**ЛАБОРАТОРНАЯ РАБОТА № 1  
Изучение портов общего назначения и исследование контроллера векторов прерываний (VIC)**

**1.1 Цель работы**

Научиться управлять работой портов ввода/вывода общего назначения, а также изучить основные возможности контроллера обработки прерываний.

**1.2 Краткие теоретические сведения**

Для линий ввода/вывода общего назначения (GPIO) микроконтроллеров семейства LPC2000 характерны следующие особенности:

* Управление направлением (вход это или выход) каждой линии индивидуально;
* Раздельное управление установкой и сбросом входного уровня линии;
* Все линии по умолчанию после сброса являются линиями GPIO, настроенными как входы.
* Линии GPIO могут использоваться для следующих приложений:
* Ввод/вывод общего назначения;
* Управление светодиодами, или другими индикаторами;
* Управление внешними устройствами, подключенными к микро контроллеру;
* Опрос цифровых входов.

Входящий в состав всех устройств микроконтроллеров LPC2000 контроллер векторов прерываний (VIC) имеет следующие особенности:

* архитектура ARM PrimeCell;
* 32 входа запроса прерываний;
* 16 векторов прерываний (IRQ);
* 16 уровней приоритета, которые динамически назначаются запросам прерываний;
* генерация программного прерывания.

Быстрое прерывание может срабатывать всего за 12 машинных циклов (200 нс при 60 МГц) в отличие от обычного векторного прерывания, которое срабатывает за 25 циклов (416 нс при 60 МГц). Это достигается за счет того, что обработчик FIQ располагается сразу после таблицы векторов прерываний, поэтому нет необходимости делать лишние переходы.

Существует 16 векторных прерываний. Эти прерывания могут произвести только IRQ прерывание. Векторные и невекторные IRQ прерывания вызывают обработчик прерывания (ISR). Чтение из векторного регистра адреса прерывания VICVectAddr дает адрес ISR, и обновляет настройки контроллера, который маскирует любые запросы на прерывание, имеющие такой же или более низкий приоритет. Запись в регистр VICVectAddr указывает контроллеру, что текущее прерывание обслужено, что позволяет прерываниям с более низким приоритетом активизироваться.

Прерывание FIQ имеет самый высокий приоритет. Невекторные IRQ прерывания имеют самый низкий приоритет.

* 1. **Порядок выполнения работы**
     1. Изучите до начала выполнения лабораторной работы основные возможности использования портов ввода/вывода общего назначения, а также работу контроллера прерываний.
     2. Изучите примеры программирования работы портов ввода/вывода общего назначения, а также контроллера обработки прерываний, приведенные в разделе 2 данных методических указаний.
     3. Разработайте схему алгоритма управления светодиодным индикатором с учетом установленного варианта задания по таблице 1.1.
     4. Запустите интегральную среду разработки Keil uVision.
     5. Создайте проект, в котором опишите разработанный алгоритм управления при помощи языка С.

Таблица 1.1 — Вариант задания

|  |  |
| --- | --- |
| № | Задание |
| 1 | Реализовать алгоритм управления светодиодным индикатором при помощи джойстика SW2. При нажатии джойстика влево или вправо некоторый светодиод перемещается в выбранном направлении, при нажатии вверх или вниз – остаётся на прежней позиции. Перемещение светодиода происходит по срабатыванию прерывания от таймера (частота выбирается произвольно). |

* + 1. Откомпилируйте созданный проект. При наличии сообщений об ошибках или предупреждениях вернитесь к предыдущему пункту и внесите необходимые изменения.
    2. Включите учебно-отладочный стенд. При помощи утилиты Flash Magic загрузите файл с расширением \*.hex, находящийся в папке проекта, в стенд.
    3. Визуально оцените правильность функционирования разработанного алгоритма.

**1.4 Результаты выполнения работы**

Схема алгоритма реализованной программы представлена на рисунке 1.1.



Рисунок 1.1 – Схема алгоритма реализованной программы

Код программы, выполненный в соответствии с вариантом задания представлен ниже.

#include "lpc21xx.h"

#define LEFT 1

#define RIGHT 2

#define MASK 0x0000FF00

int flag = 0;

\_\_irq void Timer0ISR(void) {

flag = 1;

T0IR = 0x01;

VICVectAddr = 0;

return;

}

char new\_led\_place(int direction, char place){

if (direction == LEFT){

if (place == 0) {

return 7;

} else {

return --place;

}

} else {

if (place == 7) {

return 0;

} else {

return ++place;

}

}

}

void InitTimer0(void) {

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* Eieoeaeecaoey VIC \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

VICDefVectAddr = (unsigned int) &Timer0ISR;

VICIntEnable = 0x10; //Channel#4 is the Timer0

VICIntSelect = 0x00; //all interrupts are IRQs

T0MR0 = 18000000; //Timer match (~ 0.4 second)

T0MCR = 0x03; //Interrupt on Match0, reset timer on match

T0PC = 0x01; // Prescaler to 2

T0TC = 0x00; // reset Timer counter

T0TCR = 0x01; // enable Timer

return;

}

int delay(int \_slp) {

int i, j = 0;

for (i = 0; i < \_slp \* 1000; i++)

j = i + 1;

return j;

}

int main(void) {

int direction = LEFT;

char place = 4;

int allowed\_flag = 1;

IODIR0 |= MASK;

IOSET0 |= 0x0000FF00;

IOCLR0 |= ((1 << place) << 8);

InitTimer0();

while(1) {

if (flag && allowed\_flag){

place = new\_led\_place(direction, place);

IOSET0 |= 0x0000FF00;

IOCLR0 |= ((1 << place) << 8);

flag = 0;

}

switch ((IOPIN0 & (0x1F<<16)) >> 16) {

case 0x17: direction = LEFT; break;

case 0x1B: direction = RIGHT; break;

case 0x1D: allowed\_flag = !allowed\_flag; break;

case 0x0F: allowed\_flag = !allowed\_flag; break;

default: break;

}

}

}

**1.5 Особенности функционирования интегрированной среды разработки Keil uVision , выявленные в ходе выполнения лабораторной работы**

В ходе выполнения лабораторной работы, особенностей функционирования среды разработки Keil uVision, выявлено не было.

**Выводы**

В данной лабораторной работе научились работать с портами общего назначения, а также изучили основные возможности контроллера прерываний (VIC). Ознакомились с возможностями и особенности функционирования среды разработки Keil uVision 3. Была разработана программа согласно заданному варианту.