



НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ
ІНСТИТУТ імені ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

ЗВІТ ДО ПРАКТИЧНОЇ РОБОТИ

з дисципліни

«Технології програмування на C/Embedded»
на тему «Analog to Digital Converter»

Перевірив:

Каплунов Артем
Володимирович

Виконав:

студент групи ІК-34
Могильний Микола

Київ 2025

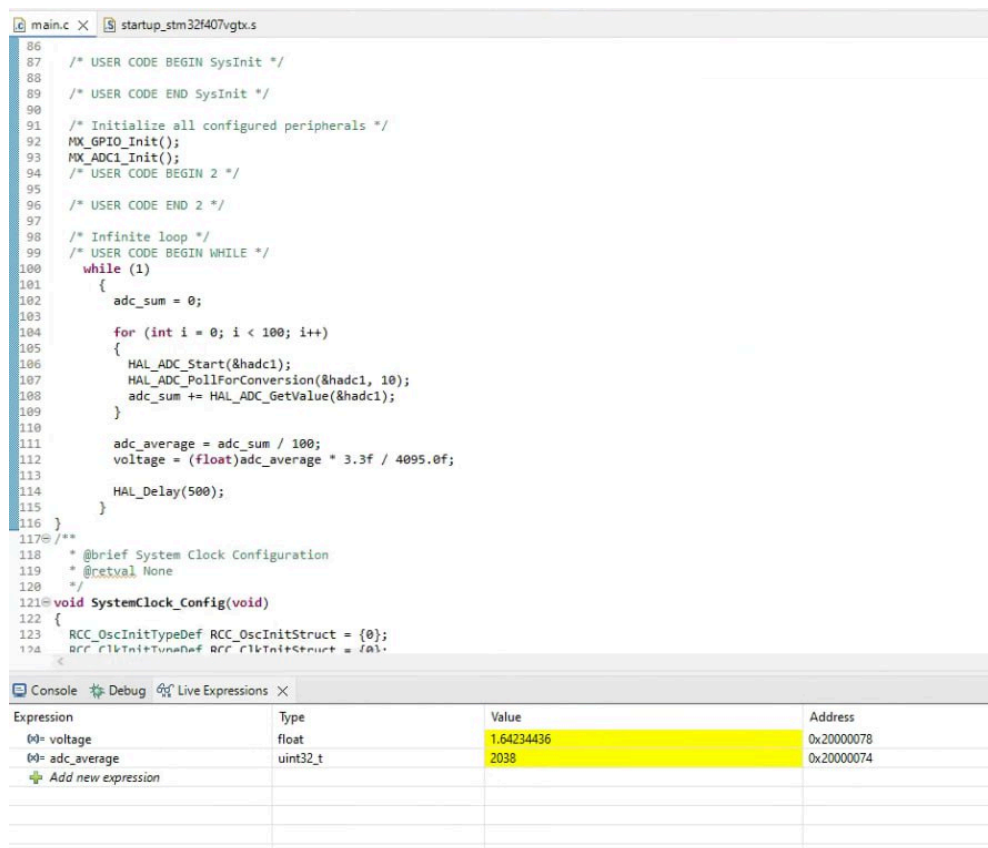
Analog to Digital Converter

Мета роботи: зрозуміти як працюють АЦП та як їх налаштовувати.

План роботи: Створити програму що, виміряє значення напруги на потенціометрі підключеного до плати. Для отримання більш точних значень, провести більш ніж 100 вимірювань і вирахувати середнє значення. До звіту додати отриманне значення - цифрове значення (отримане з ADC) і відповідне значення напруги (обраховане вами програмно).

Хід роботи

Посилання на GitHub репозиторій: [Посилання](#)



The screenshot shows an IDE with a C program for an STM32 microcontroller. The code initializes the ADC, performs 100 readings, calculates an average, and converts it to a voltage. Below the code, a 'Live Expressions' table displays the current values of 'voltage' and 'adc_average'.

```
86
87 /* USER CODE BEGIN SysInit */
88
89 /* USER CODE END SysInit */
90
91 /* Initialize all configured peripherals */
92 MX_GPIO_Init();
93 MX_ADC1_Init();
94 /* USER CODE BEGIN 2 */
95
96 /* USER CODE END 2 */
97
98 /* Infinite Loop */
99 /* USER CODE BEGIN WHILE */
100 while (1)
101 {
102     adc_sum = 0;
103
104     for (int i = 0; i < 100; i++)
105     {
106         HAL_ADC_Start(&hadc1);
107         HAL_ADC_PollForConversion(&hadc1, 10);
108         adc_sum += HAL_ADC_GetValue(&hadc1);
109     }
110
111     adc_average = adc_sum / 100;
112     voltage = (float)adc_average * 3.3f / 4095.0f;
113
114     HAL_Delay(500);
115 }
116 }
117 /**
118  * @brief System Clock Configuration
119  * @retval None
120  */
121 void SystemClock_Config(void)
122 {
123     RCC_OscInitTypeDef RCC_OscInitStruct = {0};
124     RCC_ClkInitTypeDef RCC_ClkInitStruct = {0};
125 }
```

| Expression | Type | Value | Address |
|--------------------|----------|------------|------------|
| (*) voltage | float | 1.64234436 | 0x20000078 |
| (*) adc_average | uint32_t | 2038 | 0x20000074 |
| Add new expression | | | |
| | | | |
| | | | |
| | | | |
| | | | |

Зчитування сигналу: Налаштовано АЦП (ADC) для вимірювання напруги на потенціометрі (0...3.3 В). Фільтрація даних: Застосовано алгоритм усереднення (середнє арифметичне зі 100 вимірів) для стабілізації показників та усунення шумів. Генерація ШІМ (PWM): Налаштовано таймер TIM4 у режимі генерації широтно-імпульсної модуляції. Період таймера (ARR) синхронізовано з розрядністю АЦП (4095), що спрощує розрахунки. Керування яскравістю: Реалізовано пряму залежність — відфільтроване значення з АЦП записується в регістр порівняння таймера (CCR). Це змінює шпаруватість імпульсів пропорційно повороту ручки:

0 В → Шпаруватість 0% (світлодіод вимкнений).

3.3 В → Шпаруватість 100% (максимальна яскравість).

