

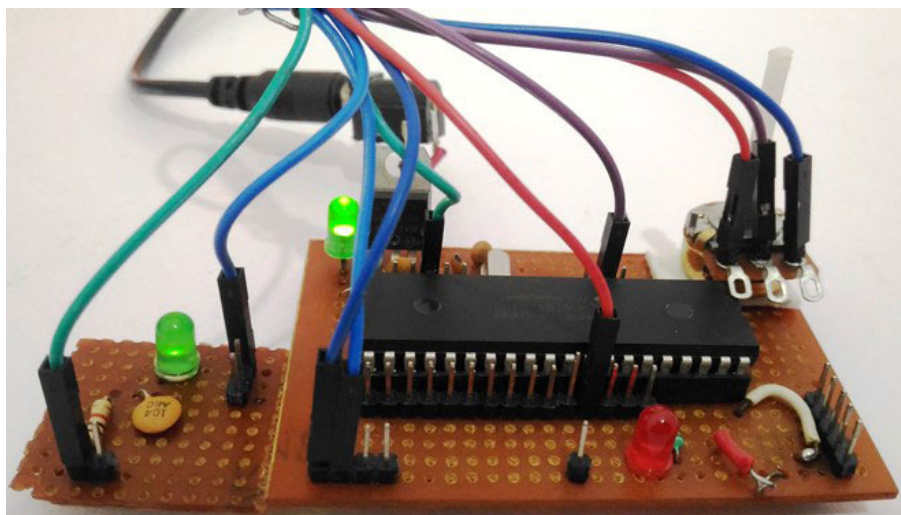
Генерация ШИМ с использованием микроконтроллера PIC с MPLAB и XC8 (/microcontroller-projects/pic-microcontroller-pic16f877a-pwm-tutorial)

По В.Асвинт Раджа (/users/baswinth-raj)
tutorial/#comments)

25 комментариев (/microcontroller-projects/pic-microcontroller-pic16f877a-pwm-tutorial/#comments)

Competitive price LCD display - LCM Manufacturer

WINSTAR- High quality of standard and total/semi custom LCD display modules. winstar.com.tw



Генерация ШИМ с использованием микроконтроллера PIC с MPLAB и XC8

This is our 10th tutorial of **Learning PIC microcontrollers using MPLAB and XC8**. Till now, we have covered many basic tutorials like [LED blinking with PIC](http://circuitdigest.com/microcontroller-projects/led-blinking-with-pic-microcontroller) (<http://circuitdigest.com/microcontroller-projects/led-blinking-with-pic-microcontroller>), [Timers in PIC](http://circuitdigest.com/microcontroller-projects/pic-microcontroller-timer-tutorial) (<http://circuitdigest.com/microcontroller-projects/pic-microcontroller-timer-tutorial>), [interfacing LCD](https://circuitdigest.com/microcontroller-projects/16x2-lcd-interfacing-with-pic-microcontroller) (<https://circuitdigest.com/microcontroller-projects/16x2-lcd-interfacing-with-pic-microcontroller>), [interfacing 7-segment](https://circuitdigest.com/microcontroller-projects/7-segment-display-interfacing-with-pic16f877a) (<https://circuitdigest.com/microcontroller-projects/7-segment-display-interfacing-with-pic16f877a>), [ADC using PIC](https://circuitdigest.com/microcontroller-projects/pic-microcontroller-pic16f877a-adc-tutorial) (<https://circuitdigest.com/microcontroller-projects/pic-microcontroller-pic16f877a-adc-tutorial>) etc. If you are an absolute beginner, then please visit the complete list of [PIC tutorials here](http://circuitdigest.com/pic-microcontroller-projects) (<http://circuitdigest.com/pic-microcontroller-projects>) and start learning.

Ads by Google

Code

PWM Motor Control Circuit

In this tutorial, we will learn **How to generate PWM signals using PIC PIC16F877A**. Our PIC MCU has a special module called **Compare Capture module (CCP)** which can be used to generate PWM signals. Here, we will generate a PWM of 5 kHz with a variable duty cycle from 0% to 100%. To vary the duty cycle we are using a potentiometer, hence it is recommended to learn [ADC tutorial](https://circuitdigest.com/microcontroller-projects/pic-microcontroller-pic16f877a-adc-tutorial) (<https://circuitdigest.com/microcontroller-projects/pic-microcontroller-pic16f877a-adc-tutorial>) before starting with PWM. PWM module also uses timers to set its frequency hence learn [how to use timers](http://circuitdigest.com/microcontroller-projects/pic-microcontroller-timer-tutorial) (<http://circuitdigest.com/microcontroller-projects/pic-microcontroller-timer-tutorial>) beforehand here. Further, in this tutorial we will use a RC circuit and a LED to convert the PWM values to Analog voltage and use it for dimming the LED light.

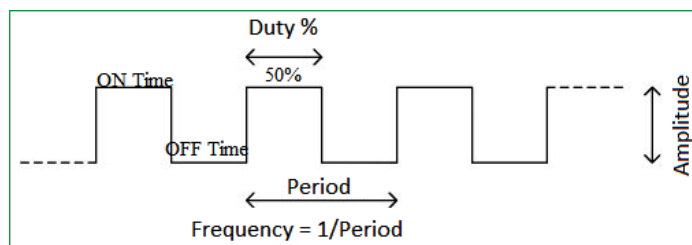
What is a PWM Signal?

Широтно-импульсная модуляция (PWM) - это цифровой сигнал, который наиболее часто используется в схемах управления. Этот сигнал устанавливается высоким (5 В) и низким (0 В) в заданное время и скорость. Время, в течение которого сигнал остается высоким, называется временем включения, а время, в течение которого сигнал остается низким, называется временем выключения. Для PWM есть два важных параметра:

Рабочий цикл ШИМ:

Процент времени, в течение которого PWM-сигнал остается HIGH (по времени), называется рабочим циклом. Если сигнал всегда включен, он работает в 100% рабочем цикле, и если он всегда выключен, это 0% рабочий цикл.

Рабочий цикл = время включения / (время включения + время выключения)



Частота ШИМ:



Частота ШИМ-сигнала определяет, как быстро PWM завершает один период. Один период является полным и выключенным сигналом ШИМ, как показано на рисунке выше. В нашем учебнике мы установим частоту 5 кГц.

PWM с использованием PIC16F877A:

ШИМ-сигналы могут быть сгенерированы в нашем микроконтроллере PIC с помощью модуля **CCP (Compare Capture PWM)**. Разрешение нашего ШИМ-сигнала 10-бит, то есть для значения 0 будет рабочий цикл 0%, а для значения 1024 (2^{10}) - рабочий цикл 100%. В нашем PIC MCU (CCP1 и CCP2) есть два модуля CCP, это означает, что мы можем генерировать два сигнала ШИМ на двух разных контактах (контакты 17 и 16) одновременно, в нашем учебнике мы используем CCP1 для генерации сигналов ШИМ на выводе 17.

Следующие регистры используются для генерации сигналов PWM с использованием нашего PIC MCU:

1. CCP1CON (регистр управления CCP1)
2. T2CON (регистр управления таймером 2)
3. PR2 (Регистр периода с таймером 2)
4. CCPR1L (CCP Register 1 Low)

Программирование PIC для генерации сигналов ШИМ:

В нашей программе мы будем читать аналоговое напряжение 0-5v от потенциометра и отображать его на 0-1024 с использованием нашего модуля ADC. Затем мы генерируем сигнал ШИМ с частотой 5000 Гц и меняем свой рабочий цикл на основе входного аналогового напряжения. То есть 0-1024 будет преобразован в 0% -100%. Этот учебник предполагает, что вы уже научились использовать [ADC в PIC](https://circuitdigest.com/microcontroller-projects/pic-microcontroller-pic16f877a-adc-tutorial) (<https://circuitdigest.com/microcontroller-projects/pic-microcontroller-pic16f877a-adc-tutorial>) если нет, прочитайте его здесь, потому что мы пропустим подробности об этом в этом учебнике.

Итак, как только биты конфигурации установлены (<https://circuitdigest.com/microcontroller-projects/writing-your-first-pic-microcontroller-program>) и программа записывается для чтения аналогового значения, мы можем продолжить работу с PWM.

При настройке модуля CCP для работы с PWM следует предпринять следующие шаги:

1. Установите период PWM, записав регистр PR2.
2. Установите рабочий цикл PWM, выполнив запись в регистр CCPR1L и CCP1CON <5: 4>.
3. Сделайте вывод CCP1 выводом, очистив бит TRISC <2>.
4. Установите значение предварительной шкалы TMR2 и включите Timer2, выполнив запись в T2CON.
5. Настройте модуль CCP1 для работы PWM.

В этой программе есть две важные функции для генерации сигналов ШИМ. Одна из них - это **функция PWM_Initialize ()**, которая будет инициализировать регистры, необходимые для настройки модуля PWM, а затем установить частоту, с которой должна работать PWM, а другая функция - функция **PWM_Duty ()**, которая установит рабочий цикл сигнала ШИМ в требуемые регистры.

```
PWM_Initialize ()
{
    PR2 = (_XTAL_FREQ / (PWM_freq * 4 * TMR2PRESCALE)) - 1; // Установка формул PR2 с использованием Datasheet // Делает работу PWM в 5KHz
    CCP1M3 = 1; CCP1M2 = 1; // Настройка модуля CCP1
    T2CKPS0 = 1; T2CKPS1 = 0; TMR2ON = 1; // Настройка модуля таймера
    TRISC2 = 0; // сделать вывод порта на C в качестве вывода
}
```

Вышеупомянутая функция - это функция инициализации PWM, в этой функции. Модуль CCP1 настроен на использование PWM, делая бит CCP1M3 и CCP1M2 высоким.

CCP1CON REGISTER/CCP2CON REGISTER (ADDRESS 17h/1Dh)							
U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	CCPxX	CCPxY	CCPxM3	CCPxM2	CCPxM1	CCPxM0
bit 7		bit 0					

bit 3-0 **CCPxM3:CCPxM0:** CCPx Mode Select bits

0000 = Capture/Compare/PWM disabled (resets CCPx module)
0100 = Capture mode, every falling edge
0101 = Capture mode, every rising edge
0110 = Capture mode, every 4th rising edge
0111 = Capture mode, every 16th rising edge
1000 = Compare mode, set output on match (CCPxIF bit is set)
1001 = Compare mode, clear output on match (CCPxIF bit is set)
1010 = Compare mode, generate software interrupt on match (CCPxIF bit is set, CCPx pin is unaffected)
1011 = Compare mode, trigger special event (CCPxIF bit is set, CCPx pin is unaffected); CCP1 resets TMR1; CCP2 resets TMR1 and starts an A/D conversion (if A/D module is enabled)
11xx = PWM mode

Предупредитель модуля таймера устанавливается, делая бит T2CKPS0 высоким, а T2CKPS1 - как низкий бит TMR2ON установлен для запуска таймера.

T2CON: TIMER2 CONTROL REGISTER (ADDRESS 12h)							
U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0
bit 7		bit 0					
bit 1-0		T2CKPS1 (1) T2CKPS0: Timer2 Clock Prescale Select bits					
		00 = Prescaler is 1					
		01 = Prescaler is 4					
		1x = Prescaler is 16					

Теперь мы должны **установить частоту сигнала ШИМ**. Значение частоты должно быть записано в регистр PR2. Желаемую частоту можно установить, используя приведенные ниже формулы

$$PWM \text{ Period} = [(PR2) + 1] * 4 * TOSC * (\text{значение пресета TMR2})$$

Переупорядочение этих формул для получения PR2 даст

$$PR2 = (Period / (4 * TOSC * TMR2 \text{ Prescale})) - 1$$

акций

Мы знаем, что $Period = (1 / PWM_freq)$ и $TOSC = (1 / _XTAL_FREQ)$. Следовательно.....

$$PR2 = (_XTAL_FREQ / (PWM_freq * 4 * TMR2PRESCALE)) - 1;$$

Как только частота установлена, эта функция не требуется вызывать снова, пока и пока нам не понадобится снова менять частоту. В нашем учебнике я назначил $PWM_freq = 5000$; так что мы можем получить рабочую частоту 5 КГц для нашего сигнала ШИМ.

Теперь зададим **рабочий цикл ШИМ**, используя следующую функцию

```

PWM_Duty (беззнаковый int duty)
{
    если (долг <1023)
    {

        duty = ((float) duty / 1023) * (_XTAL_FREQ / (PWM_freq * TMR2PRESCALE)); // Об уменьшении // duty = (((float) duty / 1023) * (1 / PWM_fr
        CCP1X = duty & 1; // Сохраняем 1-й бит
        CCP1Y = duty & 2; // Хранить 0-й бит
        CCP1L = duty >> 2; // Хранить остаток 8 бит
    }
}

```

Наш PWM-сигнал имеет 10-битное разрешение, поэтому это значение не может быть сохранено в одном регистре, так как наш PIC имеет только 8-битные строки данных. Таким образом, мы используем другие два бита CCP1CON <5: 4> (CCP1X и CCP1Y) для хранения последних двух LSB, а затем сохраняем оставшиеся 8 бит в регистре CCP1L.

Время цикла PWM может быть рассчитано с использованием приведенных ниже формул:

$$\text{PWM Duty Cycle} = (\text{CCP1L} : \text{CCP1CON} \langle 5: 4 \rangle) * T_{\text{osc}} * (\text{значение пресета TMR2})$$

Переупорядочение этих формул для получения значения CCP1L и CCP1CON даст:

$$\text{CCP1L} : \text{CCP1CON} \langle 5: 4 \rangle = \text{PWM Duty Cycle} / (T_{\text{osc}} * \text{TMR2 Prescale Value})$$

Значение нашего АЦП будет 0-1024, нам нужно, чтобы оно составляло 0% -100%, следовательно, $\text{PWM Duty Cycle} = \text{duty} / 1023$. Далее, чтобы преобразовать этот рабочий цикл в промежуток времени, мы должны умножить его на период $(1 / \text{PWM_freq})$

Мы также знаем, что $T_{\text{osc}} = (1 / \text{PWM_freq})$, следовательно ..

$$\text{Duty} = (((\text{float}) \text{duty} / 1023) * (1 / \text{PWM_freq})) / ((1 / _ \text{XTAL_FREQ}) * \text{TMR2PRESCALE});$$

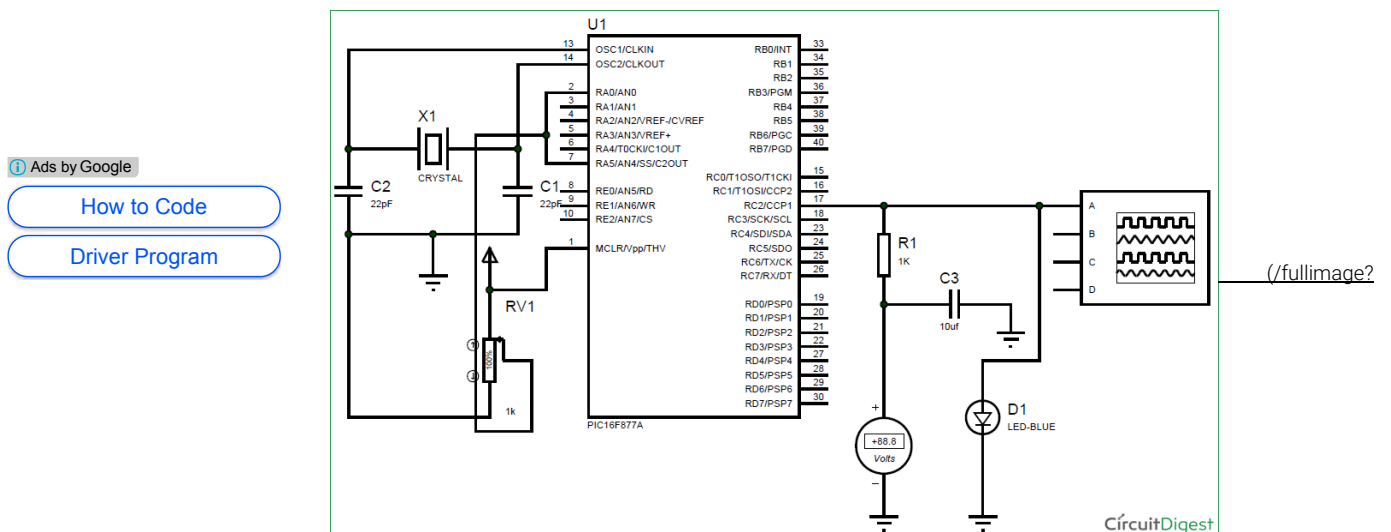
Решение вышеуказанного уравнения даст нам:

$$\text{Duty} = ((\text{float}) \text{duty} / 1023) * (_ \text{XTAL_FREQ} / (\text{PWM_freq} * \text{TMR2PRESCALE}));$$

Вы можете проверить **полную программу** в разделе «Код» ниже вместе с подробным **видео** .

Схемы и тестирование:

Как обычно, проверим вывод с использованием моделирования Proteus. **Принципиальная схема** показана ниже.



[i=circuitdiagram_mic/PWM-with-PIC-Microcontroller-MPLAB-XC8-circuit-diagram.gif](#)

1 Rheostat

Sliding, Rotary, Motorized Standard or Special 16W to 7kW+
coudoint.com

2 Kendo UI for React

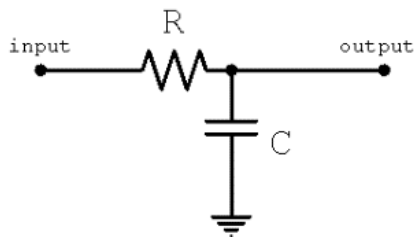
The Kendo UI library provides everything you need to integrate
with React out of the box. telerik.com

Подключите потенциометр к 7-му контакту, чтобы подать напряжение 0-5. Модуль CCP1 имеет контакт 17 (RC2), здесь генерируется PWM, который может быть проверен с помощью цифрового осциллографа. Кроме того, чтобы преобразовать это в переменное напряжение, мы использовали RC-фильтр и светодиод для проверки вывода без возможности.

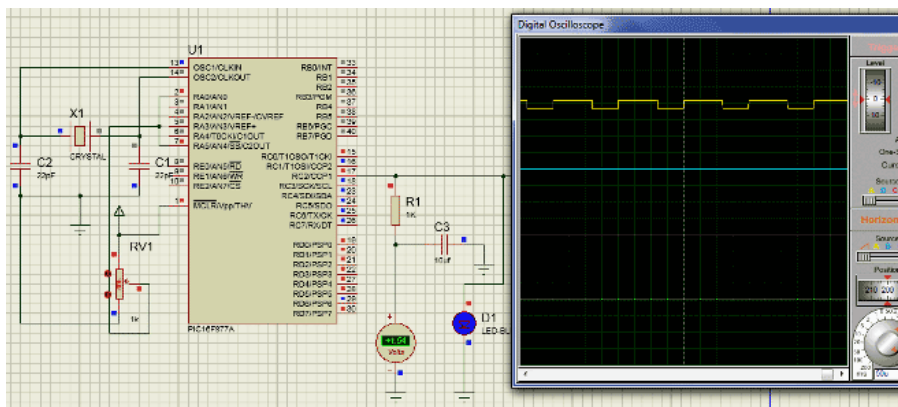
Что такое RC-фильтр?

RC - фильтр или фильтр нижних частот представляет собой простой контур с двумя пассивными элементами, а именно резистором и конденсатором. Эти два компонента используются для фильтрации частоты нашего ШИМ-сигнала и обеспечения его постоянного напряжения постоянного тока.

Если мы рассмотрим схему, когда переменное напряжение подается на вход R, конденсатор C начнет заряжаться. Теперь, основываясь на значении конденсатора, конденсатор займет некоторое время, чтобы полностью зарядиться, после зарядки он блокирует ток постоянного тока (помните, что конденсаторы блокируют постоянный ток, но пропускают переменный ток), поэтому входное напряжение постоянного тока будет отображаться на выходе. Высокочастотный ШИМ (сигнал переменного тока) будет заземлен через конденсатор. Таким образом, чистый конденсатор получается через конденсатор. Было установлено, что значение 1000 Ом и 1μf является подходящим для этого проекта. Вычисление значений R и C включает в себя анализ схемы с использованием передаточной функции, которая выходит за рамки данного руководства.

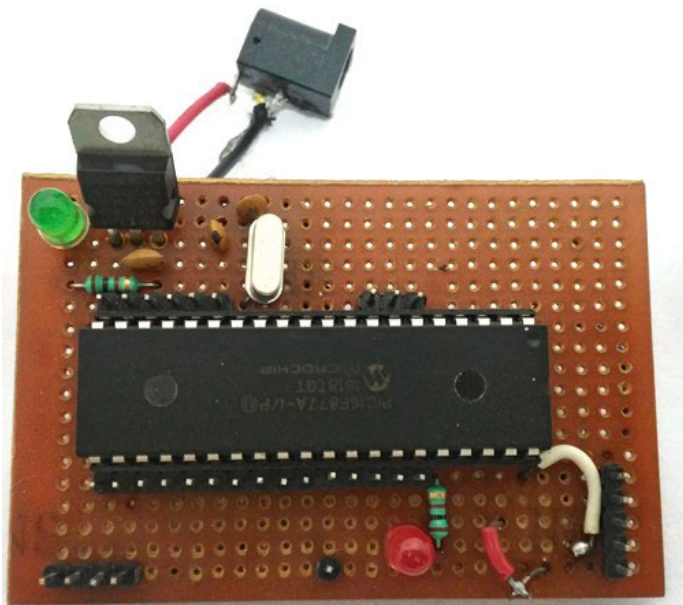


Выходной сигнал программы можно проверить с помощью цифрового осциллографа, как показано ниже, изменить потенциометр, а рабочий цикл PWM должен измениться. Мы также можем заметить выходное напряжение схемы RC с помощью вольтметра. Если все работает должным образом, мы можем продолжить наше оборудование. Далее проверьте видео в конце для полного процесса.

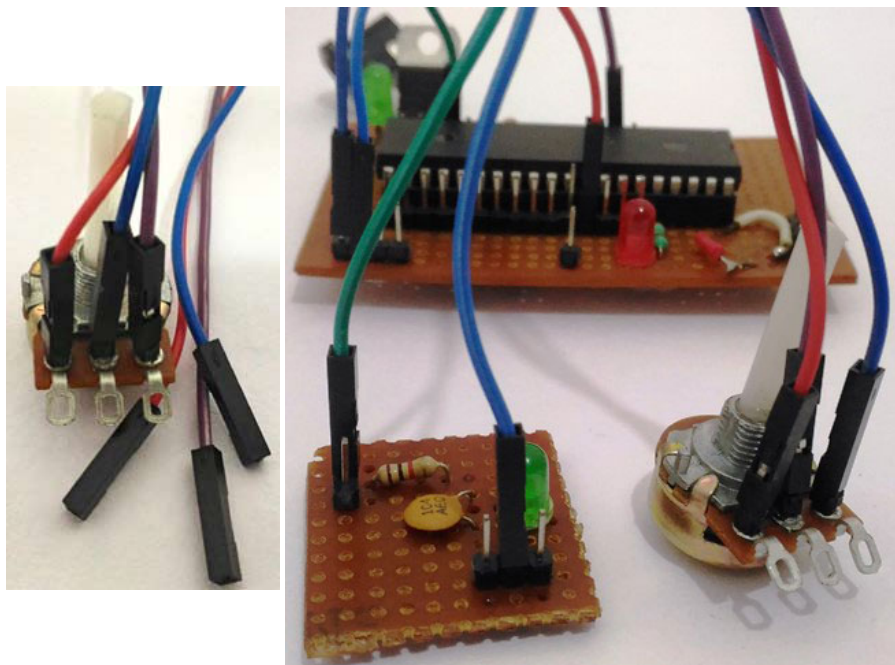


Работа с оборудованием:

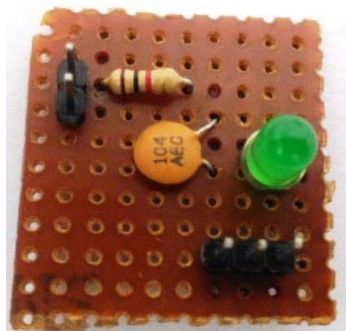
Аппаратная настройка проекта очень проста, мы просто собираемся повторно использовать нашу [панель PIC Perf](http://circuitdigest.com/microcontroller-projects/led-blinking-with-pic-microcontroller) (<http://circuitdigest.com/microcontroller-projects/led-blinking-with-pic-microcontroller>) показанную ниже.



Нам также понадобится **потенциометр для подачи аналогового напряжения**, я подключил некоторые женские концевые провода к моему банку (показано ниже), чтобы мы могли напрямую подключить их к плате PIC Perf.

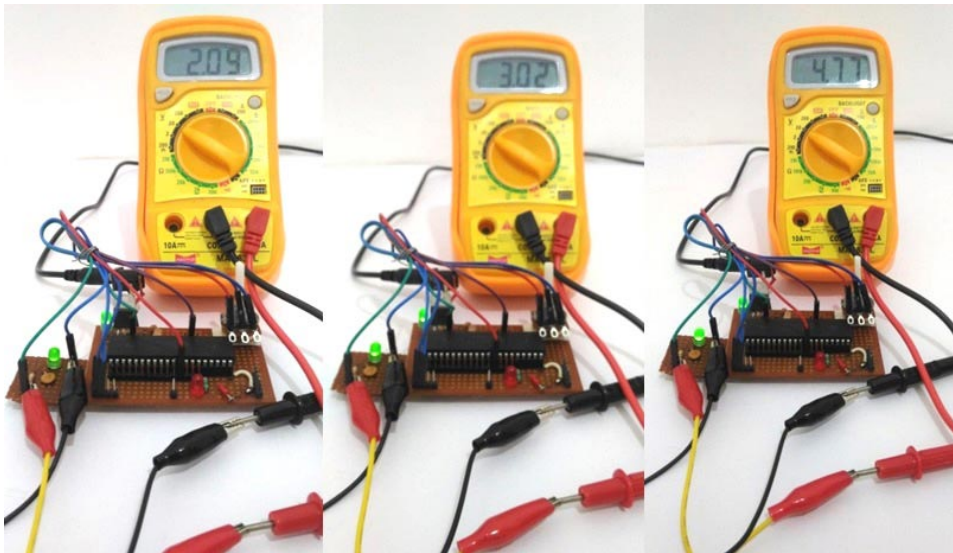


Наконец, чтобы проверить выход, нам нужна **RC-схема и светодиод**, чтобы увидеть, как работает сигнал PWM. Я просто использовал небольшую перфорированную плату и спаял RC-схему и светодиод (для управления яркостью) на нем, как показано ниже



Мы можем использовать простую соединительную проушину для женщин и женщин и подключать их согласно схемам, показанным выше. Как только соединение будет выполнено, загрузите программу в PIC, используя наш pickit3, и вы сможете получить переменное напряжение, основанное на вводе вашего потенциометра. Переменный выход используется для управления яркостью светодиода.

Я использовал свой мультиметр для измерения переменных выходов, мы также можем заметить, что яркость светодиода меняется на разные уровни напряжения.



Это мы запрограммировали на чтение аналогового напряжения от POT и преобразование в PWM-сигналы, которые, в свою очередь, были преобразованы в переменное напряжение с использованием RC-фильтра, и результат проверяется с использованием нашего оборудования. Если у вас есть какие-то сомнения или застряли где-нибудь, пожалуйста, используйте раздел комментариев ниже, мы будем рады вам помочь.
Полный рабочий работает в видео .

Также ознакомьтесь с нашими другими учебниками PWM на других микроконтроллерах:

- [Учебное пособие по малине Pi PWM \(https://circuitdigest.com/microcontroller-projects/raspberry-pi-pwm-tutorial\)](https://circuitdigest.com/microcontroller-projects/raspberry-pi-pwm-tutorial)
- [PWM с Arduino Due \(https://circuitdigest.com/microcontroller-projects/arduino-due-pwm-tutorial\)](https://circuitdigest.com/microcontroller-projects/arduino-due-pwm-tutorial)
- [Светодиодный диммер Arduino на основе PWM \(https://circuitdigest.com/microcontroller-projects/arduino-pwm-with-led-dimmer\)](https://circuitdigest.com/microcontroller-projects/arduino-pwm-with-led-dimmer)
- [Светодиодный индикатор питания с использованием микроконтроллера ATmega32 \(https://circuitdigest.com/microcontroller-projects/power-led-dimmer-using-atmega32-pwm\)](https://circuitdigest.com/microcontroller-projects/power-led-dimmer-using-atmega32-pwm)

Код:

```
// CONFIG
#pragma config FOSC = HS // Биты выбора осциллятора (генератор HS)
#pragma config WDTE = OFF // Сторожевой таймер Включить бит (WDT отключен)
#pragma config PWRT = ON // Включить бит включения питания (PWRT включено)
#pragma config BOREN = OFF // Браун-выход Сброс Разрешающий бит (BOR отключен)
#pragma config LVP = ON // Низковольтное (однополярное) последовательное программирование бит включения (бит RB3 / PGM имеет PGM функция, низковольтное программирование включено)
#pragma config CPD = OFF // Data EEPROM Память кода защиты бит (защита данных EEPROM отключена)
#pragma config WRT = OFF // Flash Program Memory Write Enable bits (Защита от записи выключена, вся программа память может быть записана с помощью элемента управления EECON)
#pragma config CP = OFF // Бит памяти программной защиты Бит защиты (защита кода выключена)

#define _XTAL_FREQ 20000000
#define TMR2PRESCALE 4

#include <xc.h>

long PWM_freq = 5000;

PWM_Initialize ()
{
    PR2 = (_XTAL_FREQ / (PWM_freq * 4 * TMR2PRESCALE)) - 1; // Установка формул PR2 с использованием Datasheet // Делает работу PWM в 5KHZ
    CCP1M3 = 1; CCP1M2 = 1; // Настройка модуля
    CCP1 T2CKPS0 = 1; T2CKPS1 = 0; TMR2ON = 1; // Настройка модуля таймера
    TRISC2 = 0; // сделать вывод порта на C в качестве вывода
}

PWM_Duty (беззнаковый int duty)
{
    if (duty < 1023)
    {
        duty = ((float) duty / 1023) * (_XTAL_FREQ / (PWM_freq * TMR2PRESCALE)); // Об уменьшении // duty = (((float) duty / 1023) * (1 / PWM_freq)) / ((1 / _XTAL_FREQ) * TMR2PRESCALE);
        CCP1X = duty & 1; // Хранить первый бит
        CCP1Y = duty & 2; // Хранить 0-й бит
        CCP1L = duty >> 2; // Хранить остаток 8 бит
    }
}

void ADC_Initialize ()
{
    ADCON0 = 0b01000001; // ADC ON и Fosc / 16 выбрано
```

```
ADCON1 = 0b11000000; // опорное напряжение внутреннего выбран
}
без знака Int ADC_Read (неподписанные символ канала)
{
    ADCON0 &= 0x11000101; // Очистка битов выбора канала
    ADCON0 |= канал << 3; // Установка необходимых битов
    __delay_ms (2); // Время сбора зарядового конденсатора
    GO_nDONE = 1; // Инициализирует преобразование A / D
    во время (GO_nDONE); //
    Ожидаем конверсии A / D для завершения возврата ((ADRESH << 8) + ADRESL); // Возвращает результат
}

void main ()
{
    int adc_value;
    TRISC = 0x00; // PORTC как выход
    TRISA = 0xFF; // PORTA в качестве входного
    TRISD = 0x00;
    ADC_Initialize (); // Инициализирует модуль АЦП
    PWM_Initialize (); // Это устанавливает частоту PWM PWM1

    do
    {
        adc_value = ADC_Read (4); // Чтение аналогового канала 0
        PWM_Duty (adc_value);

        __delay_ms (50);

    } в то время как (1); // Бесконечная петля
}
```

Видео:

How to Generate PWM using PIC Microcontroller




JLPCB - Prototype PCB за 2 доллара США + Бесплатная доставка по первому заказу

(<https://jlcpcb.com/>)

Крупнейший изготовитель прототипов печатных плат в Китае, 290 000+ клиентов и 8 000+ онлайн-заказов в день (<https://jlcpcb.com/>)

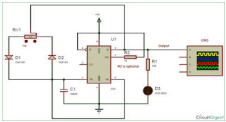
10 печатных плат Цена: 2 доллара за 2 слоя, 15 долларов США за 4 слоя, 74 доллара за 6-слойную

(<https://jlcpcb.com/quote>)




Rheostat

Ad [www.coudoint.com](#)



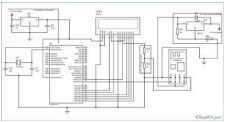
555 Timer PWM Generator Circuit Diagram

[circuitdigest.com](#)




LCM Manufacturer - Competitive price LCD display

Ad [winstar.com.tw](#)




Interfacing PIC Microcontroller with ESP8266 WiFi...

[circuitdigest.com](#)



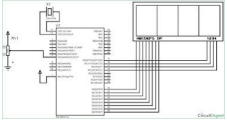
Arduino Project Builder

Ad [Cayenne](#)



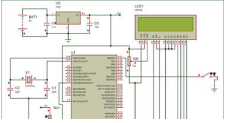
Understanding Timers in PIC Microcontroller...

[circuitdigest.com](#)



PIC microcontroller PICF877A ADC Tutorial using...

[circuitdigest.com](#)




Tutorial to Use PIC16F877A Microcontroller...

[circuitdigest.com](#)

[Добавить комментарий \(/microcontroller-projects/pic-microcontroller-pic16f877a-pwm-tutorial#comment-form\)](#)

Комментарии (25)



Arkan MR

Hi sir,

I am electrical engineering and I need more simple practical circuits about (delay time cct) in my work (electronic change over equipment) so as to make more safety in the work

Asimple practical cct please

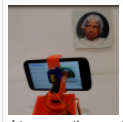
And I would like to thank you very much about all these informatios

thank you very much

Arkan M R .

[Ответить \(/comment/reply/827/12172\)](#)

Mar 17, 2017



[\(/users/baswinth-raj\)](#)

B.Aswinth Raj (/users/baswinth-raj)

Hi Arkan,

imple practical circuits about (delay time cct) in my work (electronic change over equipment) so as to make more safety in the work

Kindly explain your requirements in detail, we will be happy to help you out!!


And I would like to thank you very much about all these informatios

thank you very much

Thanks Arkan, your comments encourage us..

[reply \(/comment/reply/827/12304\)](#)

Mar 19, 2017



Ann

Hi. I am writing a PWM code for PIC18f4550 micrococontroller. I am using MPLAB X IDE with XC8. I am referring to your code to write a simple PWM code to test it on oscilloscope and later on build on it to do more things. I don't understand what are:

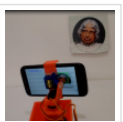
CCP1X = duty & 1; //Store the 1st bit

CCP1Y = duty & 2; //Store the 0th bit

What are CCP1X and CCP1Y? What is their role?

[reply \(/comment/reply/827/12835\)](#)

Mar 30, 2017



[\(/users/baswinth-raj\)](#)

B.Aswinth Raj (/users/baswinth-raj)

Hi Ann,

I will explain the following lines from my code

[reply \(/comment/reply/827/12871\)](#)

```
CCP1X = duty & 1; //Store the 1st bit
CCP1Y = duty & 2; //Store the 0th bit
CCPR1L = duty>>2; // Store the remaining 8 bit
```

As you can see the variable duty contains the value of the PWM duty cycle. This value has to be used to set the CCP1X and CCP1Y bit and also the CCPR1L register.

The following lines are from the datasheet

The PWM duty cycle is specified by writing to the CCPR1L register and to the CCP1CON<5:4> bits. Up to 10-bit resolution is available. The CCPR1L contains the eight MSBs and the CCP1CON<5:4> contains the two LSbs.

CCP1CON<5:4> is nothing but the CCP1X and CCP1Y bits respectively. The value of duty after this formulae (shown below) can be up to 10-bit resolution

$$\text{duty} = ((\text{float})\text{duty}/1023)*(_XTAL_FREQ/(\text{PWM_freq}*\text{TMR2PRESCALE}));$$

Since the Registers in our PIC MCU can hold only up to 8-bits (PIC16F877A is a 8-bit MCU) the additional two bits (CCP1X and CCP1Y) are used to store the least two significant bits and the remaining 8 bits are stored into the CCPR1L register.

Hope this answered your question.

Mar 31, 2017



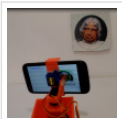
Thejo

reply (/comment/reply/827/15304)

Hi,

I checked your code for my project and it worked well. now the problem is only the led is dimming but the fan or bulb doesn't dimming. Help me in this.

Jun 16, 2017



(/users/baswinth-raj)

B.Aswinth Raj (/users/baswinth-raj)

reply (/comment/reply/827/15308)

Hi Thejo,

Glad that the code worked for you...

I am not sure how you have interfaced the fan or bulb with the PIC MCU. Let me know what your driver circuit is. If you have used a relay it will not work. Hence make sure you use a power electronic device like MOSFET and you switch it properly using the PWM signals.

Jun 16, 2017



emon

reply (/comment/reply/827/15487)

the timer0 in PIC18F is used to get 13ms delay. can u show steps to get the value of T0CON based on TMR0H=0x02 and TMR0L=0x12 and Fosc=20MHz.

Jun 22, 2017



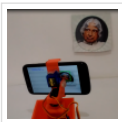
Abel

reply (/comment/reply/827/16599)

hello! please from "duty = ((float)duty/1023)*(_XTAL_FREQ/(PWM_freq*TMR2PRESCALE));" what is "float" doing in there? is it a variable or what??

why did you have to add float to the formulae??

Aug 03, 2017



(/users/baswinth-raj)

B.Aswinth Raj (/users/baswinth-raj)

reply (/comment/reply/827/16875)

No Float is not a variable.

Here the data type of duty is integer. So when this integer value is divided by 1023 it will also be an integer. But we need it in terms of decimal for better accuracy. So we prefix (float) to ask the compiler not to truncate the obtained result to integer but keep it in float as such.

Hope this clears your doubt.

Thanks,

Aswinth

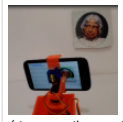
Aug 14, 2017

**Abel**[reply \(/comment/reply/827/16970\)](#)

please give me an example of this... am a bit confuse.. from PWM_Duty(adc_value); i understand that, voltage from 0-5v will also be converted from 0-1023

take for instance adc_value=1020 which in our duty formulae has been divided by 1023.duty=((float)duty/1023) saying (1020/1023)=0.9971 which will result everything to be zero.. so please explain to me how (float) works in that code with an example of duty result between 0-5v.. Thank you!(i want to move to the next tutorial after getting full understanding of this)

Aug 18, 2017



(/users/baswinth-raj)

B.Aswinth Raj (/users/baswinth-raj)[reply \(/comment/reply/827/16971\)](#)

Yes you got it partially correct.

$(1020/1023)=0.9971$

But, the value 0.9971 will not be converted to 0 yet. Because we still have the remaning formulae to complete.

so,

$(duty/1023) * (20000000/(5000*4))$

Since you took an example of 1020 as duty

$0.9971 * (20000000/(5000*4))$

Which will be $0.9971*(1000) = 997$

PWM_Duty(997);

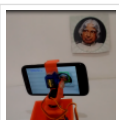
Hope this clears things for you

Aug 18, 2017

**Abel**[reply \(/comment/reply/827/16973\)](#)

wow!! Thanks ... confusion cleared!

Aug 18, 2017



(/users/baswinth-raj)

B.Aswinth Raj (/users/baswinth-raj)[reply \(/comment/reply/827/17041\)](#)

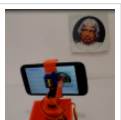
Always Welcome!!!!

Aug 20, 2017

**zain**[reply \(/comment/reply/827/16884\)](#)

hi, can i use my keypad as an input to generate pwm to control the angle of the servo motor ?e.g opening a door or locking the door with the input on the keypad

Aug 14, 2017



(/users/baswinth-raj)

B.Aswinth Raj (/users/baswinth-raj)[reply \(/comment/reply/827/16892\)](#)

Yes, you can...

Interface a keyboard with microcontroller and read the entered values. Then write that value to PWM_duty() function. This way you should be able to control the Servo motor based on your entered value.

Aug 15, 2017



(/users/skumar)

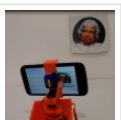
S.Kumar (/users/skumar)[reply \(/comment/reply/827/18142\)](#)

iam intersted this pwm generating ckt.

first of all i thanks you lot about this tutorials .

can you help me how to modify this ckt instead of pot. make it digital through Up & Down push buttons.... is it possible?

Sep 17, 2017



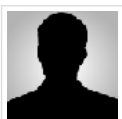
(/users/baswinth-raj)

B.Aswinth Raj (/users/baswinth-raj)[reply \(/comment/reply/827/18159\)](#)

Hi Kumar,

Yes, it is very much possible. Here in this code above. The variable adc_value controls the duty cycle of the output PWM signal. You simply have to connect two push buttons and use some If statements to check if they are pressed. If pressed simply increment/decrement the value of adc_value. note the range of this variable should only be from 0-1024

Sep 18, 2017

**Yashashree Jadhav**[reply \(/comment/reply/827/18329\)](#)

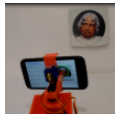
Hi,
This code is very helpful to get the basic idea for coding for a pwm output.
Can you guide me to design a pwm output for pic16f18857.
Thanks

Sep 23, 2017

**Yashashree Jadhav**[reply \(/comment/reply/827/18330\)](#)

hhow do i make this run for pic16f18857

Sep 23, 2017

**B.Aswinth Raj (/users/baswinth-raj)**[reply \(/comment/reply/827/18375\)](#)

The method is the same. Only the name of registers might change. Read the datasheet of PIC16F18857 and modify the code accordingly

Sep 24, 2017

(/users/baswinth-raj)

**adam**[reply \(/comment/reply/827/19319\)](#)

hi sir
,
this project helped me.
nice and great.....
but
i want to make this project on mickro c compiler..plz..pic16f877a

Oct 21, 2017

**Jack**[reply \(/comment/reply/827/21283\)](#)

Hi,
I am using a PIC16F873A with a variation of your code that uses a for() loop to increase and decrease the duty cycle. I am using a LED component that effectively has RGB LED's, connected to PORTC 1, 2, and 3. Is there any way of changing which pin the pwm signal outputs on? have tried altering the TRISC2=0 to TRIS1=0 etc to no avail. is it possible to output the signal on multiple pins at the same time?

Dec 18, 2017

**Sonya**[reply \(/comment/reply/827/21753\)](#)

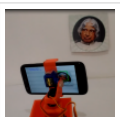
Hello!
Never have I used PWM signals before so it's kinda tricky to me. I am using PIC18F8722 and am supposed to build a function with the following structure : void config_PWM(int en, int ch, int freq, int period, int c_duty) . //en=enable the module, ch=PWM output channel, freq=osc. freq., period=PWM period(0-127), c_duty=cycle duty(0-1024) .
All the information here has been really helpful, I am just having difficulties in putting all this stuff together within one function. If you could please land me a hand I would be very grateful, since I won't have access to the microcontroller til next week. Thank you!

Jan 03, 2018

**CB**[reply \(/comment/reply/827/22209\)](#)

If I may ask, why is the signal from the potentiometer being fed to both RA0 and RA5 ?
Thanks

Jan 16, 2018

**B.Aswinth Raj (/users/baswinth-raj)**[reply \(/comment/reply/827/22247\)](#)

Sorry for the mistake CB, you can connect it either to RA0 or to RA5, the code here reads voltage from pin RA5. So if you are using this code then you can ignore the connection to RA0

Jan 18, 2018

(/users/baswinth-raj)

Leave a commentYour name * E-mail *

The content of this field is kept private and will not be shown publicly.

Subject

Comment *

No HTML tags allowed.

Web page addresses and e-mail addresses turn into links automatically.

Lines and paragraphs break automatically.

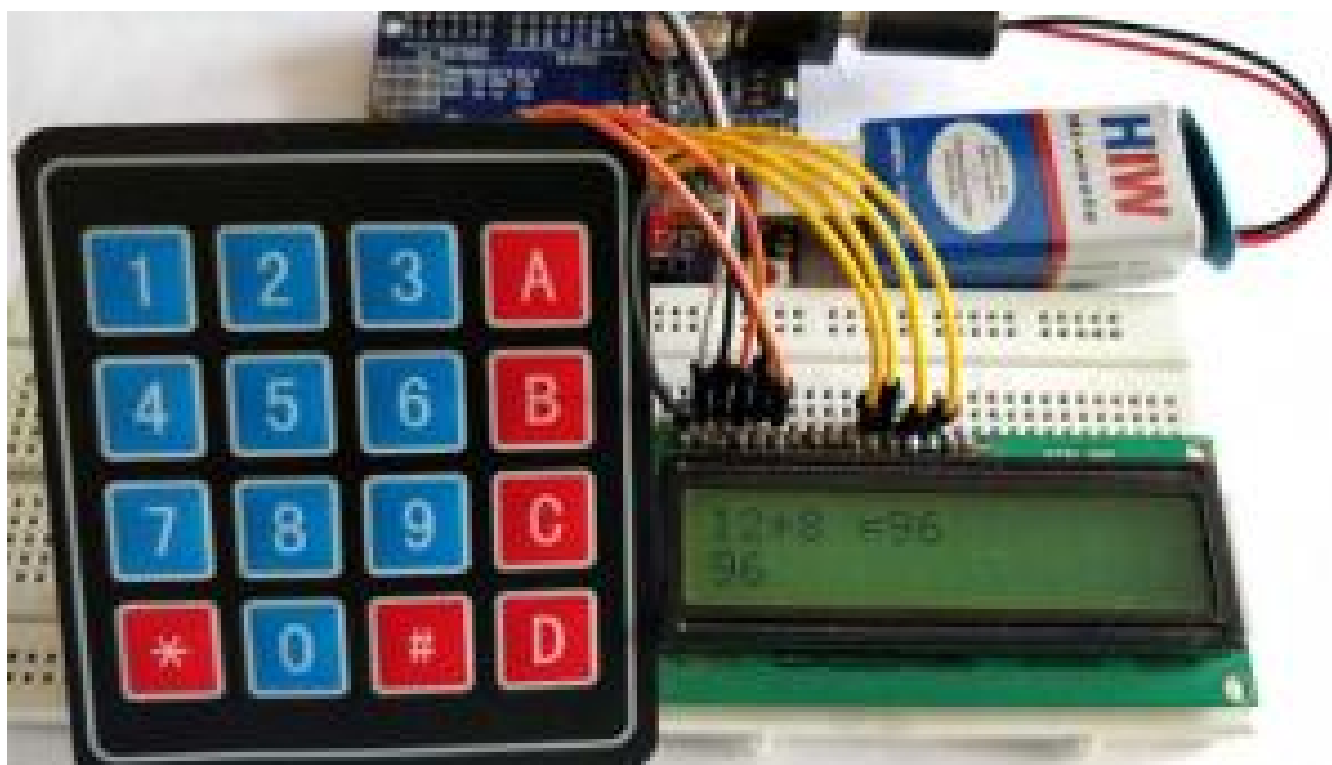
Save

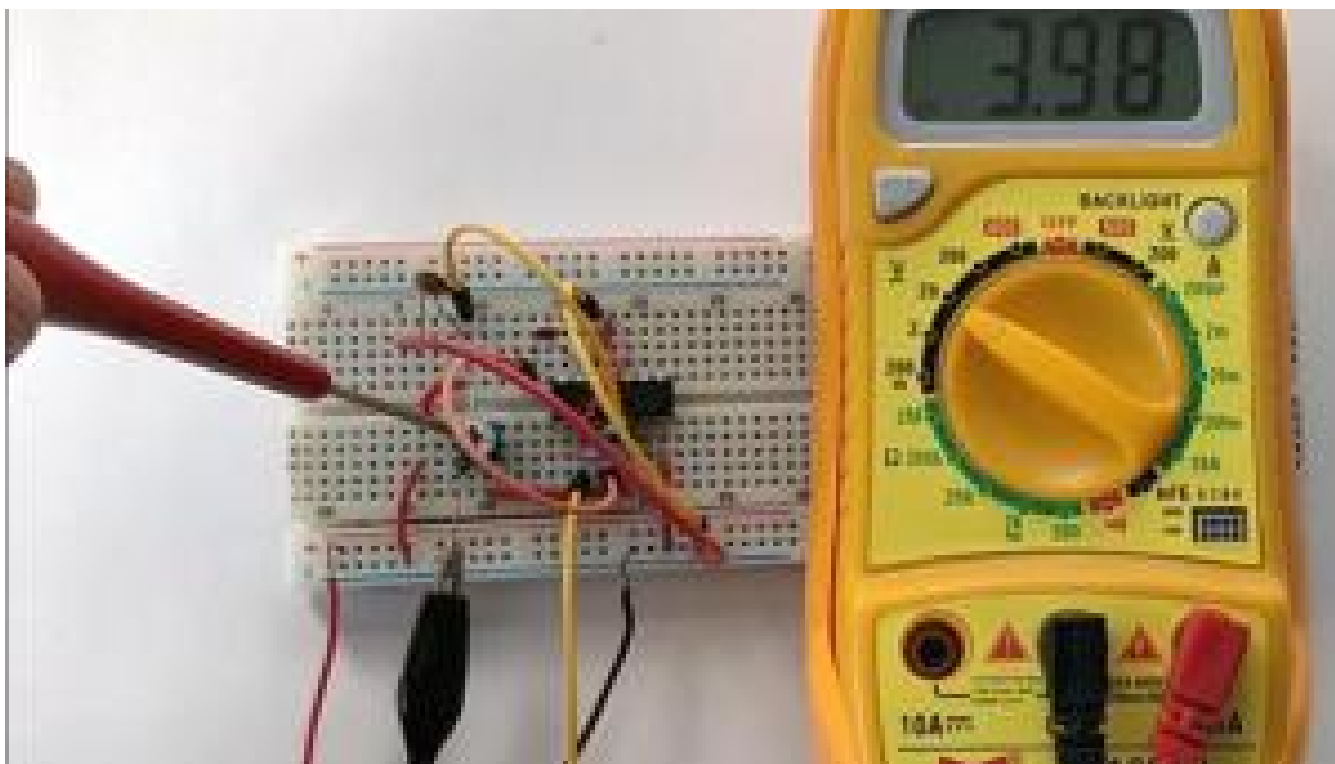
Preview

RELATED CONTENT

[Line Follower Robot using PIC Microcontroller \(/microcontroller-projects/line-follower-robot-using-pic16f877a\)](/microcontroller-projects/line-follower-robot-using-pic16f877a)[Obstacle Avoiding Robot using PIC Microcontroller \(/microcontroller-projects/obstacle-avoiding-robot-using-pic16f877a\)](/microcontroller-projects/obstacle-avoiding-robot-using-pic16f877a)[Interfacing Ultrasonic Sensor HC-SR04 with PIC Microcontroller \(/microcontroller-projects/interfacing-ultrasonic-sensor-hc-sr04-with-pic16f877a\)](/microcontroller-projects/interfacing-ultrasonic-sensor-hc-sr04-with-pic16f877a)[GSM module Interfacing with PIC Microcontroller - Make and Receive Calls \(/microcontroller-projects/gsm-interfacing-with-pic16f877a\)](/microcontroller-projects/gsm-interfacing-with-pic16f877a)[Цифровой амперметр с использованием микроконтроллера PIC и ACS712 \(/microcontroller-projects/digital-ammeter-circuit-using-pic16f877a-ac127\)](/microcontroller-projects/digital-ammeter-circuit-using-pic16f877a-ac127)<https://www.wellpcb.com>

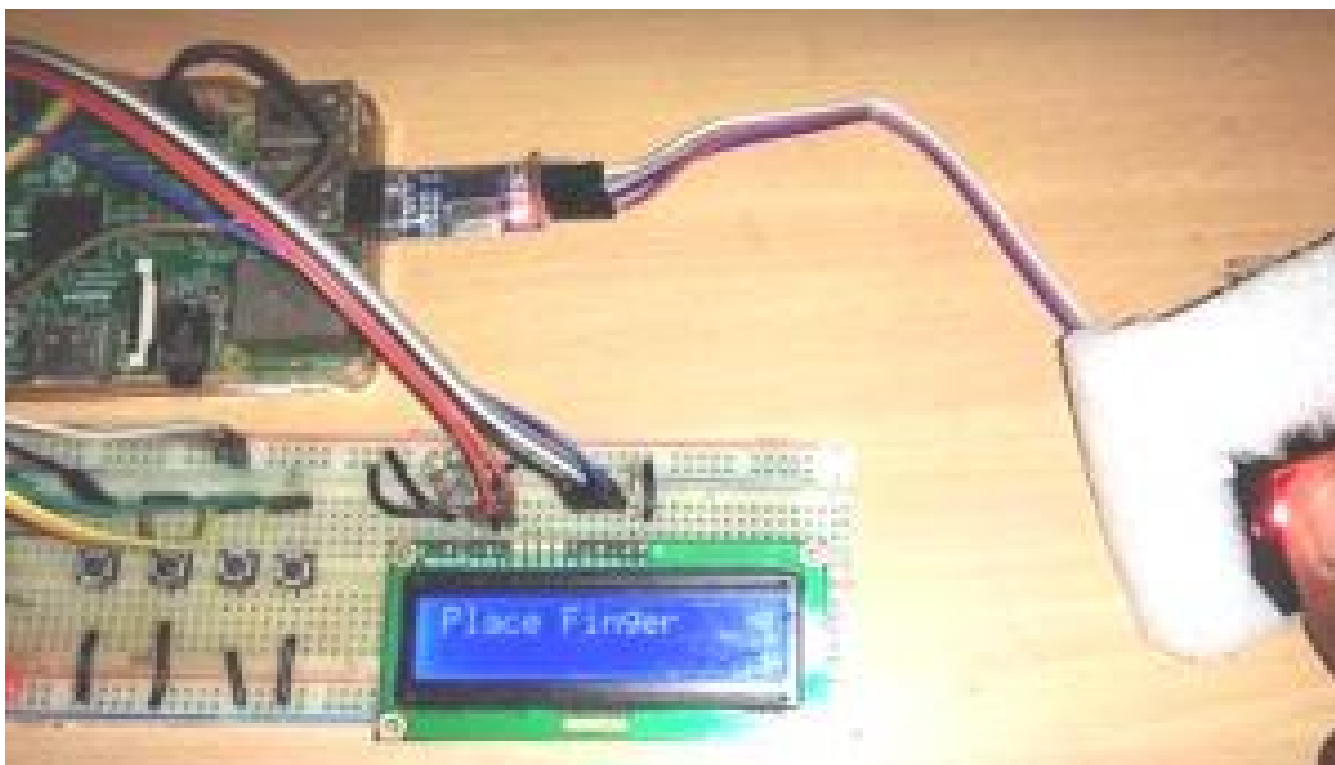
ПОСЛЕДНИЕ ПОСТЫ

[\(/microcontroller-projects/arduino-calculator-using-4x4-keypad\)](/microcontroller-projects/arduino-calculator-using-4x4-keypad)Калькулятор **Arduino** с использованием клавиатуры **4x4** (/microcontroller-projects/arduino-calculator-using-4x4-keypad)



(/electronic-circuits/lm723-voltage-regulator-circuit-diagram)

Цепь регулятора напряжения LM723 (/electronic-circuits/lm723-voltage-regulator-circuit-diagram)



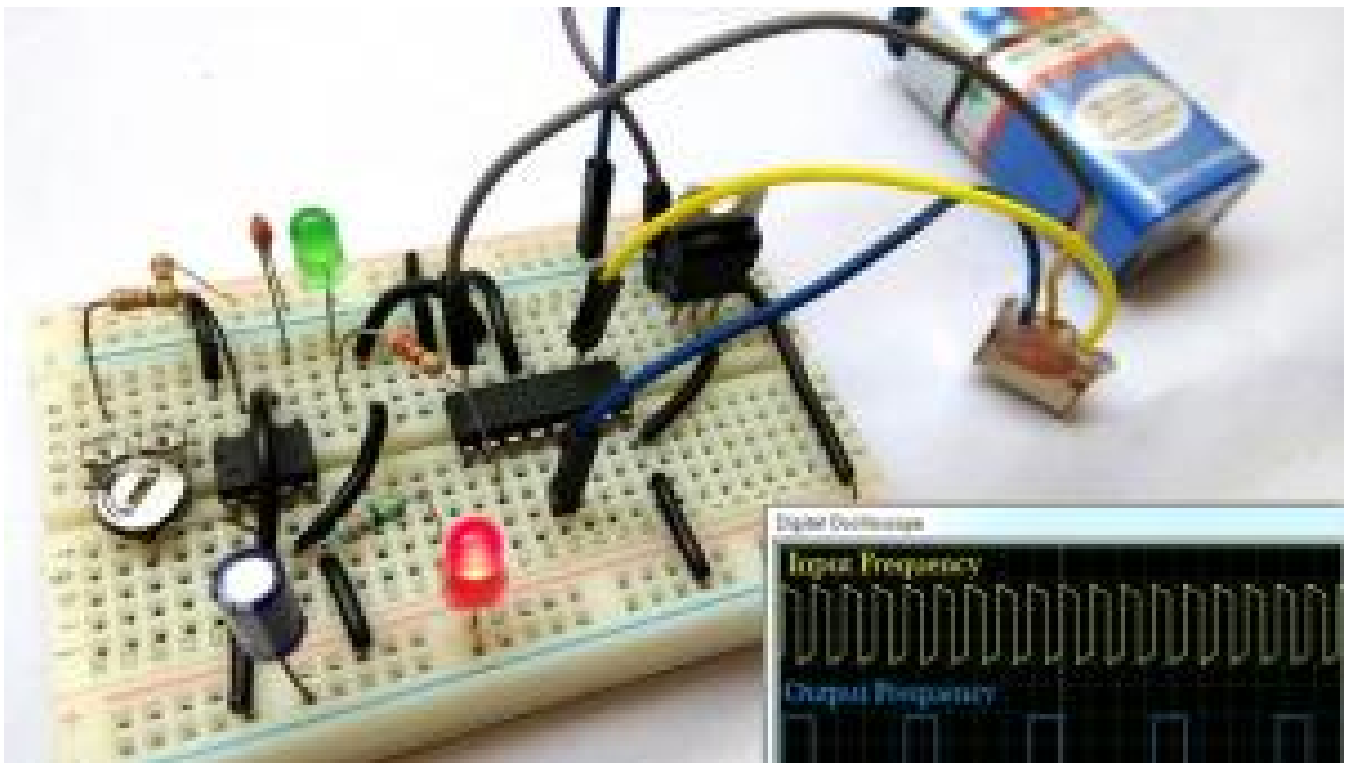
(/microcontroller-projects/raspberry-pi-fingerprint-sensor-interfacing)

Датчик отпечатка пальца в сочетании с малиной Pi (/microcontroller-projects/raspberry-pi-fingerprint-sensor-interfacing)



(/news/samsung-electronics-starts-producing-industrys-first-16-gigabit-GDDR6-for-advanced-graphics-systems)

Samsung Electronics начинает производство первой 16-гигабитной GDDR6 в отрасли для усовершенствованных графических систем
(/news/samsung-electronics-starts-producing-industrys-first-16-gigabit-GDDR6-for-advanced-graphics-systems)



(/electronic-circuits/frequency-divider-circuit-diagram)

Цепь делителя частоты с использованием таймера 555 и CD4017 (/electronic-circuits/frequency-divider-circuit-diagram)