897, 1894, 1890,
1887, 1884, 1880, 1877, 1873, 1870, 1867, 1863,
1860, 1856, 1853, 1849, 1846, 1843, 1839, 1836,
1832, 1829, 1826, 1822, 1819, 1815, 1812, 1808,
1805, 1802, 1798, 1795, 1791, 1788, 1785, 1781,
1778, 1774, 1771, 1767, 1764, 1761, 1757, 1754,
1750, 1747, 1743, 1740, 1737, 1733, 1730, 1726,
1723, 1720, 1716, 1713, 1709, 1706, 1702, 1699,

```
1696, 1692, 1689, 1685, 1682, 1679, 1675, 1672,
1668, 1665, 1661, 1658, 1655, 1651, 1648, 1644,
1641, 1638, 1634, 1631, 1627, 1624, 1620, 1617,
1614, 1610, 1607, 1603, 1600, 1596, 1593, 1590,
1586, 1583, 1579, 1576, 1573, 1569, 1566, 1562,
1559, 1555, 1552, 1549, 1545, 1542, 1538, 1535,
1532, 1528, 1525, 1521, 1518, 1514, 1511, 1508,
1504, 1501, 1497, 1494, 1491, 1487, 1484, 1480,
1477, 1473, 1470, 1467, 1463, 1460, 1456, 1453,
1449, 1446, 1443, 1439, 1436, 1432, 1429, 1426,
1422, 1419, 1415, 1412, 1408, 1405, 1402, 1398,
1395, 1391, 1388, 1385, 1381, 1378, 1374, 1371,
1367, 1364, 1361, 1357, 1354, 1350, 1347, 1344,
1340, 1337, 1333, 1330, 1326, 1323, 1320, 1316,
1313, 1309, 1306, 1302, 1299, 1296, 1292, 1289,
1285, 1282, 1279, 1275, 1272, 1268, 1265, 1261,
1258, 1255, 1251, 1248, 1244, 1241, 1238, 1234,
1231, 1227, 1224, 1220, 1217, 1214, 1210, 1207,
1203, 1200, 1197, 1193, 1190, 1186, 1183, 1179,
1176, 1173, 1169, 1166, 1162, 1159, 1155, 1152,
1149, 1145, 1142, 1138, 1135, 1132, 1128, 1125,
1121, 1118, 1114, 1111, 1108, 1104, 1101, 1097,
1094, 1091, 1087, 1084, 1080, 1077, 1073, 1070,
1067, 1063, 1060, 1056, 1053, 1050, 1046, 1043,
1039, 1036, 1032, 1029, 1026, 1022, 1019, 1015,
1012, 1008, 1005, 1002,  998,  995,  991,  988,
 985,  981,  978,  974,  971,  967,  964,  961,
 957,  954,  950,  947,  944,  940,  937,  933,
 930,  926,  923,  920,  916,  913,  909,  906,
 903,  899,  896,  892,  889,  885,  882,  879,
 875,  872,  868,  865,  861,  858,  855,  851,
 848,  844,  841,  838,  834,  831,  827,  824,
 820,  817,  814,  810,  807,  803,  800,  797,
 793,  790,  786,  783,  779,  776,  773,  769,
 766,  762,  759,  756,  752,  749,  745,  742,
 738,  735,  732,  728,  725,  721,  718,  714,
 711,  708,  704,  701,  697,  694,  691,  687,
 684,  680,  677,  673,  670,  667,  663,  660,
 656,  653,  650,  646,  643,  639,  636,  632,
 629,  626,  622,  619,  615,  612,  609,  605,
 602,  598,  595,  591,  588,  585,  581,  578,
 574,  571,  567,  564,  561,  557,  554,  550,
 547,  544,  540,  537,  533,  530,  526,  523,
 520,  516,  513,  509,  506,  503,  499,  496,
 492,  489,  485,  482,  479,  475,  472,  468,
 465,  462,  458,  455,  451,  448,  444,  441,
 438,  434,  431,  427,  424,  420,  417,  414,
 410,  407,  403,  400,  397,  393,  390,  386,
 383,  379,  376,  373,  369,  366,  362,  359,
 356,  352,  349,  345,  342,  338,  335,  332,
 328,  325,  321,  318,  315,  311,  308,  304,
 301,  297,  294,  291,  287,  284,  280,  277,
 273,  270,  267,  263,  260,  256,  253,  250,
 246,  243,  239,  236,  232,  229,  226,  222,
 219,  215,  212,  209,  205,  202,  198,  195,
 191,  188,  185,  181,  178,  174,  171,  168,
```

```
164,  161,  157,  154,  150,  147,  144,  140,
137,  133,  130,  126,  123,  120,  116,  113,
109,  106,  103,   99,   96,   92,   89,   85,
 82,   79,   75,   72,   68,   65,   62,   58,
 55,   51,   48,   44,   41,   38,   34,   31,
 27,   24,   21,   17,   14,   10,    7,    3 };
```