ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
СРЕДНЕГО (ПОЛНОГО) ОБЩЕГО ОБРАЗОВАНИЯ
ЛИЦЕЙ ПРИ СПБГУТ

Вендор-ориентированный учебный курс в системе
«Старшая профильно-профессиональная школа-ВУЗ-Работодатель»:
«Программирование микроконтроллеров Microchip»

Богураев М.В.

«ЯЗЫК СИ ДЛЯ МИКРОКОНТРОЛЛЕРОВ PIC ПЕРИФЕРИЙНЫЙ МОДУЛЬ НУЛЕВОЙ ТАЙМЕР»

Методические указания к выполнению лабораторной работы

Санкт - Петербург
2012

Богураев М.В. «ЯЗЫК СИ ДЛЯ МИКРОКОНТРОЛЛЕРОВ PIC ПЕРИФЕРЙИНЫЙ МОДУЛЬ НУЛЕВОЙ ТАЙМЕР». Методические указания к выполнению лабораторной работы № 4. СПб: ГОУ «Лицей при СПбГУТ», 2012.

ЛАБОРАТОРНАЯ РАБОТА №12
«ЯЗЫК СИ ДЛЯ МИКРОКОНТРОЛЛЕРОВ PIC
ПЕРИФЕРИЙНЫЙ МОДУЛЬ НУЛЕВОЙ ТАЙМЕР»

Цель работы

Научиться включать периферийные модули. Ознакомиться с работой периферийного модуля нулевой таймер. Освоить моделирование этого модуля в MPLAB IDE. Овладеть навыками программирования таймеров PIC микроконтроллеров.

Теоретические основы

Периферийные модули – это устройства, которые располагаются на одном кристалле и имеют связь с ядром микроконтроллера. Периферийные модули могут быть подключены и к ядру микроконтроллера и к ножкам корпуса микроконтроллера. Каждый периферийный модуль выполняет одну или несколько функций. Настройка и включение этих модулей производятся программно.

Счётчики и таймеры один из видов периферийных модулей. Счётчик – это периферийный модуль, на вход которого подают напряжения логических уровней. При каждом спаде или каждом нарастании напряжения содержимое счётчика меняется на единицу. Таймер – это частный случай счётчика. При помощи таймеров измеряют время. Счётчик может стать таймером, если изменения напряжений на входе счётчика будут периодическими.

Модуль TIMER0 описывается в документации на микроконтроллер (Data Sheet – доступен на сайте производителя: файл 39582b.pdf). Режим работы TIMER0 выбирается битом T0CS в регистре OPTION_REG. Счётчик TIMER0 производит подсчёт каждого спада (нарастания) напряжения на ножке RA4/T0CKI. Модуль TIMER0 можно настроить или как счётчик или как таймер. Когда T0CS=0, модуль TIMER0 работает в режиме таймера и содержимое регистра TMR0 изменяется на единицу на каждом машинном цикле. Импульсы на вход TIMER0 подаются с генератора микроконтроллера. Для увеличения отсчитываемого времени к таймеру можно подключить предделитель. Предделитель увеличивает период (делит «частоту») импульсов, поступающих на вход TIMER0. Коэффициент деления устанавливают битами <PS2:PS0> в регистре OPTION_REG.

В этой лабораторной работе TIMER0 конфигурируется следующим образом: источник тактовых импульсов – основной генератор микроконтроллера (T0CS=0). Предделитель не подключен к TIMER0 (PSA=1).

Чтобы пользоваться нулевым таймером его нужно настроить и включить. Затем в программе отслеживать бит T0IF. Этот бит устанавливается, когда таймер переполняется. Под переполнением таймера понимается следующее: при каждом изменении напряжения на входе счётчика содержимое ячейки памяти данных TMR0 увеличивается на единицу. Когда содержимое ячейки становится равным 0xFF и приходит очередной импульс – значение в ячейке TMR0 обнуляется (TMR0=0x00) и устанавливается бит (флаг) T0IF (другое название этого бита TMR0IF).

В этой лабораторной работе при каждой установке флага T0IF катод одного из семисегментных индикаторов соединяется с общим проводом источника напряжения. Когда подключается один индикатор, выводится одна цифра, а когда подключается другой индикатор, выводится другая цифра. Каждое число отображается лишь часть времени. Поскольку человеческий глаз не замечает быстрой смены образов, то создаётся впечатление, что индикаторы отображают одновременно, хотя на самом деле они горят по очереди. Такой способ отображения информации на семисегментном индикаторе получил

название динамической индикации. В этой лабораторной работе используются три семисегментных индикатора, для коммутации катодов используются биты PORTC<4:2>.

Задание

Изучите теоретические основы этой лабораторной работы (дополнительно можно изучить файл 39582b.pdf, раздел TIMER0). Заполните таблицу включения и настройки нулевого таймера. На диске C в папке C_Projects создайте папку C_Project4. В эту папку скопируйте файл C_Project4.c. Затем создайте проект C_Project4. Откомпилируйте текст программы C_Project4.c. В симуляторе MPLAB SIM промоделируйте работу устройства.

На лабораторном макете подключите три семисегментных индикатора к разъёму PORTD микроконтроллера. Выводы DIGIT1, DIGIT2, DIGIT3 подключите к разъёму PORTC к битам 2, 3, 4 соответственно. Запрограммируйте микроконтроллер и продемонстрируйте работу программы на лабораторном макете.

Порядок выполнения

Откройте файл 39582b.pdf, изучите раздел TIMER0. Заполните таблицу включения и настройки нулевого таймера (табл. 1).

Табл. 1. Включение и настройка нулевого таймера.

Название регистра	Название, номер бита, значение бита.								После POR	После сброса
	7	6	5	4	3	2	1	0		
INTCON	GIE	PEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	0000 000u
	0	0	0	0	0		0	0		
OPTION_REG	RBPU	INT EDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	1111 1111
	1	1		1		1	1	1		
TMR0	Регистр модуля Timer0									
	0	0	0	0	0	0	0	0	xxxx xxxx	uuuu uuuu

Прим. POR (power on reset) перезагрузка при включении питания; x = неизвестно, u = не изменяется.

На диске C в папке C_Projects создайте папку C_Project4. В эту папку скопируйте файл C_Project4.c. Запустите MPLAB IDE и создайте проект C_Project4. Откомпилируйте текст программы C_Project4.c.

Выберите отладчик MPLAB SIM. Откройте окно Watch и выберите регистры PORTC, PORTD, TMR0, INTCON и переменную DispState (рис. 1).

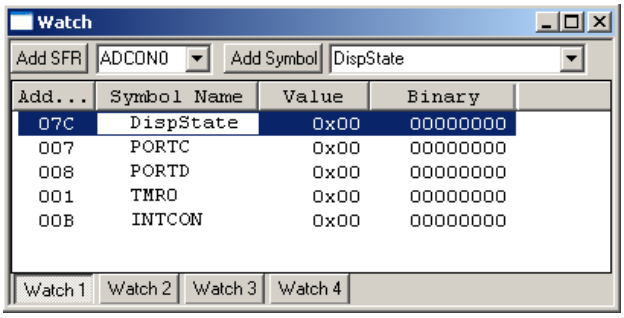

Рис. 1. Окно Watch с выбранными параметрами.

Затем из меню View откройте Simulator Logic Analyzer (рис. 2).

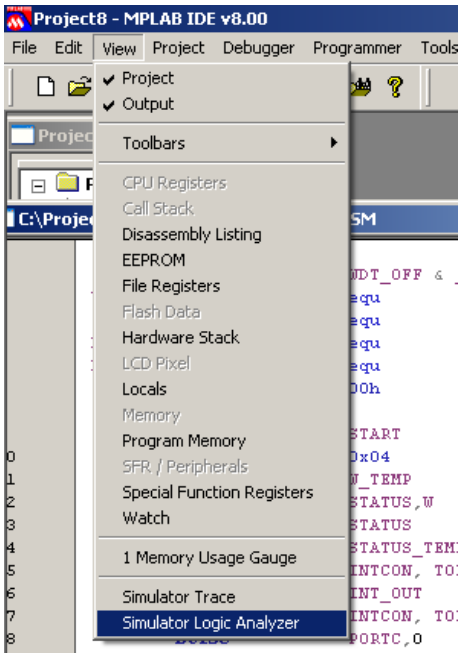

Рис. 2. Запуск логического анализатора.

В появившемся окне (рис. 3) нажмите кнопку Channels. Появится окно Configure Channels. Для отображения каналов RC2, RC3, RC4 нужно выделить добавляемый канал и нажать кнопку Add (рис. 4). Выберите каналы RC2, RC3, RC4.

Для отображения порта PORTD нажмите кнопку Configure Bus(s) – появится окно Configure Bus (рис. 5). В этом окне нажмите кнопку New Bus и в появившемся окне Bus Name введите имя шины: PORTD (рис. 6). Нажмите кнопку OK. Теперь нужно указать какие выводы будут входить в шину. Для этого в окне Configure Bus нужно добавить ножки микросхемы. Добавлять нужно в следующем порядке: выделить ножку, нажать кнопку Add. Добавляйте ножки от младших разрядов к старшим (от lsb к msb): RD0, RD1, RD2, RD3, RD4, RD5, RD6, RD7 (рис. 7). Теперь нажмите кнопку OK. Затем в окне Configure Channels выберите PORTD и нажмите кнопку Add (рис. 8). Затем нажмите кнопку OK.

Перейдите в окно с текстом программы. Напротив функции Disp() поставьте точку останова. Для этого дважды щёлкните левой кнопкой мыши на строке, в которой написана функция Disp(). Запустите симулятор и наблюдайте содержимое окна Logic Analyzer. Изменения в регистрах PORTC, PORTD, TMR0, INTCON и переменной DispState будут видны и в окне Watch (Рис. 1). Проанализируйте полученные результаты (рис. 9).

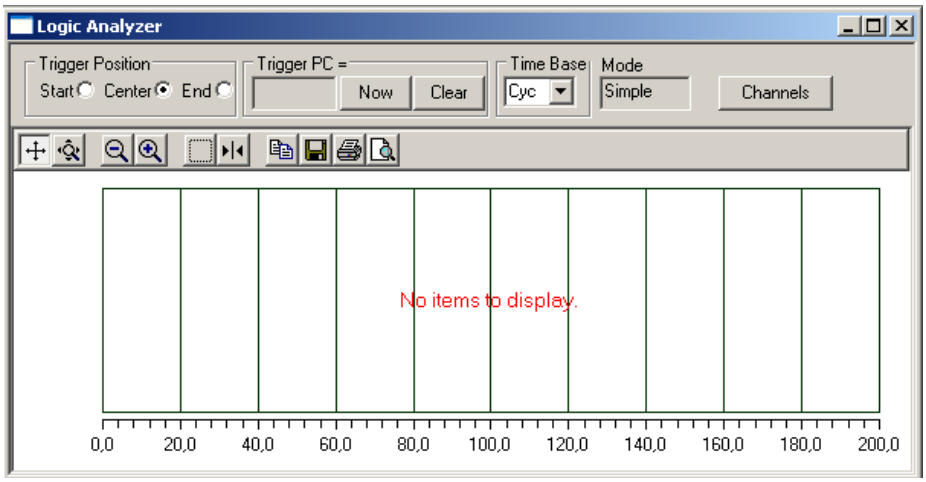

Рис. 3. Вид окна логического анализатора.

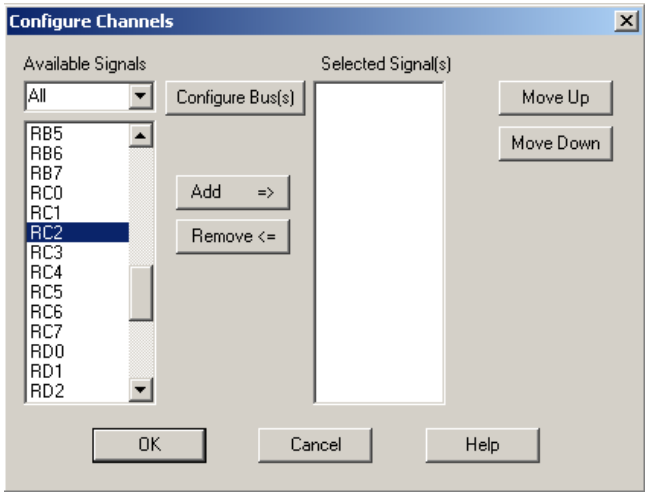

Рис. 4. Выбор каналов.

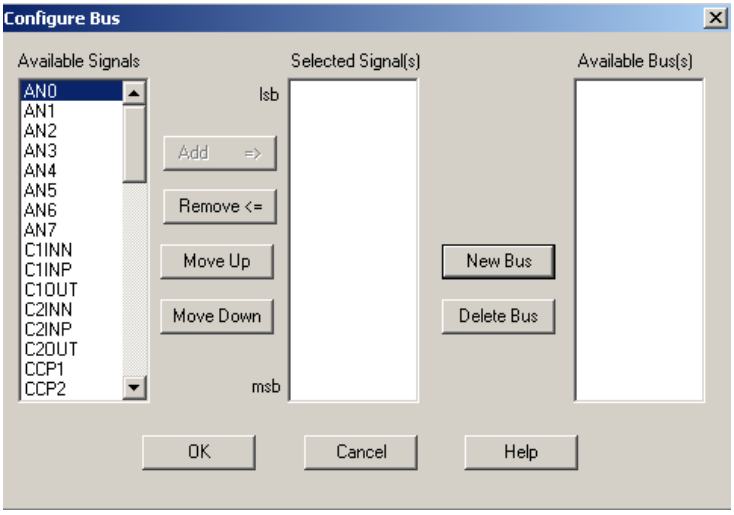

Рис. 5. Вид окна Configure Bus.

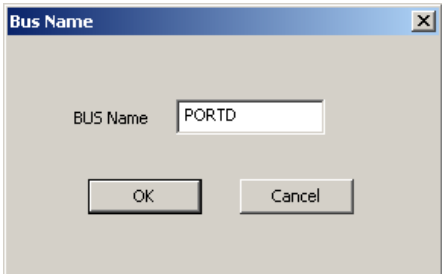

Рис. 6. Вид окна Bus Name.

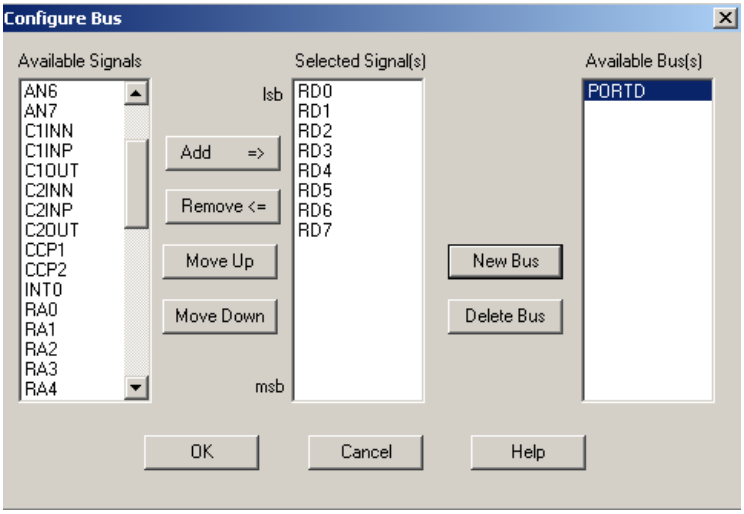

Рис. 7. Вид окна Configure Bus после формирования шины PORTD.

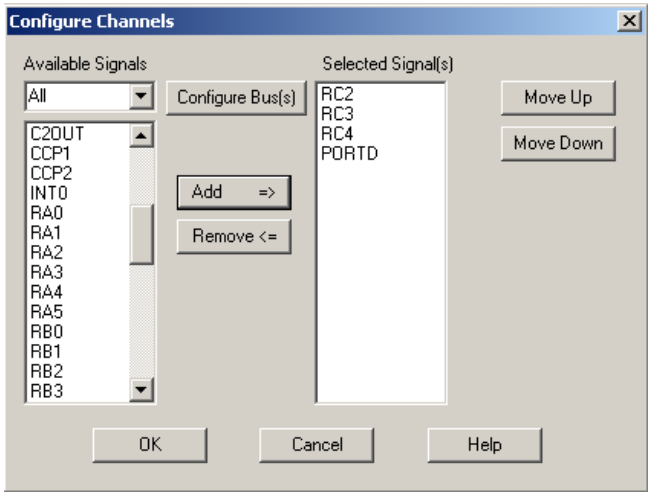

Рис. 8. Окно Configure Channels – каналы для наблюдения в Logic Analyzer.

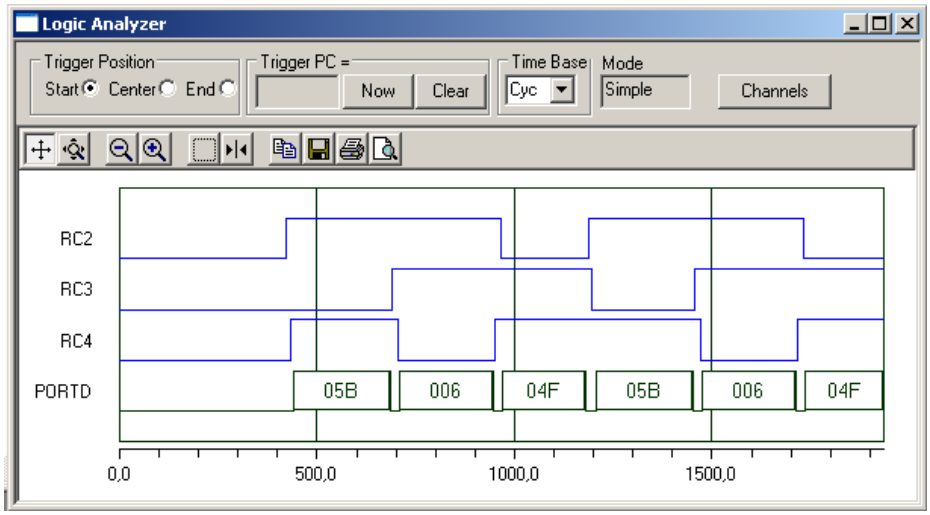

Рис. 9. Результаты симуляции программы.

Проанализируйте полученный результат: постарайтесь понять, какое число и когда будет подано на PORTD; какие при этом уровни будут на ножках RC2, RC3 и RC4, что в это время будет содержаться в переменной DispState. Какие символы будут выводиться на семисегментные индикаторы. Предскажите результат работы программы на макете.

После симуляции соберите схему на лабораторном макете в соответствии с рис. 15. После этого в среде разработки выберите имеющийся у вас отладчик, например PICkit 2 (рис. 10).

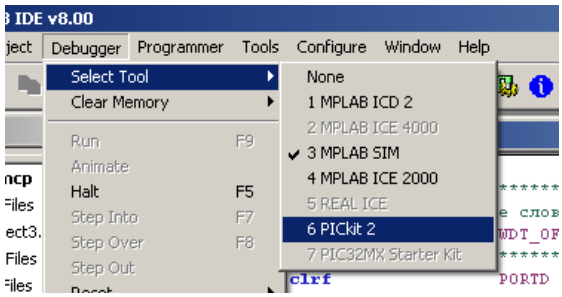

Рис. 10. Выбор отладчика в среде разработки.

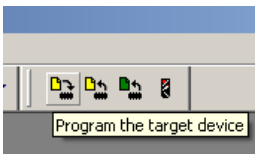

Рис. 11. Кнопка для программирования лабораторного макета.

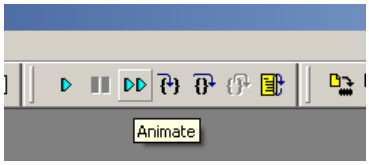

Рис. 12. Кнопка для запуска программы в режиме Animate.

Рис. 13. Кнопка для запуска программы в режиме Run.

Запрограммируйте лабораторный макет (рис. 11). Запустите программу в режиме Animate (рис. 12). Затем запустите программу в режиме RUN (рис. 13) на макете и предъявите результат.

Аппаратное обеспечение

В этой работе используется три семисегментных индикатора. Семисегментный индикатор состоит из восьми светодиодов и каждый из них работает как обычный светодиод. Только в семисегментном индикаторе светодиоды выполнены в виде полосок. Полоски имеют определённое положение на плоскости индикатора. Комбинируя горящие и не горящие светодиоды можно получать различные символы. Катоды у всех диодов объединены и подключаются к выводам PORTC. Принципиальная электрическая схема семисегментных индикаторов лабораторного макета изображена на рис. 14. Принципиальная электрическая схема для выполнения лабораторной работы изображена на рис. 15. На рис. 16 показаны соединения на плате лабораторного макета.

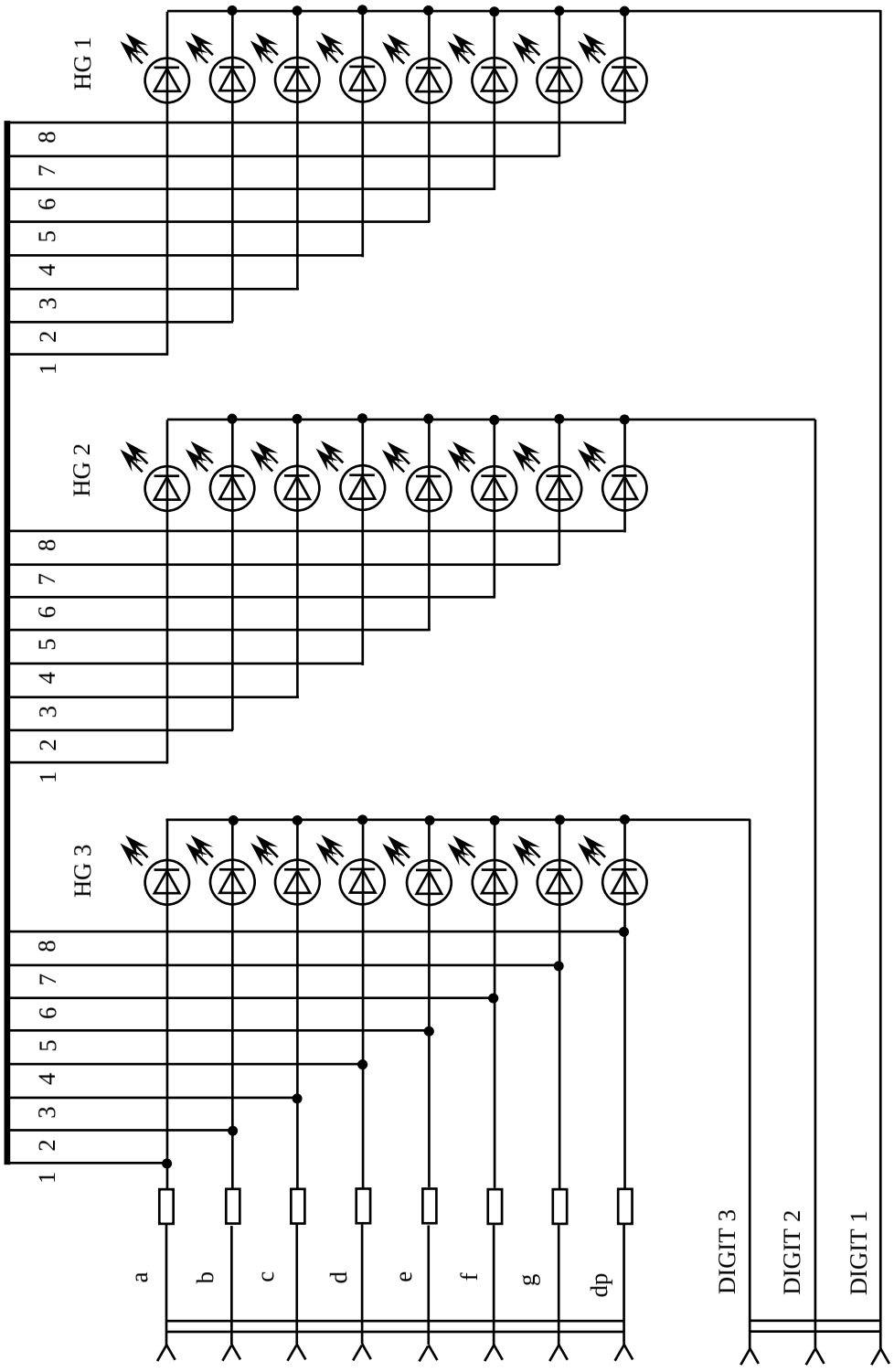

Рис. 14. Принципиальная электрическая схема семисегментных индикаторов.

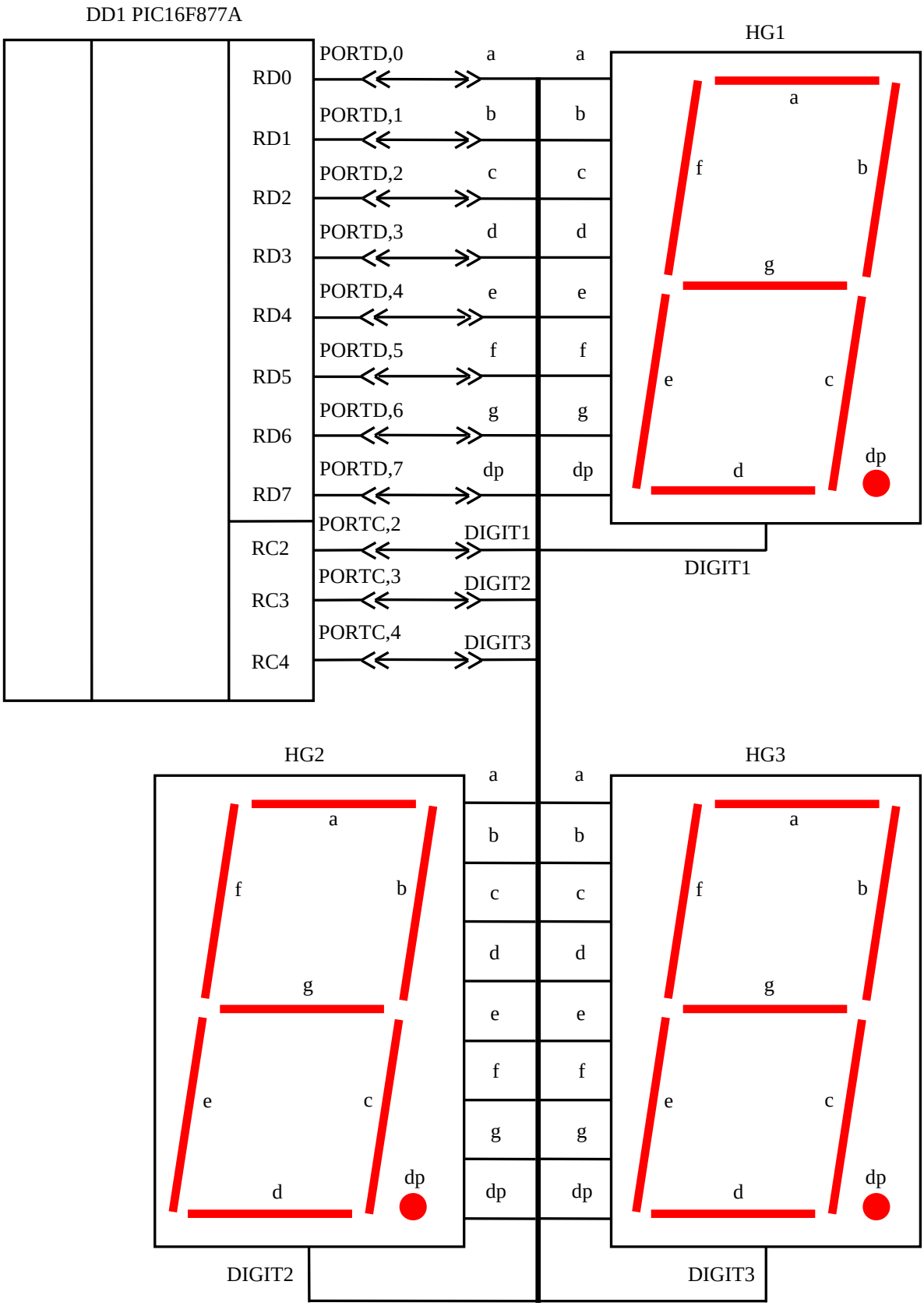

Рис. 15. Принципиальная электрическая схема для выполнения лабораторной работы.

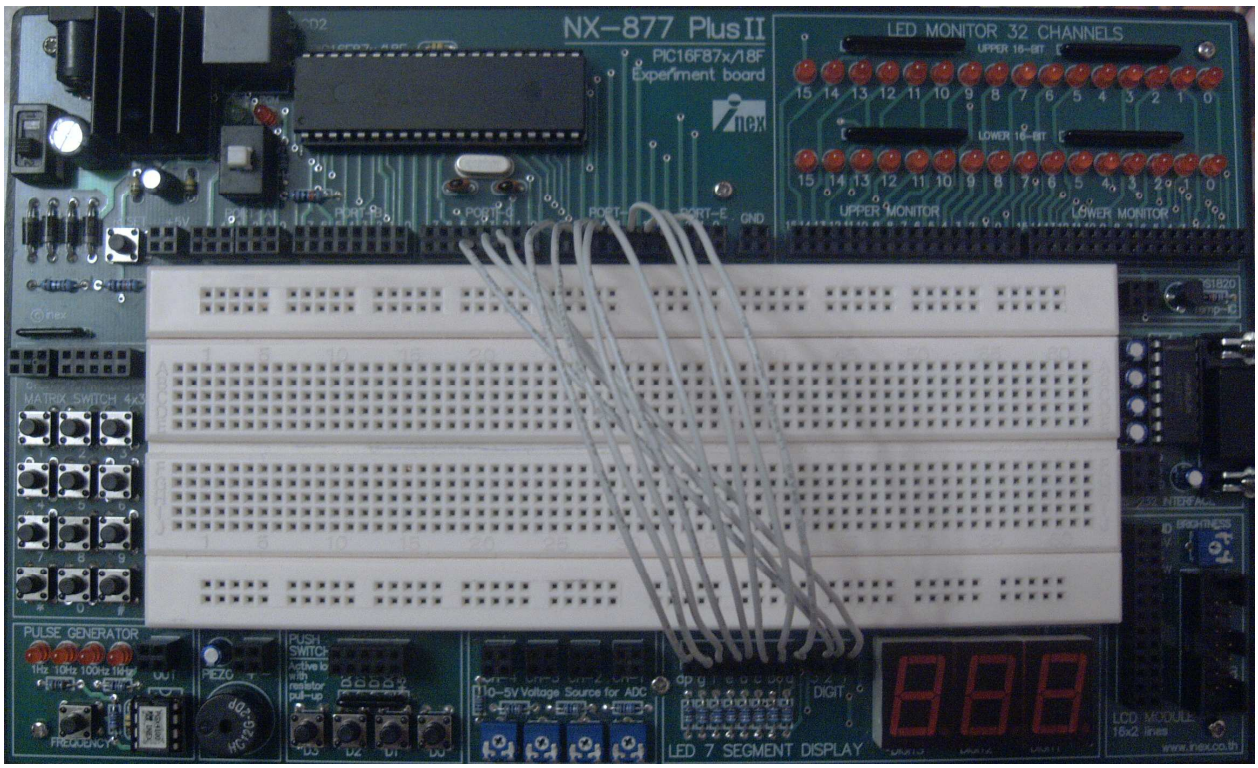

Рис. 16. Схема, собранная на лабораторном макете.

Программное обеспечение

Текст файла Project4.C

```
#include <pic.h>
#define ONE         0x06
#define TWO         0x5B
#define THREE     0x4F
#define FOUR       0x66
#define FIVE        0x6D
#define SIX          0x7D
#define SEVEN     0x07
#define EIGHT     0x7F
#define NINE       0x6F
#define ZERO       0x3F
 __CONFIG (HS & WDTDIS & LVPDIS & DEBUGEN);
 unsigned char DispState              = 0;
 unsigned char DispSeconds         = THREE;
 unsigned char DispDesSeconds   = TWO;
 unsigned char DispMinutes         = ONE;
   void init (void)
      {
         PORTD = 0x00;
         PORTC = 0x00;
         TRISC = 0x03;
         TRISD = 0x00;
      }
   void initTMR0 (void)
      {
          TMR0=0x00;
          T0CS=0;
          PSA=1;
      }
   void Disp (void)
      {
          TMR0IF=0;
             if (DispState==0)//RC2==0
             {
                    PORTD = 0x00;
                    RC2=1;
                    RC3=0;// Вывод DesSeconds
                    RC4=1;
                    PORTD = DispDesSeconds;
             }
             else if (DispState==1)//RC3==0
             {
                    PORTD = 0x00;
                    RC3=1;
                    RC2=1;
                    RC4=0;//Вывод Minutes
                PORTD = DispMinutes;
             }
             else if (DispState==2)//RC4==0
             {
```

```
                PORTD = 0x00;
                RC4=1;
                RC3=1;
                RC2=0;//Вывод Seconds
            PORTD = DispSeconds;
          }
          DispState++;
            if (DispState > 2) DispState = 0;
    }
void main (void)
{
  init();
  initTMR0();
  while (1)
  {
  while (!TMR0IF) continue;
    Disp();
  }
}
```

Индивидуальные задания

Выведите три разных символа или цифры на семисегментные индикаторы в режиме динамической индикации.

Напишите программу для подсчёта числа нажатий кнопки в диапазоне от 0 до 24 нажатий.

Контрольные вопросы

1. Что такое периферийные модули? Назовите несколько периферийных модулей.
2. Как настраивают и включают периферийные модули?
3. Что такое счётчик? Что такое таймер?
4. Как счётчик может стать таймером?
5. Для чего нужен предделитель?
6. Когда устанавливается бит T0IF?
7. Что такое переполнение таймера?
8. Что такое динамическая индикация?
9. Прокомментируйте заполненную таблицу включения и настройки нулевого таймера.

Оглавление:

Список рисунков: