Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Кафедра ЭВМ

Лабораторная работа № 2

«Прерывания. Таймеры»

Вариант №2

Выполнил:

ст. гр. 050503 Русель А.А.

Проверил:

ассистент Шеменков В.В.

Минск 2023

1. Цель работы

Цели работы:

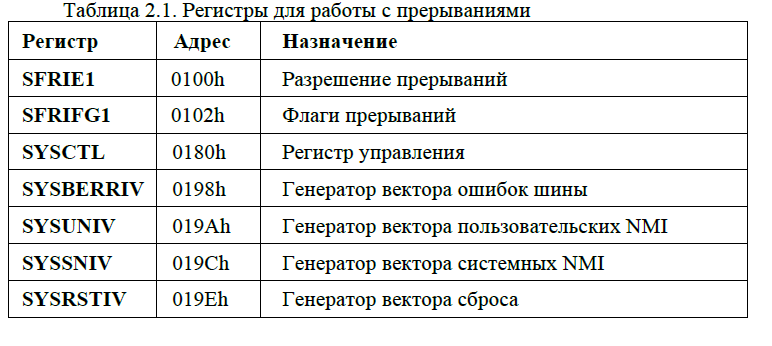
Ознакомиться с работой подсистемы прерываний и таймерами микроконтроллера MSP430F5529.

1. Исходные данные

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

1. Теоретические сведения

Различают системные немаскируемые (SMNI), пользовательские немаскируемые (UNMI) и маскируемые прерывания. К системным немаскируемым относятся: сигнал RST/NMI в режиме NMI, сбой генератора, ошибка доступа Flash памяти. К пользовательским немаскируемым – сбой напряжения питания (от подсистемы PMM), доступ к несуществующей (vacant) памяти, события с буфером (mailslot) JTAG интерфейса. Маскируемые прерывания могут быть отключены (замаскированы) индивидуально или все сразу (бит GIE регистра состояния SR).





MSP430F5529 содержит 32-разрядный сторожевой таймер WDT (базовый адрес 015Сh), 3 таймера TAx (базовые адреса соответственно 0340h, 0380h, 0400h), таймер TBx (базовый адрес 03C0h) и таймер часов реального времени RTC\_A (базовый адрес 04A0h).

1. Выполнение работы
   1. Задание

В соответствии с вариантом, используя прерывания и таймеры, запрограммировать кнопки и светодиоды. Для работы с кнопками использовать только прерывания. Не использовать опросы флагов состояния в цикле и циклы задержки (активное ожидание). Не допускается использовать иные заголовочные файлы, кроме msp430, не допускается также использовать высокоуровневые библиотеки. При выполнении задания особое внимание уделить грамотному выбору режима работы таймера. Комментарии в тексте программы обязательны, они должны пояснять что именно делает данные фрагмент.

* 1. Программа по управлению цифровым вводом-выводом с использованием прерываний

#include <msp430.h>

#define LED1 BIT0

#define LED2 BIT1

#define LED3 BIT2

#define BUTTON BIT7

static volatile unsigned int button\_intr = 0;

static volatile unsigned int timer\_counter = 0;

void main(void) {

// отключаем сторожевой таймер

WDTCTL = WDTPW | WDTHOLD;

// устанавливаем пины LED1, LED2 и LED3 как выходы

P1DIR |= LED1;

P8DIR |= (LED2 | LED3);

P8OUT &= ~(LED2 | LED3);

// выключаем все светодиоды

P1OUT &= ~LED1;

P8OUT &= ~(LED2 | LED3);

// устанавливаем пин BUTTON как вход

P1DIR &= ~BUTTON;

// включаем подтягивающий резистор на пине BUTTON

P1REN |= BUTTON;

// устанавливаем подтягивающий резистор на пине BUTTON

P1OUT |= BUTTON;

P1IE |= BUTTON; // Разрешение прерывания для пина BUTTON

P1IES |= BUTTON; // Настройка прерывания по спаду

P1IFG = 0;

TA1CCTL0 = CCIE; // Разрешение прерывания для таймера TA1

TA1CCR0 = 1047;

TA1CTL = TASSEL\_2 + MC\_1 + TACLR;

\_\_enable\_interrupt(); // Разрешение глобальных прерываний

\_\_no\_operation();

}

#pragma vector=PORT1\_VECTOR

\_\_interrupt void Port1\_ISR(void) {

button\_intr = 1;

P1IFG &= ~BIT7;

}

#pragma vector=TIMER1\_A0\_VECTOR

\_\_interrupt void Timer1\_A0\_ISR(void) {

static int is\_button\_pressed = 0;

static int debounce\_total\_counter = 0;

static int debounce\_pressed\_counter = 0;

static int pressed\_count = 0;

static int is\_counter\_allowed = 0;

static int last\_pressed\_time\_counter = 0;

timer\_counter++;

if(button\_intr){

P1IE &= ~BIT7;

debounce\_total\_counter++;

debounce\_pressed\_counter += (P1IN & BIT7) == 0;

if(debounce\_total\_counter == 10){

is\_button\_pressed = debounce\_pressed\_counter > 7;

if (is\_button\_pressed) {

is\_counter\_allowed = 1;

last\_pressed\_time\_counter = timer\_counter;

pressed\_count++;

}

button\_intr = 0;

debounce\_total\_counter = 0;

debounce\_pressed\_counter = 0;

if(is\_button\_pressed)

P1IES &= ~BIT7;

else

P1IES |= BIT7;

P1IE |= BIT7;

}

}

if (!is\_counter\_allowed) {

timer\_counter = 0;

}

if (timer\_counter - last\_pressed\_time\_counter > 2000) {

if (pressed\_count == 1) {

P1OUT ^= LED1;

} else if (pressed\_count == 2) {

P8OUT ^= LED2;

} else if (pressed\_count == 3) {

P8OUT ^= LED3;

}

pressed\_count = 0;

timer\_counter = 0;

is\_counter\_allowed = 0;

last\_pressed\_time\_counter = 0;

}

}

1. Выводы

В ходе выполнения лабораторной работы удалось ознакомиться с интегрированной средой разработки Code Composer Studio и с основными функциональными возможностями платы MSP-EXP430F5529. Удалось написать программу по управлению цифровым вводом-выводом (светодиодами и кнопками) в соответствии с вариантом №2 с использованием прерываний.