

## generador PWM

estudiante : Jhonatan Alexander Gomez Gamboa

profesor : Luis René Vela García

### **1. Introducción experimental.**

---

La creación e implementación del siguiente diseño digital tiene la función de generar una señal PWM (Pulse Width Modulation) la cual se utilizará para variar la velocidad de un motor de 5 Volts DC. Dicho control de la velocidad será variable mediante 3 pulsadores, uno para sumar en una unidad la referencia, otro para restar una unidad la referencia y el ultimo para reiniciar la funcionalidad, dicha referencia podrá visualizarse en tiempo real utilizando 3 display 7 segmentos propios de la tarjeta. Nos apoyaremos para el control del motor del controlador L298n, así como un módulo convertidor de nivel lógico BSS 138 encargado de convertir el nivel lógico de trabajo de la tarjeta de 3.3v a 5v. Cabe mencionar que todos los módulos a excepción del BCDCConvert son parametrizables.

### **2. Ambiente experimental.**

---

Quartus (Quartus Prime 17.7 Lite Edition)

FPGA Cyclone IV, ALINX family EP4CE6F17C8N.

### **3. Programa.**

---

El módulo top o principal el cual se carga en la tarjeta FPGA consta de 12 módulos, cuyas interconexiones se representan en la (figura 1). La entrada rst\_top se encarga de inicializar los contadores o fijar los registros a cero dependiendo el funcionamiento del módulo conectado, clk\_top es una entrada conectada a la señal de reloj de 50 M Hz propia de la tarjeta, las entradas sum\_top permite sumar en una unidad la referencia, rest\_top se utiliza para restar en una unidad la referencia la cual es utilizada por el módulo comparador para variar el ciclo útil de la señal. La salida seg\_data\_top es una salida de 8 bits la cual se conectaría a los segmentos del display para poder mostrar la información de la referencia, la salida de 6 bits seg\_sel\_top se utiliza para poder iterar sobre 3 valores cada uno representando la posición de un display y poder configurar que salida se mostrará en que display. La salida pwm\_top representara el tren de pulsos para poder variar la velocidad de motor. Por último, las salidas in1\_top e in2\_top se utilizar para poder configurar la dirección de giro del motor, deben tener valores contrarios. Es importante mencionar que la sección dentro del rectángulo morado es una alternativa a no contar con una tarjeta FPGA que tenga 8 switches, la tarjeta utilizada para la implementación solo cuenta con 4 pushbuttons, esta área puede fácilmente remplazarse por el módulo switchesEntrada (tabla 9).

El módulo prescaler (figura 2) nos permite poder reducir la frecuencia de operación a una menor para el tipo de evaluación o acción que lleva a cabo un módulo en específico, este modulo es parametrizable por lo cual al variar el tamaño del contador variara el tiempo en que una señal de habilitación es obtenida. Las entradas y salidas pueden consultarse en la tabla 1.

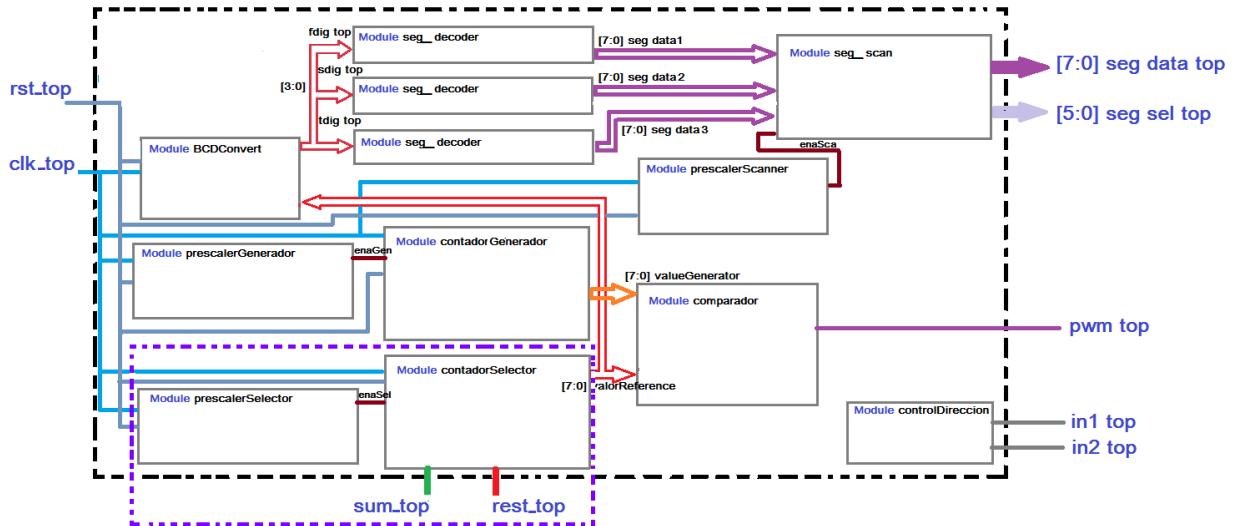


Figure 1: Diagrama de flujo para la creación del módulo **moduloTop** utilizado para poder hacer las conexiones entre las instancias de cada módulo, además de la asignación de parámetros para los módulos que sean parametrizables.

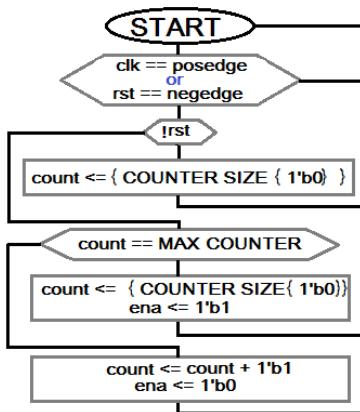


Figure 2: Diagrama de flujo para la creación del módulo **prescaler** parametrizado utilizado para la reducción de la frecuencia de la tarjeta FPGA.

Table 1: módulo prescaler

Señal	Dirección	Tamaño	Descripción
<i>clk</i>	in	1	señal de reloj de 50 Mhz de la tarjeta FPGA
<i>rst</i>	in	1	señal de reset para inicializar el contador
<i>ena</i>	out	1	señal para realizar acciones con una frecuencia menor

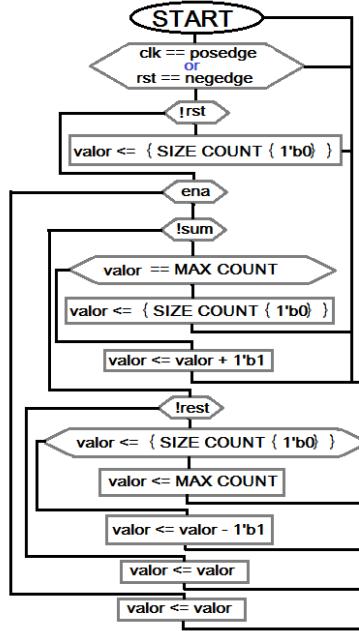


Figure 3: Diagrama de flujo para la creación del módulo **contadorSelector** parametrizado utilizado para la obtención del valor de referencia usado para reducir o aumentar el ciclo de trabajo útil de la señal PWM.

El módulo contadorSelector (figura 3) nos permite poder variar la referencia utilizada para generar la señal PWM, haciendo una de 3 operaciones, sumar una unidad a la referencia, restar una unidad a la referencia o restaurar los registros a cero, este modulo utiliza una señal de habilitación propia de un prescaler para poder tener una frecuencia de trabajo más adecuada con la tarea, las entradas y las salidas pueden consultarse en la tabla 2.

Table 2: módulo contadorSelector

Señal	Dirección	Tamaño	Descripción
<i>clk</i>	in	1	señal de reloj de 50 Mhz de la tarjeta FPGA
<i>rst</i>	in	1	señal de reset para inicializar el contador del selector
<i>ena</i>	in	1	señal para realizar acciones con una frecuencia menor
<i>sum</i>	in	1	señal que permite sumar en un bit la referencia
<i>rest</i>	in	1	señal que permite restar un bit a la referencia
<i>valor</i>	out	SIZE COUNT	señal parametrizada como referencia

El módulo ContadorGenerador (figura 4) nos permite poder contar en una unidad un registro a una cantidad máxima establecida como la resolución de la señal pwm apoyándose de un prescaler para poder variar la

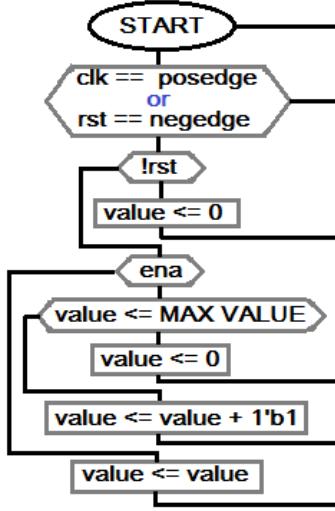


Figure 4: Diagrama de flujo para la creación del módulo **contadorGenerador** parametrizado utilizado para generar una sucesión aritmética cuya salida será conectada al modulo comparador para generar las señales PWM.

frecuencia de la señal, este módulo es parametrizable por lo cual la resolución es editable y afecta la frecuencia de la salida del módulo. Las entradas y las salidas del modulo pueden consultarse en la tabla 3.

Table 3: módulo contadorGenerador

Señal	Dirección	Tamaño	Descripción
clk	in	1	señal de reloj de 50 Mhz de la tarjeta FPGA
rst	in	1	señal de reset para inicializar el contador del selector
ena	in	1	señal para realizar acciones con una frecuencia menor
value	out	RESOLUTION BIT	salida del contador parametrizada

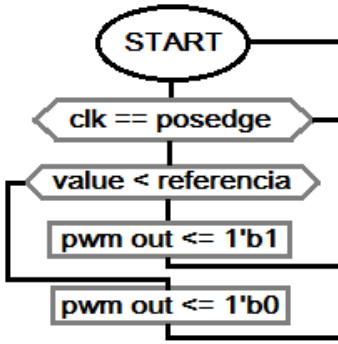


Figure 5: Diagrama de flujo para la creación del módulo **comparador** parametrizado utilizado para recibir un valor de referencia y una secuencia aritmética con el fin de generar la señal PWM.

El módulo comparador (figura 5) nos permite poder generar salidas de 1 si el valor recibido por el contadorGenerador se encuentra por debajo del valor de referencia recibido por el contadorSelector, y devolverá 0 cuando este valor se encuentre sobre la referencia. Puede consultar las entradas y salidas en la tabla 4.

Table 4: módulo comparador

Señal	Dirección	Tamaño	Descripción
<i>clk</i>	in	1	señal de reloj de 50 Mhz de la tarjeta FPGA
<i>value</i>	in	RESOLUTION BITS	entrada del contador
<i>referencia</i>	in	RESOLUTION BITS	entrada del selector como referencia
<i>pwm_out</i>	out	1	salida del generador del PWM

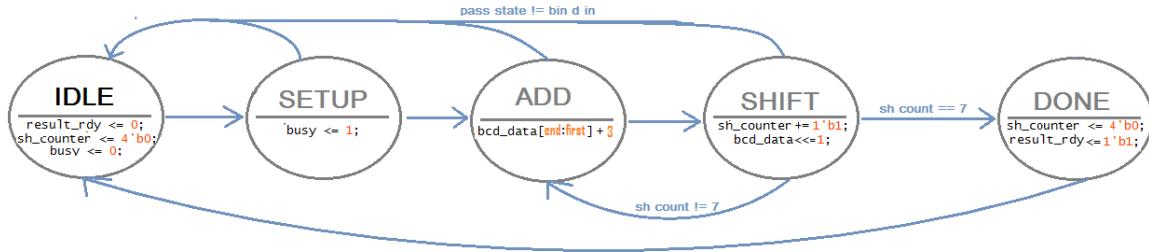


Figure 6: Diagrama del módulo **BCDCConvert** para una entrada de 8 bits utilizado para describir una maquina de estados finitos del algoritmo double dabble.

El módulo BCDCConvert (figura 6) es una máquina de estados finito que nos permite poder describir y ejecutar el algoritmo double dabble, comúnmente utilizado para poder convertir un numero binario de 8 bits, en una salida la cual cada 4 bits es un numero en binario representado las unidades, decenas y centenas. Para conocer todas las entradas y las salidas consultar la tabla 5.

Table 5: módulo BCDCConvert

Señal	Dirección	Tamaño	Descripción
<i>clk</i>	in	1	señal de reloj de 50 Mhz de la tarjeta FPGA
<i>bin_d_in</i>	in	8	entrada de 8 bits de la referencia para convertir a seg
<i>fdig</i>	out	4	salida del numero utilizado para el primer display
<i>sdig</i>	out	4	salida del numero utilizado para el segundo display
<i>tdig</i>	out	4	salida del numero utilizado para el tercer display
<i>rdy</i>	out	1	señal como bandera cuando se logre la conversión

El módulo seg\_decoder (figura 7) nos permite tomar una entrada de 4 bits que representa un numero entre 0-9, y lo codifica en una salida de 8 bits BCD que puede conectarse a un display de 7 segmentos, este módulo hará la conversión cuando reciba una señal de 1 en la entrada ena, que viene del modulo BCDCConvert cuando logra hacer la conversión, si no la ha realizado no hace la conversión en este módulo. Las entradas y salidas se plantean en la tabla 6.

Table 6: módulo seg\_decoder

Señal	Dirección	Tamaño	Descripción
<i>clk</i>	in	1	señal de reloj de 50 Mhz de la tarjeta FPGA
<i>ena</i>	in	1	señal de habilitación que entra del BCDCConvert

<b>Señal</b>	<b>Dirección</b>	<b>Tamaño</b>	<b>Descripción</b>
<i>bin_data</i>	in	4	entrada en binario de un número entre 0-9
<i>seg_data</i>	out	8	salida codificada a BCD para conectar al display

El módulo *seg\_scan* (figure 8) nos permite tomar las 3 entradas de 8 bits y los conecta con los display, al iterar cada señal sobre un display dependiendo un identificador que permite representar en que display se mostrara, en el digito de las unidades, decenas y centenas. Las entradas y salidas se plantean en la tabla 7.

Table 7: módulo *seg\_scan*

<b>Señal</b>	<b>Dirección</b>	<b>Tamaño</b>	<b>Descripción</b>
<i>clk</i>	in	1	señal de reloj de 50 Mhz de la tarjeta FPGA
<i>rst_n</i>	in	1	señal reseta para inicializar los valores de registros
<i>seg_data_0</i>	in	8	entrada del primer digito del BCDConvert
<i>seg_data_1</i>	in	8	entrada del segundo digito del BCDConvert
<i>seg_data_2</i>	in	8	entrada del tercer digito del BCDConvert
<i>seg_sel</i>	out	6	salida para asignar el display a mostrar
<i>seg_data</i>	out	8	salida con el código en BCD para el display iterado

Módulo *controlDireccion* (figura 9) nos permite configurar dos salidas, una colocada en 0 y la otra en 1, estas salidas se conectarán con el controlador para poder configurar la dirección del motor. Para conocer todas las entradas y las salidas consultar la tabla 8.

Table 8: módulo *controlDireccion*

<b>Señal</b>	<b>Dirección</b>	<b>Tamaño</b>	<b>Descripción</b>
in1	out	1	señal 1 para la dirección del motor que usa el driver L298N
in2	out	1	señal 2 para la dirección del motor que usa el driver L298N

El módulo *switchesEntrada* (figura 10) este módulo nos permitirá poder hacer el cambio con el área morada (figura 1) para poder utilizar 8 switches en lugar de 3 botones. No se utilizará si se utilizan los módulos dentro de la zona morada. Para conocer todas las entradas y salidas consultar tabla 9.

Table 9: módulo *switchesEntrada*

<b>Señal</b>	<b>Dirección</b>	<b>Tamaño</b>	<b>Descripción</b>
sw0	in	1	entrada del switch 1 para establecer el bit 0
sw1	in	1	entrada del switch 2 para establecer el bit 1
sw2	in	1	entrada del switch 3 para establecer el bit 2
sw3	in	1	entrada del switch 4 para establecer el bit 3
sw4	in	1	entrada del switch 5 para establecer el bit 4
sw5	in	1	entrada del switch 6 para establecer el bit 5
sw6	in	1	entrada del switch 7 para establecer el bit 6
sw7	in	1	entrada del switch 8 para establecer el bit 7
valueReference	out	1	salida para la referencia del generador

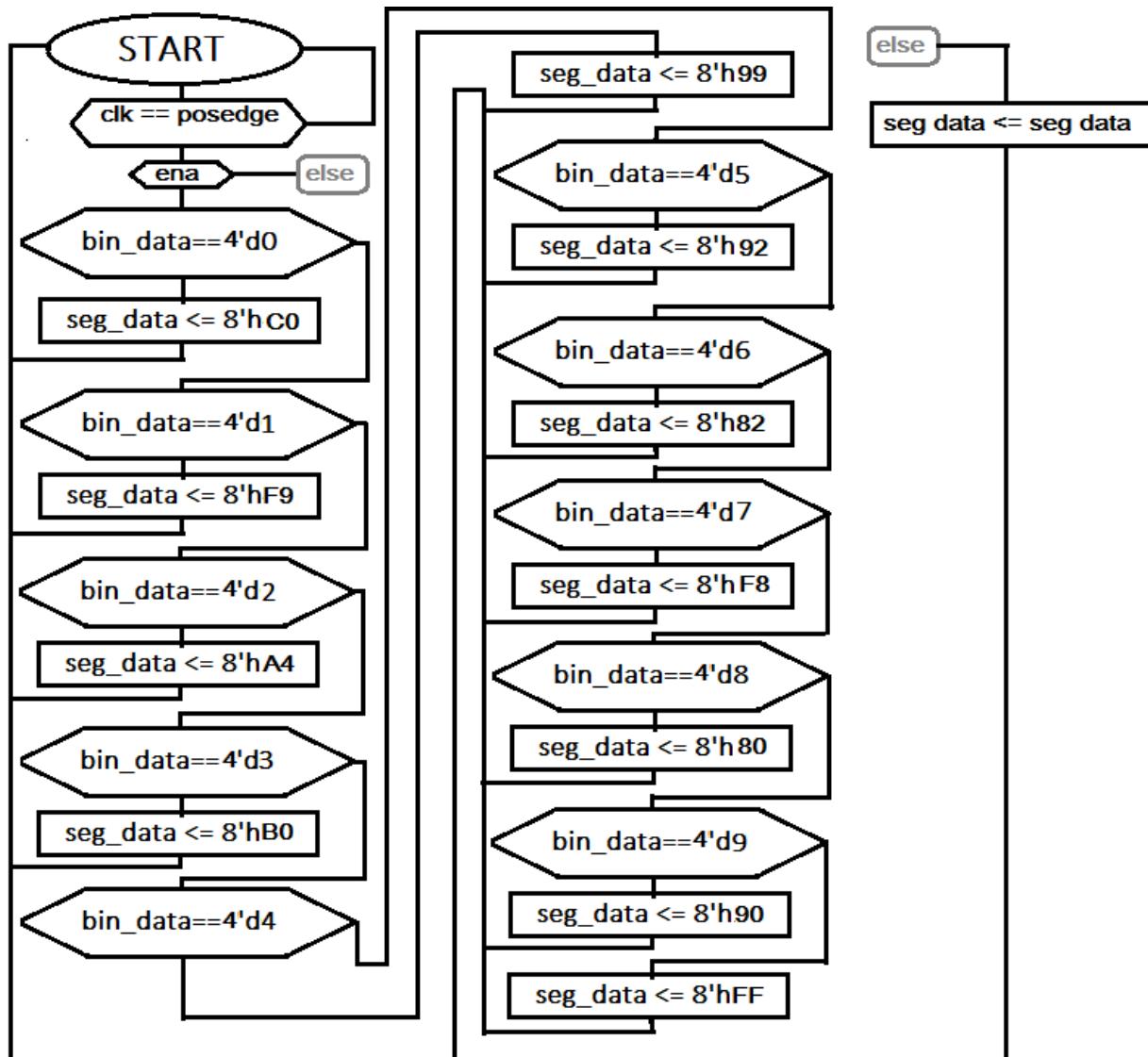


Figure 7: Diagrama de flujo para la creación del módulo seg decoder utilizado para poder codificar una entrada de 4 bits que representa un numero en binario de 0-9 a código BCD de 8 bits para un display de 7 segmentos.

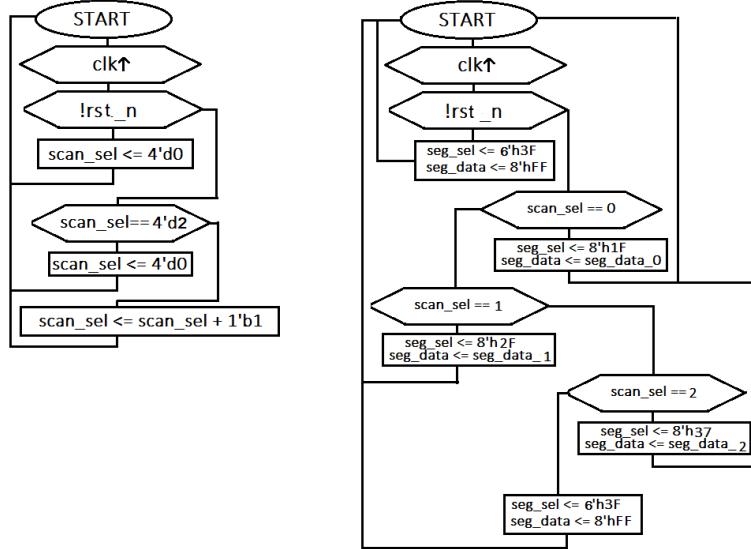


Figure 8: Diagrama de flujo para la creación del módulo **seg scan** utilizado para poder mostrar información a los displays de la tarjeta, recibiendo 3 entradas una para cada digito y sale una salida en código BCD con un selector del display.

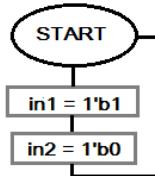


Figure 9: Diagrama de flujo para la creación del módulo **controlDireccion** utilizado para poder configurar la dirección de giro de un motor DC conectado al driver L298N.

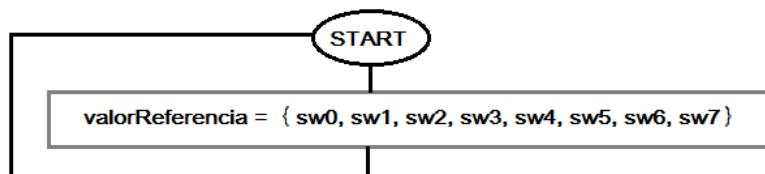


Figure 10: Diagrama de flujo para la creación del módulo **switchesReference** utilizado para poder ingresar y variar el valor de referencia utilizando 8 switches de una tarjeta FPGA.

## 4. Resultados simulados.

---

Para un mejor entendimiento de las imágenes consultar las tablas 10 y 11 para conocer la representación en hexadecimal de la salida codificada para la representación BCD.

Table 10: valores del selector de display

Selector binario	selector hexadecimal	selección
6'b111_111	8'h3F	N/A
6'b011_111	8'h1F	Display 1
6'b101_111	8'h2F	Display 2
6'b110_111	8'h37	Display 3

Table 11: codificacion para activacion de segmentos

Selector binario	selector hexadecimal	selección
8'b1111_1111	8'hFF	OFF
8'b1100_0000	8'hC0	0
8'b1111_1001	8'hF9	1
8'b1010_0100	8'hA4	2
8'b1011_0000	8'hB0	3
8'b1001_1001	8'h99	4
8'b1001_0010	8'h92	5
8'b1000_0010	8'h82	6
8'b1111_1000	8'hF8	7
8'b1000_0000	8'h80	8
8'b1001_0000	8'h90	9

En la siguiente recopilación de figuras podremos apreciar como la salida pwm\_top varia conforme la referencia varia, desde el instante 0 ns a 56000 ns. La variable valorReference\_tb es la referencia que nos permite varias el ciclo útil de la señal pwm\_tb, fdig\_tb, sdig\_tb y tdig\_tb son los números en binario que representan la referencia, pero cada uno es un dígito (unidades, decenas y centenas) y pwm\_tb es la señal pwm de salida. Como ejemplo en la figura 11, tenemos como valorReference\_tb=10, tenemos fdig\_tb=0, sdig\_tb=1, tdig\_tb=0 lo que representa 010, y si observamos las variables seg\_data1\_tb=c0, seg\_data2=f9, seg\_data3=c0 lo que representan la codificación a código bdc, uno y cero como corresponde. En la figura 18 podemos ver como se representan los valores de data1\_tb, seg\_data2, seg\_data3 y como se asociación a cada uno de los displays.

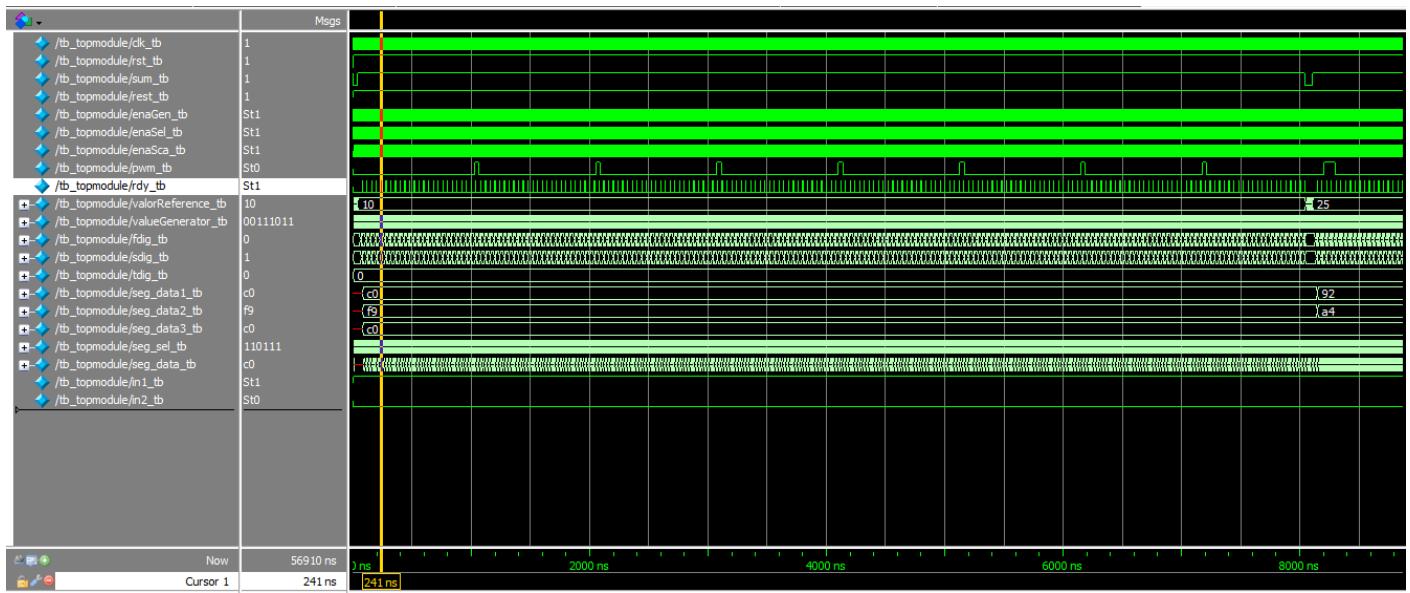


Figure 11: Simulación de 0-8000 ns con un valor de referencia de 10 y la respectiva salida pwm con el ciclo util correspondiente

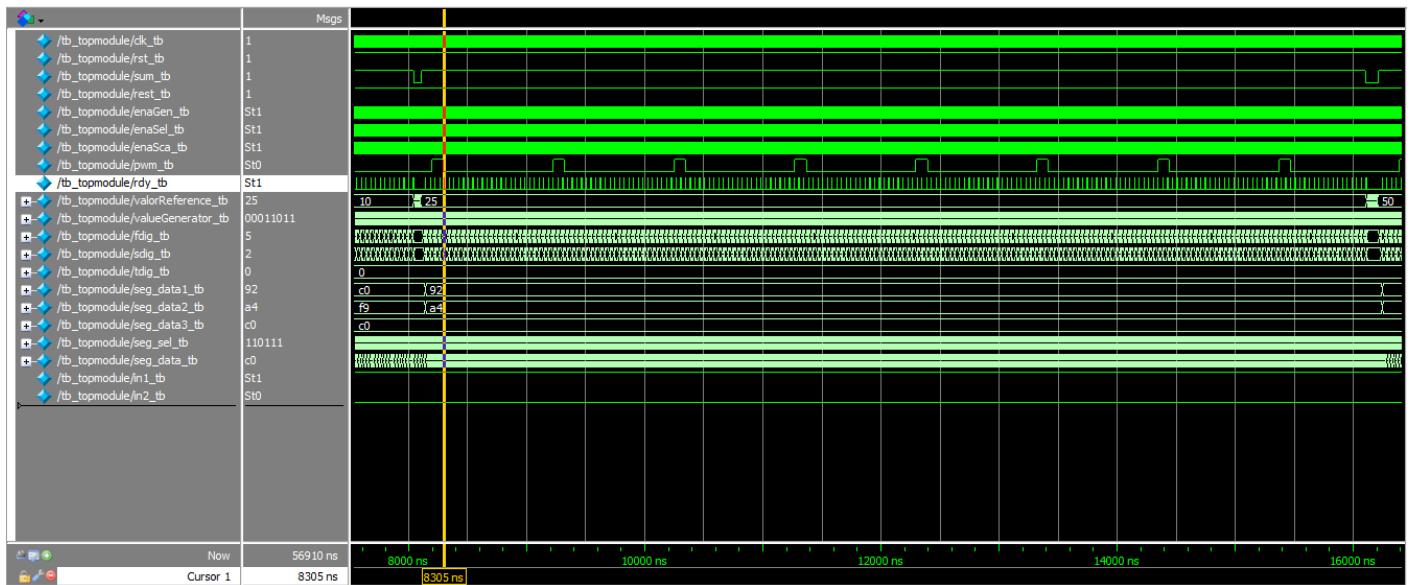


Figure 12: Simulación de 8000-16000 ns con un valor de referencia de 25 y la respectiva salida pwm con el ciclo util correspondiente

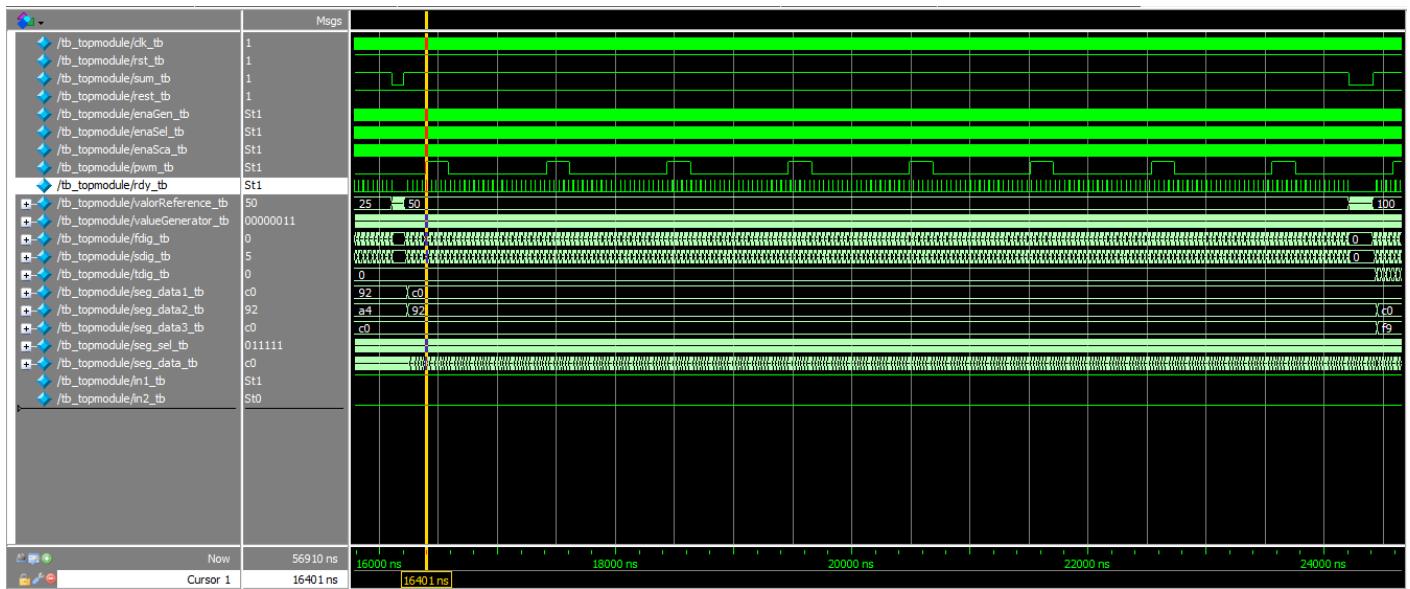


Figure 13: Simulación de 16000-24000 ns con un valor de referencia de 50 y la respectiva salida pwm con el ciclo util correspondiente

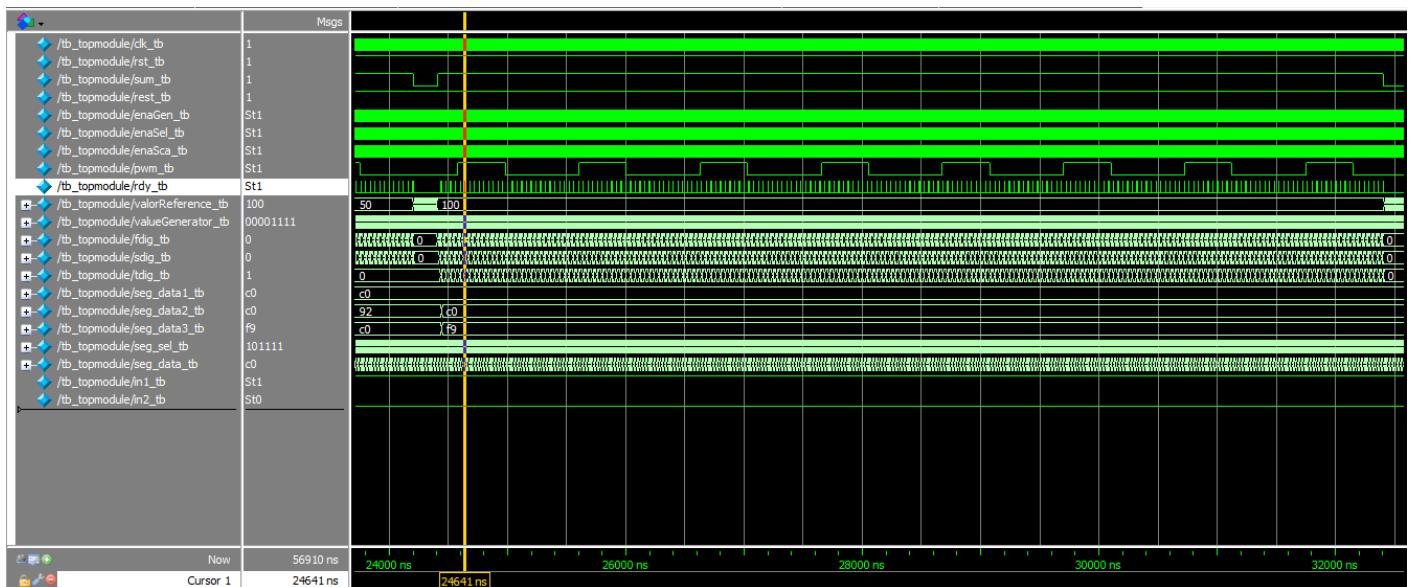


Figure 14: Simulación de 24000-32000 ns con un valor de referencia de 100 y la respectiva salida pwm con el ciclo util correspondiente



Figure 15: Simulación de 32000-40000 ns con un valor de referencia de 150 y la respectiva salida pwm con el ciclo util correspondiente

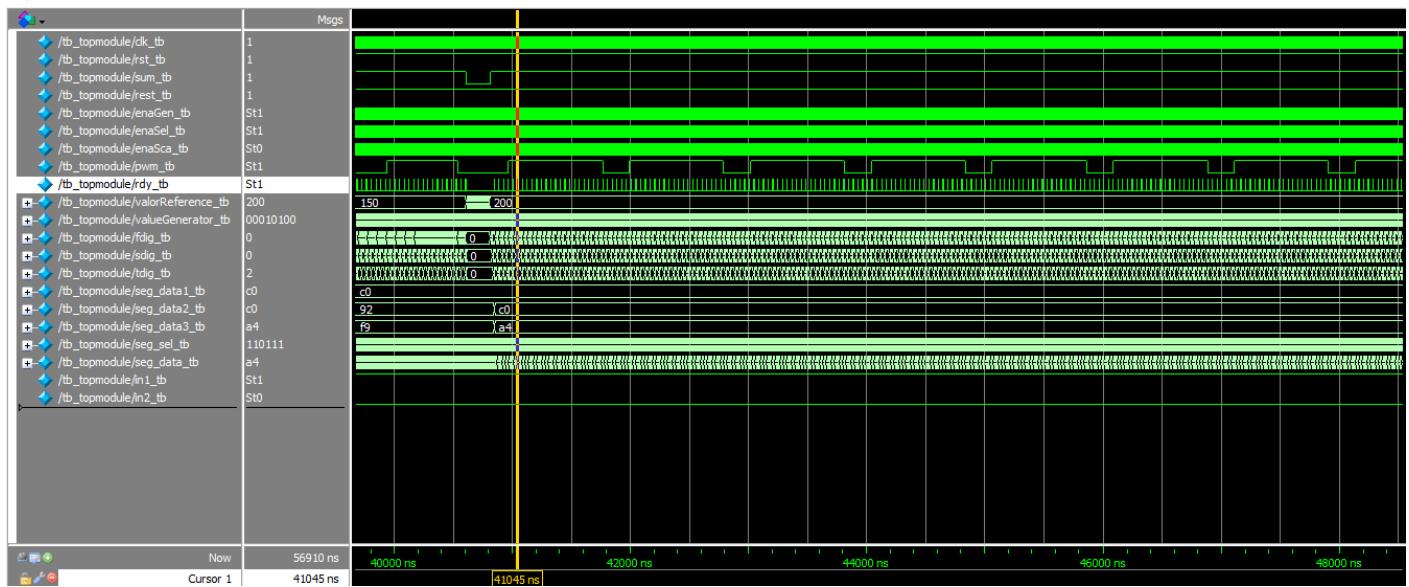


Figure 16: Simulación de 40000-48000 ns con un valor de referencia de 200 y la respectiva salida pwm con el ciclo util correspondiente

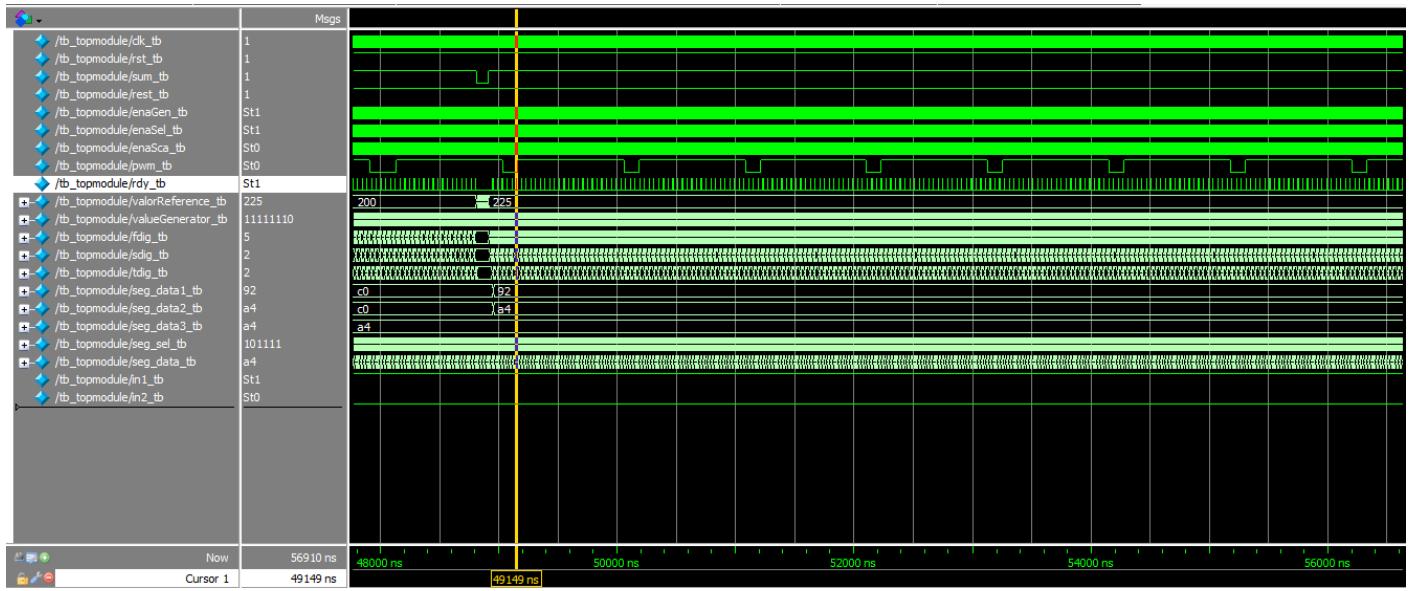


Figure 17: Simulación de 48000-56000 ns con un valor de referencia de 225 y la respectiva salida pwm con el ciclo util correspondiente

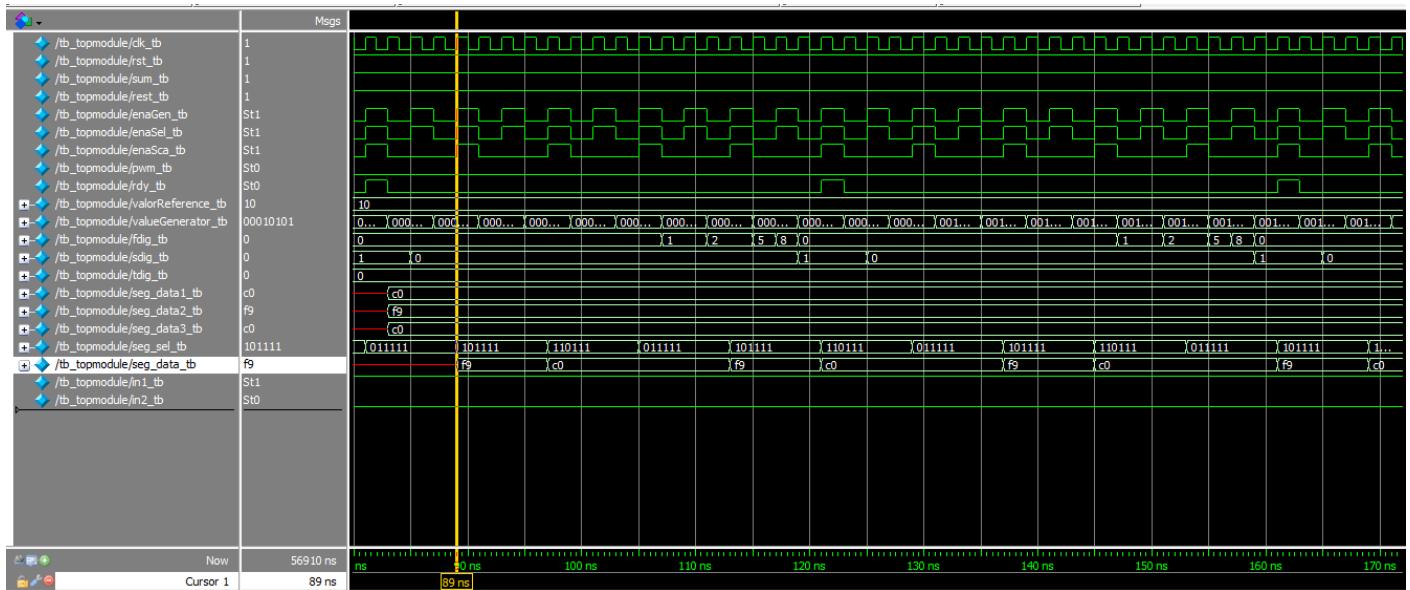


Figure 18: Simulación tomando como valor de referencia el número 10 mostrando las salidas conectadas a los displays

## 5. Conexiones experimentales.

A continuación, se presentar el circuito digital utilizado para implementar el modulo generador PWM a una tarjeta y utilizarlo para control la velocidad de un motor DC (figura 19), así como su implementación física (figura 20 y 21) si se desea conocer las entradas y salidas consultar la tabla 12.

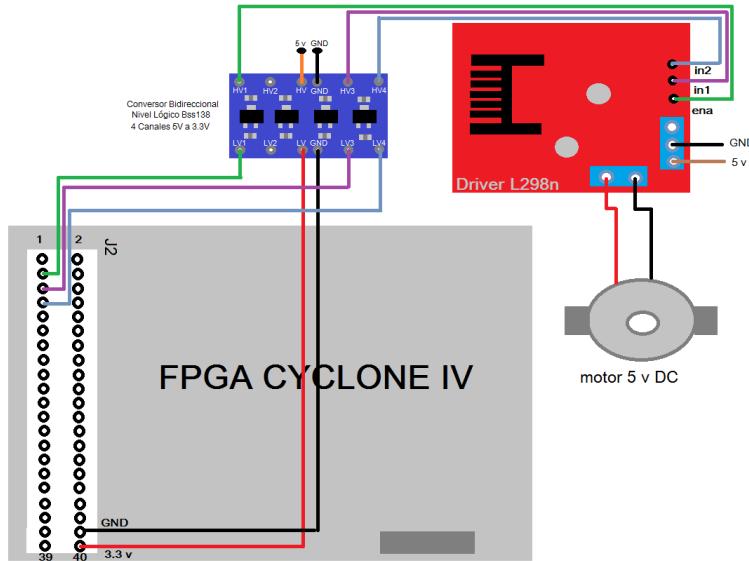


Figure 19: *Diagrama de conexiones para la implementación práctica para controlar la velocidad de un motor*

Table 12: conexión de los elementos para uso experimental.

nombre	color	ubicación	uso
pin 3	verde	FPGA	señal PWM utilizada para variar velocidad
pin 5	morado	FPGA	señal en 0 o 1 para dirección de motor
pin 7	azul	FPGA	señal el negado del pin 5 para dirección de motor
pin 38	negro	FPGA	tierra
pin 40	rojo	FPGA	voltaje de 3.3v utilizado como referencia BSS 138
<i>LV</i>	rojo	BSS 138	voltaje de referencia en bajo para convertidor lógico
<i>GND</i>	negro	BSS 138	tierra común
<i>HV</i>	naranja	BSS 138	voltaje de referencia alto para convertidor lógico
<i>LV1</i>	verde	BSS 138	señal pwm del FPGA a convertir en 5 volts
<i>LV3</i>	morado	BSS 138	señal de 3.3v de la señal 1 para dirección a 5v
<i>LV4</i>	azul	BSS 138	señal de 3.3v de la señal 2 para dirección a 5v
<i>in1</i>	morado	L298n	entrada del driver para dirección del motor
<i>in2</i>	azul	L298n	entrada del driver para dirección del motor
<i>vcc</i>	marrón	L298n	entrada de voltaje para alimentar motores
<i>GND</i>	negro	L298n	tierra común
<i>motorOut</i>	negro y rojo	L298n	conexión del motor

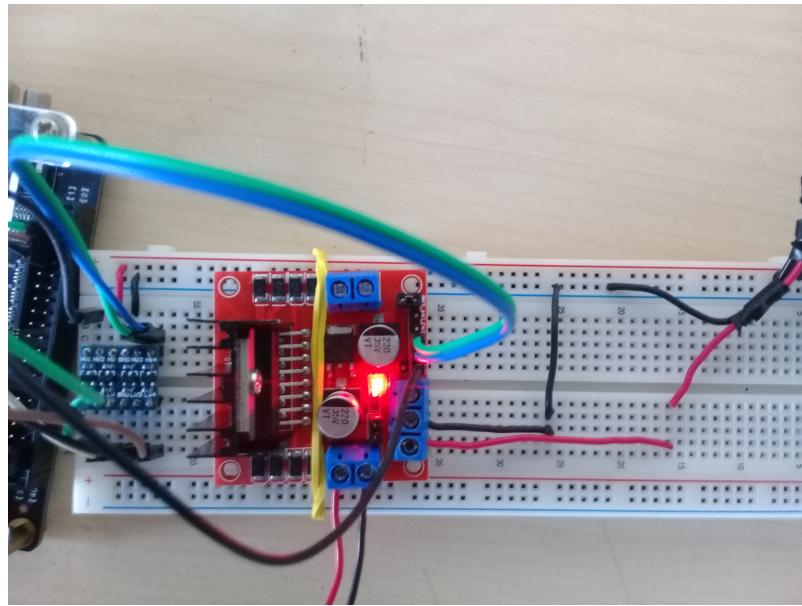


Figure 20: *conexiones entre Driver L298n, convertidor de nivel lógico y tarjeta FPGA*

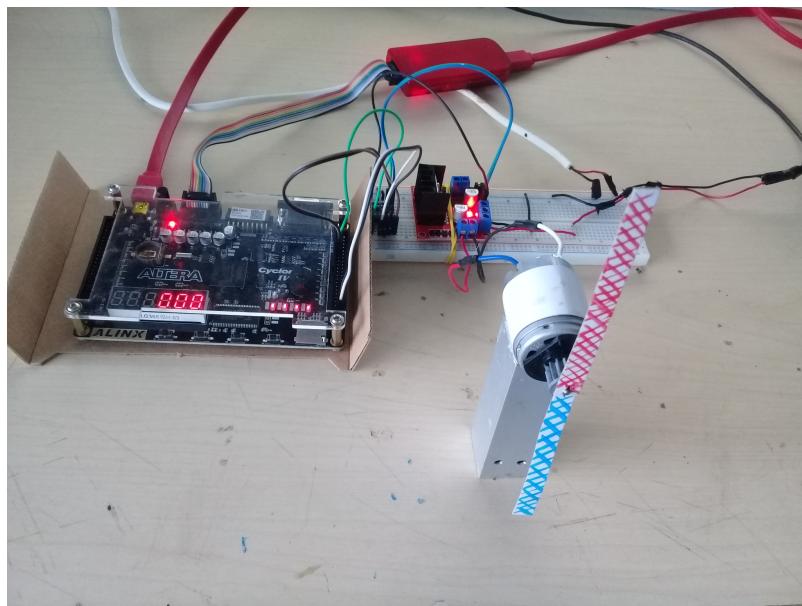


Figure 21: *Implementación completa del modulo cargado en la tarjeta, el driver, convertidor de nivel y motor DC*

## **6. Resultados experimentales.**

---

Se presentarán evidencias del funcionamiento del generador PWM, para poder acceder a evidencia en forma de videos acceder a la siguiente liga:

[Clickear aqui : Evidencia del funcionamiento](#)

### **6.1 Frecuencia de 1.531 KHz.**

Se presenta el desempeño del circuito al configurar el módulo para generar los pulsos PWM a una frecuencia de 1.5 KHz en la figura 22 mostramos el valor de referencia donde el motor muestra movimiento, considerando que ha alcanzado el voltaje mínimo para que gire, en la figura 23 aumentamos la referencia hasta un valor de 100 mientras el valor aumenta puede escucharse como aumentan las revoluciones del motor. En la figura 24 utilizamos el botón de reset para colocar todos los registros a cero y comenzar desde cero la referencia esta función es útil como un botón de apagado rápido. En la figura 25 podemos apreciar como utilizando el botón de restar una unidad a la referencia estando esta en cero podemos colocar en su máximo valor la referencia y por lo tanto el voltaje que recibe el motor es todo el que suministra la fuente conectada al controlador del motor. En la figura 26 podemos apreciar como aun valor de referencia de 42 el voltaje promedio que recibe el motor no es el mínimo para permitir el movimiento, sin embargo, puede escucharse como conducen las bobinas en su interior.

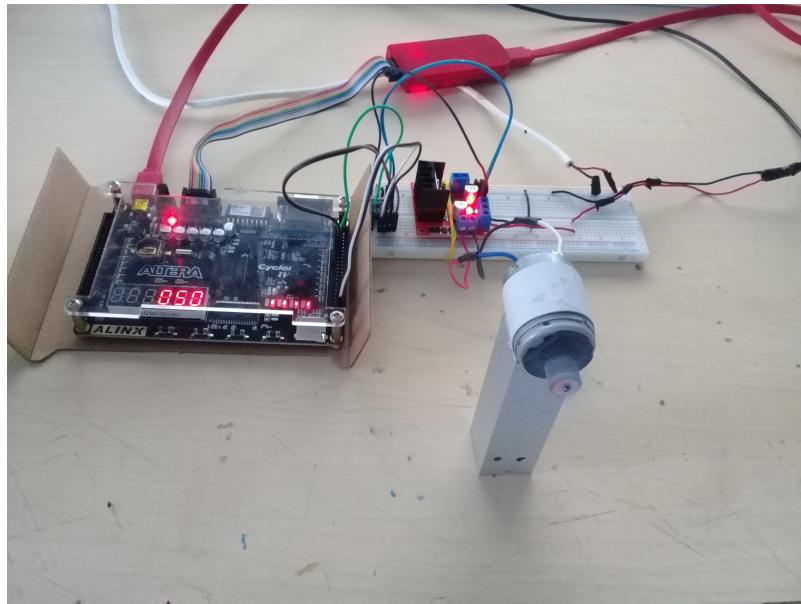


Figure 22: Módulo generador PWM a 1.5 KHz con una referencia de 50 con el motor en movimiento

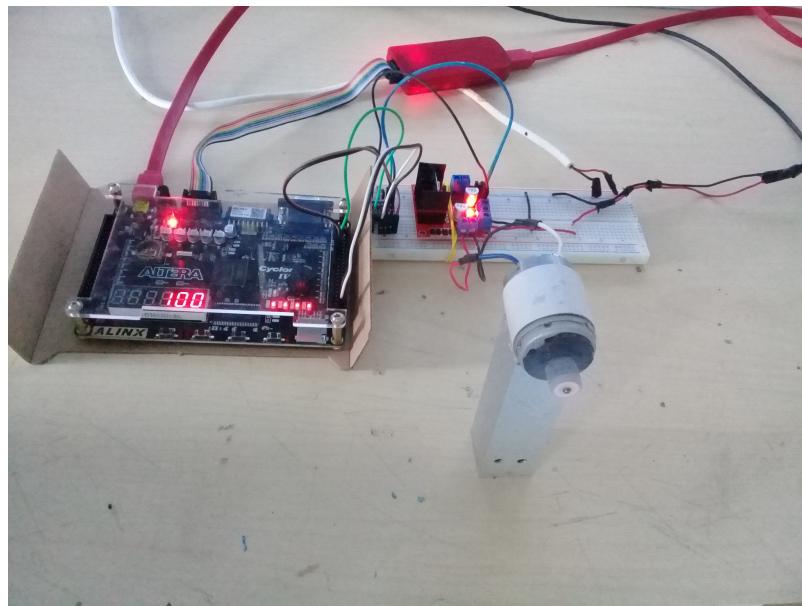


Figure 23: Módulo generador PWM a 1.5 KHz con una referencia de 100 con el motor en movimiento

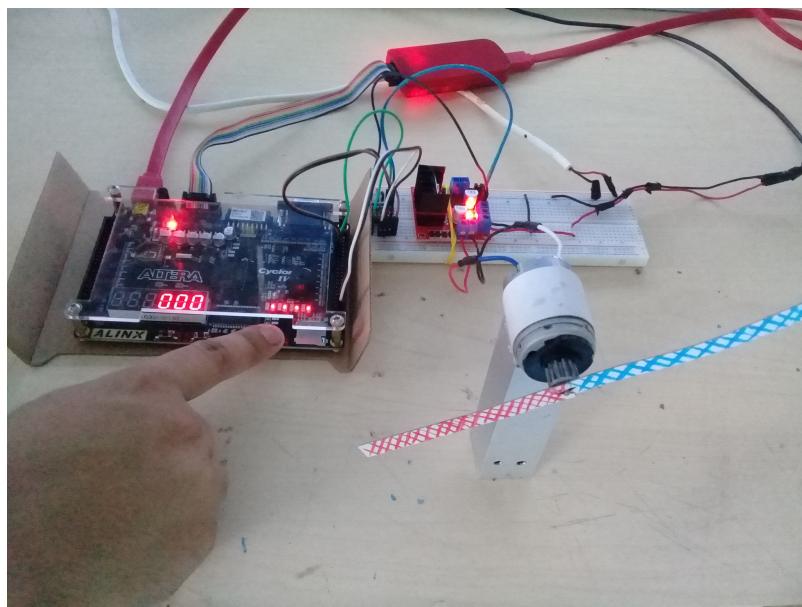


Figure 24: Módulo generador PWM a 1.5 KHz con una referencia de 0 con el motor estatico al reiniciar los registros

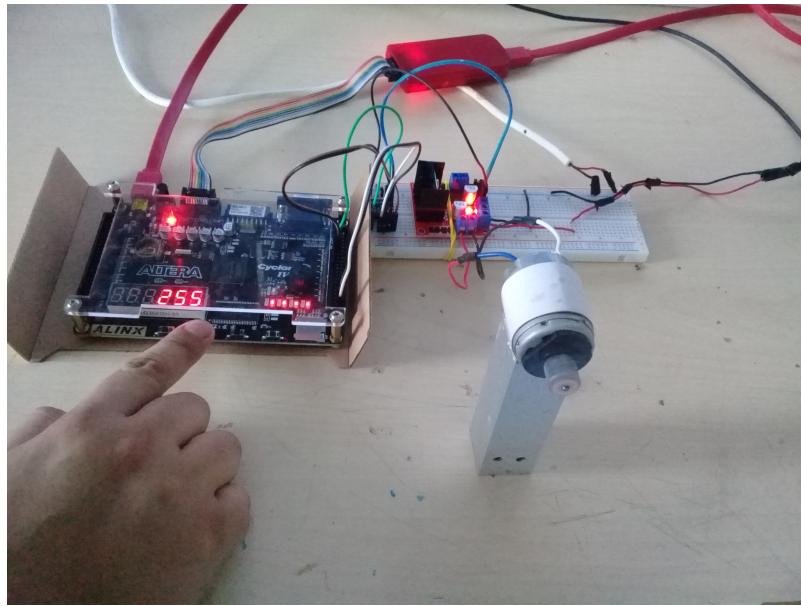


Figure 25: Módulo generador PWM a 1.5 KHz con una referencia de 255 con el motor en movimiento, dicha referencia obtenida al pasar de 0 a 255 utilizando el botón para restar

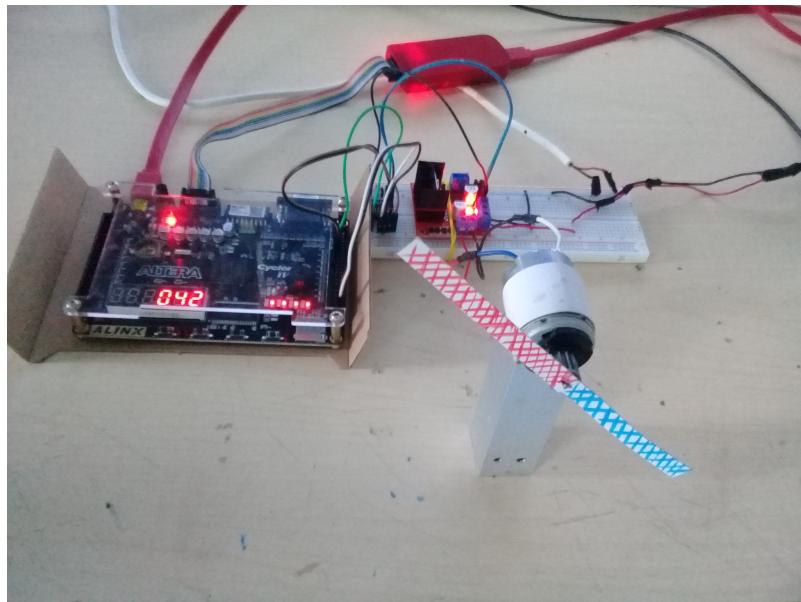


Figure 26: Módulo generador PWM a 1.5 KHz con una referencia de 42 con el motor estatico el voltaje promedio no es suficiente para permitir el movimiento del motor

## 6.2 Frecuencia de 24.5 KHz.

Aumentando la frecuencia de la señal PWM a 24.5 KHZ, podemos apreciar como un valor de referencia de 140 el motor no tiene el voltaje necesario para producir un movimiento, por lo tanto, podemos concluir que, a mayor frecuencia, mayor será el valor de referencia (ciclo de trabajo) necesario para provocar que la señal pwm permita el movimiento para este motor en específico.

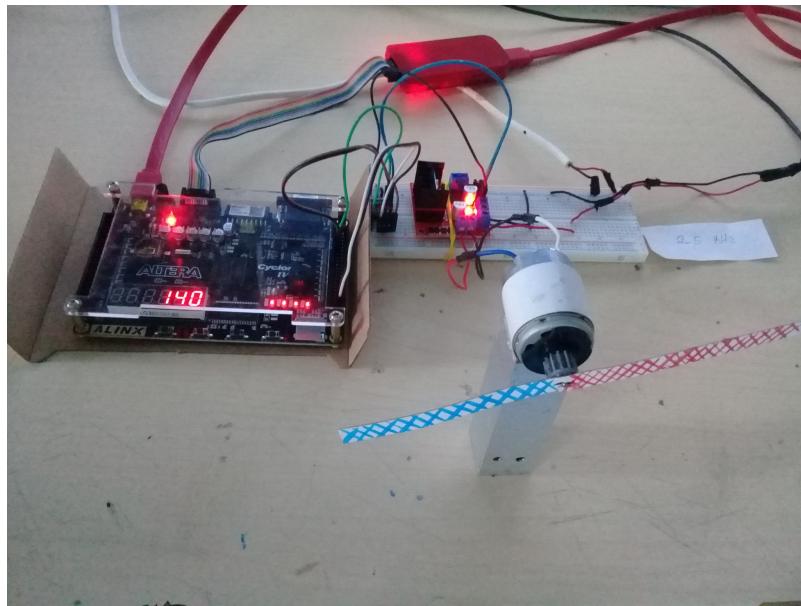


Figure 27: Módulo generador PWM a 24.5 KHz con una referencia de 140 con el motor estatico, el voltaje promedio a esta frecuencia es insuficiente para permitir el movimiento

### 6.3 Utilizando un led blanco para variar su intensidad.

En las siguientes figuras podemos apreciar como se hizo el mismo proceso experimental utilizando un led y variando su intensidad lumínica con la señal pwm. Es importante señalar que el cambio, aunque mínimo es perceptible ya que al aumentar el valor de referencia puede observarse como la zona iluminada por debajo del led aumenta su área y la intensidad de este.

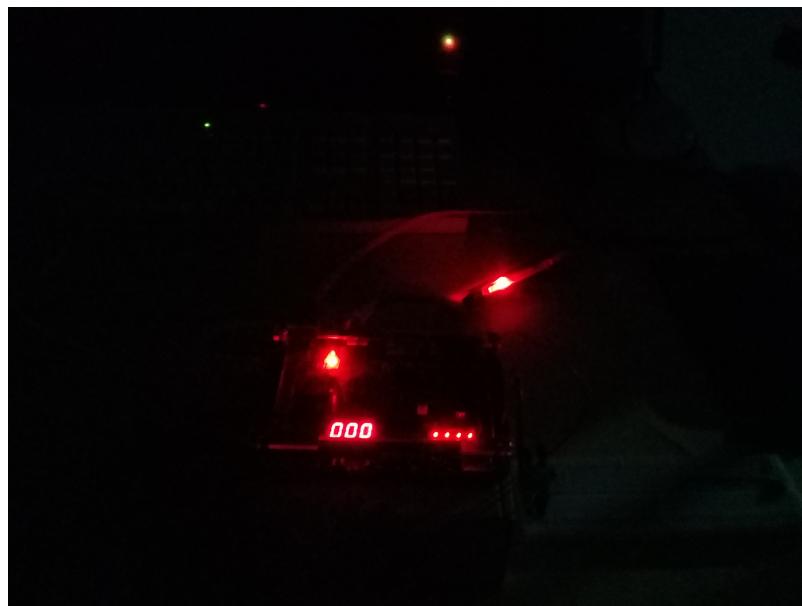


Figure 28: Módulo generador PWM a 1.5 KHz con una referencia de 0 con el led apagado

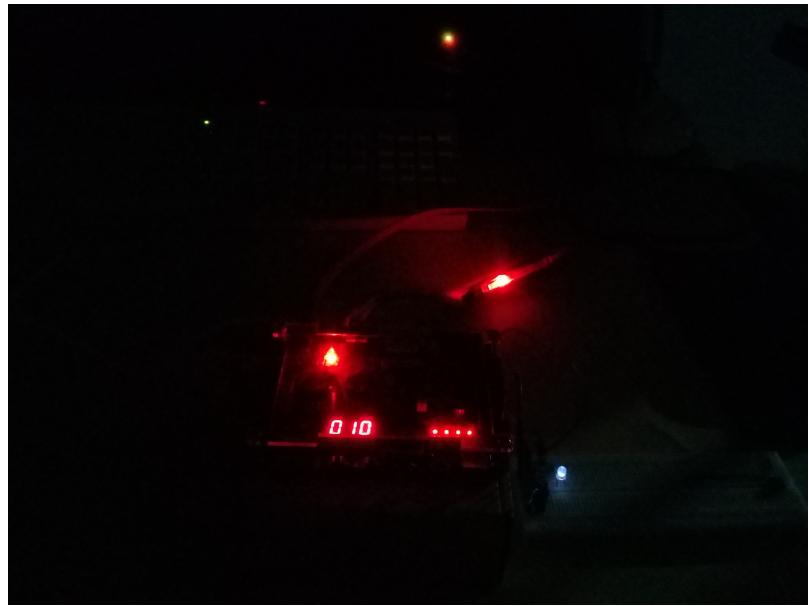


Figure 29: Módulo generador PWM a 1.5 KHz con una referencia de 10 con el led encendido con una intensidad tenue

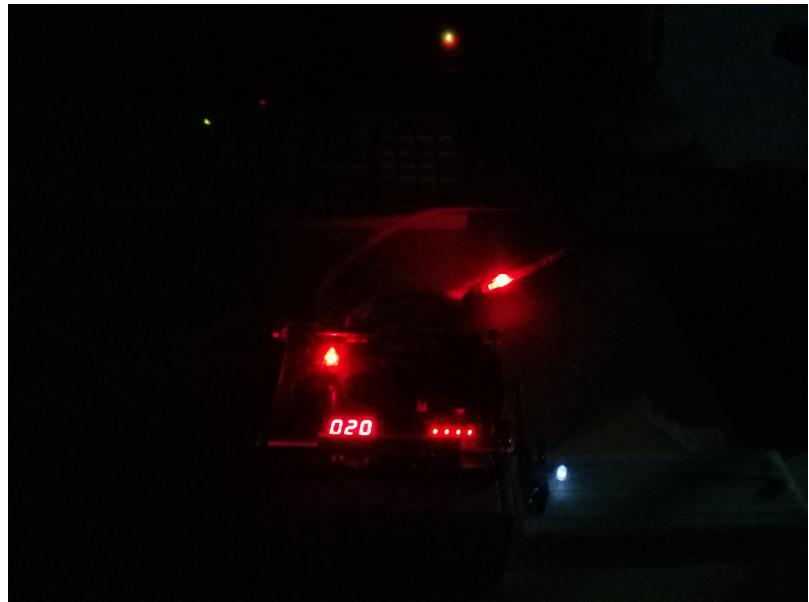


Figure 30: Módulo generador PWM a 1.5 KHz con una referencia de 20 con el led encendido con una intensidad tenue pero con mayor intensidad que el la imagen anterior

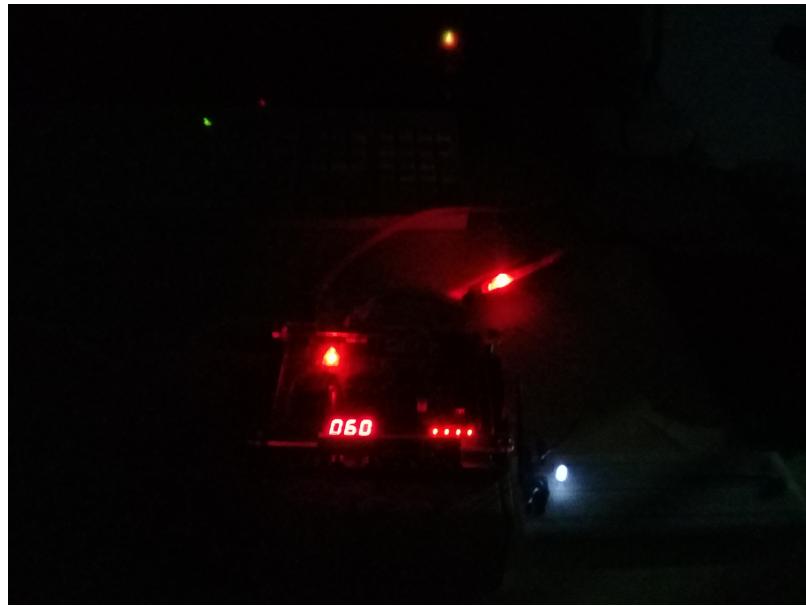


Figure 31: Módulo generador PWM a 1.5 KHz con una referencia de 60 con el led encendido con una intensidad baja

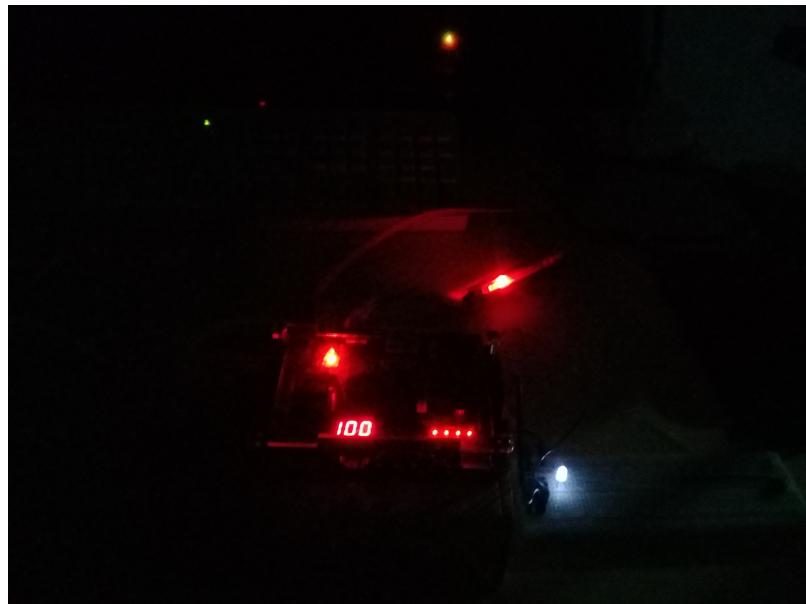


Figure 32: Módulo generador PWM a 1.5 KHz con una referencia de 100 con el led encendido con una intensidad baja pero de mayor intensidad

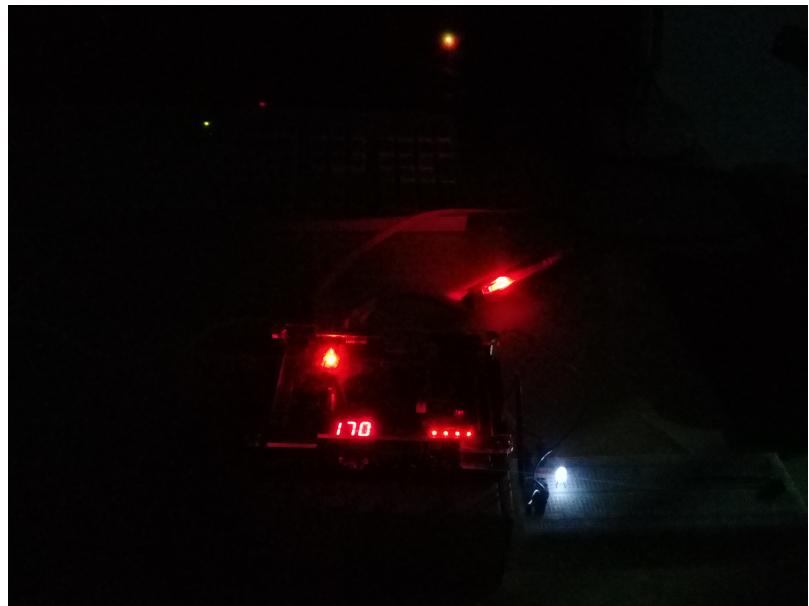


Figure 33: Módulo generador PWM a 1.5 KHz con una referencia de 170 con el led encendido con una intensidad media

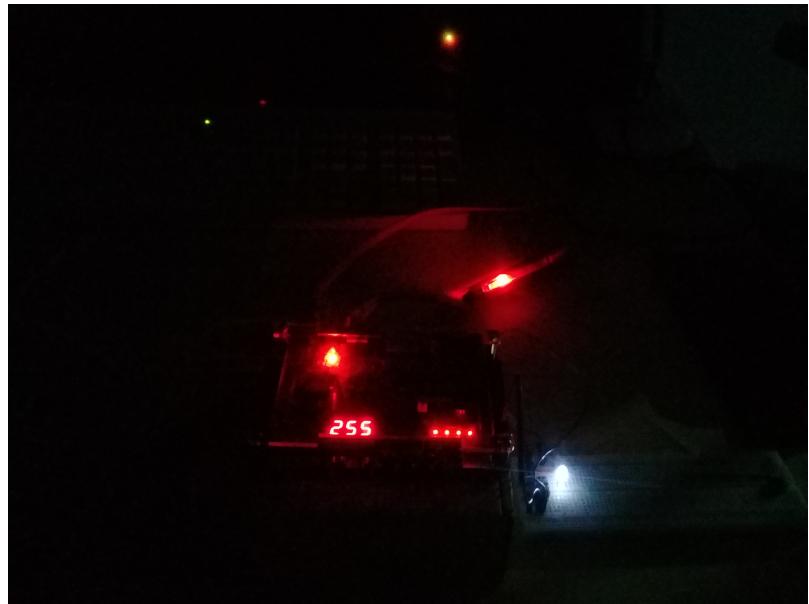


Figure 34: Módulo generador PWM a 1.5 KHz con una referencia de 2500 con el led encendido con una intensidad alta