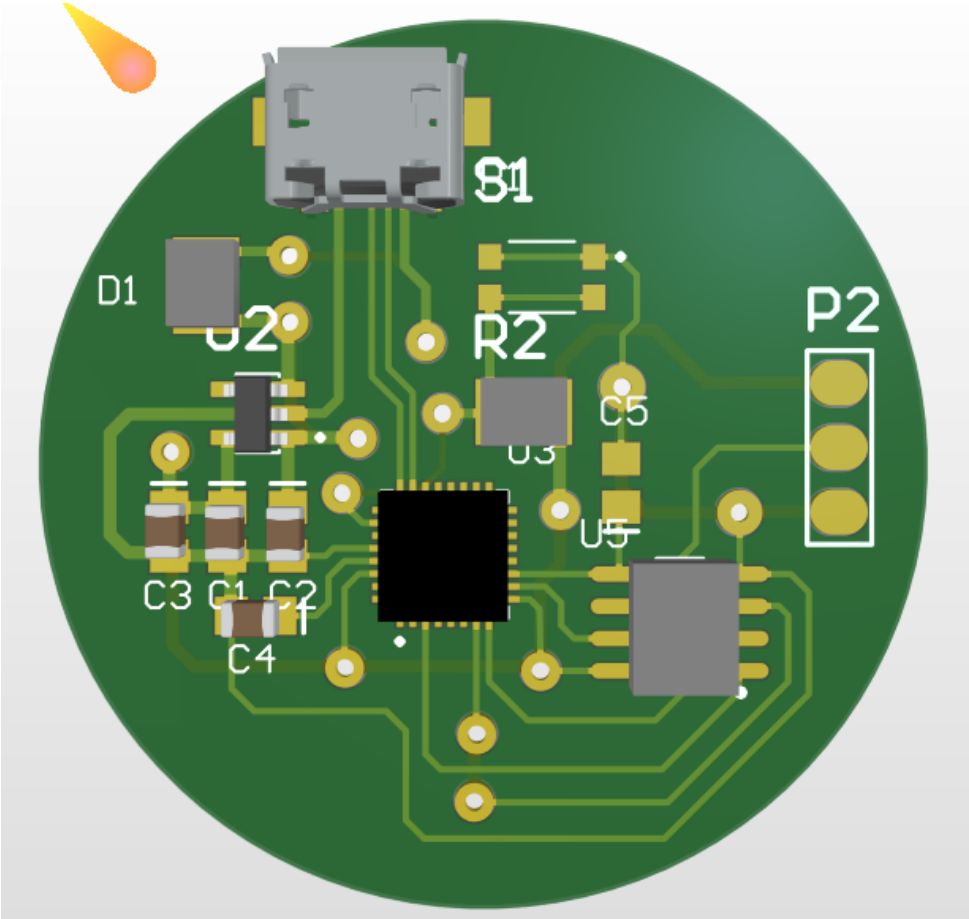
Estimulador VNS – Guía rápida de inicio

v. 2020-1

Gustavo Adolfo Ordoñez

November 23, 2020



Introducción

El estimulador VNS es capaz de generar una señal de voltaje con frecuencia, ancho de pulso, amplitud, tiempo de encendido y apagado configurables mediante software. Además se proporciona la configuración de un módulo convertidor de voltaje a corriente para entregar la señal de terapia al electrodo que posteriormente se adhiere al nervio vago. En el presente documento se resume la instalación de las librerías y configuraciones necesarias para programar el microcontrolador SAMD21E18A que funciona como generador de la señal de estimulación. También se incluye la forma de modificar los parámetros de la señal en el respectivo código y como cambiar los valores de estimulación incluidos previamente en el código.

Contents

1 SAMD21E18A Overview

El SAMD21E18A es un microcontrolador (MCU) de arquitectura ARM y 32-bits. La descripción física y el diagrama de pines de salida es mostrado en la Figura 1. Tiene un total de 32 pines de entrada-salida, específicamente para el empaquetado TQFP/QFN.

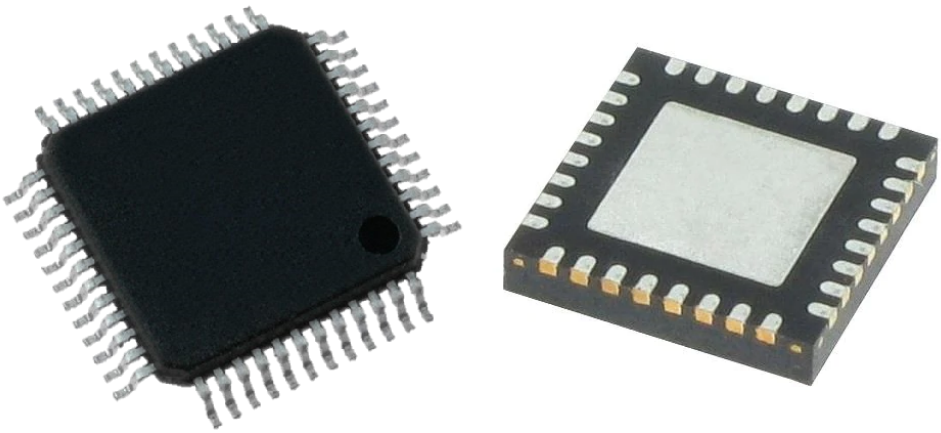


Figure 1: Diagrama de SAMD21E18A para encapsulado TQFP/QFN.

La funcionalidad de cada uno de los pines se describe en la Figura 2. Se puede observar que el microcontrolador cuenta con varios pines configurables como entrada/salida digital, seis pines analógicos para lectura de voltajes mediante el módulo ADC, cinco pines de alimentación distribuidos en lados opuestos del empaquetado y dos pines configurables como entrada/salida para colocar un oscilador externo y aumentar la velocidad/precisión del reloj principal.

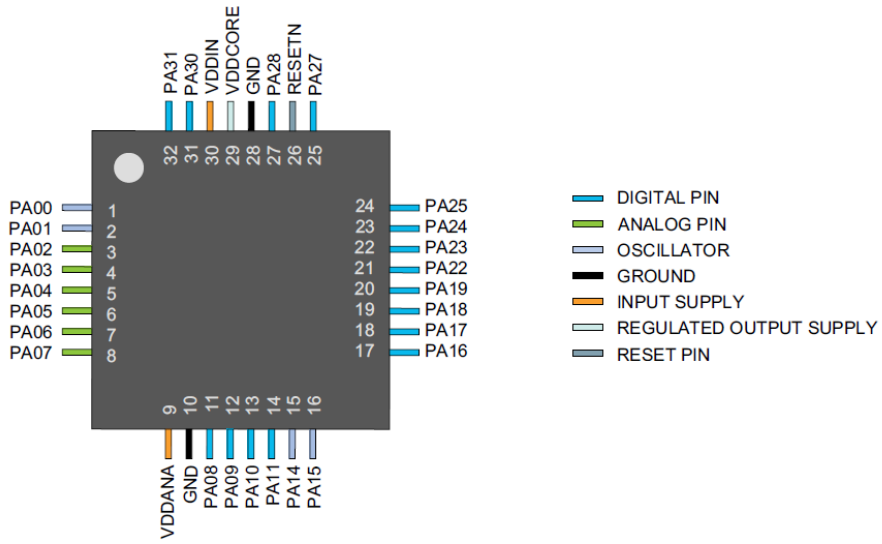


Figure 2: Clasificación pines del SAMD21E18A.

Al ser un microcontrolador de 32-bit ofrece una mayor cantidad de memoria RAM disponible, registros de configuración y módulos con mayor resolución. Por ejemplo, los módulos temporizadores de precisión con contadores de 16 y 24-bit. Además que este MCU es capaz de operar a una frecuencia nominal de operación de 48 MHz y hasta 96 MHz con overclock.

2 Programando el SAMD21E18A

El SAMD21E18A puede ser utilizado ya sea implementado mediante una plataforma como el Trinket M0 de Adafruit o el Arduino Zero. Una alternativa es adquirir el MCU directamente de un distribuidor de dispositivos electrónicos como Digikey o Mouser para posteriormente programar el IC en el ambiente de ATMEL Studio o cargarle un *bootloader* y poder trabajar con el IDE de Arduino. Por fines de simplicidad y tiempo limitado, se procedió a trabajar con la plataforma Trinket M0 y Arduino.

2.1 Trinket M0 con Arduino

El Trinket M0, como se mencionó anteriormente, es una plataforma desarrollada por Adafruit con una placa de dimensiones reducidas que implementa el SAMD21, un regulador de voltaje de 3.3V, un LED RGB y un puerto micro USB para alimentación y programación entre otros, como se muestra en la Figura 3

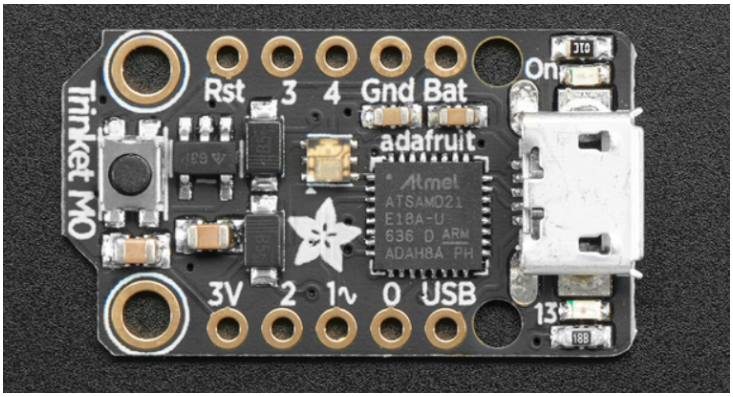


Figure 3: Plataforma Trinekt M0.

Para emplear el IDE de Arduino lo primero que deberá hacerse es descargar la última versión de Arduino. Una vez descargado e instalado, se debe iniciar el IDE y seleccionar la opción File › Preferences. Aparecerá la ventaja mostrada en la Figura 4 y debe agregarse la dirección URL encerrada en el cuadro rojo. Una vez cargado el URL, las librerías de Adafruit y otras actualizaciones se habilitaran para la instalación. Un procedimiento aconsejable es revisar que las librerías de las placas de Adafruit necesarias para el Trinket M0 se encuentren instaladas en Tools › Board Manager.

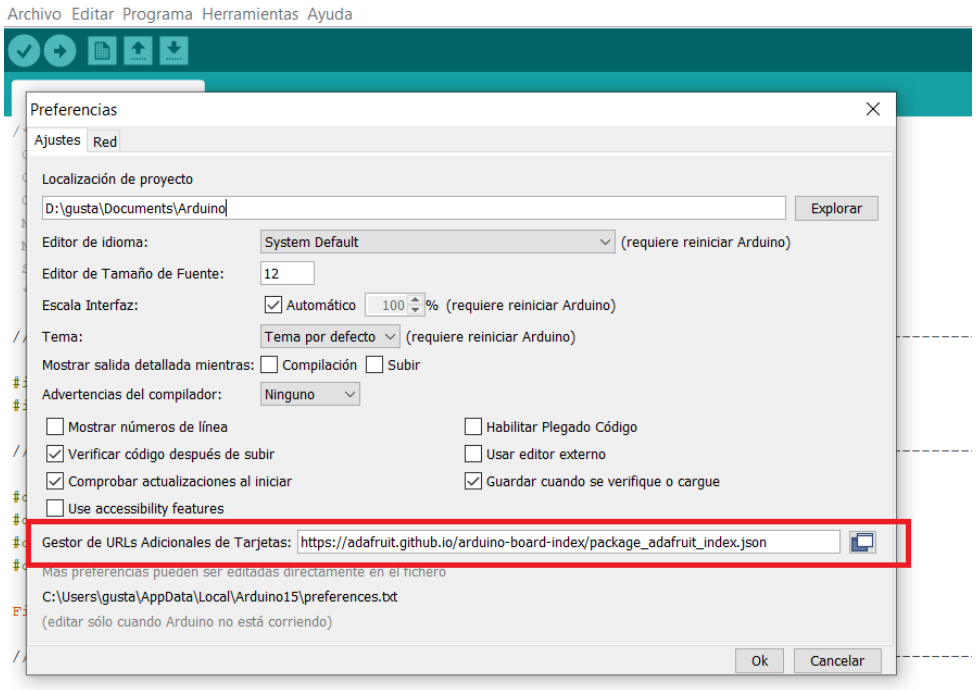


Figure 4: Configuración de Preferencias en Arduino.

3 Conexiones del Módulo Generador

El modulo generador se compone del microcontrolador SAMD21E18A y el control de amplitud mediante un potenciómetro digital X9C104 como se muestra en la Figura 5. El programa principal para el módulo estimulador se encuentra en el repositorio de Github bajo la dirección Estimulador > Códigos > PWM Module.ino.

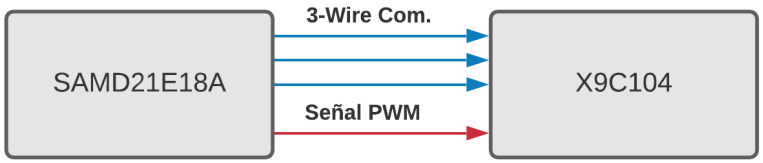


Figure 5: Configuración de Preferencias en Arduino.

Los pines para la comunicación 3-Wire con el potenciómetro X9C104 se detallan al inicio del código como se observa en la Figura 6. Donde **UD** es el pin para seleccionar si la resistencia del potenciómetro incrementar o decrementar, **INC** para incrementar/decrementar según UD y **CS** para seleccionar modificar el valor del potenciómetro digital.

```
// ..................................... PWM and Timer Values Selection ........................


volatile unsigned int period;          // = PWM_freq[0][3];
volatile float pulse_width;            // = period*0.25; //PWM_pw[2][0];
volatile int ON_Time;                  // = ON_OFF_TIME[13];
volatile int OFF_Time;                 // = ON_OFF_TIME[13];


// ............................. Variables for PWM, Timer and Digital Potentiometer .................


// Select Gen Clock to setting the waveform generator clock or sample rate
const unsigned char gClock = 4;
byte PWM_CLK;

const byte UD = 1;
const byte INC = 2;
const byte CS = 3;

bool flag = 0;
byte mode = 0;
bool Ramp_mode = false;
bool go_to_sleep = false;
bool save_energy_mode;
```

Figure 6: Declaración de pines para comunicación 3-Wire con X9C104.

4 Conexión Convertidor Voltaje-Corriente

El convertidor de voltaje a corriente utiliza el circuito integrado (IC) XTR111. Se utilizó la aplicación estándar que se propone en la hoja de datos del convertidor como se observa en la Figura 7. El divisor de voltaje resaltado en el cuadro azul de la figura anterior se omitió ya que no fue necesario implementarlo para estas pruebas. Es recomendable revisar la corriente que puede suministrar y alimentar el resto del circuito en caso la corriente sea suficiente.

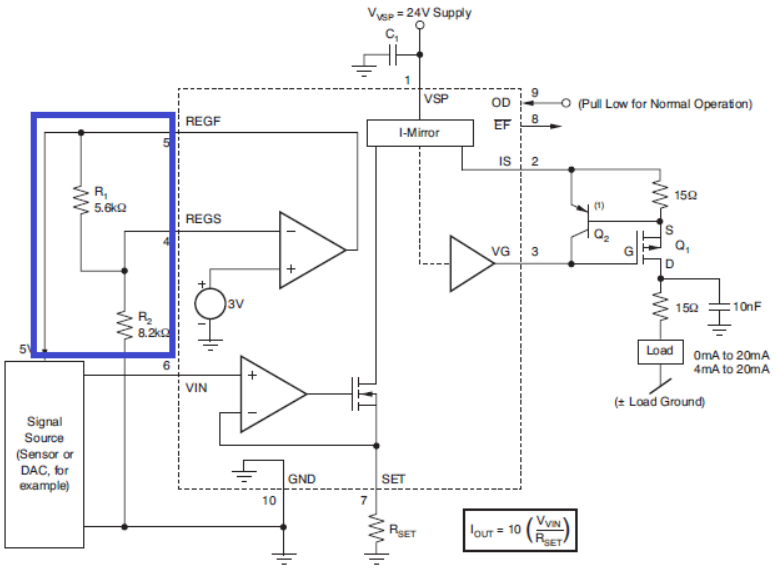


Figure 7: Conexión del convertidor de voltaje a corriente.

La resistencia de ajuste R_{set} se fijó en un valor de 10 KΩ para realizar una conversión del voltaje de entrada con el mismo valor pero diferente orden de magnitud, específicamente al orden de los miliamperios. Como recomienda el manual del XTR111 que se puede encontrar en el repositorio Estimulador > Documentación, se empleó un transistor Mosfet canal P NTF2955 y un transistor PNP BC327.

5 Configuración de parámetros de estimulación

Los parámetros de la señal de estimulación son los siguientes:

- Frecuencia
- Ancho de pulso
- Tiempo encendido (estimulación)
- Tiempo apagado (reposo)
- Amplitud

Los parámetros se encuentran declarados y pueden ser modificados en la sección superior del código de “PWM Module.ino" como se muestra en la Figura 8.

```
// ............................. Variables for PWM, Timer and Digital Potentiometer ..


// Select Gen Clock to setting the waveform generator clock or sample rate
const unsigned char gClock = 4;
byte PWM_CLK;

const byte UD = 1;
const byte INC = 2;
const byte CS = 3;

bool flag = 0;
byte mode = 0;
bool Ramp_mode = false;
bool go_to_sleep = false;
bool save_energy_mode;

int pulsos = 0;
int amplitude_max = 98;
int amplitude_min = 1;

int amplitude;
int frecuencia = 5;
int Stimulation_time = 1;
int Repose_time = 1;

float ramp_interval;
int timer_interval;

// ..............................................................................
```

Figure 8: Configuración de los parámetros de estimulación.

Los valores de los parámetros anteriores están limitados a los valores del Cuadro 1 y son seleccionados a partir de las variables de la Figura 8 mediante estructuras “Switch Case" en la sección de *Setup()*.

Parámetros	Valores	Tolerancia
Corriente de salida (mA)	0-3.5 *steps of 0.25	$\pm 0.25 \leq 1, \pm 10\% > 1$
Frecuencia (Hz)	1,2,5,10,15,20,25,30	$\pm 6\%$
Ancho de pulso (μs)	130,250,500,750,100	$\pm 10\%$
Tiempo encendido (s)	7,14,21,30,60	$\pm 15\%$
Tiempo apagado (m)	0.2,0.3,0.5,0.8,1.1,1.8,3,5,180	$+4,4s$ / $-8.4s$

Table 1: Parámetros de la señal de estimulación.

5.1 Valores preestablecidos

Los valores de frecuencia, ancho de pulso y tiempo de encendido/apagado especificados en el Cuadro 1, se calcularon empleando los temporizadores denominados TCC0, TCC1 y TC5. Los módulos anteriores rigen la frecuencia del PWM y la duración del tiempo de estimulación/reposo mediante las Ecuaciones 1 y 2.

$$ValorCCx = (\frac{GCLK_{frec}}{TCCx_{presc} * PWM_{frec}}) - 1 \tag{1}$$

$$TemporizadorCCx = (\frac{tiempo(s) * GCLK_{frec}}{TCCx_{presc}}) - 1 \tag{2}$$

Los valores obtenidos con las ecuaciones anteriores se organizaron en arreglos de datos en la parte superior del código. Los arreglos contienen los valores para los parámetros del Cuadro 1 variando la frecuencia de reloj para valores de 48 MHz, 8MHz y 32KHz, como se puede observar en la Figura 9.

```
// ..................................... PWM and Timer Values Data Base.....................................

// Frequency configuration of the Stimulation Signal (PWM) for 30, 25, 20, 15, 10, 5, 2 and 1Hz
const unsigned int PWM_freq[3][8] = {{1562, 1874, 2343, 3124, 4687, 9374, 23437, 46874},    // 48MHz
                                     {1041, 1249, 1562, 2082, 3124, 6249, 15624, 31249},    // 8MHz
                                     {1066, 1279, 1599, 2132, 3199, 6399, 15999, 31999}};   // 32KHz

// CCx values for 1ms, 750us, 500us, 250us and 130us respectively, for pulse
// width or hight state in the square signal (PWM) during the Stimulation Time
const unsigned char PWM_pw[3][5] = {{47, 35, 23, 12, 6},    // 48MHz
                                    {31, 23, 16, 8, 4},     // 8MHz
                                    {32, 24, 16, 8, 4}};    // 32KHz

// Array for CCx values for 600, 300, 180, 108, 66, 60 48, 30, 21, 18, 14, 7, 2, 1s respectively, for
// On/Off time for Stimulation Signal. Clock at 32KHz
const unsigned int ON_OFF_TIME[14] = {18749, 9374, 5624, 3374, 2062, 1874, 1499, 937, 655, 562, 437, 218, 62, 30};

// ..................................... PWM and Timer Values Selection .....................................

volatile unsigned int period;          // = PWM_freq[0][3];
volatile float pulse_width;            // = period*0.25; //PWM_pw[2][0];
volatile int ON_Time;                  // = ON_OFF_TIME[13];
volatile int OFF_Time;                 // = ON_OFF_TIME[13];
```

Figure 9: Valores precargados para la terapia VNS.