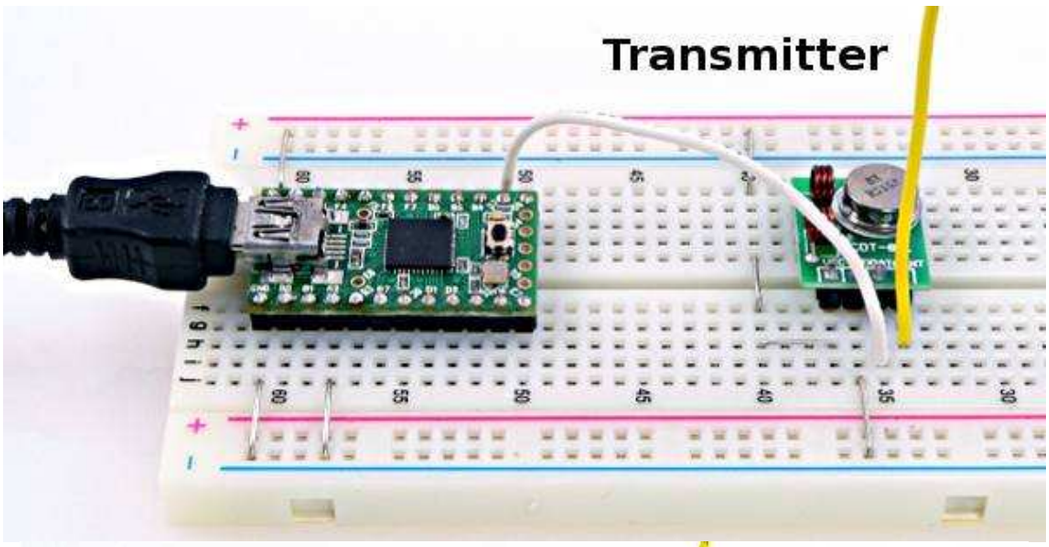
VirtualWire Library

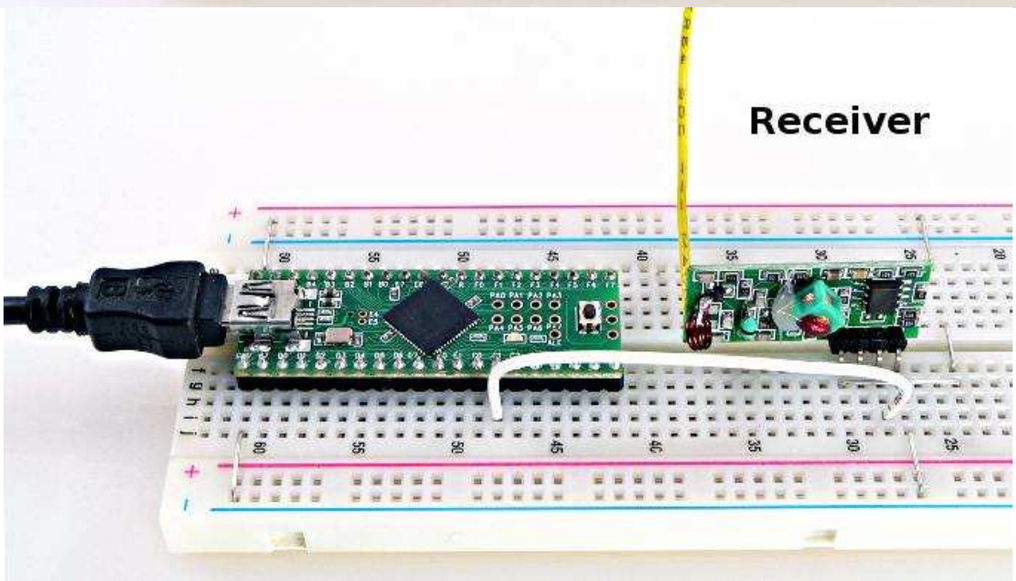
VirtualWire, de Mike McCauley, le ayuda a utilizar módulos de radio inalámbricos de muy bajo costo.

Los módulos RF de muy bajo costo requieren datos con formato especial, con patrones de sincronización, equilibrio de 0 y 1 bits y comprobación de errores. VirtualWire proporciona todas estas características, permitiendo el mejor rendimiento de circuitos de radio muy baratos.

Hardware Requirements

VirtualWire requiere módulos RF. En estas fotos, se utilizó un kit de enlace de RF de 315 MHz.



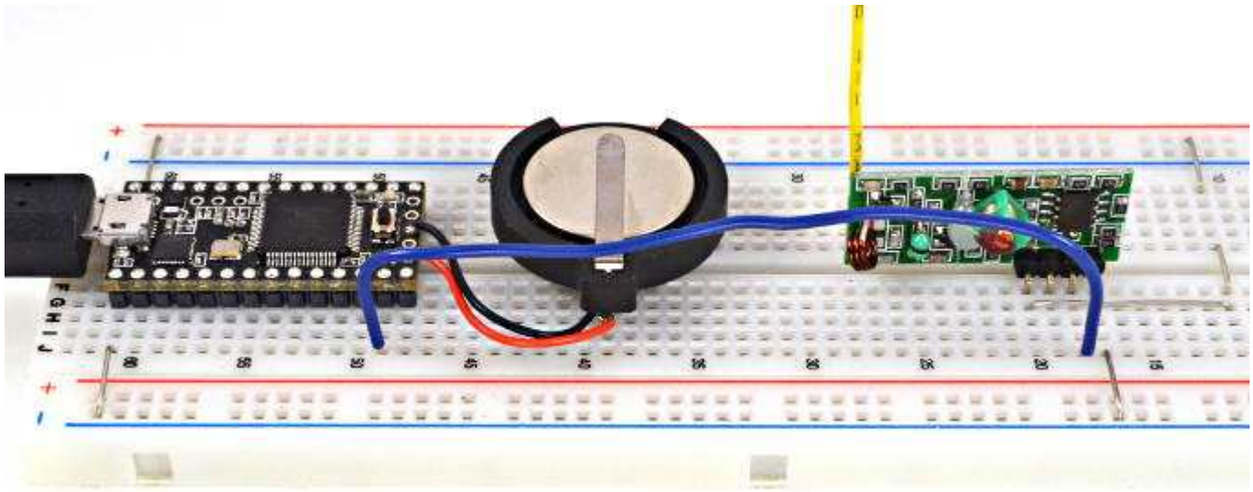


Estos módulos de bajo costo son muy simples. La señal de salida del receptor indica si se escucha la energía de un transmisor. Puede ser conectado a cualquier perno, pero no puede conducir los pernos con un LED atado.

Los transmisores suelen tener sólo un pin de datos, que activan la salida de RF cuando está alta. Algunos módulos Los módulos de 2 vías, con un transmisor y un receptor, también tienen un pin de habilitación del transmisor.

VirtualWire utiliza Timer1, lo que significa que algunos pins PWM que requieren Timer1 no funcionarán. Otras bibliotecas que utilicen Timer1 serán incompatibles con VirtualWire.

Board	Receive Pin	Transmit Pin	Transmit Enable Pin	PWM Pins Affected
Teensy 3.0	Any, except 13	Any	Any	-
Teensy 2.0	Any, except 11	Any	Any	4, 14, 15
Teensy 1.0	Any, except 6	Any	Any	15, 17, 18
Teensy++ 2.0	Any, except 6	Any	Any	25, 26, 27
Teensy++ 1.0	Any, except 6	Any	Any	25, 26, 27



Basic Usage

VirtualWire funciona de forma algo diferente a la mayoría de las bibliotecas de Arduino. Muchas funciones individuales se utilizan, y sus nombres son algo diferentes. Afortunadamente, cada uno es simple.

Configuration Functions

vw_set_tx_pin(transmit_pin)
Configure el pin de transmisión. El valor predeterminado es pin 12. Blah
vw_set_rx_pin(receive_pin)
Configure el pin de recepción, por defecto es el pin 11.
vw_set_ptt_pin(transmit_en_pin)
Configure el pin de habilitación de transmisión, o "presione para hablar". El valor predeterminado es pin 10..
vw_set_ptt_inverted(true)
Configure la polaridad "empujar para hablar". ("push to talk" polarity).
vw_setup(2000)
Begin using all settings and initialize the library. This is similar to the "begin" function of other libraries. All pins should be configured before calling this function.

*Comience a usar todos los ajustes e inicialice la biblioteca. Esto es similar a la función "empezar" de otras bibliotecas. Todos los pines deben configurarse antes de llamar a esta funció*n.

Transmission Functions (Funciones de transmisión)

vw_send(message, length)
Transmit a message. "message" is an array of the bytes to send, and "length" is the number of bytes stored in the array. This function returns immediately and the message is sent slowly by an interrupt-based background process.

Transmitir un mensaje. "mensaje" es una matriz de los bytes para enviar, y "longitud" es el número de bytes almacenados en la matriz.Esta función devuelve inmediatamente y el mensaje es enviado lentamente por un proceso de fondo basado en interrupción.

vw_tx_active()
Returns true if the message is being sent, or false if the transmitter is not active. You can use this after sending a message

to test when it has finished being transmitted.

Devuelve true si el mensaje se está enviando o false si el transmisor no está activo. Puede usar esto después de enviar un mensaje para probar cuando se ha terminado de transmitir.

vw_wait_tx()
Wait for a message to be fully transmitted. Often the simplest approach is to call this after vw_send.

Espere a que se transmita completamente un mensaje. A menudo, el enfoque más sencillo es llamar a esto después de vw_send.

Reception Functions (Funciones de recepción)

vw_rx_start()
Activate the receiver process. You must call this function before any reception can occur. An interrupt-based background process is started which monitors the reception of data.

Activar el proceso del receptor. Debe llamar a esta función antes de que pueda producirse una recepción. Se inicia un proceso de fondo basado en interrupciones que supervisa la recepción de datos.

vw_have_message()
Returns true if message has been received. This is similar to the "available" function of most other libraries.

Devuelve true si se ha recibido el mensaje. Esto es similar a la función "disponible" de la mayoría de las otras bibliotecas.

vw_wait_rx()
Wait for a message to be received. This will only return when a message has been received, otherwise it will wait forever.

Espere a que se reciba un mensaje. Esto sólo regresará cuando se reciba un mensaje, de lo contrario se esperará para siempre.

vw_wait_rx_max(timeout_ms)
Wait for a message, but give up after "timeout_ms". Returns true if a message was received, or false if the timeout period elapsed.

Espere un mensaje, pero abandone después de "timeout_ms". Devuelve true si se recibió un mensaje o false si transcurría el tiempo de espera.

vw_get_message(buf, &buflen))
Read the last received message. This should be called only when a message is known to be received with any of the 3 functions above. "buf" is an array where the message is copied. "buflen" should have the array's maximum size upon input, and upon return the number of bytes actually copied is retured. The function itself returns true if the message was verified correct, or false if a message was received but appears to have been corrupted.

Lea el último mensaje recibido. Esto debe llamarse sólo cuando se sabe que un mensaje se recibe con cualquiera de las 3 funciones anteriores. "buf" es una matriz donde se copia el mensaje. "buflen" debe tener el tamaño máximo de la matriz al ingresar, y a su regreso se reescribe el número de bytes realmente copiados. La función en sí devuelve true si el mensaje se verificó correctamente o false si se recibió un mensaje pero parece haberse corrompido.

vw_rx_stop()
Disable the receiver process.

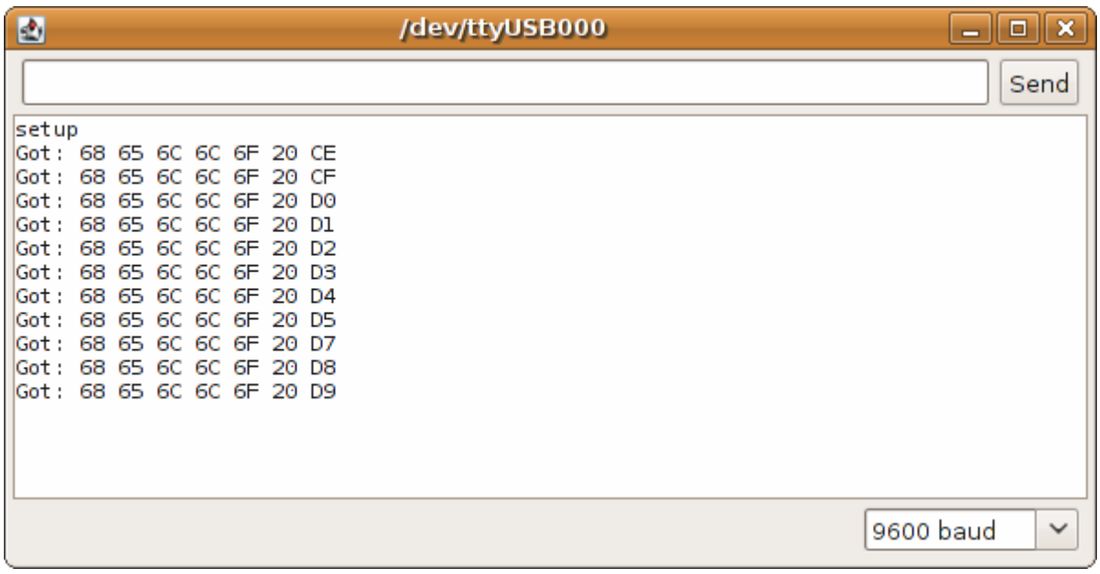
Desactive el proceso del receptor.

The official VirtualWire documentation (PDF), explains these functions and other usage considerations.

Example Program – Transmit

```
#include <VirtualWire.h>

const int led_pin = 11;
const int transmit_pin = 12;
const int receive_pin = 2;
const int transmit_en_pin = 3;

void setup()
{
  // Initialise the IO and ISR
  vw_set_tx_pin(transmit_pin);
  vw_set_rx_pin(receive_pin);
  vw_set_ptt_pin(transmit_en_pin);
  vw_set_ptt_inverted(true); // Required for DR3100
  vw_setup(2000);       // Bits per sec
}

byte count = 1;

void loop()
{
  char msg[7] = {'h','e','l','l','o',' ','#'};

  msg[6] = count;
  digitalWrite(led_pin, HIGH); // Flash a light to show transmitting
  vw_send((uint8_t *)msg, 7);
  vw_wait_tx(); // Wait until the whole message is gone
  digitalWrite(led_pin, LOW);
  delay(1000);
  count = count + 1;
}
```

Example Program - Receive

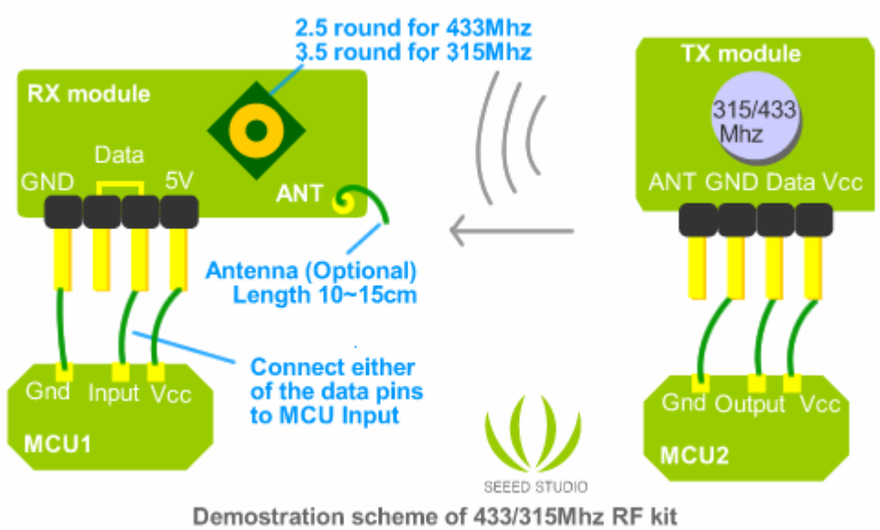



```
#include <VirtualWire.h>

const int led_pin = 6;
const int transmit_pin = 12;
const int receive_pin = 11;
const int transmit_en_pin = 3;

void setup()
{
    delay(1000);
    Serial.begin(9600);        // Debugging only
    Serial.println("setup");

    // Initialise the IO and ISR
```

```
    vw_set_tx_pin(transmit_pin);
    vw_set_rx_pin(receive_pin);
    vw_set_ptt_pin(transmit_en_pin);
    vw_set_ptt_inverted(true); // Required for DR3100
    vw_setup(2000);     // Bits per sec

    vw_rx_start();       // Start the receiver PLL running
}

void loop()
{
    uint8_t buf[VW_MAX_MESSAGE_LEN];
    uint8_t buflen = VW_MAX_MESSAGE_LEN;

    if (vw_get_message(buf, &buflen)) // Non-blocking
    {
        int i;

        digitalWrite(led_pin, HIGH); // Flash a light to show received good message
        // Message with a good checksum received, print it.
        Serial.print("Got: ");

        for (i = 0; i < buflen; i++)
        {
            Serial.print(buf[i], HEX);
            Serial.print(' ');
        }
        Serial.println();
        digitalWrite(led_pin, LOW);
    }
}
```

Expectativas realistas de rendimiento

Muchos de los módulos de RF más baratos se venden con demandas muy poco realistas de velocidad de datos y máxima distancia de comunicación, ya veces con muy poca documentación (o incluso incorrecta). VirtualWire ayudará a estos módulos a funcionar tan bien como puedan, pero el viejo dicho se aplica: "obtienes lo que pagas".
Por ejemplo, los módulos de 315 MHz mostrados anteriormente se documentaron con sólo esta imagen.



The only "documentation" for $4.90 module pair shown above

Estos módulos trabajaron muy confiablemente cuando se sentaban sólo cerca uno del otro

sobre una mesa. Cuando se separaron por unos 20 pies con muebles de oficina ordinarios, y un cable de 13 cm unidos a cada uno (en el medio de los 10 a 15 cm sugerido), fueron capaces de comunicarse, pero aproximadamente el 20% de los mensajes fueron corrompidos.

Tal vez el uso de mejores antenas podría ayudar, pero cada placa tiene una bobina de carga que parece estar diseñada para antenas relativamente cortas, y no hay otra documentación parece existir en relación con las mejores antenas.

Estos módulos pueden funcionar bien para aplicaciones no críticas de bajo rendimiento. Para aplicaciones más exigentes, se deben considerar módulos RF más sofisticados (y más caros).

RF Modules

- 315Mhz RF kit <-- has schematics and antenna advice!
- 315Mhz RF link kit
- 433Mhz RF link kit
- RF Link Transmitter - 315MHz
- RF Link 4800bps Receiver - 315MHz

FUENTE: https://www.pjrc.com/teensy/td_libs_VirtualWire.html

Ampliando el tema

Re: Virtual Wire y PWM
 # 2

15 de octubre de 2010, 13:09

Parece que la biblioteca virtual de alambre cambia de alguna manera los ajustes TIMER1 y básicamente la señal PWM para el pin 9 y 10 se efectúa. Pero no he probado a cambiar pines todavía como mi aplicación es construir y tengo que hacer algo de trabajo para llegar a los pines. Así que si alguien puede confirmar mi pensamiento sería genial.
Mi única esperanza es que otros pines de PWM no se efectúen.

Re: Virtual Wire y PWM
 # 3

15 de octubre de 2010, 16:28

Bueno, después de probar por mi cuenta ahora puedo confirmar que PWM en pin 5 y 6 funciona bien cuando se ejecuta la biblioteca virtual de alambre.
Pero los pines 9 y 10 no lo harán en absoluto.