

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
«ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»
ДГТУ

Кафедра «Приборостроение»

**ПРОГРАММИРОВАНИЕ МИКРОКОНТРОЛЛЕРА
PIC16F628A**

Методические указания к лабораторной работе по дисциплинам
«Автоматизированные системы сбора и обработки измерительной
информации», «Микропроцессорная техника»

Ростов-на-Дону 2012

УДК 004.382.7

Составители: к.т.н., доц. А.В. Литвин

к.т.н., доц. К.А. Мороз

ст. преподаватель И.Н. Нестеренко

ст. преподаватель В.Н. Сыроватка

Программирование микроконтроллера PIC16F628A: Метод. указания к лабораторной работе по дисциплинам «Автоматизированные системы сбора и обработки измерительной информации» и «Микропроцессорная техника». – Ростов н/Д: Издательский центр ДГТУ. 2012. – 20 с.

В методических указаниях приводятся цель работы, краткое описание порядка составления программы, содержание и порядок выполнения лабораторной работы, рассмотрен пример написания программы для микроконтроллера. Предназначены для студентов очной и заочной форм обучения специальностей «Приборостроение» и «Стандартизация и сертификация».

Печатается по решению методической комиссии факультета «Приборостроение и техническое регулирование»

Рецензент к.т.н., доцент П. С. Обухов

Научный редактор к.т.н., проф. В.Н. Ананченко

© Издательский центр ДГТУ, 2012

1. Цель работы. Приобретение практических навыков по созданию управляющей программы для микроконтроллера (МК) PIC16F628A на языке ассемблера.

2. Краткие сведения из теории.

Создание программы для МК состоит из следующих этапов:

1. Формирование технического задания (ТЗ), с подробным описанием всех функций проектируемого устройства. Описываются все варианты действия управляющей программы.
2. Для реализации требований ТЗ выбирается МК необходимой производительности и аппаратной конфигурации. Создается схема электрическая принципиальная устройства.
3. Составляется алгоритм управляющей программы.
4. По составленному алгоритму пишется программа на языке низкого уровня (ассемблере) или высокого уровня (Си).
5. Написанная программа отлаживается и тестируется в программе-симуляторе, которая программным путем воссоздает работу МК.
6. Окончательная отладка программы производится на реальных МК или на внутрисхемных отладчиках.

В качестве примера рассмотрим следующую задачу:

Пример. Создать устройство, которое циклически выдает сигналы управления на восемь линий согласно временной диаграмме на рисунке 1.

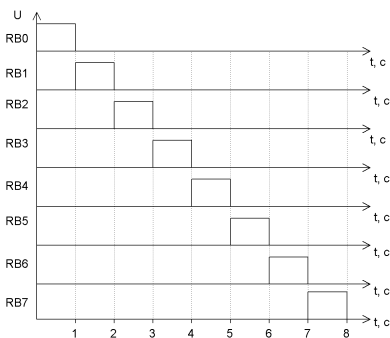


Рисунок 1 – Временная диаграмма работы устройства

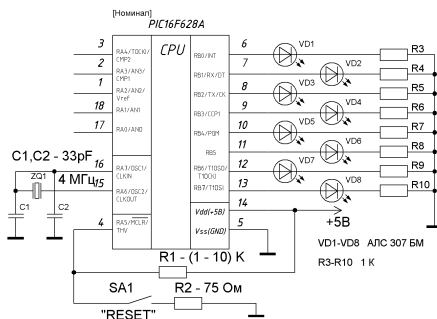


Рисунок 2 – Схема электрическая принципиальная устройства

Для реализации устройства необходим один восьми разрядный порт ввода-вывода. Выберем для реализации устройства МК PIC16F628A, который помимо порта, содержит минимальный набор периферийных модулей. Все неиспользуемые модули будут отключены.

Создаем схему электрическую принципиальную устройства. (Рис. 2).

Для индикации состояния линий порта используются светодиоды VD1-VD8, подключенные через токоограничивающие резисторы R3-R10 к соответствующим линиям порта. Для задания частоты тактового генератора используется кварцевый резонатор с частотой 4 МГц (совместно с конденсаторами C1,C2), что позволит получить частоту машинных циклов в 1 МГц, т.е. время выполнения простейшей инструкции составит 1 мкс, что удобно для вычисления временных характеристик разрабатываемого распределителя импульсов. Используется внешняя схема сброса, представленная резистором R1 номиналом от 1 до 10 Ком, который включается в схему между плюсом источника питания и выводом -MCLR. Вывод -MCLR с чертой, что говорит о том, что сброс наступает при поступлении низкого уровня напряжения на данный вывод. Используя данное свойство вывода можно реализовать схему для ручного принудительного сброса микроконтроллера с помощью кнопки SA1 и токоограничивающего резистора R2 подключенного к "земле".

Составляем алгоритм работы программы (см. рисунок 3).

При составлении алгоритма необходимо учитывать, что после включения питания, МК непрерывно выполняет программу, которая расположена в памяти программ, и никогда не останавливается (кроме режима SLEEP). Поэтому, все программы имеют циклическую структуру.

Первым действием в любой программе является настройка модулей МК, необходимых для работы и отключение не используемых. Планируется отключить все на задействованные в реализации замысла модули, с целью однозначного определения выводов МК как выводов портов. Далее, в силу того, что по условию задания выводы порта В должны служить источником сигнала, необходимо весь порт В настроить на выход.

После завершения настроек, начинается

Рисунок 3 основной цикл программы, который будет выполняться непрерывно, пока МК включен. Сначала, формируется логическая единица ТТЛ уровня (+5 В) на линии RB0 в течение заданного интервала времени в 1 секунду. Остальные линии порта В обнулены. Далее, формируется сигнал логической единицы на линии RB1 в течение 1 секунды. Остальные линии, в том числе и линия

RB0 – обнулены. Таким образом, последовательно формируются сигналы уровня логической единицы на всех линиях порта В (с RB0 по RB7) с интервалом между переключениями в 1 секунду. По завершении формирования логической единицы на линии RB7, происходит переход на формирование логической единицы на линии RB0 и цикл повторяется.

Составляем управляющую программу. Текст программы условно можно разделить на “шапку” и основную часть, собственно программу. Основная часть помещается в память программ при программировании МК. Содержимое “шапки” в непосредственном виде не входит в программу, помещаемую в микроконтроллер, однако оно влияет на параметры работы МК, его внутреннюю связь. Эти параметры учитываются ассемблером при создании прошивки для МК.

Текст программы приведен в приложении А. Перечислим основные синтаксические правила написания программы. Программа состоит из трех значащих столбцов и четвертого столбца – комментариев. Первый левый столбец начинается обязательно с первой позиции. Между столбцами обязателен как минимум один пробел. Рекомендуется в целях удобства чтения форматировать текст программы таким образом, чтобы положение первых позиций в столбце совпадали. Комментарии начинаются после знака ‘;’ , после которого ассемблер никакие символы не воспринимает. Перенос строк не допускается.

В ‘шапке’ программы первый столбец предназначен для указания наименования регистров, констант и битов регистров, если они прописываются. Во втором столбце указываются директивы ассемблера. В третьем столбце – адреса регистров, а также значение битов регистров (если они прописываются), а также значение числовых констант. В основной части программы в первом столбце указываются имена подпрограмм и меток. Метки и имена могут содержать латинские буквы и цифры, знак подчеркивания. Начинаются всегда с буквы. Во втором столбце, располагаются команды. В третьем - названия регистров (а также номера их битов) с которыми выполняют операции команды. Также в этом столбце указывается место сохранения результата (в регистр или в аккумулятор), а также константы и их числовые значения.

Директивы ассемблера отличаются от команд. Они не входят в конечный текст программы, загружаемый в МК. Они лишь предписывают ассемблеру выполнять те или иные действия, устанавливать связи, определять границы программы и т.д. Результат их выполнения ассемблером заключается в том, что при программировании МК программатором происходит исходная аппаратная настройка МК, которая выполняется только один раз в момент программирования. Изменение этой настройки возможно только при следующих

перепрограммированиях. В лабораторной работе используются некоторые из множества директив (см. таблица 1).

Текст программы (см. приложение Б) начинается с 'шапки', в которой указывается тип МК (p=16F628A). Далее, директивой include подключается файл с описанием регистров специального назначения (PCH) данного МК. При помощи директивы __CONFIG (впереди два знака подчеркивания) устанавливаются значения слова конфигурации.

Таблица 1 – Директивы ассемблера

№	Директива	Описание	Пример
1	END	Окончание программы (конец всех команд) end.
2	EQU	Присваивает неизменное значение константе	AAA equ 71h ; присвоить константе AAA число 71h
3	SET	Присваивает значение константе, которое в последствии можно переопределить.	AAA set 71h
4	INCLUDE	Подключение дополнительного исходного файла.	include <p16f628a.inc>; подключение файла с описанием регистров специального назначения.
5	__CONFIG	Установка битов конфигурации.	См. таблицу символов конфигурации в приложении Б.
6	ORG	Установить начальный адрес программы.	org 0; начальный адрес программы 0 PC.

Слово конфигурации. (См. приложение А). При помощи слова конфигурации происходит исходная настройка аппаратной части МК. Оно расположено в памяти программ по адресу 2007h. Этот адрес не доступен пользователю, и к нему можно обратиться только в режиме программирования МК. Слово конфигурации имеет 14 битов конфигурации. Согласно назначению битов конфигурации установим в программе следующее сочетание их значений (0 или 1) в двоичной системе счисления: b'10000101100001', также в программе допускается записывать слово конфигурации в шестнадцатеричной системе счисления (__CONFIG 2161h). Расшифруем значение данного слова конфигурации.

13 бит – установлен в 1. Это означает, что защита памяти программ выключена. Если бы был установлен 0, то из Flash памяти программ микроконтроллера было бы невозможно считать программу на

внешний носитель. Данная функция позволяет защитить программу от несанкционированного тиражирования.

12-9 биты не реализованы, и их значения могут быть любыми.

8 бит – установлен в 1. Это означает, что защита EEPROM памяти данных выключена. Если бы был установлен 0, то при помощи программатора было бы невозможно извлечь данные из EEPROM памяти данных. Данная функция защищает эти данные от несанкционированного считывания.

7 бит – установлен в 0. Данным битом был отключен режим низковольтного программирования (LVP), благодаря чему, вывод RB4 используется как цифровой порт ввода/вывода.

6 бит – установлен в 1. Данным значением шестого бита разрешен сброс по снижению питания BOR. Теперь, если в процессе работы МК значение напряжения питания уменьшится до величины менее 4 В, то произойдет сброс МК. При восстановлении величины напряжения питания до 4 В и более, произойдет запуск МК и работа программы начнется сначала.

5 бит – установлен в 1. Данным значением пятого бита настраиваем режим работы вывода RA5 как вывода сброса –MCLR, поскольку мы используем внешний сброс, посредством подключения резистора между выводом –MCLR и плюсом источника питания.

3 бит – установлен в 0. Данным действием разрешается работа таймера сброса по включению питания PWRT. Будет происходить выдержка в 72 мс после поступления сигнала от схемы POR или BOD.

2 бит – установлен в 0. Данным действием запрещается работа сторожевого таймера WDT. Таймер отключен, и не требует периодического сброса в программе. Также, в отключенном состоянии он не защищает МК от зависания.

4,1,0 биты – установлены соответственно в 001, что соответствует выбору стандартного кварцевого XT генератора, поскольку в схеме используется кварцевый резонатор на 4 МГц.

Режимы работы тактового генератора. МК PIC16F628A поддерживает 8 режимов работы тактового генератора. Микроконтроллер может тактироваться внешними импульсами (режим EC). Или же сам генератор может генерировать тактовую частоту с помощью внешней или внутренней RC цепочек или внешнего кварцевого резонатора. Использование RC цепочек позволяет получить широкий диапазон частот, при низкой стоимости. Недостаток – низкая стабильность тактовой частоты, её зависимость от напряжения питания и температуры.

Внутренние RC цепи позволяют генерировать две частоты 4 МГц или 37 кГц (выбирается битами регистра PCON). Внешние элементы не

требуется. При этом, выводы RA6 и RA7 могут функционировать как порты ввода/вывода.

Использование внешнего кварцевого резонатора позволяет получить стабильную тактовую частоту. В зависимости от значения частоты, выбирается один из режимов – см. таблицу 2.

Таблица 2 – Режимы генератора при использовании кварцевого резонатора.

№	Название режима	Обозначение	Значение частоты	Единица измерения
1.	Низкочастотный	LP	32...200	кГц
2.	Обычный кварцевый	XT	0,2...4	МГц
3.	Высокочастотный	HS	свыше 4	МГц

Продолжаем составлять управляющую программу. (см. приложение Б) После слова конфигурации в ‘шапке’ программы описываются все необходимые в работе регистры специального назначения. Директивой equ присваивается неизменное значение константе. В данном случае, именам регистров специального назначения (PCH) присваиваются их адреса согласно карты памяти для МК PIC16F628A. Таким образом, программа, при обращении к константе по имени, всегда будет ставить ей в соответствие её адрес. В работе используются следующие PCH: Status (03h) - регистр выбора банка памяти; далее, в силу того что сигнал будет выводиться на выводы порта В то прописывается два регистра управляющих этим портом - TrisB (86h) - регистр настройки направления работы выводов порта В, PortB (06h) - регистр управления защелками порта В, то есть уровнем сигнала на его линиях; далее, прописываются все регистры, с помощью которых будут отключены неиспользуемые модули МК - VRCON (9Fh), CMCON (1Fh), T1CON (10h), T2CON (12h), CCP1CON (17h), RCSTA (18h).

Далее, в ‘шапке’ программы указываются названия пользовательских ячеек памяти с присвоением им адресов. Данную часть заполняют в процессе написания программы, поскольку заранее достоверно не известно, сколько ячеек памяти может понадобиться. В данном случае, в силу того, что программа составлена, то необходимо прописать три ячейки памяти с именами n, n1, n2 с адресами 70h, 71h, 72h соответственно. Все перечисленные ячейки используются для хранения переменных счетчиков проходов при организации подпрограмм задержки. Пользовательской ячейкой может быть любая белая (не окрашенная) ячейка из карты памяти. Рекомендуется, наиболее употребляемые в программе пользовательские переменные прописывать в регистрах с адресами с 70h по 7Fh. Данные 16 ячеек памяти отображаются в каждом банке памяти, и при обращении к их

содержимому, не важно в каком банке памяти происходит работа МК. При обращении к остальным регистрам общего назначения необходимо находиться в том банке, в котором эти регистры расположены.

В конце шапки программы, с помощью директивы `org`, указывается точка входа в программу. Основная программа всегда начинается с нулевого адреса. Следом, указывается команда `goto Start`. Этим осуществляется безусловный переход на начало программы.

Собственно программа начинается с метки `Start`. И далее следует последовательность команд программы. Командой `bcf` обнуляется бит 7 регистра `RCSTA`, данным действием отключается модуль `USART`. Далее отключаем модуль `CCP1`. Для этого необходимо обнулить биты 3-0 регистра `CCP1CON`. Данную операцию можно осуществить, скопировав ноль из аккумулятора в регистр `CCP1CON`, при этом в силу того, что модуль выключен, состояния оставшихся битов 7-4 значения не имеют. Для осуществления этой операции необходимо использовать две команды. Но обнуление регистра можно воспользоваться и одной командой `clrf`.

Формат команды:

`clrf f` – очистить регистр `f`.

Пример обнуления регистра :

`clrf CCPCON` ; очистка (обнуление) регистра `CCPCON`.

Примечание: очистка (обнуление) аккумулятора производится командой `clrw`, очистка сторожевого таймера - командой `clrwtd`.

Далее, производится выключение модулей `TMR1` и `TMR2`, посредством обнуления бита 0 регистра `T1CON` и бита 2 - `T1CON`. Отключение модуля компараторов осуществляется установкой в единицу битов 0,1,2 регистра `CMCON`. Данную операцию можно выполнить, записав соответствующую константу `b'00000111'` или `b'11111111'` (для модуля компараторов в отключенном состоянии значение битов 3-7 не имеет значения) в аккумулятор, а затем скопировав его содержимое в регистр `CMCON`. Но данную операцию можно осуществить путем обнуления содержимого регистра с помощью команды `clrf` и последующей инверсией этого содержимого командой `comf` что и представлено в программе.

Формат команды:

comf f,d – инвертировать все биты в регистре f. Если d = 0, результат сохраняется в аккумуляторе (регистр W), если d = 1, результат сохраняется в регистре f

Пример инверсии содержимого регистра :

comf CMCON,1 ; инверсия битов регистра CMCON с сохранением результата в нем же (d = 1) . Если содержимое регистра CMCON до инверсии было '00000000', то после выполнения данной команды будет '11111111'

При помощи следующих двух команд bcf и bsf происходит переход в первый банк. В первом банке расположен регистр VRCON, установкой седьмого бита которого в ноль отключается модуль источника опорного напряжения. Далее, константа ноль записывается в аккумулятор, а далее в регистр TrisB, который также расположен в нулевом банке. Данным действием мы записываем нули во все разряды регистра TrisB, и следовательно, настраиваем все линии порта B на выход, вывод сигнала из МК. При помощи команды bcf Status,5 происходит возврат в нулевой банк памяти.

Далее, командой clrf производится исходная очистка регистра PortB. И на выводах данного порта гарантировано устанавливаются нулевые значения напряжения.

Следующая строка начинается с метки TACH1 в крайнем левом столбце. С данной метки начинается основной цикл программы, который будет повторяться снова и снова пока напряжение питания подается на МК. По условию задачи требуется последовательно выдавать импульсы на линии порта, начиная с линии RB0. При этом на всех остальных линиях должен находиться уровень напряжения логического нуля. Установка единичного уровня производится установкой единицы в соответствующем разряде регистра PortB, в данном случае в нулевом. Для этого, в программе производится копирование константы b'00000001' в аккумулятор, а затем из него в регистр PortB. По завершении исполнения команды записи в регистр PortB происходит установка логической единицы на линии RB0, при этом все остальные линии порта будут обнулены, так как в биты управляющие состояниями остальных линий порта записаны нули. После установки логической единицы на линии RB0 необходимо чтобы это состояние на линии сохранилось в течение 1 секунды. Защелки портов сохраняют установленное состояние сколь угодно долго, пока включено питание, а процессор в это время может выполнять любые операции не связанные с изменением состояния линий портов. Поддерживать состояние линий порта не нужно. Также, состояние защелки, а, следовательно, и линии порта будет оставаться

неизменным, до тех пор пока соответствующая линия не будет переустановлена с помощью команды программы. Таким образом, для обеспечения на линии RB0 единичного импульса в течение 1 секунды, необходимо спустя 1 секунду после установки линии RB0 в единицу, переустановить её в ноль. В это время МК может выполнять любые необходимые действия, но в силу того, что в задании никакие иные действия, кроме изменения состояния линий порта не предусмотрены, необходимо организовать задержку в 1 секунду при помощи подпрограммы задержки. Задержка, или пауза, понятие условное, поскольку МК никогда не останавливается. Следовательно, МК должен выполнять определенные команды в течение заданного интервала времени. Командой call Pause вызывается на исполнение подпрограмма задержки в 1 с.

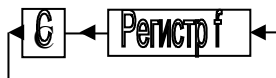
После того, как программа отработает подпрограмму задержки Pause, исполняются две пустые команды пор. Данная команда называется пустой, не выполняет никаких действий, однако время её исполнения составляет 1 машинный цикл. Используется для организации задержек и для уравнивания времени выполнения различных ветвей программы при её ветвлении.

Далее, следует команда, записывающая константу b'00000010' в аккумулятор, а затем в регистр PortB. Данным действием происходит обнуление бита 0 регистра PortB, и установка бита 1 данного порта в единицу. Следом, выполняется вызов и отработка подпрограммы задержки Pause, а также выполнение двух калибровочных команд пор.

Представленный способ изменения состояний линий порта не является единственным. Далее в программе, представлен вариант использования для этой цели команды rlf, которая производит циклический сдвиг содержимого регистра PortB влево на один бит. При этом, старший, седьмой бит сдвигаемого регистра помещается во флаг переноса-заема C (нулевой бит регистра STATUS), а младший, нулевой бит регистра загружается содержимым флага C, которое было в нем на момент выполнения команды rlf.

Формат команды:

rlf f,d – циклический сдвиг влево содержимого регистра f через перенос (бит C рег. STATUS). При этом результат сдвига сохраняется в аккумуляторе W, если d= 0, или же в самом сдвигаемом регистре f, если d = 1.



rff f,d – циклический сдвиг вправо содержимого → регистра f через перенос.

Пример циклического сдвига :

rlf PortB,1 ; циклический сдвиг влево содержимого регистра PortB через
; перенос с сохранением результата операции в регистре
; PortB. Если до операции содержимое регистра PortB было
; равно b'00000010', а содержимое флага C было равно 0, то
; после операции, содержимое регистра PortB будет равно
; b'00000100', а содержимое флага
; C также будет равно 0, поскольку седьмой бит регистра
; PortB до операции имел нулевое значение.

Для однозначного определения содержимого флага C, в программе производится обнуление его содержимого командой bcf STATUS,0. (флаг C является нулевым флагом регистра STATUS). После чего, производится циклический сдвиг содержимого регистра PortB влево. По завершении данной операции, единица в регистре PortB смещается с первой позиции на вторую, таким образом, обнуляется линия RB1, и выставляется логическая единица на линии RB2. Далее следует вызов подпрограммы задержки и двух калибровочных команд por. После чего опять производится обнуление флага C, сдвиг регистра, выполнение задержки и калибровочных констант. Данные операции повторяются до тех пор, пока не будет сформирован импульс на линии RB7. По завершении времени формирования импульса на линии RB7, происходит переход на новый виток формирования импульсной последовательности на линиях RB0 – RB7 при помощи команды goto TACH1. После чего, при помощи команды movlw b'00000001' константа помещается в аккумулятор, затем командой movwf PortB копируется в регистр PortB, после чего линия RB7 обнуляется, а на линии RB0 вновь появляется уровень логической единицы. И цикл повторяется.

Временные характеристики программы.

Пустые команды por используются в программе для уравнивания времени между переключениями линий. Переключение линий происходит по завершении команды movwf PortB или rlf PortB,1 . Поскольку команды mov, bcf, rlf, por выполняются за 1 машинный цикл (1мкс.) каждая, а подпрограмма задержки - за 1 с, то следовательно между переключениями линий пройдет 1с и 4 мкс: ((call Pause)+2*1(nor)+2*1(mov)) или (((call Pause)+2*1(nor)+ 1(bcf)+ 1(rlf)). При переходе на новый цикл программы используется команда goto, которая выполняется за 2 машинных цикла. Следовательно, между переключениями линий с RB7 на RB0 выполнится также 4 машинных цикла + задержка : (call

Pause)+2(goto)+ 2*1(mov)). Именно из-за наличия команды goto, которая выполняется за 2 машинных цикла, возникла необходимость добавления между переключениями большинства линий по две пустые команды пор. Иначе, время между переключениями отдельных линий было бы различным.

3. Порядок выполнения работы

3.1. Выберите вариант задания по таблицам 3 и 4.

3.2. Составьте схему электрическую принципиальную устройства.

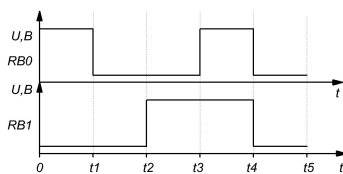
3.3. Составьте алгоритм управляющей программы для МК.

3.4. Создайте управляющую программу для МК.

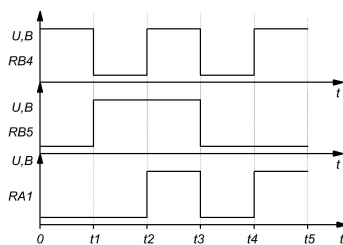
Таблица 3 – Задания на выполнение лабораторной работы.

Вар	Задание	t1	t2	t3	t4	t5	[t]	Вар.	Задание	t1	t2	t3	t4	t5	[t]
1	1	100	200	300	400	500	мкс	21	5	100	300	600	800	-	мкс
2	2	25	50	75	-	-	мс	22	6	400	500	600	700	900	мкс
3	3	70	140	210	280	320	мкс	23	7	100	200	300	400	-	мс
4	4	1,5	3,0	4,5	6	7,5	мс	24	8	70	140	210	-	-	мс
5	5	2	4	6	8	-	мс	25	1	120	180	300	360	420	мс
6	6	120	180	300	360	420	мкс	26	2	90	180	360	-	-	мкс
7	7	100	300	600	800	-	мкс	27	3	125	250	375	500	625	мкс
8	8	24	48	96	-	-	мс	28	4	30	60	90	120	150	мс
9	1	111	222	333	444	555	мкс	29	5	25	50	75	100	-	мкс
10	2	50	200	300	-	-	мс	30	6	13	26	39	52	65	мкс
11	3	10	20	30	40	70	мкс	31	7	1	2	3	4	-	мс
12	4	125	250	375	500	625	мкс	32	8	90	180	360	-	-	мс
13	5	1	2	3	4	-	мс	33	1	125	250	375	500	625	мкс
14	6	22	44	66	88	110	мкс	34	2	2	4	6	-	-	мс
15	7	90	180	270	360	-	мкс	35	3	30	60	90	120	150	мкс
16	8	1	2	3	-	-	мс	36	4	100	300	600	800	900	мс
17	1	125	250	375	500	625	мкс	37	5	70	140	210	280	-	мс
18	2	45	90	135	-	-	мс	38	6	250	500	600	700	750	мкс
19	3	120	180	300	360	420	мкс	39	7	9	18	36	45	-	мкс
20	4	400	500	600	700	900	мкс	40	8	24	48	96	-	-	мкс

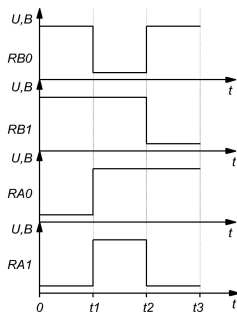
Задания для выполнения лабораторной работы: сгенерировать сигналы на линиях портов согласно графика, циклически.



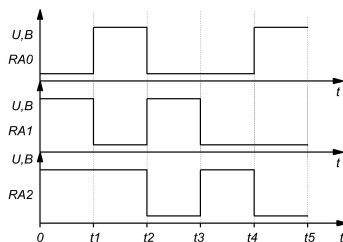
Задание 1



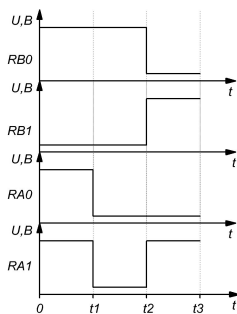
Задание 3



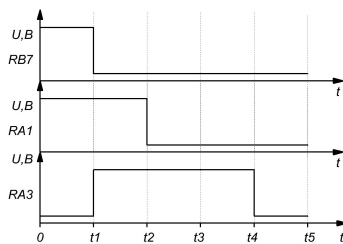
Задание 2



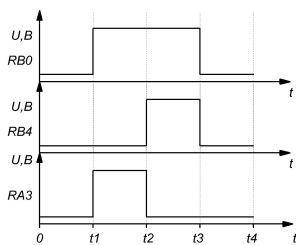
Задание 4



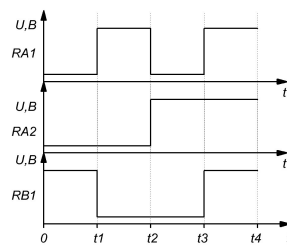
Задание 8



Задание 6



Задание 7



Задание 5

Таблица 4 – Задания на выполнение лабораторной работы –
содержимое слова Конфигурации.

Вар.	Защита памяти	Защита EEPROM	LVP	MCLR	PWRT	WDT	Генератор 4 МГц
------	------------------	------------------	-----	------	------	-----	--------------------

1	2	3	4	5	6	7	8
1	Выкл.	Выкл.	Выкл.	-MCLR	Выкл.	Выкл.	Кварцевый
2	Выкл.	Выкл.	Выкл.	Порт RA5	Вкл.	Вкл.	Внутр. INTRC
3	Выкл.	Выкл.	Выкл.	Порт RA5	Выкл.	Вкл.	Внутр. INTRC
4	Выкл.	Выкл.	Выкл.	Порт RA5	Выкл.	Выкл.	Внутр. INTRC
5	Выкл.	Выкл.	Выкл.	-MCLR	Вкл.	Выкл.	Кварцевый
6	Выкл.	Выкл.	Выкл.	-MCLR	Вкл.	Вкл.	Внутр. INTRC
7	Выкл.	Выкл.	Выкл.	Порт RA5	Выкл.	Выкл.	Кварцевый
8	Выкл.	Выкл.	Выкл.	-MCLR	Вкл.	Вкл.	Внутр. INTRC
9	Выкл.	Выкл.	Выкл.	Порт RA5	Выкл.	Выкл.	Кварцевый
10	Выкл.	Выкл.	Выкл.	-MCLR	Выкл.	Вкл.	Кварцевый
11	Выкл.	Выкл.	Выкл.	-MCLR	Вкл.	Выкл.	Кварцевый
12	Выкл.	Выкл.	Выкл.	Порт RA5	Выкл.	Выкл.	Внутр. INTRC
13	Выкл.	Выкл.	Выкл.	-MCLR	Выкл.	Вкл.	Внутр. INTRC
14	Выкл.	Выкл.	Выкл.	-MCLR	Вкл.	Вкл.	Кварцевый
15	Выкл.	Выкл.	Выкл.	-MCLR	Выкл.	Выкл.	Кварцевый
16	Выкл.	Выкл.	Выкл.	-MCLR	Выкл.	Вкл.	Кварцевый
17	Выкл.	Выкл.	Выкл.	-MCLR	Вкл.	Вкл.	Внутр. INTRC
18	Выкл.	Выкл.	Выкл.	-MCLR	Выкл.	Выкл.	Внутр. INTRC
19	Выкл.	Выкл.	Выкл.	Порт RA5	Вкл.	Выкл.	Кварцевый
20	Выкл.	Выкл.	Выкл.	-MCLR	Выкл.	Выкл.	Кварцевый
21	Выкл.	Выкл.	Выкл.	-MCLR	Вкл.	Выкл.	Кварцевый
22	Выкл.	Выкл.	Выкл.	-MCLR	Выкл.	Вкл.	Кварцевый
23	Выкл.	Выкл.	Выкл.	Порт RA5	Вкл.	Вкл.	Внутр. INTRC
24	Выкл.	Выкл.	Выкл.	Порт RA5	Выкл.	Выкл.	Внутр. INTRC
25	Выкл.	Выкл.	Выкл.	Порт RA5	Выкл.	Выкл.	Кварцевый
26	Выкл.	Выкл.	Выкл.	-MCLR	Выкл.	Вкл.	Кварцевый
27	Выкл.	Выкл.	Выкл.	-MCLR	Вкл.	Вкл.	Внутр. INTRC
28	Выкл.	Выкл.	Выкл.	-MCLR	Выкл.	Вкл.	Кварцевый
29	Выкл.	Выкл.	Выкл.	Порт RA5	Вкл.	Вкл.	Внутр. INTRC
30	Выкл.	Выкл.	Выкл.	Порт RA5	Выкл.	Выкл.	Кварцевый

31	Выкл.	Выкл.	Выкл.	-MCLR	Выкл.	Выкл.	Кварцевый
32	Выкл.	Выкл.	Выкл.	-MCLR	Выкл.	Вкл.	Внутр. INTRC
33	Выкл.	Выкл.	Выкл.	-MCLR	Вкл.	Вкл.	Внутр. INTRC
1	2	3	4	5	6	7	8
34	Выкл.	Выкл.	Выкл.	Порт RA5	Выкл.	Выкл.	Внутр. INTRC
35	Выкл.	Выкл.	Выкл.	Порт RA5	Выкл.	Вкл.	Внутр. INTRC
36	Выкл.	Выкл.	Выкл.	-MCLR	Вкл.	Вкл.	Кварцевый
37	Выкл.	Выкл.	Выкл.	-MCLR	Выкл.	Вкл.	Кварцевый
38	Выкл.	Выкл.	Выкл.	-MCLR	Вкл.	Вкл.	Внутр. INTRC
39	Выкл.	Выкл.	Выкл.	Порт RA5	Выкл.	Вкл.	Кварцевый
40	Выкл.	Выкл.	Выкл.	Порт RA5	Вкл.	Выкл.	Внутр. INTRC

4. Содержание отчета

4.1. Цель работы.

4.2. Сведения о командах ассемблера, используемых в работе.

4.3. Описание порядка составления программы.

4.4. Задание на выполнение работы.

4.5. Схема электрическая принципиальная создаваемого распределителя импульсов.

4.6. Алгоритм работы программы.

4.7. Текст программы для МК.

5. Контрольные вопросы для проверки готовности студентов к выполнению лабораторной работы.

5.1. Опишите состав и особенности портов ввода-вывода?

5.2. Опишите порядок работы с портами ввода-вывода?

5.3. Как рассчитать время выполнения фрагмента программы.

5.4. Расскажите об особенностях составления подпрограмм?

5.5. Опишите команды call, goto, return?

5.6. Как сформировать задержку заданной длительности?

6. Контрольные вопросы по итогам лабораторной работы.

6.1. Опишите назначение и состав слова Конфигурации?

6.2. Расскажите о порядке составления программы для МК?

6.3. Какие директивы ассемблера Вы знаете?

6.4. Какие существуют режимы тактового генератора?

6.5. Опишите назначение и работу команд rtf и rlf?

6.6. Опишите назначение и состав «шапки» программы?

7. ЛИТЕРАТУРА.

- 7.1. Сайт компании MICROCHIP - www.microchip.ru
- 7.2. Корабельников Е.А. Руководство по конструированию устройств на микроконтроллерах. 2006. www.ikarab.narod.ru
- 7.3. Предько М.Справочник по PIC-микроконтроллерам.— М.: ДМК Пресс, 2002.
- 7.4. Ульрих В.А. Микроконтроллеры PIC16X7XX. — СПб.: Наука и техника, 2002.
- 7.5. Катцен Сид PIC-микроконтроллеры. Полное руководство. — М.: Додэка-XXI, 2010.
- 7.6. Литвин А.В. и др. Методические указания к лабораторной работе «Регистры специального назначения». Ростов-на-Дону: ДГТУ, 2007.

Редактор А.А. Литвинова

ЛР N 020639 от 26.04.96. В набор . .11. В печать . .11.
Офсет. Объем 2.50 усл.п.л., уч. - изд.л. Формат 60x80/16
Бумага тип N3. Заказ N . Тираж . Цена

Издательский центр ДГТУ
Адрес университета и полиграфического предприятия:
344010, Ростов-на-Дону, пл. Гагарина, 1

ПРИЛОЖЕНИЕ А - Слово конфигурации (адрес 2007h)

CP	-	-	-	-	CP	LVP	BODEN	MCLR	FOSC2	-	WDTE	FOSC1	FOSC0
D					D			E		PWRT			
13	12	11	10	9	8	7	6	5	4	3	2	1	0

Бит 13	CP	– Бит защиты памяти программ											
		1 = защита памяти программ выключена											
		0 = защита памяти программ включена											
Биты 12,11,10,9		– Не реализованы.											
Бит 8	CPD	– Бит защиты EEPROM памяти данных											
		1 = защита памяти данных выключена											
		0 = защита памяти данных включена											
Бит 7	LVP	– Бит разрешения низковольтного программирования											
		1 = вывод RB4/PGM работает как PGM, режим низковольтного программирования включен											
		0 = вывод RB4/PGM работает как цифровой порт ввода/вывода, вывод HV используется для программирования микроконтроллера											
Бит 6	BODEN	– Бит разрешения сброса по снижению напряжения питания											
		1 = разрешен сброс BOR											
		0 = запрещен сброс BOR											
Бит 5	MCLRE	– Бит выбора режима работы вывода RA5/-MCLR											
		1 = RA5/-MCLR работает как -MCLR											
		0 = RA5/-MCLR работает как порт ввода/вывода, используется внутренний сброс -MCLR											
Бит 3	-PWRT	– Бит разрешения работы таймера включения питания											
		1 = PWRT выключен											
		0 = PWRT включен											
Бит 2	WDTE	– Бит разрешения работы сторожевого таймера											
		1 = WDT включен											
		0 = WDT выключен											
Биты 4,1,0	FOSC2:FOSC0	– Биты выбора режима тактового генератора											
		111 = RC генератор: вывод RA6/OSC2/CLKOUT работает как CLKOUT, RC цепь подключается к выводу RA7/OSC1/CLKIN											
		110 = RC генератор: вывод RA6/OSC2/CLKOUT работает как цифровой порт ввода/вывода, RC цепь подключается к выводу RA7/OSC1/CLKIN											
		101 = INTRC генератор: вывод RA6/OSC2/CLKOUT работает как CLKOUT, RA7/OSC1/CLKIN работает как цифровой порт ввода/вывода											
		100 = INTRC генератор: выводы RA6/OSC2/CLKOUT, RA7/OSC1/CLKIN работают как цифровые порты ввода/вывода											
		011 = EC генератор: вывод RA6/OSC2/CLKOUT работает как цифровой порт ввода/вывода, вывод RA7/OSC1/CLKIN работает как CLKIN											
		010 = HS генератор: высокочастотный резонатор подключается к выводам RA6/OSC2/CLKOUT, RA7/OSC1/CLKIN											
		001 = XT генер-р: резонатор подключается к выводам OSC2, OSC1											
		000 = LP генер-р: резонатор подключается к выводам OSC2, OSC1											

ПРИЛОЖЕНИЕ Б – Текст программы

```

;*****
; primer1.asm 13 ноября 2011 года.
; Пример № 1. Программа формирователя импульсов. PIC16F628A    Кварц 4 мГц.
;*****
LIST      p=16F628A      ; Установка типа микроконтроллера
include   <p16f628a.inc>  ; Подключение файла опций
__CONFIG  b'10000101100001'; Слово конфигурации
          ; Бит защиты выкл. Вывод RA5 как MCLR,
          ; WDT выключен, низко вольтн.прогр-выкл
          ; стандартный XT - генератор,
          ; BOR разрешен, PWRT- разрешен.
;*****
; Определение положения регистров специального назначения.
;*****
Status    equ    03h      ; Регистр выбора банка.
TrisB     equ    86h      ; Рег. настройки направл. работы выводов
          ; порта B.
PortB     equ    06h      ; Регистр управления защелками порта B
VRCON     equ    9Fh      ; Регистр управления источником опорного U.
CMCON     equ    1Fh      ; Регистр управления модулем компараторов.
T1CON     equ    10h      ; Регистр управления таймером TMR1.
T2CON     equ    12h      ; Регистр управления таймером TMR2.
CCP1CON   equ    17h      ; Регистр управления модулем CCP.
RCSTA     equ    18h      ; Регистр управления модулем USART.
;*****
; Определение названия и положения регистров общего назначения.
;*****
n         equ    70h      ; Счетчик времени
n1        equ    71h      ; Счетчик времени 1
n2        equ    72h      ; Счетчик времени 2
;*****
; Вход в программу.
;*****
org       0              ; Начать выполн. программы с адр. 0 PC.
goto     Start           ; Переход в ПП Start.
;*****
;***** ; Текст программы.
;*****
; Отключение не используемых модулей, настройка порта B на выход.
;*****
Start     bcf      RCSTA,7 ; Обнуление бита 7. Выкл. модуля USART.
          clrf     CCP1CON ; Обнуление регистра. в т.ч. биты 0-3.
          ; выключение модуля CCP
          bcf      T1CON,0 ; Обнуление бита 0 регистра T1CON.
          ; Выкл. таймера TMR1.
          bcf      T2CON,2 ; Обнуление бита 2. Выкл. таймера TMR2.
          clrf     CMCON   ; Обнуление регистра CMCON.
          comf     CMCON,1 ; Инверсия содержимого рег.CMCON. Биты
          ; 0-2 в '1'. Выкл. модуля компараторов.
          bcf      Status,6 ; Обнул. бита 6. Выбор банков 0 и 1.
          bsf      Status,5 ; Установка в '1' бита 5. Выбор банка 1.
          bcf      VRCON,7 ; Обнул. бита 7 рег.VRCON.Выкл.источника
          ; опорного напряжения.
          movlw    .0      ; Запись в аккумулятор (W) константы 0.
          movwf   TrisB   ; Копирование в TrisB содержимого W.
          ; Линии порта B настраиваются на выход.
          bcf      Status,5 ; Выбор банка 0.
          clrf     PortB   ; Обнуление защелок (линий) порта B.
TACH1     movlw    b'00000001'; Запись константы 00000001 в аккумулятор.
          movwf   PortB   ; Копирование константы в PortB. RB0='1'
          call     Pause   ; Переход на подпрограмму Pause
          nop       ; Калибровочный (пустой)машинный цикл
          nop       ; Калибровочный (пустой)машинный цикл
          movlw    b'00000010'; Запись константы 00000010 в аккумулятор.
          movwf   PortB   ; Копирование константы в PortB. RB1='1'
          call     Pause   ; Переход на подпрограмму Pause
          nop       ; Калибровочный (пустой)машинный цикл
          nop       ; Калибровочный (пустой)машинный цикл
          bcf      Status,0 ; Обнул. бита 0.(флага переноса-заема C)

```

```

rif      PortB,1      ; Циклический сдвиг влево содержимого
                        ; регистра PortB. RB2='1'
call     Pause        ; Переход на подпрограмму Pause
nop      ; Калибровочный (пустой)машинный цикл
nop      ; Калибровочный (пустой)машинный цикл
bcf      Status,0     ; Обнул. бита 0.(флага переноса-заема C)
rif      PortB,1      ; Циклический сдвиг влево содержимого
                        ; регистра PortB. RB3='1'
call     Pause        ; Переход на подпрограмму Pause
nop      ; Калибровочный (пустой)машинный цикл
nop      ; Калибровочный (пустой)машинный цикл
bcf      Status,0     ; -----//-----//-----
rif      PortB,1      ; RB4='1'
call     Pause        ; -----//-----//-----
nop      ; -----//-----//-----
nop      ; -----//-----//-----
bcf      Status,0     ; -----//-----//-----
rif      PortB,1      ; RB5='1'
call     Pause        ; -----//-----//-----
nop      ; -----//-----//-----
nop      ; -----//-----//-----
bcf      Status,0     ; -----//-----//-----
rif      PortB,1      ; RB6='1'
call     Pause        ; -----//-----//-----
nop      ; -----//-----//-----
nop      ; -----//-----//-----
bcf      Status,0     ; -----//-----//-----
rif      PortB,1      ; RB7='1'
call     Pause        ; -----//-----//-----
goto     TACH1        ; Переход на повтор цикла
;-----
; Подпрограмма формирования задержки в 999994 мкс.
;-----
Pause    movlw        .249      ; Запись числа 249 в регистр W
movwf    n1             ; Скопировать 249 из W в регистр n1
X2       call         Pause1    ; Переход на подпрограмму Pause1
call     Pause1         ; Переход на подпрограмму Pause1
call     Pause1         ; Переход на подпрограмму Pause1
call     Pause1         ; Переход на подпрограмму Pause1
decfsz   n1,1          ; Декремент содержимого регистра n1 с
goto     X2             ; помещением результата в него же.
call     Pause1         ; Переход на подпрограмму Pause1
call     Pause1         ; Переход на подпрограмму Pause1
call     Pause1         ; Переход на подпрограмму Pause1
movlw    .81            ; Запись числа 81 в регистр W
movwf    n2             ; Скопировать 81 из W в регистр n2
X3       decfsz       n2,1      ; n2 = n2 - 1, Если n2 не ноль,
goto     X3             ; то переход на X3, если ноль, то
return    ; выход из подпрограммы Pause.
;-----
; Подпрограмма формирования задержки в 998 мкс.
;-----
Pause1   movlw        .199      ; Запись числа 199 в регистр W
movwf    n              ; Скопировать 199 из W в регистр n
X1       nop          ; Калибровочный (пустой)машинный цикл
nop      ; Калибровочный (пустой)машинный цикл
decfsz   n,1           ; n = n - 1, Если n не ноль, то
goto     X1            ; переход на X1, если ноль, то
return    ; выход из подпрограммы Pause1.
;-----
end      ; Директива конца программы.

```