Министерство образования и науки
Российской Федерации

Федеральное государственное бюджетное образовательное
учреждение высшего образования
ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра конструирования узлов и деталей радиоэлектронной аппаратуры
(КУДР)

А.А. Бомбизов, А.Г. Лощилов

ВНЕШНИЕ ПРЕРЫВАНИЯ

Методические указания к выполнению
лабораторной и самостоятельной работы
по дисциплине «Программирование микроконтроллеров»

Томск 2017

1 Введение

В предыдущих работах были освоены основные принципы функционирования портов ввода-вывода: организация вывода дискретных сигналов на выводы микроконтроллера; опрос во время выполнения программы состояния вывода, настроенного на вход. Если основной алгоритм достаточно сложный и/или медлительный, то опрос состояния порта (в частности кнопки) может производиться не своевременно, что может привести к пропуску событий.

Целью настоящей работы является освоение принципов работы контроллера внешних прерываний, с использованием которого программа может с малыми задержками получать информацию о внешних событиях независимо от выполняемого в текущий момент алгоритма.

2 Краткая теория

Прерывание (англ. interrupt) – сигнал от программного или аппаратного обеспечения, сообщающий процессору о наступлении какого-либо события, требующего немедленного внимания. Процессор отвечает приостановкой своей текущей активности, сохраняя свое состояние и выполняя функцию, называемую обработчиком прерывания (или программой обработки прерывания), который реагирует на событие и обслуживает его, после чего возвращает управление в прерванный код.

Преимущества прерываний

1. При использовании прерываний, нет необходимости писать код в loop (), постоянно проверяющий высокоприоритетное условие.

2. Не нужно беспокоиться о медленной реакции или пропущенных нажатиях кнопок из-за слишком долго выполняющихся подпрограмм.

3. Процессор будет автоматически останавливать то, что он делает в момент возникновения прерывания и вызвать ваш обработчик прерывания. Вам просто нужно написать код, отвечающий на прерывание всякий раз, когда оно происходит.

Свойства прерываний

1. У каждого периферийного устройства, что входит в состав микроконтроллера, есть как минимум один источник прерывания (Interrupt source).

2. За каждым прерыванием, строго закреплен вектор (ссылка) указывающий на процедуру обработки прерывания (Interrupt service routine). Все векторы прерываний, располагаются в самом начале памяти программ и вместе формируют “таблицу векторов прерываний” (Interrupt vectors table).

3. Каждому прерыванию соответствует определенный “бит активации прерывания” (Interrupt Enable bit). Чтобы использовать определенное прерывание, следует записать в его “бит активации прерывания” – лог. единицу.

4. Независимо от того активировали Вы или нет определенные прерывания, микроконтроллер не начнет обработку этих прерываний, пока в “бит всеобщего разрешения прерываний” (Global Interrupt Enable bit в регистре состояния SREG) не будет записана лог. единица. Также, чтобы запретить все прерывания (на неопределенное время), в бит всеобщего разрешения прерываний следует записать – лог. нуль (рисунок 1).

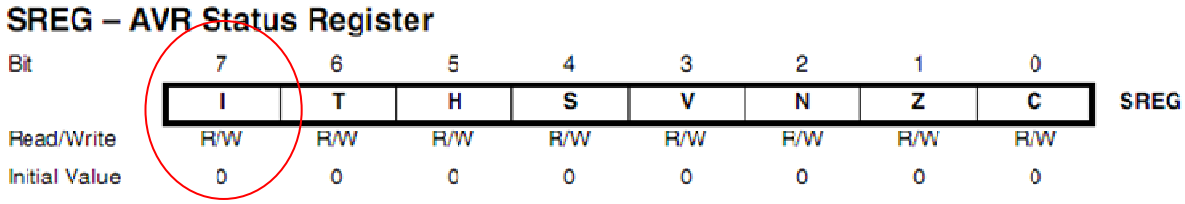

Рисунок 1– Регистр SREG

Существует альтернативный способ разрешения или запрещения внешнего прерывания – это использование следующих функций:

sei() – глобальное разрешение прерываний (установка бита 7 (I));

cli() – глобальное отключение прерываний (обнуление бита 7 (I)).

Всего микроконтроллер Atmega328 может обрабатывать прерывания от 26 источников. Все они перечислены в таблице 1 .

Таблица 1 – Таблица прерываний

Вектор	**Название**	**Метка обработчика**	**Описание**
1	RESET	RESET_vect	External Pin, Power-on Reset, Brown-out Reset and Watchdog System Reset
2	INT0	INT0_vect	Внешнее прерывание 0
3	INT1	INT1_vect	Внешнее прерывание 1
4	PCINT0	PCINT0_vect	Прерывание 0 по изменению состояния порта B
5	PCINT1	PCINT1_vect	Прерывание 1 по изменению состояния порта C
6	PCINT2	PCINT2_vect	Прерывание 2 по изменению состояния порта D
7	WDT	WDT_vect	Сторожевой таймер (если используется в качестве источника прерывания)
8	TIMER2_COMPA	TIMER2_COMPA_vect	Прерывание по сравнению, канал A таймера/счетчика 2
9	TIMER2_COMPB	TIMER2_COMPB_vect	Прерывание по сравнению, канал B таймера/счетчика 2

Таблица 1 – Продолжение

Вектор	**Название**	**Метка обработчика**	**Описание**
10	TIMER2_OVF	TIMER2_OVF_vect	Прерывание по переполнению таймера/счетчика 2
11	TIMER1_CAPT	TIMER1_CAPT_vect	Прерывание таймера/счетчика 1 по захвату
12	TIMER1_COMPA	TIMER1_COMPA_vect	Прерывание по сравнению, канал А таймера/счетчика 1
13	TIMER1_COMPB	TIMER1_COMPB_vect	Прерывание по сравнению, канал В таймера/счетчика 1
14	TIMER1_OVF	TIMER1_OVF_vect	Прерывание по переполнению таймера/счетчика 1
15	TIMER0_COMPA	TIMER0_COMPA_vect	Прерывание по сравнению, канал А таймера/счетчика 0
16	TIMER0_COMPB	TIMER0_COMPB_vect	Прерывание по сравнению, канал В таймера/счетчика 0
17	TIMER0_OVF	TIMER0_OVF_vect	Прерывание по переполнению таймера/счетчика 0
18	SPI	SPI_STC_vect	Завершение передачи по последовательному каналу SPI
19	USART_RX	USART_RX_vect	Завершение приема по каналу USART
20	USART_UDRE	USART_UDRE_vect	Регистр данных USART пуст
21	USART_TX	USART_TX_vect	Завершение передачи по каналу USART
22	ADC	ADC_vect	Преобразование АЦП завершено
23	EE_READY	EE_READY_vect	EEPROM готова
24	ANALOG_COMP	ANALOG_COMP_vect	Аналоговый компаратор переключился
25	TWI	TWI_vect	Событие двухпроводного интерфейса (I2C)
26	SPM_READY	SPM_READY_vect	Готовность SPM

В рамках настоящей темы будут освоены только внешние прерывания от портов ввода-вывода, так как они являются наглядной демонстрацией работы.

Внешние прерывания от портов ввода-вывода позволяют отслеживать изменение состояния на определенных выводах микроконтроллера и бывают двух видов:

- INT0 и INT1;
- PCINT0, PCINT1, PCINT2.

Прерывания INT0 и INT1 могут отслеживать следующие изменения состояния вывода:

- передний фронт импульса;
- задний фронт импульса;

- передний и задний фронты одновременно (любое изменение уровня);
- низкий уровень.

Ввиду сложности технической реализации этих возможностей данный вид прерываний позволяет обрабатывать изменение состояния только лишь у двух выводов микроконтроллера PD2 (INT0), PD3 (INT1). (Рисунок 2, выделено красным).

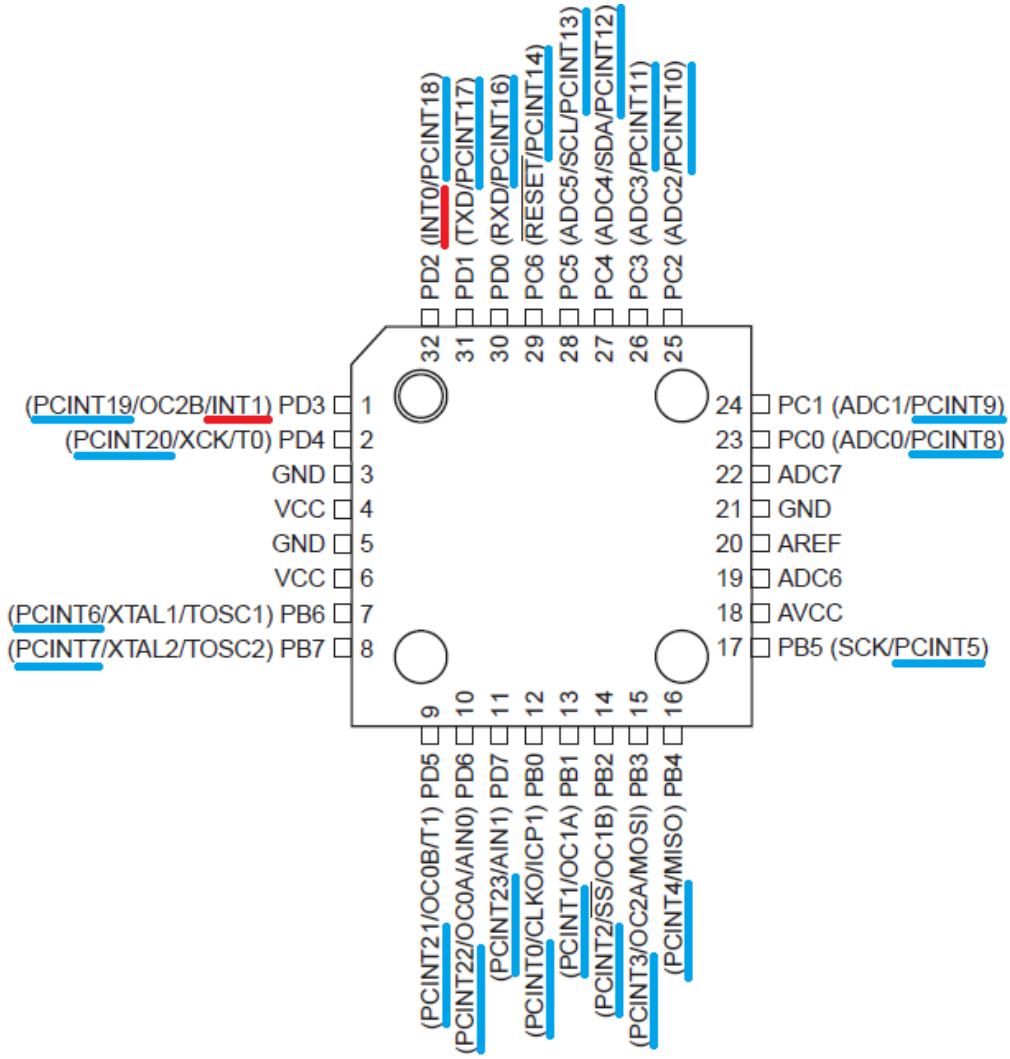

Рисунок 2– Микроконтроллер ATMEGA328 в корпусе TQFP32

Настройка данного типа прерываний осуществляется путём конфигурации регистров EIMSK и EICRA. Регистр EIMSK отвечает за выбор необходимого вывода для обработки. Для этого достаточно установить в логическую единицу требуемый бит (рисунок 3).

EIMSK – External Interrupt Mask Register

Bit	7	6	5	4	3	2	1	0	
0x1D (0x3D)	–	–	–	–	–	–	INT1	INT0	EIMSK
Read/Write	R	R	R	R	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Рисунок 3– Регистр EIMSK

Следующий регистр EICRA отвечает за выбор события, по которому будет вызвано прерывание (рисунок 4).

EICRA – External Interrupt Control Register A

The External Interrupt Control Register A contains control bits for interrupt sense control.

Bit	7	6	5	4	3	2	1	0	
(0x69)	–	–	–	–	ISC11	ISC10	ISC01	ISC00	EICRA
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Рисунок 4– Регистр EICRA

Данный регистр содержит две пары конфигурационных битов: ISC00/ ISC01 и ISC10/ ISC11. Комбинации битов для выбора необходимой реакции на внешнее воздействие описаны в таблице 2.

Таблица 2 – Режимы работы внешних прерываний (где x=0 или 1)

ISCx1	**ISCx0**	**Описание**
0	0	по появлению нижнего уровня
0	1	по изменению уровня
1	0	По спадающему фронту
1	1	По нарастающему фронту

Другим типом прерываний от портов ввода-вывода являются прерывания PCINT0, PCINT1, PCINT2. Их преимуществом перед INT0/ INT1 является то, что они подключены ко всем портам ввода-ввода. Но при этом они срабатывают только на изменение уровня, что является недостатком.

Настройка данного типа прерываний осуществляется путём конфигурации регистров PCICR и PCMSK0/PCMSK1/PCMSK2.

Разрешение внешнего прерывания от определенного порта осуществляется путем установки соответствующего бита в единицу в регистре PCICR (рисунок 5).

PCICR – Pin Change Interrupt Control Register

Bit	7	6	5	4	3	2	1	0	
(0x68)	–	–	–	–	–	PCIE2	PCIE1	PCIE0	PCICR
Read/Write	R	R	R	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Рисунок 5– Регистр PCICR

Где PCIE0 – флаг включения прерывания PCINT0 (PORT B);

PCIE1 – флаг включения прерывания PCINT1 (PORT C);

PCIE2 – флаг включения прерывания PCINT2 (PORT D).

После определения порта необходимо выбрать выводы, за которыми должен следить микроконтроллер. За обработку каждого вывода отвечает соответствующий бит в регистрах PCMSK0– PCMSK2 (рисунок 6).

PCMSK0 – Pin Change Mask Register 0

Bit	7	6	5	4	3	2	1	0	
(0x6B)	PCINT7	PCINT6	PCINT5	PCINT4	PCINT3	PCINT2	PCINT1	PCINT0	PCMSK0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Рисунок 6 – Регистр PCMSK0

Помимо настройки микроконтроллера на генерацию прерываний на внешние события необходимо создать обработчики.

Обработчик прерываний (или процедура обслуживания прерываний) – это специальная процедура, вызываемая по прерыванию для выполнения его обработки.

Во время генерации прерывания ядро микроконтроллера по вектору прерывания переходит в определенную область flash-памяти для определения адреса функции – обработчика. Порядок размещения адресов в памяти должен соответствовать таблице 1 . Разрабатывая программу в среде Arduino, разработчику не требуется описывать всю таблицу прерываний. Она уже определена в используемых библиотеках. Поэтому для определения обработчика прерываний ISR (Interrupt Service Routine) достаточно только описать функцию обработки в следующем формате:

```
ISR(Метка обработчика)
{
    // код обработки
    // код обработки
    // код обработки
}
```

Где *ISR* – специальный макрос, определенный системой, а *Метка обработчика* – это идентификатор, взятый из соответствующего столбца в таблице 1 .

Например, обработчик для прерывания INT0 должен быть записан следующим образом.

```
ISR(INT0_vect)
{
    // код обработки
    // код обработки
    // код обработки
}
```

3 Порядок выполнения работы

Практическая часть является продолжением предыдущей работы по реализации программы счетчика. В рамках данной темы необходимо снять нагрузку опроса состояния кнопок с основного рабочего цикла программы и переложить её на внешние прерывания.

Для этого необходимо выполнить следующие действия:

3.1 Изучите предложенный в п. 2 теоретический материал.

3.2 Скопируйте проект, созданный на предыдущем занятии в новую папку.

3.3 Удалите из функции *loop()* обработку кнопок.

3.4 Определите по [1] номера прерываний PCINT для осуществления обработки кнопок S1–S3.

3.5 В функции setup() сконфигурируйте необходимые регистры для обработки прерывания PCINT, соответствующие нажатию кнопок S1–S3.

3.6 Разрешите прерывание глобально.

3.7 Создайте функцию – обработчик прерывания PCINT.

3.8 В обработчике прерывания выполните чтение состояний выводов микроконтроллера, соответствующих кнопкам S1–S3.

3.9 Если по результатам чтения состояния определено, что некоторая кнопка была нажата, то выполните соответствующее изменение значения переменной счетчика.

3.10 Оформите отчет, содержащий титульный лист и разделы: введение, ход выполнения работы, ответы на контрольные вопросы и выводы.

3.11 Защитите отчет у преподавателя.

4 Контрольные вопросы

4.1 Какой бит и в каком регистре отвечает за глобальное разрешение прерываний?

4.2 Какой требуется макрос для определения прерывания?

4.3 Возможно ли обработать прерывание типа INTx от вывода PB2 микроконтроллера?

4.4 Сколько различных событий возможно обрабатывать с использованием PCINTx?

4.5 Сколько различных событий возможно обрабатывать с использованием INTx?

Список литературы

1. UNO Schematic – Arduino.– URL: https:\\www.arduino.cc\en\uploads\Main\arduino-uno-schematic.pdf (дата обращения: 10.01.2017).

2. ATMEL 8-BIT MICROCONTROLLER WITH 4/8/16/32KBYTES, IN-SYSTEM PROGRAMMABLE FLASH.– URL: http:\\www.atmel.com\images\Atmel-8271-8-bit-AVR-Microcontroller-ATmega48A-48PA-88A-88PA-168A-168PA-328-328P_datasheet_Complete.pdf (дата обращения: 10.01.2017).