Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Кафедра ЭВМ

Лабораторная работа № 3

«Тактирование. Питание. Энергопотребление»

Вариант №1

Выполнил:

Студент группы 150504 Горбачевский М.В.

Проверил:

ассистент каф. ЭВМ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Шеменков В.В.

Минск 2024

1. Цель работы

**Цель:** изучить принципы построения систем с низким энергопотреблением на базе микроконтроллера MSP430F5529.

**Задача:** написать программу, переключающую режимы тактирования, питания и энергопотребления в соответствии с заданием.

1. Исходные данные

Для выполнения работы используется плата MSP-EXP430F5529 и интегрированная среда разработки Code Composer Studio.

Необходимо выполнить задание варианта № 1.

| **LPM** | **Vcore, В** | **Частота** |
| --- | --- | --- |
| 0 | 1,9 — 1,8 | DCOCLK = 0,3 МГц, ACLK = смена источника VLOCLK / DCOCLK |

1. Выполнение работы
   1. Изменение частоты питания и энергопотребления

Для выполнения лабораторной работы следовало измерить Vcc, ток и частоту. Vcc = 3.6 B (см. рисунок 3.2.1).



Рисунок 3.1.1 – Напряжение



Рисунок 3.1.2 – Ток в энергосбережении



Рисунок 3.1.3 – Ток не в энергосбережении

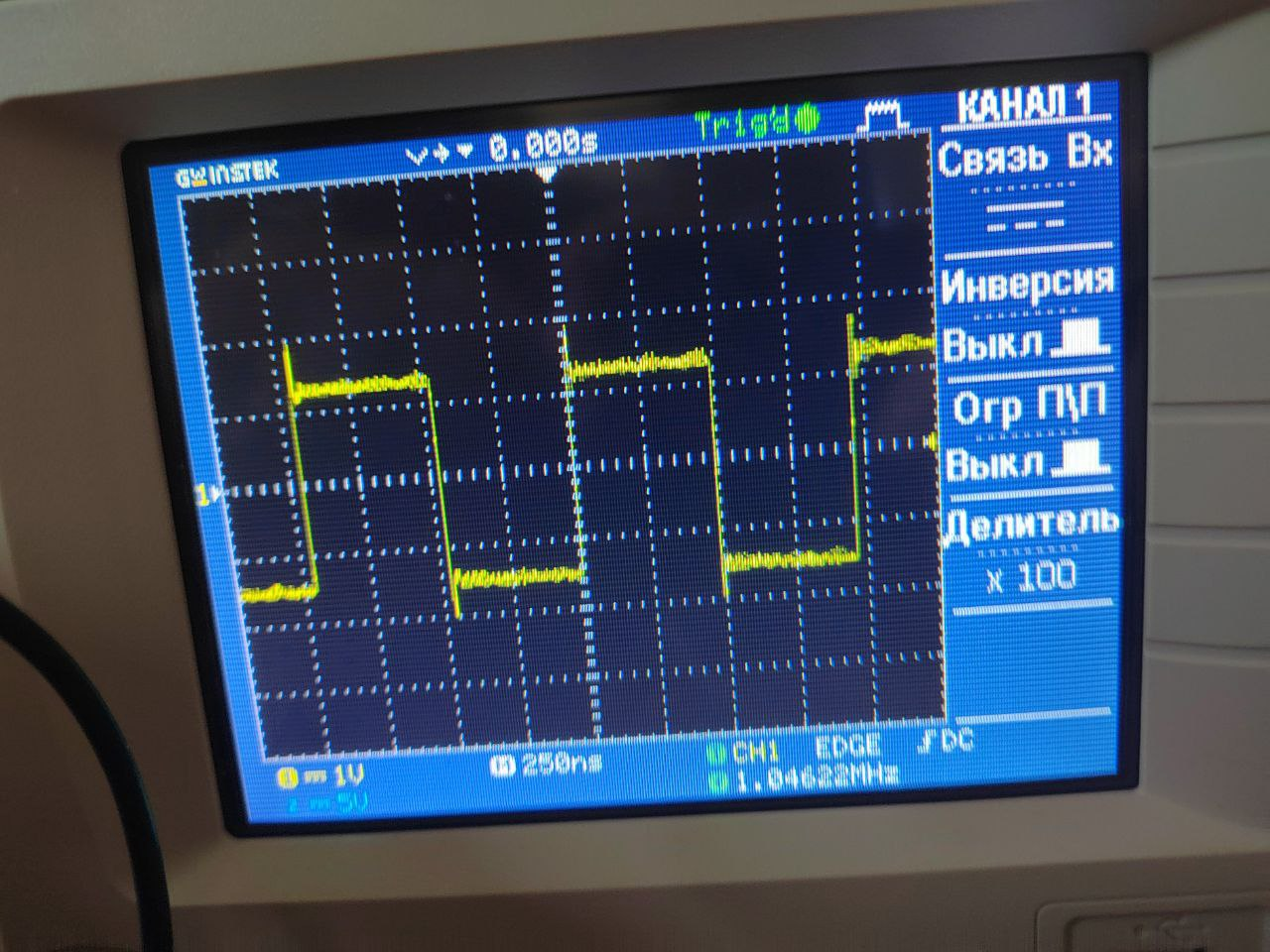


Рисунок 3.1.4 – Частота

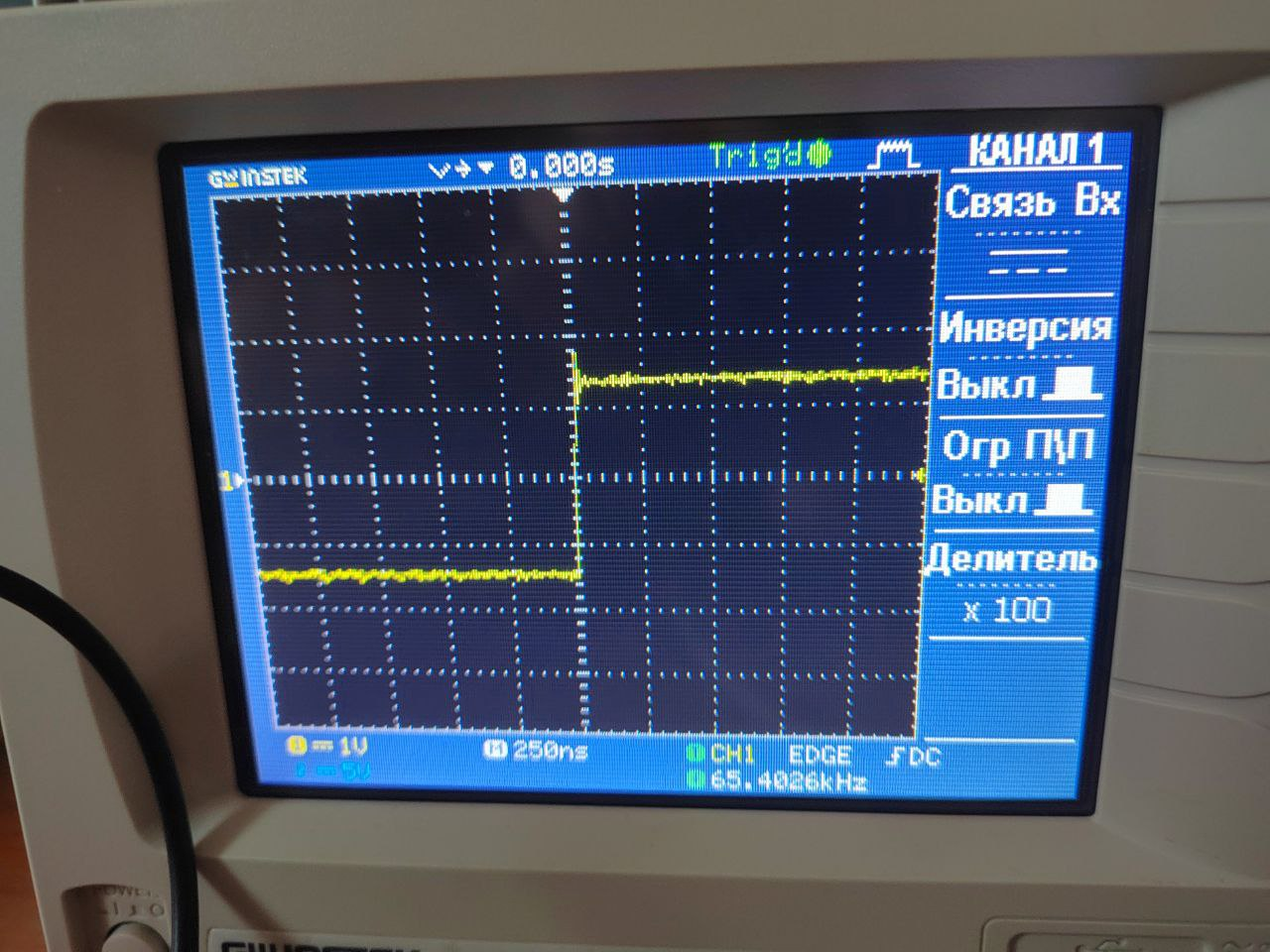


Рисунок 3.1.5 – Частота

* 1. Код программы

**#include** <msp430.h>

// Определение светодиодов

**#define** LED1 BIT0 // P1.0 Активный режим

**#define** LED2 BIT1 // P8.1

**#define** LED3 BIT2 // P8.2 Пониженное энергопотребление

**#define** LED4 BIT1 // P1.1 Высокое напряжение

**#define** LED5 BIT2 // P1.2 Низкое напряжение

**#define** LED6 BIT3 // P1.3

**#define** LED7 BIT4 // P1.4

**#define** LED8 BIT5 // P1.5 Управление через таймер

// Кнопки

**#define** BUTTON\_S1 BIT7 // P1.7

**#define** BUTTON\_S2 BIT2 // P2.2

**volatile** **unsigned** **int** isLowPowerMode = 0; // Флаг для режима пониженного энергопотребления

**void** **setupClocks**() {

// Настройка тактовых генераторов

UCSCTL3 |= SELREF\_2; // FLL reference = REFO

UCSCTL4 |= SELA\_2; // ACLK = REFO

**\_\_bis\_SR\_register**(SCG0); // Disable the FLL control loop

UCSCTL0 = 0x0000; // Set DCO to lowest Tap

UCSCTL1 = DCORSEL\_4; // Set DCO range for 3.1 MHz

UCSCTL2 = FLLD\_1 + 0x1F4; // Set DCOCLK = 3.1 MHz

**\_\_bic\_SR\_register**(SCG0); // Enable the FLL control loop

**\_\_delay\_cycles**(250000);

}

**void** **setupLEDs**() {

P1DIR |= LED1 + LED4 + LED5 + LED6 + LED7 + LED8; // Настраиваем светодиоды на вывод

P8DIR |= LED2 + LED3; // Настраиваем светодиоды на вывод для P8

P1OUT &= ~(LED1 + LED4 + LED5 + LED6 + LED7 + LED8); // Выключаем все светодиоды

P8OUT &= ~(LED2 + LED3);

}

**void** **setupTimer**() {

TA0CTL = TASSEL\_2 | MC\_1 | TACLR; // SMCLK, Up Mode

TA0CCR0 = 12000; // Установка периода таймера

TA0CCTL0 |= CCIE; // Разрешаем прерывание от таймера

}

**#pragma** vector=TIMER0\_A0\_VECTOR

\_\_interrupt **void** **Timer\_A**(**void**) {

P1OUT ^= LED8; // Переключаем LED8

}

// Функция переключения частоты и напряжения

**void** **switchFrequency**() {

**static** **int** freq\_step = 0;

**if** (freq\_step == 0) {

// Устанавливаем низкое напряжение Vcore = 1.8V и частоту DCO / 16

PMMCTL0\_H = PMMPW\_H; // Разблокируем регистр управления питанием

PMMCTL0\_L = PMMCOREV\_0; // Устанавливаем низкое напряжение (Vcore = 1.8V)

UCSCTL1 = DCORSEL\_2; // Устанавливаем DCO на низкую частоту

UCSCTL2 = FLLD\_4 + 0x1F4; // DCOCLK / 16

P1OUT |= LED5; // Включаем LED5 для индикации низкого напряжения

P1OUT &= ~LED4; // Отключаем LED4 (высокое напряжение)

} **else** {

// Устанавливаем высокое напряжение Vcore = 1.9V и частоту DCO = 3.1 МГц

PMMCTL0\_H = PMMPW\_H; // Разблокируем регистр управления питанием

PMMCTL0\_L = PMMCOREV\_1; // Устанавливаем высокое напряжение (Vcore = 1.9V)

UCSCTL1 = DCORSEL\_4; // Устанавливаем DCO на высокую частоту

UCSCTL2 = FLLD\_1 + 0x1F4; // DCOCLK = 3.1 MHz

P1OUT |= LED4; // Включаем LED4 для индикации высокого напряжения

P1OUT &= ~LED5; // Отключаем LED5 (низкое напряжение)

}

freq\_step = (freq\_step + 1) % 2;

}

**void** **enterLowPowerMode**() {

P8OUT |= LED3; // Включаем LED3 (энергосбережение)

P1OUT &= ~LED1; // Выключаем LED1

isLowPowerMode = 1; // Устанавливаем флаг

**\_\_bis\_SR\_register**(LPM1\_bits + GIE); // Входим в LPM1 с разрешением прерываний

}

**void** **exitLowPowerMode**() {

P1OUT |= LED1; // Включаем LED1 (активный режим)

P8OUT &= ~LED3; // Выключаем LED3

isLowPowerMode = 0;

}

**int** **main**(**void**) {

WDTCTL = WDTPW | WDTHOLD; // Останавливаем сторожевой таймер

setupClocks(); // Настраиваем тактовые частоты

setupLEDs(); // Настраиваем светодиоды

setupTimer(); // Настраиваем таймер

P1DIR &= ~BUTTON\_S1; // Настраиваем P1.7 как вход (Кнопка S1)

P2DIR &= ~BUTTON\_S2; // Настраиваем P2.2 как вход (Кнопка S2)

P1IE |= BUTTON\_S1; // Прерывание по кнопке S1 (P1.7)

P1IES |= BUTTON\_S1;

P1IFG &= ~BUTTON\_S1;

P2IE |= BUTTON\_S2; // Прерывание по кнопке S2 (P2.2)

P2IES |= BUTTON\_S2;

P2IFG &= ~BUTTON\_S2;

**\_\_bis\_SR\_register**(GIE); // Разрешаем глобальные прерывания

**while** (1) {

**if** (isLowPowerMode == 0) {

**\_\_no\_operation**(); // Основной цикл

}

}

}

**#pragma** vector=PORT1\_VECTOR

\_\_interrupt **void** **Port\_1**(**void**) {

**if** (P1IFG & BUTTON\_S1) { // Обработка нажатия кнопки S1

switchFrequency(); // Переключаем частоту и напряжение

P1IFG &= ~BUTTON\_S1;

**return**;

}

}

**#pragma** vector=PORT2\_VECTOR

\_\_interrupt **void** **Port\_2**(**void**) {

**if** (P2IFG & BUTTON\_S2) {

P2IFG &= ~BUTTON\_S2;

**if** (isLowPowerMode) {

exitLowPowerMode();

**return**;// Выходим из LPM1

} **else** {

enterLowPowerMode();

**return**; // Входим в LPM1

}

}

}

1. Выводы

В ходе выполнения лабораторной работы удалось изучить принципы построения систем с низким энергопотреблением на базе микроконтроллера MSP430F5529. Удалось написать программу, переключающую режимы тактирования, питания и энергопотребления в соответствии с заданием.