МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «ЛИПЕЦКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Институт компьютерных наук
Кафедра автоматизированных систем управления

Лабораторная работа

по микропроцессорным системам №3

“Подсистема прерываний и таймеры”

Студент АС-21-1 _____ Станиславчук С. М.

(подпись, дата)

Руководитель

Ст. преподаватель _____ Болдырихин О. В.

(подпись, дата)

Задание кафедры:

Исследовать порядок обработки прерываний и работы таймеров. Разобрать последовательность инициализирующих действий для использования прерываний. Подробно рассмотреть последовательность действий микроконтроллера при возникновении прерывания для перехода к его обработчику, а также при возврате из обработчика. Изучить характеристики таймеров, их структуру, регистры, источники прерываний, режимы работы, порядок их инициализации и использования.

Для проведения исследования написать программу на языке ассемблера, выполняющую основную задачу в обработчике прерывания таймера. Общая схема программы:

— В начале памяти команд расположена таблица векторов прерываний, в которой, по крайней мере, имеются Reset-вектор и вектор заданного прерывания таймера;

— Reset-обработчик проводит необходимую инициализацию и вводит процессор в бесконечный цикл;

— Ввод-обработка-вывод выполняются в обработчике прерывания.

Вариант 7

Задача, выполняемая в обработчике прерывания: преобразование числа в код с контролем по нечетности

Прерывание таймера: совпадение T0

Режим работы таймера: Normal

Листинг программы:

```
.include "m8535def.inc" ; Includes a file with register and constant definitions for the m8535 microcontroller.

.cseg ; Beginning of a code segment.

.org 0 ; Instructs the compiler to start program code at address 0.

   ; Set the address of the TIMER0_COMPA interrupt vector.
   .org 0x0013
   rjmp TIMER0_COMPA_vect

init: ; Label of the entry point into the program.

   ldi r16, 0xFF      ; Load the value 0xFF into register r16.
   out DDRA, r16      ; Configure Port A to Output.

   ldi r16, 0         ; Load the value 0 into register r16.
   out DDRC, r16      ; Configure port C for input.

   ; Timer 0 setup.
   ldi r16, (1<<CS02) | (1<<CS00)  ; Set the prescaler to 1024.
   out TCCR0, r16                  ; Write the settings to TCCR0.
   ldi r16, 100                    ; Load the comparison value.
   out OCR0, r16                   ; Write to comparison register.

   ; Enable interrupts from Timer 0 Counter Matches.
   ldi r16, (1<<OCIE0)  ; Enable interrupt by comparison.
   out TIMSK, r16       ; Write settings to TIMSK.

   sei                  ; Global interrupt enable.
```

```
    ; Stack initialization.
    ldi r16, low(0x22F)   ; Load the low byte of the stack address into register r16.
    out spl, r16          ; Set the low byte of the stack address.

    ldi r16, high(0x22F)  ; Load the high byte of the stack address into register r16.
    out sph, r16          ; Set the high byte of the stack address.

main: ; The main part of the program.

    rjmp main             ; Endless cycle.

; TIMER0_COMPA interrupt handler.
TIMER0_COMPA_vect:
    ; Here's your code to handle the interrupt.
    ; Reading a number from port C and converting it to odd parity.
    in r16, PINC        ; Reading a number from port C.

    ; Converting a number to an odd parity code.
    ldi r18, 0          ; Initialize the counter to count single bits.
    mov r17, r16        ; Copy the value of the number into register r17.

    ; Loop counting unit bits in a number.
    ccl:                ; Cycle label.
        sbrc r17, 0     ; Checking if the least significant bit of a number is one.
        inc r18         ; Increment the counter if the least significant bit is one.
        lsr r17         ; Shift the number to the right.
        brne ccl        ; Repeat the loop if the number is not zero.
```

; Set the most significant bit in register r16 depending on the parity of the number of unit bits.

```
    sbrs r18, 0         ; Checking if the number of one bits is odd.
    ori r16, 0x80       ; Setting the most significant bit in register r16 if the number of one bits is odd.

    ; Output the resulting code with odd parity to port A.
    out PORTA, r16      ; Output the result to port A.

    reti                ; Return from interrupt.
```

Обработка прерываний:

Архитектура AVR насчитывает 21 источник прерываний, которые представлены в таблице 1. Таблица векторов прерываний находится в начале адресного пространства памяти команд, однако может быть перемещена в начало загрузочной секции, и чем меньше адрес, тем больше приоритет прерывания.

Таблица 1 - Таблица векторов прерываний и сброса

№	Адрес	Источник	Описание
1	0x00	RESET	Сброс
2	0x01	INT0	Запрос внешнего прерывания INTO
3	0x02	INTI	Запрос внешнего прерывания INT1
4	0x03	Таймер T2, COMP	Совпадение таймера T2
5	0x04	Таймер T2, OVF	Переполнение таймера T2
6	0x05	Таймер T1, CAPT	Захват таймера T1
7	0x06	Таймер T1, COMPA	Совпадение канала A таймера T1
8	0x07	Таймер T1, COMPB	Совпадение канала B таймера T1
9	0x08	Таймер T1, OVF	Переполнение таймера T1
10	0x09	Таймер T0, OVF	Переполнение таймера T0
11	0x0A	SPI. STC	Завершение передачи SPI
12	0x0B	USART. RXC	Завершение приема USART
13	0x0C	USART, UDRE	Регистр данных USART пуст
14	0x0D	USART, TXC	Завершение передачи USART
15	0x0E	ADC, Conversion Complete	Завершение преобразования ADC
16	0x0F	EEPROM	Готовность EEPROM
17	0x10	Аналоговый компаратор	Событие аналогового компаратора
18	0x11	TWI	Завершение текущей операции TWI
19	0x12	INT2	Запрос внешнего прерывания INT2
20	0x13	Таймер T0	Совпадение таймера T0
21	0x14	SPM RDY	Готовность памяти команд для записи

Прерывания могут быть разрешены или запрещены с помощью установки бита 1 командами CLI (запрет прерываний) и SEI (разрешение прерываний). Каждому прерыванию присвоен свой бит разрешения, который

должен быть установлен совместно с битом 1 регистра статуса. При возникновении прерывания бит 1 очищается и адрес возврата записывается в стек. Чтобы разрешить вложенные прерывания, может установить бит 1 внутри подпрограммы обработки прерывания. Выход из подпрограммы обработки прерывания происходит по команде RETI, при этом адрес возврата выгружается из стека и устанавливается бит 1.

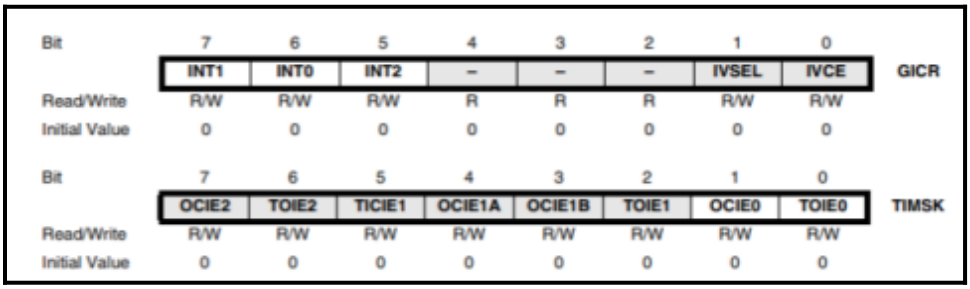

Рисунок 1 – Регистры масок прерываний

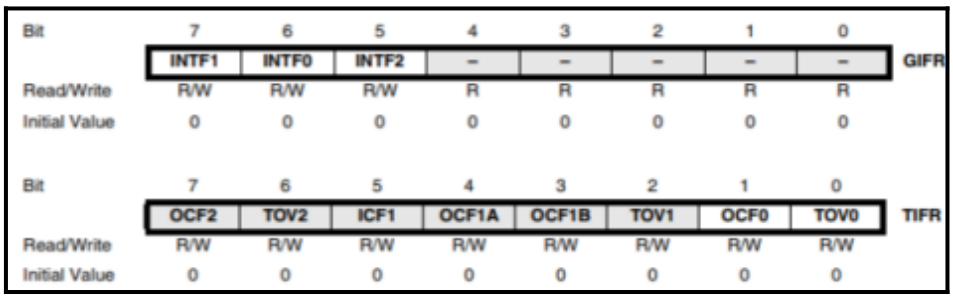

Рисунок 2 – Регистры флагов прерываний

Микроконтроллер содержит два 8-ми разрядных регистра масок прерываний: регистр масок внешних прерываний GICR и регистр масок прерываний по таймеру/счетчику TIMSK (смотреть рисунок 1), а также два 8-ми разрядных регистра флагов прерываний: регистр флагов внешних прерываний GIFR и регистр флагов прерываний по таймерам/счетчикам TIFR (смотреть рисунок 2). При обработке прерывания, соответствующий флаг, вызвавший прерывание, аппаратно очищается. Некоторые флаги прерываний можно очистить, записав в соответствующий бит очищаемого флага логическую единицу.

Таймеры:

Микроконтроллер оснащен тремя таймерами общего назначения - двумя 8-разрядными T0 и T2 и одним 16-разрядным T1. T0, в дополнение к обычному режиму, может тактироваться асинхронно от внешнего генератора.

T0 оснащен своим собственным предварительным делителем.

T1 и T2 используют выходы ступеней деления общего 10-разрядного предварительного делителя. Эти два таймера можно использовать как таймеры с встроенной временной базой или как счетчики, переключаемые по состоянию на внешнем выводе.

Предварительный делитель T1 и T2 содержит четыре ступени деления: CK/8, CK/64, CK/256 и CK/1024, где CK - входной тактовый сигнал. Кроме того, в качестве источников тактовых сигналов могут быть использованы сигналы от внешних источников, тактовый сигнал CK и нулевой тактовый сигнал (stop).

8-разрядные таймеры получают тактовый сигнал непосредственно от CK, после прохождения его через предварительный делитель или от внешнего вывода. В регистре флагов прерывания таймеров TIFR хранятся различные флаги состояния регистров (переполнения, совпадения при сравнении и захвата события). Установки управляющих сигналов хранятся в регистрах управления таймерами/счетчиками TCCR0 и TCCR2 (смотреть рисунок 3). Установка разрешения/запрещения прерываний производится в регистре масок прерываний таймеров/счетчиков TIMSK.

Биты	7	6	5	4	3	2	1	0	
$33 ($53)	-	PWM0	COM01	COM00	CTC0	CS02	CS01	CS00	TCCR0
Чтение/Запись	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Начальное состояние	0	0	0	0	0	0	0	0	

Рисунок 3 – Регистр управления таймером T0

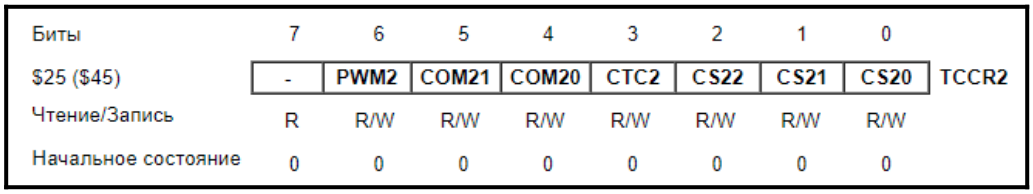

Рисунок 4 – Регистр управления таймером T2

7 бит FOCn (Force Output Compare) предназначен для принудительной установки логического уровня на выходе OC. Он работает только для режимов Normal и CTC. При установке бита FOCn в единицу состояние выхода меняется в соответствии со значениями битов COMn1 и COMn0.

6 и 3 бит – WGMn (Wave Generator Mode) определяют режим работы таймера-счетчика. Комбинации этих битов представлены в таблице 1.

Таблица 2 – Выбор режима работы таймера

WGMn1	WGMn0	Режим работы таймера/счётчика
0	0	Нормальный режим счётчика (normal)
1	0	Сброс таймера при совпадении регистров OCRn и TCNTn (CTC)
0	1	ШИМ с коррекцией фазы (Phase Correct PWM)
1	1	Быстрая ШИМ (Fast PWM)

5 и 4 бит – COMn1, COMn0 (Compare Match Output Mode) задают режим сравнения выхода, биты 1 и 0. Комбинации этих битов представлены в таблице 2.

Таблица 3 – Выбор режима сравнения

COMn1	COMn0	Описание
0	0	Таймер/счетчик отсоединен от выходного вывода OCn/PWMn
1	0	Переключение выходной линии OCn/PWMn
0	1	Очистка выходной линии OCn/PWMn (установка в состояние 0)
1	1	Установка выходной линии OCn/PWMn (установка в состояние 1)

Биты CSn2, CSn1, CSn0 (Clock Select) задают для таймера Tn коэффициент предделителя. Комбинации этих битов представлены в таблице 3.

Таблица 4 - Выбор коэффициента деления предделителя

CSn2	CSn1	CSn0	Описание
0	0	0	Источника тактирования нет, таймер остановлен
0	0	1	Тактовая частота МК
0	1	0	Тактовая частота МК/8
0	1	1	Тактовая частота МК/32
1	0	0	Тактовая частота МК/64
1	0	1	Тактовая частота МК/128
1	1	0	Тактовая частота МК/256
1	1	1	Тактовая частота МК/1024

Регистры TCNT0 и TCNT2 хранят состояние счетчика. Оба таймера реализованы как счетчики по нарастанию или реверсивные (в ШИМ режиме) счетчики с возможностью чтения/записи. Если в таймер/счетчик записано некоторое значение и выбран источник тактового сигнала, то он продолжит счет с записанного значения с тактовой частотой счетчика.

Оба таймера/счетчика поддерживают две функции сравнения выхода OCR0 и OCR2 как источники данных, сравниваемых с содержимым таймеров/счетчиков. В функции сравнения выхода входит и опция очистки счетчика при совпадении и формирование, при совпадении, сигнала на выводах - PB4(OC0/PWM0) и PB7(OC2/PWM2).