Batch: A1 **Roll No.: 16010322014**
Experiment / assignment / tutorial No. 4
Grade: AA / AB / BB / BC / CC / CD /DD
Signature of the Staff In-charge with date

Title RTOS implementation using STM32 microcontroller

Aim:-To perform software configuration in STM32 Cube software
Execute simple programs using IDE
OUTCOME: Understand architecture of STM 32 microcontroller

Hardware tools required and its details:-
The STM32 family of 32-bit microcontrollers based on the Arm Cortex®-M processor is designed to offer new degrees of freedom to MCU users.

Core:

- **CPU**: ARM® 32-bit Cortex®-M4 with FPU, up to 84 MHz
- **Accelerator**: ART Accelerator™ for fast Flash memory access (0-wait state)
- **Performance**:
  - 105 DMIPS
  - 1.25 DMIPS/MHz (Dhrystone 2.1)
- **DSP Instructions**: Yes

Memory:

- **Flash**: Up to 512 KB
- **SRAM**: Up to 96 KB

Clock & Power Management:

- **Voltage**: 1.7 V to 3.6 V
- **Oscillators**:
  - 16 MHz internal RC
  - 32 kHz RTC oscillator with calibration

Power Consumption:

- **Run**: 146 µA/MHz
- **Stop**:
  - 42 µA (typical)
  - 10 µA in deep power-down mode
- **Standby**:
  - 2.4 µA (RTC off)
  - 12 µA at 85 °C
- **VBAT RTC**: 1 µA

ADC:

- **A/D Converter**: 12-bit, 2.4 MSPS, up to 16 channels

Timers & DMA:

- **Timers**:
  - 6x 16-bit
  - 2x 32-bit
  - SysTick
  - Watchdog
- **DMA**: 16 streams with FIFOs and burst support

Debug:

- **Debug Mode**: SWD, JTAG, Embedded Trace Macrocell™

I/O & Communication:

- **I/O Ports**: Up to 81 with interrupt capabilities (78 fast up to 42 MHz)
- **Interfaces**:
  - 3x I2C
  - 3x USART
  - 4x SPI (up to 42 Mbps)
  - USB 2.0 (Full-speed, Device/Host/OTG)
  - SDIO
  - IrDA
  - Modem Control
  - LIN

Other:

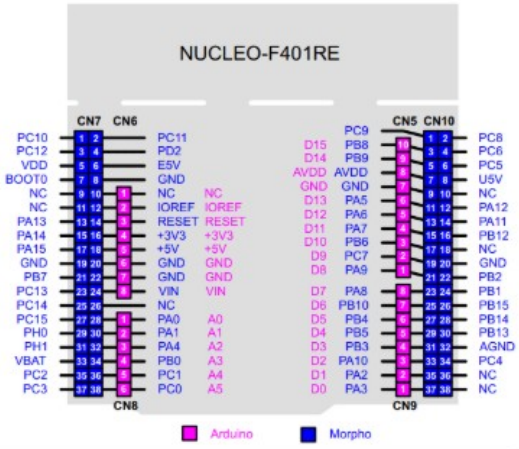
- **CRC**: Calculation unit
- **Unique ID**: 96-bit
- **RTC**: Subsecond accuracy, hardware calendar
- **Package Options**: WLCSP49, LQFP64/100, UFQFPN48, UFBGA100 (ECOPACK®2 compliant)

Software tools required and details:-

- **Keil MDK ARM**
- **IAR WORKBENCH**
- **STM32CUBE IDE**
- **ARM Mbed**

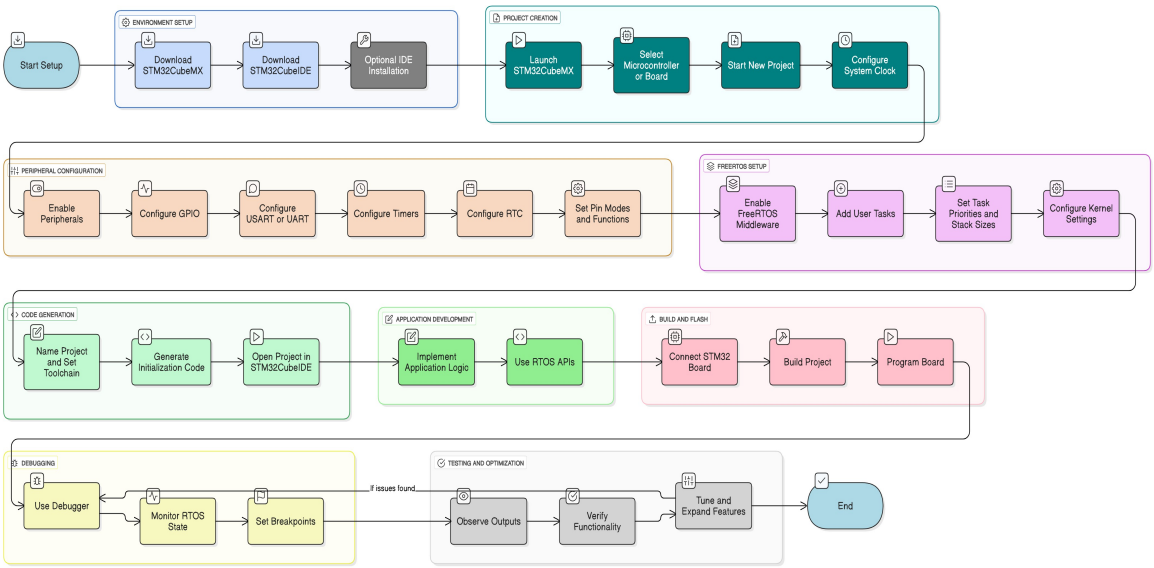
Diagram:

STM32 Nucleo F401RE Development Board



STM32 Nucleo-F401RE Pinout

Procedure:



Source code:

```
/* USER CODE BEGIN Header */
/**
  ******************************************************************************
  * @file           : main.c
  * @brief          : Main program body
  ******************************************************************************
  * @attention
  *
  * Copyright (c) 2025 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"

/* Private includes ----------------------------------------------------------*/
/* USER CODE BEGIN Includes */

/* USER CODE END Includes */

/* Private typedef -----------------------------------------------------------*/
/* USER CODE BEGIN PTD */

/* USER CODE END PTD */

/* Private define ------------------------------------------------------------*/
/* USER CODE BEGIN PD */
/* USER CODE END PD */

/* Private macro -------------------------------------------------------------*/
/* USER CODE BEGIN PM */

/* USER CODE END PM */

/* Private variables ---------------------------------------------------------*/

/* USER CODE BEGIN PV */

/* USER CODE END PV */

/* Private function prototypes -----------------------------------------------*/
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
/* USER CODE BEGIN PFP */

/* USER CODE END PFP */

/* Private user code ---------------------------------------------------------*/
/* USER CODE BEGIN 0 */

/* USER CODE END 0 */

/**
  * @brief  The application entry point.
  * @retval int
  */
int main(void)
{
  /* USER CODE BEGIN 1 */

  /* USER CODE END 1 */

  /* MCU Configuration--------------------------------------------------------*/
```

```
  /* Reset of all peripherals, Initializes the Flash interface and the Systick. */
  HAL_Init();

  /* USER CODE BEGIN Init */

  /* USER CODE END Init */

  /* Configure the system clock */
  SystemClock_Config();

  /* USER CODE BEGIN SysInit */

  /* USER CODE END SysInit */

  /* Initialize all configured peripherals */
  MX_GPIO_Init();
  /* USER CODE BEGIN 2 */

  /* USER CODE END 2 */

  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
    /* USER CODE END WHILE */

    /* USER CODE BEGIN 3 */

 HAL_GPIO_TogglePin (GPIOA,GPIO_PIN_5);
 HAL_Delay(500);

  }
  /* USER CODE END 3 */
}

/**
  * @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
  */
   __HAL_RCC_PWR_CLK_ENABLE();
  __HAL_PWR_VOLTAGESCALING_CONFIG(PWR_REGULATOR_VOLTAGE_SCALE2);
  /** Initializes the RCC Oscillators according to the specified parameters
  * in the RCC_OscInitTypeDef structure.
  */
  RCC_OscInitStruct.OscillatorType = RCC_OSCILLATORTYPE_HSI;
  RCC_OscInitStruct.HSIState = RCC_HSI_ON;
  RCC_OscInitStruct.HSICalibrationValue = RCC_HSICALIBRATION_DEFAULT;
  RCC_OscInitStruct.PLL.PLLState = RCC_PLL_NONE;
  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_HSI;
  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_0) != HAL_OK)
  {
    Error_Handler();
  }
}
/**
  * @brief GPIO Initialization Function
```

```
  * @param None
  * @retval None
  */
static void MX_GPIO_Init(void)
{
  GPIO_InitTypeDef GPIO_InitStruct = {0};

  /* GPIO Ports Clock Enable */
  __HAL_RCC_GPIOA_CLK_ENABLE();

  /*Configure GPIO pin Output Level */
  HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, GPIO_PIN_RESET);

  /*Configure GPIO pin : PA5 */
  GPIO_InitStruct.Pin = GPIO_PIN_5;
  GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
  HAL_GPIO_Init(GPIOA, &GPIO_InitStruct);
}
/* 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 */
```

\

```
while(1)
{
    HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_5);
    HAL_Delay(500);
}
```

This line toggles the state of pin PA5:

- If the pin is currently **HIGH (on)** → it becomes **LOW (off)**
- If the pin is **LOW (off)** → it becomes **HIGH (on)**
- It changes the output level of the LED pin each time it's called.
- Repeated toggling creates a **blinking effect** for the LED.

Internally:

- GPIOA: This refers to **GPIO Port A**
- GPIO_PIN_5: This specifies **pin 5** of that port (PA5)
- So, this command flips the logic level on **pin PA5**

This line pauses the program for 500 milliseconds (i.e., half a second).

- Without this delay, the toggle would happen so fast you wouldn't see the LED blink.
- The delay creates a visible **on/off cycle** that's slow enough for human eyes.
- Uses the **SysTick timer** to block the code execution for the specified time (500 ms).
- While this delay is active, the CPU is mostly idle (it's a blocking delay).

Output:

```
      -------------------------------------------------------------------
                        STM32CubeProgrammer v2.9.0-RC01
      -------------------------------------------------------------------

ST-LINK SN  : 066AFF544949878667203224
ST-LINK FW  : V2J39M27
Board       : NUCLEO-F401RE
Voltage     : 3.25V
SWD freq    : 4000 KHz
Connect mode: Under Reset
Reset mode  : Hardware reset
Device ID   : 0x433
Revision ID : Rev Z
Device name : STM32F401xD/E
Flash size  : 512 KBytes
Device type : MCU
Device CPU  : Cortex-M4
BL Version  : --



Memory Programming ...
Opening and parsing file: ST-LINK_GDB_server_a02916.srec
  File          : ST-LINK_GDB_server_a02916.srec
  Size          : 5740 Bytes
  Address       : 0x08000000

Erasing memory corresponding to segment 0:
Erasing internal memory sector 0
Download in Progress:


File download complete
Time elapsed during download operation: 00:00:00.441


Verifying ...



Download verified successfully


Shutting down...
Exit.
```

Conclusion:

The RTOS implementation on the STM32F401RE microcontroller using FreeRTOS and STM32CubeIDE was successfully completed. It demonstrated efficient multitasking by managing multiple tasks with precise timing and synchronization. This project provided hands-on experience with configuring peripherals, creating RTOS tasks, and debugging in an embedded environment. Overall, it deepened understanding of real-time operating systems and the STM32 architecture, serving as a foundation for more complex real-time embedded applications.

References:

file:///C:/Users/kjsce_extc208/Downloads/stm32f401re.pdf
https://www.st.com/en/development-tools/stm32cubemx.html
https://reversepcb.com/stm32cubeide/#elementor-toc__heading-anchor-1

Signature of faculty in-charge