Proyecto 1

Microcontrolador con CPU AVR

Bohórquez P. Duvier, Vargas F. Darmael, Loaiza C. Braulio

***Resumen*— El presente documento contiene todas las especificaciones, descripciones e indicaciones pertinentes sobre el funcionamiento de cada componente implementado en los módulos, que hacen parte del microcontrolador propuesto para su respectiva instauración y simulación, por medio del software proteus, mostrando los aspectos más relevantes encontrados en este proceso, que fueron útiles para la emisión de las conclusiones más apropiadas con respecto a este proyecto.**

***Índice de términos***— **Timer, Puerto In/out, Puertos, Señal PWM, Microprocesador, Memoria RAM, CPU AVR.**

# Introducción

l

os microcontroladores conocidos en la actualidad son elementos bastante útiles y fundamentales debido a su contribución al desarrollo de toda clase de sistemas digitales que, a su vez, suplen necesidades existentes en diversas áreas del conocimiento, al cumplir con las tareas requeridas en las mismas, teniendo en cuenta una configuración previa a nivel de software. Esto también ha representado un ahorro bastante prudencial en términos de costos y, por tal razón, es importante poseer un dominio práctico y conceptual con respecto al funcionamiento de los mismos.

Debido a esto se lleva a cabo el desarrollo consecutivo del presente proyecto, el cual consiste en la implementación y posterior simulación de un microcontrolador basado en una CPU AVR de 8 bits, y este contendrá diversos módulos que se encargan de realizar múltiples tareas, al estar interconectados, estableciendo comunicación continua entre los mismos con tal CPU para cumplir con las instrucciones dadas por dicho dispositivo.

# Objetivos

***Objetivo General***

* Realizar la implementación de un microcontrolador basado en CPU AVR, por medio del software proteus.

***Objetivos Específicos***

* Implementar módulos para entrada y salida del microcontrolador.
* Implementar un módulo para la memoria de datos SRAM
* Construir un temporizador de 8 bits que posea la configuración normal, CTC y Fast PWM.
* Añadir una memoria del programa o Flash Program Memory.
* Verificar el funcionamiento de cada módulo contenido por el microcontrolador.

# marco teórico

Es importante entender, primeramente, algunos conceptos que serán necesarios y se deben tener presentes a lo largo de este proyecto. Por tal razón se define el concepto de procesador, a partir de la sigla en inglés CPU o unidad central de procesamiento y este es un componente encargado de ejecutar una secuencia de instrucciones almacenadas, que se denominan como “programa”.

En un microprocesador, la función principal del núcleo central de procesamiento es garantizar la ejecución del programa; para lo cual, específicamente gestiona y permite el acceso adecuado a recursos como memoria, establece el control de periféricos, realiza cálculos y maneja las interrupciones.

El AVR es una CPU de arquitectura Harvard que pertenece a una familia de microcontroladores RISC. Cabe notar que con la arquitectura RISC se posibilita la segmentación y el paralelismo en la ejecución de instrucciones, minimizando este tiempo y se reducen los accesos a memoria.

Con la arquitectura AVR se implementan 32 registros de 8 bits que tienen un tiempo de acceso de solo un ciclo de reloj. Esto permite que cada operación de la ALU se haga en un ciclo de reloj. En una operación típica de la ALU, dos operandos son obtenidos desde el banco de registros, la operación se ejecuta, y el resultado se almacena nuevamente en uno de ellos en un solo ciclo de reloj.

El núcleo AVR fue diseñado inicialmente para la ejecución óptima de código de lenguaje C compilado.

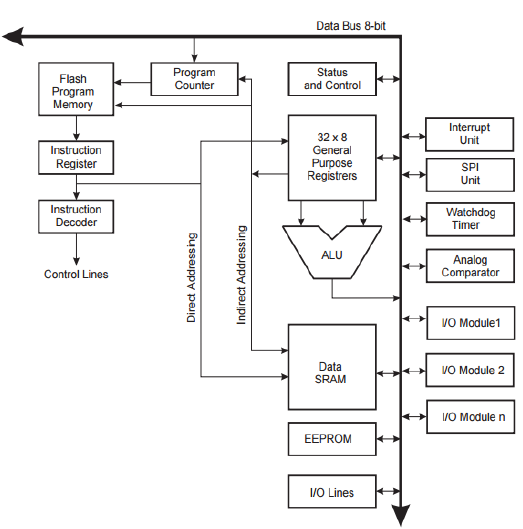
El espacio de memoria de la arquitectura AVR es totalmente lineal con un mapa regular de distribución.

**Configuración de pines**

Los pines de cada puerto tienen asociados individualmente tres bits que se encuentran ubicados en los registros de entrada/salida respectivos: Ddxn, PORTxn y PINxn; los primeros se acceden desde los registros DDRx, los segundos desde PORTx y los últimos desde PINx.

# Procedimiento

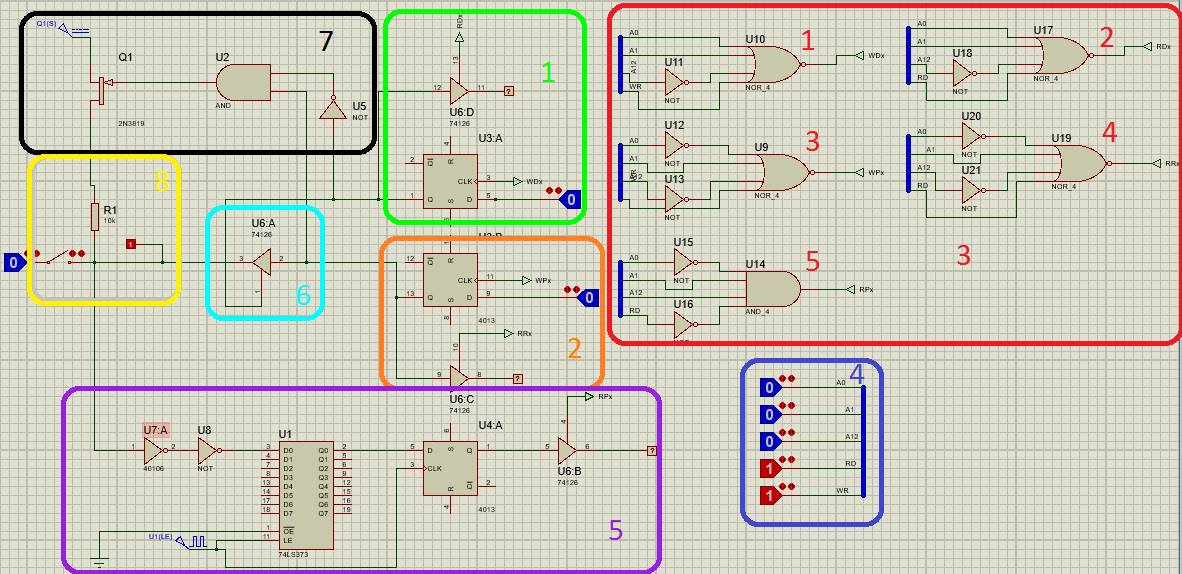
Para dar inicio al proyecto, se procede a realizar la implementación de los módulos propuestos, por la guía dispuesta para este trabajo, los cuales se identifican por medio del diagrama de bloques que describe el funcionamiento de todo el microcontrolador, como se observa a continuación, en la figura #.



**Fig. 1** Diagrama de bloques para el microcontrolador

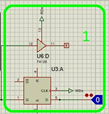
En este caso el proyecto se centra en la implementación de los módulos denominados Data SRAM, el cual corresponde a la memoria de datos, I/O Module n, quien hace referencia a los módulos de entrada y salida que serán requeridos en este caso y son, en efecto, puertos dispuestos para establecer comunicación entre la máquina y el usuario, un temporizador de 8 bits, el cual tendrá disponibles tres modos de operación, como se observara posteriormente. Como se ha mencionado, anteriormente, estos módulos se interconectaran con un microprocesador AVR de 8 bits, el cual se encargara de enviar las instrucciones correspondientes a cada módulo instaurado.

Primeramente, para realizar la implementación de los puertos de entrada y salida, se lleva a cabo la configuración y conexión de un circuito que se encargara de controlar el funcionamiento de cada pin compuesto por dichos puertos.



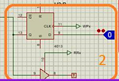
**Fig. 2** Circuito requerido para controlar un pin del puerto

Como se puede observar, en la figura 2, este circuito se encuentra dividido por secciones puesto que cada una cumple con una función específica y posee ciertas características, además de que trabaja de cierta manera, como se describe a continuación.



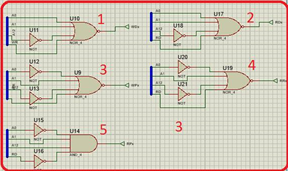
**Fig. 3** Sección del circuito que controla DDRx

La figura 3 muestra el segmento del circuito encargado de configurar el pin, ya sea para entrada o salida de acuerdo a la configuración ingresada por el usuario, donde el WDx es la entrada de reloj la cual activa el flip-flop, con 1 el pin queda de escritura y con un 0 el pin queda en modo lectura.



**Fig. 4** Sección del circuito que controla PORTx

En este caso la figura 4 muestra el segmento de escritura en el cual se ingresan los datos para mostrar en la salida del pin ya sea un 1 o un 0, donde para mostrar el valor a la salida debió ser configurado en el segmento anterior con un 1 para configurarlo En modo escritura. Para que el flip-flop deje pasar el número a mostrar es necesario el flanco de reloj que es WPx.



**Fig. 5** Compuertas que controlan el funcionamiento de cada sección del circuito

Como se muestra en la figura 5 en este segmento se encuentran las compuertas lógicas que, debido a su configuración, garantizan la activación de un solo canal del circuito en un mismo instante de tiempo, ya sea uno de los tres tristates, que se encuentran conectados a dichas compuertas, o los dos registros.

Explicado de manera más puntual, para las entradas al circuito (RDx, WDx, WPx, RPx y RRx) se tendrán productos de operaciones lógicas, de modo que solo se ejecutara una acción por flanco de reloj, de la siguiente manera:

Para que se active RDx:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A0 | A1 | A12 | WD | RD | RDx |
| 0 | 0 | 1 | 1 | 0 | 1 |

Para activar RRx

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A0 | A1 | A12 | RD | WD | RRx |
| 1 | 0 | 1 | 0 | 1 | 1 |

Para activar WDx

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A0 | A1 | A12 | RD | WD | WDx |
| 0 | 0 | 1 | 1 | 0 | 1 |

Para activar WPx

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A0 | A1 | A12 | RD | WD | WPx |
| 1 | 0 | 1 | 1 | 0 | 1 |

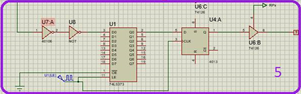
Para activar RPx

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A0 | A1 | A12 | RD | WD | RPx |
| 0 | 1 | 1 | 0 | 1 | 1 |



**Fig. 6** Probadores lógicos

Los probadores lógicos que se observan, en la figura 6, son utilizados para enviar los valores a las compuertas lógicas, donde A0 es para activar el flip-flop 2, A1 es para leer del puerto, A12 es para configurar el pin, RD es para leer del pin y WR es para escribir en el pin.



**Fig. 7** Seccion del circuito que controla PINx

El segmento mostrado, en la figura 7, es el encargado de leer la señal de entrada desde el pin, para cuando se configura en modo lectura.



**Fig. 8** Compuerta tristate

La compuerta tristate observada, en la figura 8, es utilizada para controlar el paso de la información suministrada por el registro de escritura, el cual solo debe funcionar cuando el pin se encuentre configurado como salida.



**Fig. 9** Compuertas conectadas a transistor MOSFET

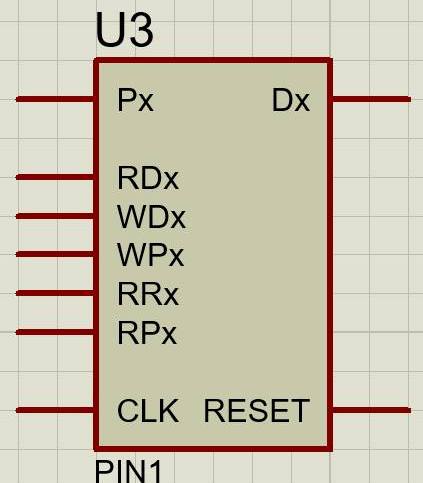
El segmento de la figura 9 contiene las compuertas lógicas para activar el transistor MOSFET, de acuerdo a la configuración que posea el pin, el cual se activa únicamente con un valor cero.



**Fig. 10** Resistencia de Pull-up

La resistencia presentada, en la figura 10, se denomina como resistencia de Pull-up, la cual es de protección para el pin, para cuando este está configurado en modo de lectura, se activa cuando el transistor se cortocircuite con un cero dejando pasar el voltaje a la resistencia.

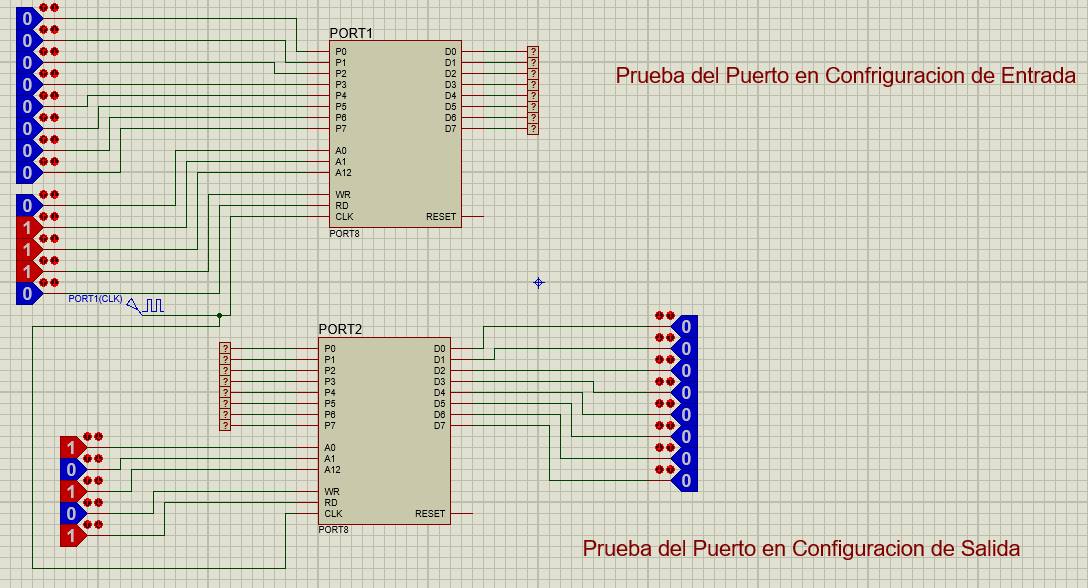
Al tener claras las características y el funcionamiento del circuito que controla un pin, se procede a realizar la debida implementación de la pastilla que contendrá dicho circuito.



**Fig. 11** Pastilla diseñada para empaquetar el circuito del pin

El integrado que se observa, en la figura 11 posee 9 pines, 7 de entrada y dos bidireccionales, entre los pines de entrada se encuentra: RDx, WDx, WPx, RRx, una entrada de reloj y una de reset cuya función es reiniciar el funcionamiento de sus flipflop internos; entre los pines bidireccionales se encuentra Px y Dx cuya configuración depende del manejo de datos que se le da al circuito del pin, pueden se tanto de entrada como de salida, pero no podrán tener la misma configuración, ya que esto ocasionaría un cortocircuito y quemaría el microcrontrolador.

Al realizar las pruebas correspondientes sobre este componente, para verificar su correcto funcionamiento y efectividad, se procede a realizar el montaje del integrado que contendrá 8 de estos circuitos, puesto que se requiere la manipulación continua de 8 bits a lo largo de los diferentes módulos que contiene el microcontrolador.

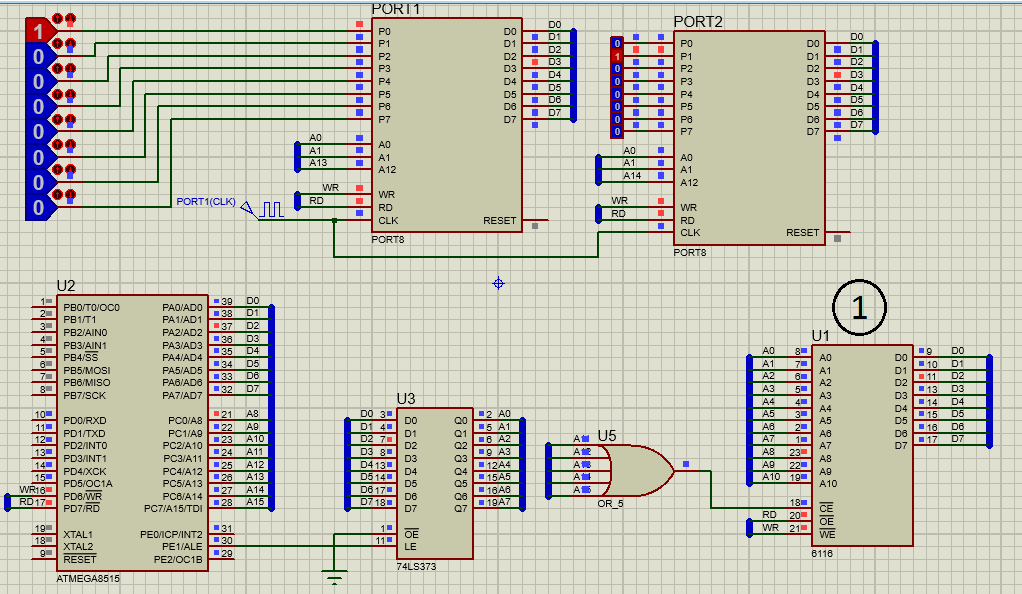


**Fig. 12** Integrado del puerto para el módulo de entrada y salida

La imagen anterior muestra el funcionamiento del componente que representa el puerto de entrada y salida a simular, por medio del software proteus. Como se puede observar en la misma, es necesario implementar dos puertos para realizar pruebas independientes, al configurarlo como entrada o salida, respectivamente. Analizando a detalle dicho componente, este contiene un conjunto de 8 pines bidireccionales [P0..P7] que se utilizan para establecer la conexión entre el usuario y la máquina.

Además de esto, el componente también posee un conjunto de 8 pines bidireccionales [D0..D7] que funcionan en pro de la máquina, para recibir datos enviados por el usuario, cuando el pin se encuentra configurado como entrada, para escribir datos en cada registro del circuito, y para obtener la información contenida en los mismos. Junto a estos se encuentran situados otros pines de configuración, para enviar la orden, al puerto, de realizar una escritura (WR) o leer información de alguna dirección (RD) la cual se especifica, en binario, por medio de los pines A0, A1 y A12.

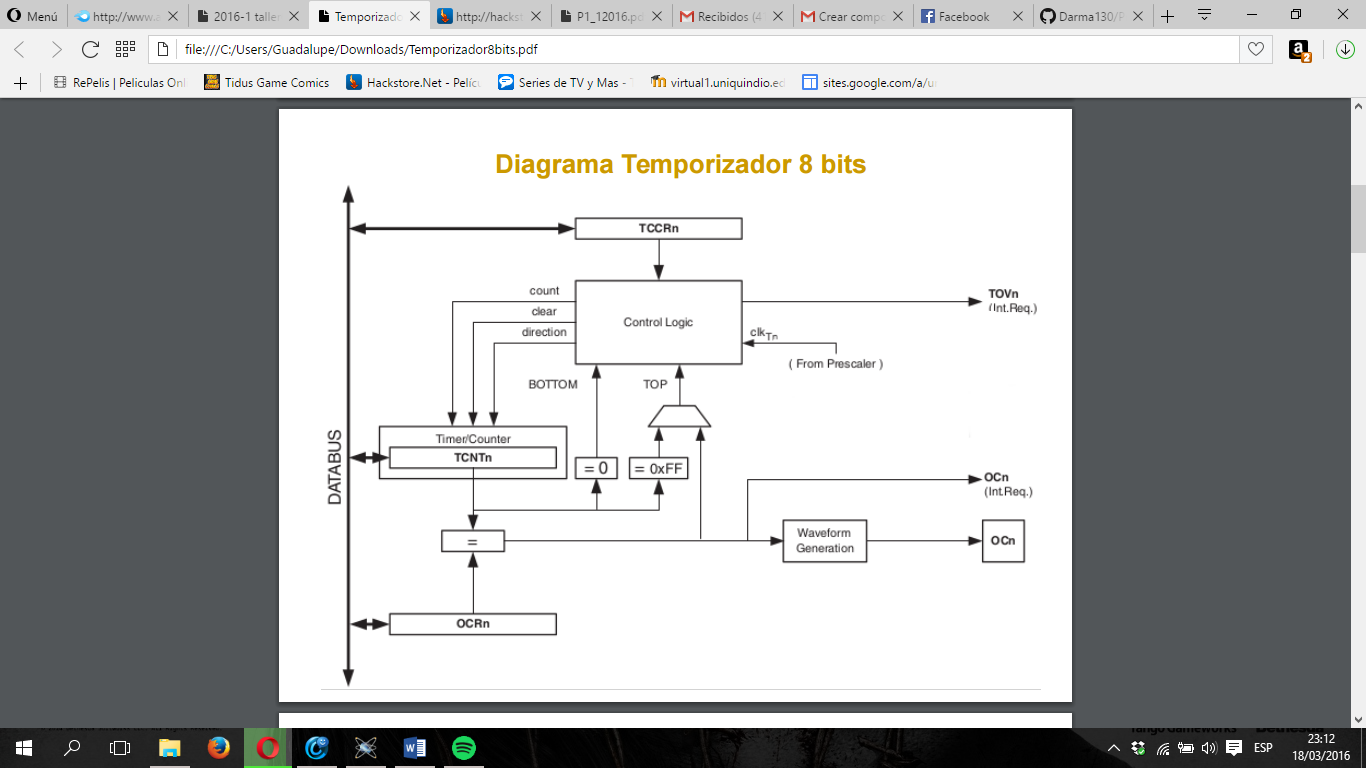
Con este módulo debidamente implementado y probado, verificando su correcto funcionamiento, se procede a realizar la implementación del módulo correspondiente a la memoria de datos, o memoria RAM, no sin antes hacer uso del microprocesador atmega 8515, el cual se encargara de compilar los algoritmos a utilizar para probar el funcionamiento de cada módulo.



**Fig. 13** Montaje de la memoria RAM junto con los módulos de entrada/salida y el microprocesador

La compuerta lógica, junto con el integrado señalado con 1, en la figura 13 representan el modulo para la memoria de datos. Esta memoria RAM puede leer y escribir según lo declare el microprocesador y esta cuenta con 10 líneas de memoria de las cuales 7 son del latch y las otras tres salen del microprocesador. Es activado con un enable el cual es designado por una compuerta OR administrada por el microprocesador, ya que el circuito solo puede hacer una tarea a la vez. Esto se traduce al hecho de que, en este caso, los puertos tienen asignadas direcciones que se encuentran por fuera de estas líneas de dirección y, por lo tanto, si se trabaja con estos componentes, se encontrara activo uno de las direcciones posicionadas en la compuerta or, garantizando un uno que evitara la activación de la memoria RAM en ese instante de tiempo.

Ahora, para realizar la implementación del módulo correspondiente al temporizador de 8 bits, es necesario realizar un análisis previo del diagrama de bloques que se muestra a continuación, en la figura 14.

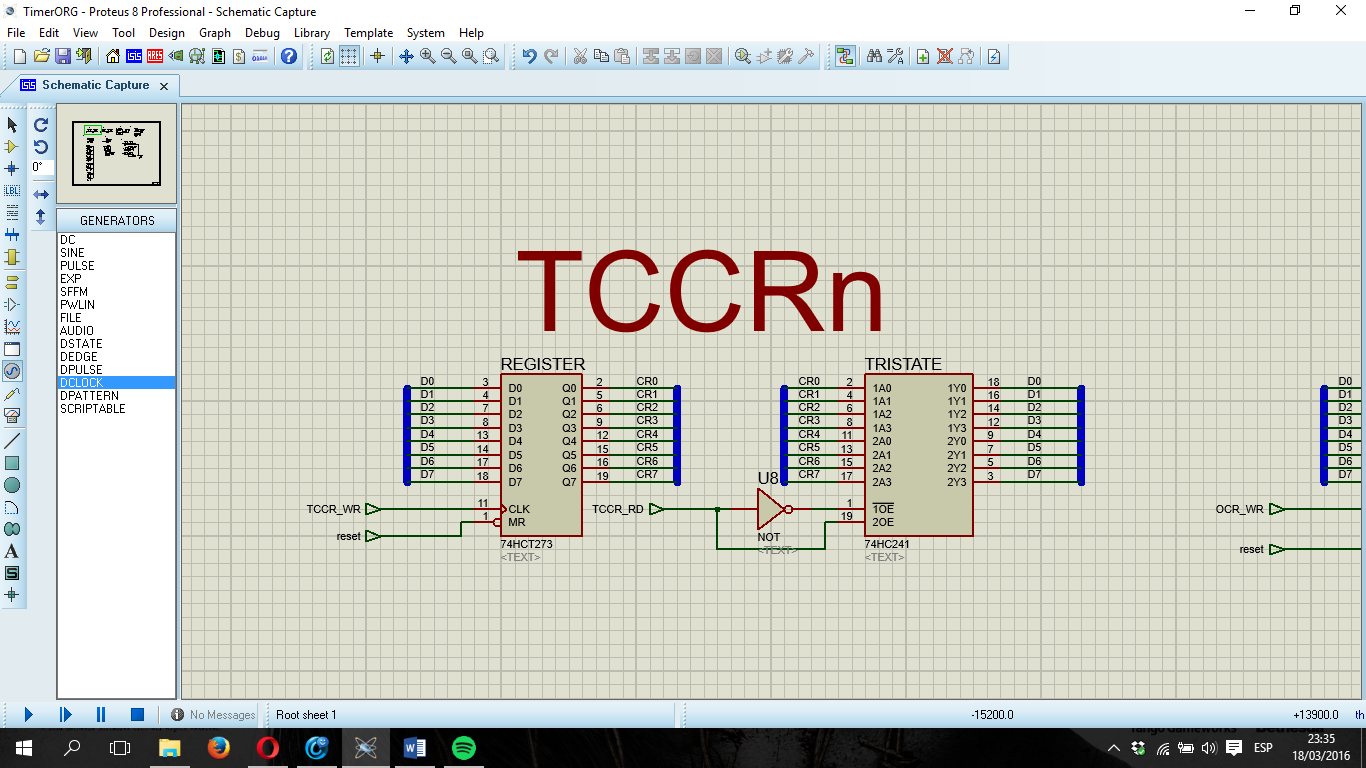


**Fig. 14** Diagrama de bloques para el temporizador

Como se puede observar, en la figura 14, este módulo contiene diversos componentes y conexiones, los cuales corresponden a operaciones de comparación y ejecución, puesto que este módulo tiene tres modos de operación, quienes a su vez requieren de varias condiciones.

En la parte superior se tiene a TCCRn que es una de las partes más fundamentales del circuito la cual se encarga de recibir comandos directamente desde el microprocesador y configura el modo de operación del TOP y funcionamiento del preescalador, también se encarga de controlar el generador de forma de onda.

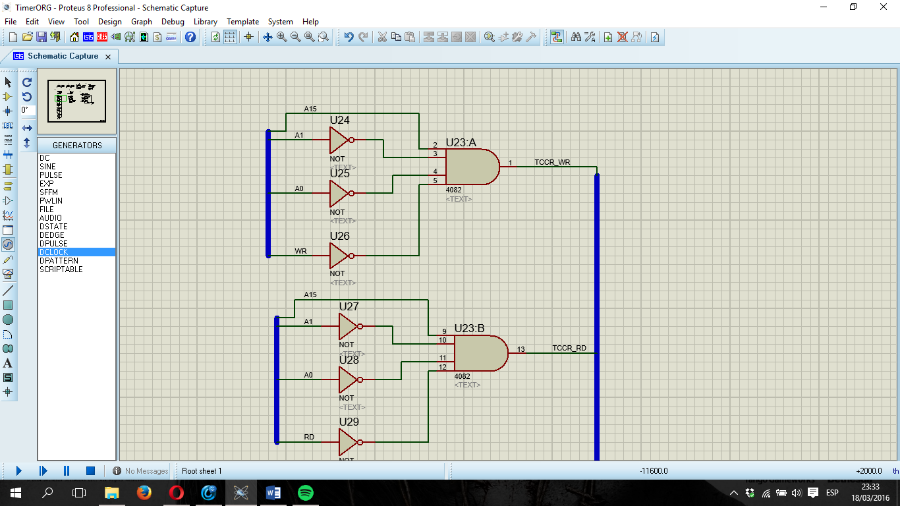
Su configuración se puede observar a continuación, en la figura 15.



**Fig. 15** Circuito de escritura y lectura para TCCRn

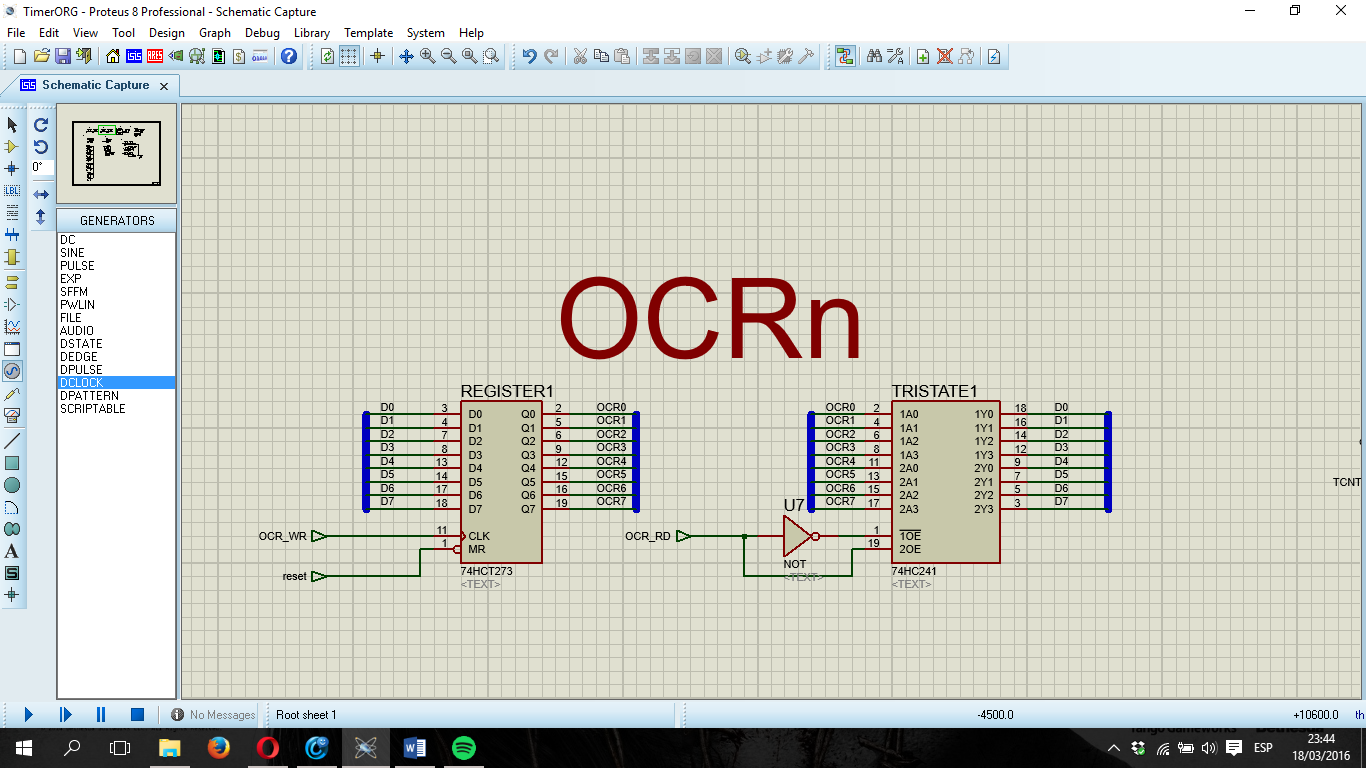
Las salidas CR0, CR1, CR2, CR3, CR4, CR5, CR6, CR7 corresponden respectivamente a los valores de CS00, CS01, CS02, WGM01, COM00, COM01, WGM00.

Sus datos pueden ser bidireccionales, así que puede ser de lectura o escritura, su posición en bloque de memoria se asume como 0x8000 y su configuración depende de A0, WR y RD de la siguiente forma:



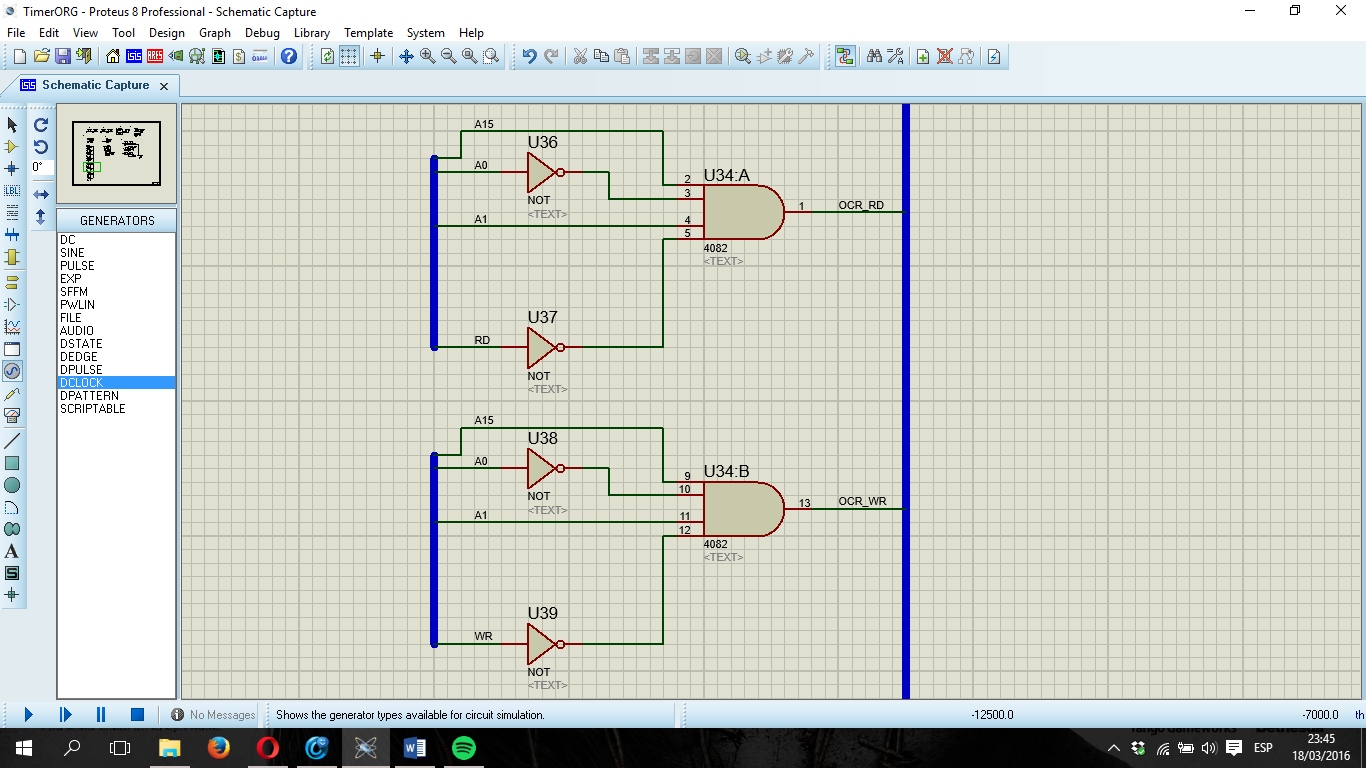
**Fig. 16** Compuertas lógicas para la dirección de TCCRn

OCRn de la misma manera que TCCRn tiene datos tanto de entrada como de salida, dependiendo de la configuración de lectura o escritura del mismo, posteriormente estos datos se comparan con los datos del Timer/counter originando el valor de TOP. Su estructura es de la siguiente manera:



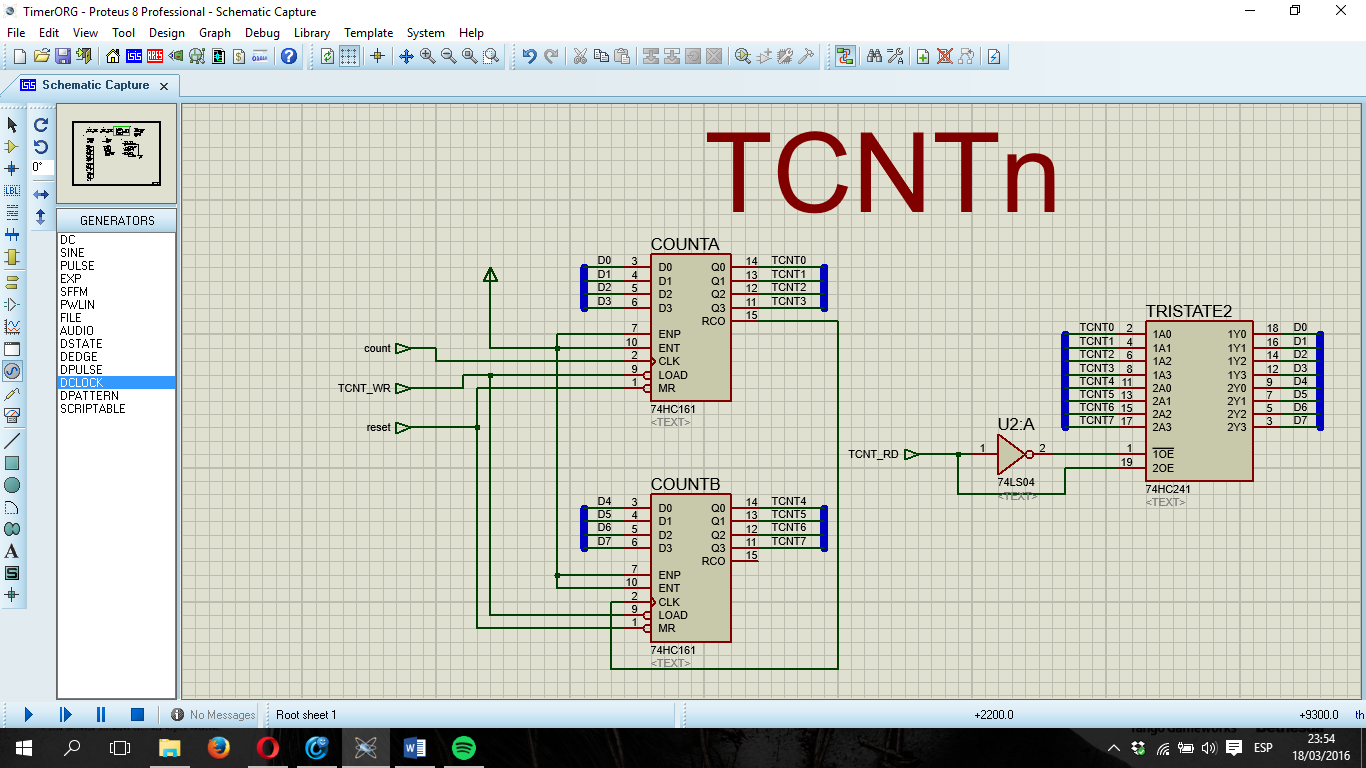
**Fig. 17** Circuito de escritura y lectura para OCRn

Su ubicación en el bloque de memoria corresponde a la dirección 0x8002 dependiendo de A15, A1 y A0 y su configuración lógica está dada por:



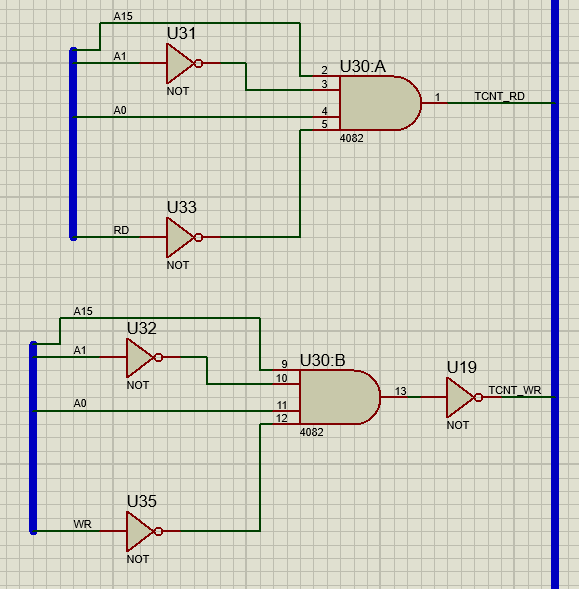
**Fig. 18** Compuertas lógicas para la dirección de OCRn

TCNTn corresponde al temporizador y contador del circuito recibiendo órdenes directas del bloque de control lógico y sus salidas se comparan con los datos del bloque anterior, siendo fundamental en la creación de BOTTOM y TOP, su configuración puede ser de lectura y de escritura, su posición en el bloque de memoria es 0x8001 dependiendo ésta de A15, A1, A0, WR y RD. Su estructura e de la siguiente forma:



**Fig. 19** Circuito de escritura y lectura para TCNTn

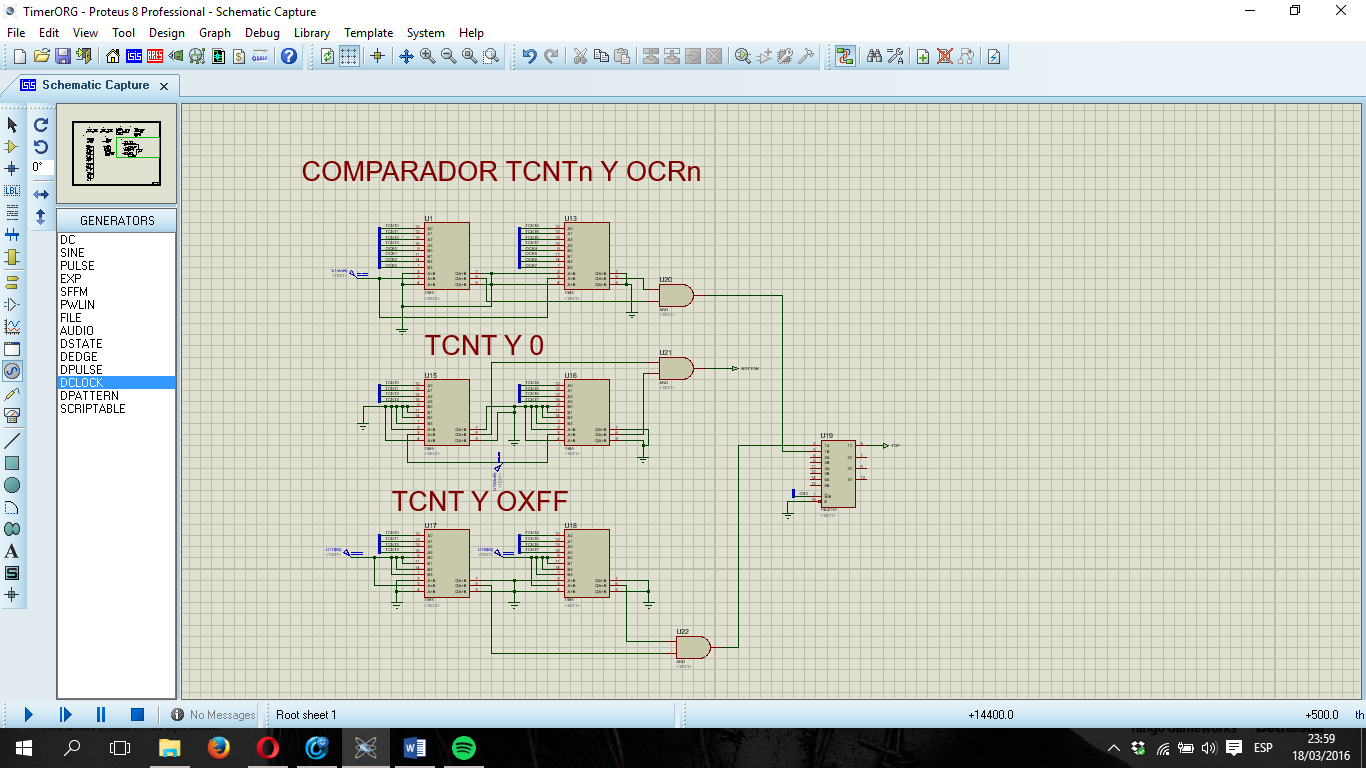
Y su estructura de control se observa en la figura 20.



**Fig. 20** Compuertas lógicas para la dirección de TCNTn

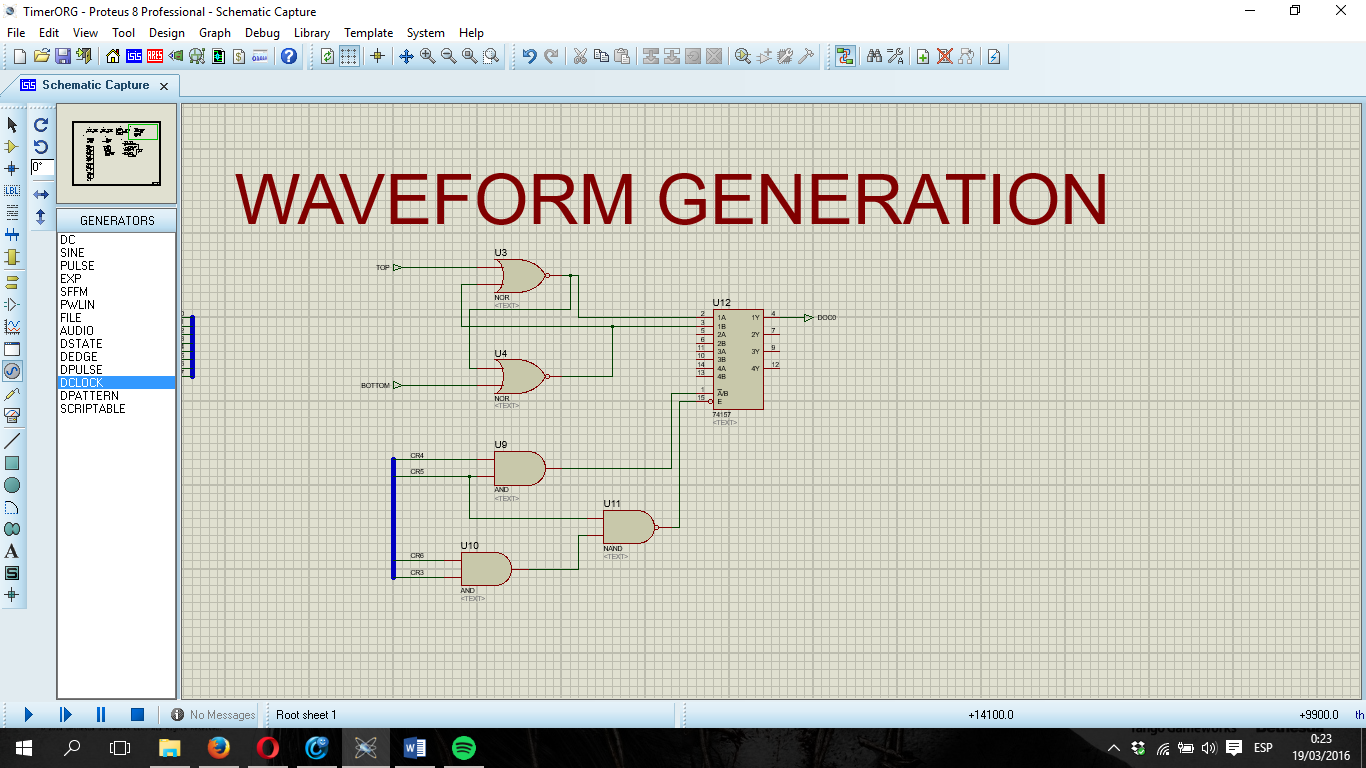
***Estructura de comparación***

Se constituye por la comparación directa entre los datos de TCNT y OCR, TCNT y 0x00, y finalmente TCNT y 0xFF, para esto se usaron comparadores 7845 configurados en cascada, como se observa en la figura #, esto porque la máxima capacidad individual de comparación es de 4 bits por componente, además solo se mira si los datos son iguales, ignorando sus otras formas de comparación, el resultado de comparar TCNT y 0x00 define el valor de BOTTOM, mientras que las dos comparaciones restantes van a un multiplexor cuyo modo de operación define WGM01 también llamado CR3 y este sería el comando TOP que iría a la estructura de control lógico principal.



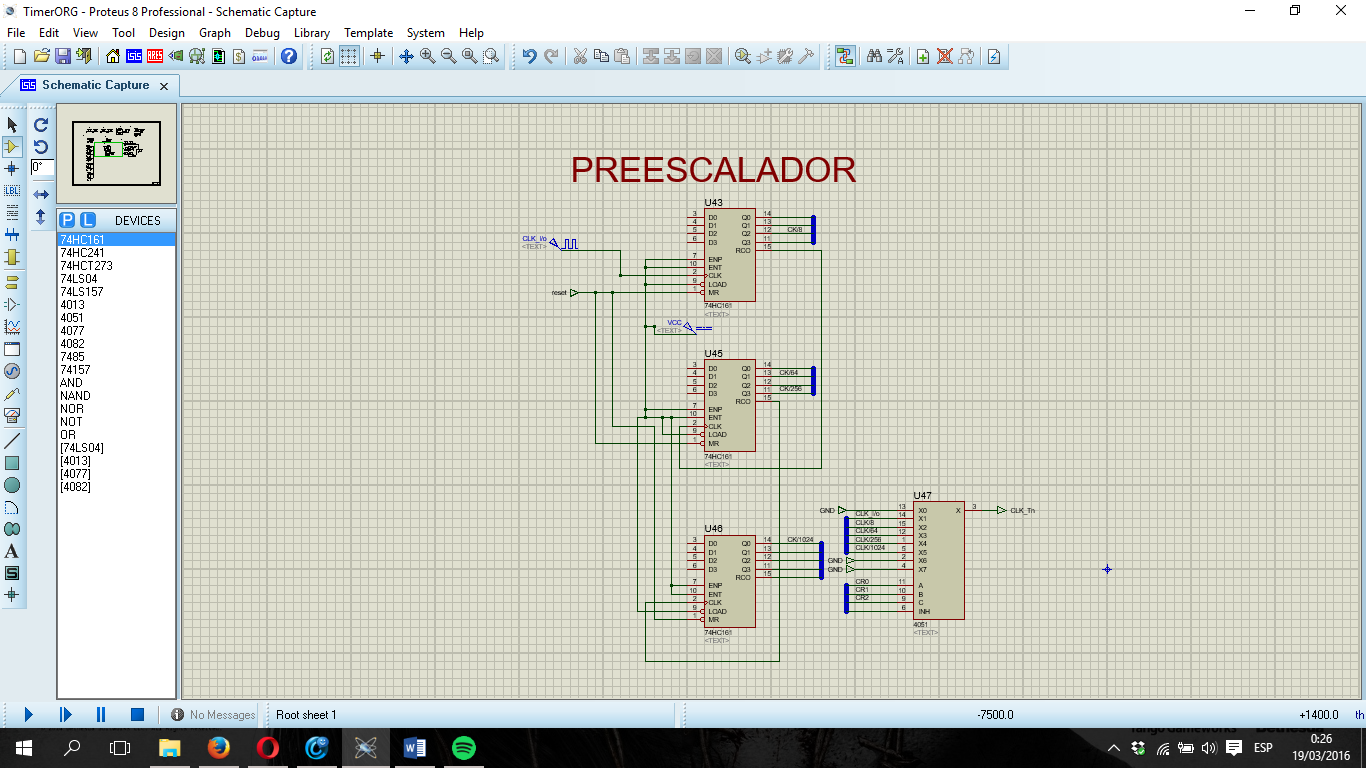
**Fig. 21** Montaje de comparadores

El generador de forma de onda toma los datos de TOP y BOTTOM y los lleva a un multiplexor cuyo modo de operación está definido por los valores de COM00, COM01 y COM02 encendiendo y apagando dependiendo de los valores el multiplexor, creando una forma de onda cuadrada llamada DOC0. El generador de forma de onda tiene la siguiente estructura:



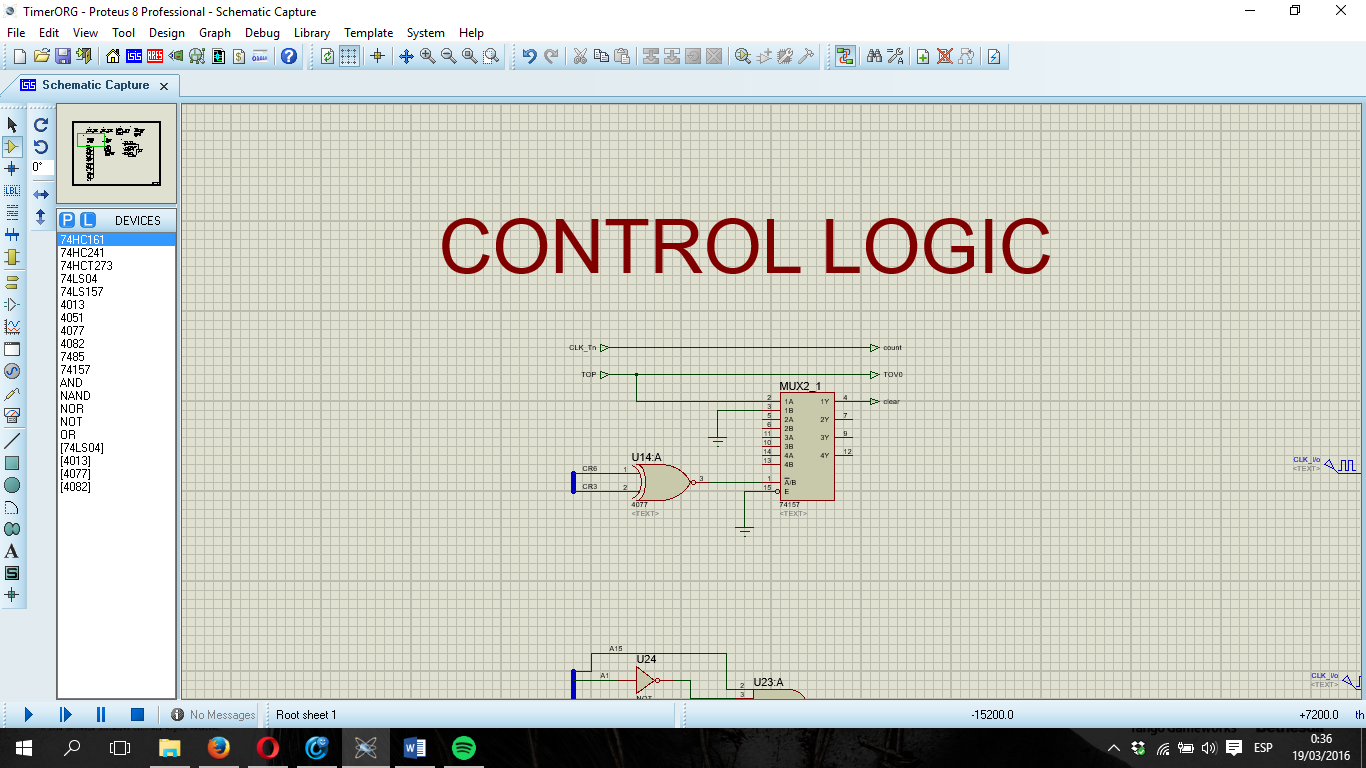
**Fig. 22** Circuito generador de forma de onda

El preescalador tiene la función de aumentar o disminuir la frecuencia del temporizador del sistema a partir de un contador y un multiplexor de 8bits cuyo modo de operación depende de CS00, CSO1 y CS02. Su estructura se observa en la figura 23.



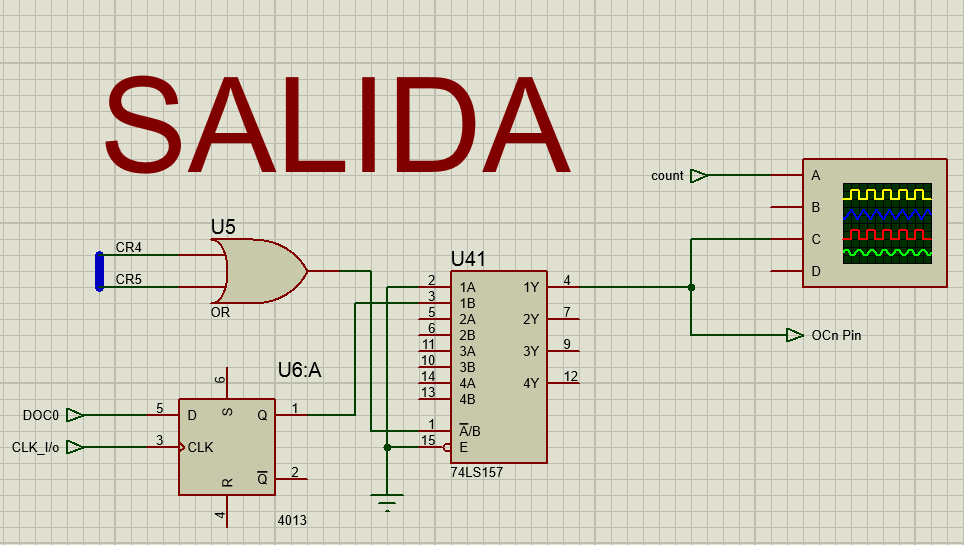
**Fig. 23** Circuito Preescalador

El bloque de control lógico tiene la función de tomar la señal dada por el preescalador y mandarla hacia la entrada TCNTn en forma de count, también convierte el valor de TOP en la salida TOV0 y genera la señal clear a partir del multiplexor y los valores que asuman CR6 y CR3. Es el bloque más importante y se puede apreciar a continuación, en la figura 24.



**Fig. 24** Circuito para la lógica de control.

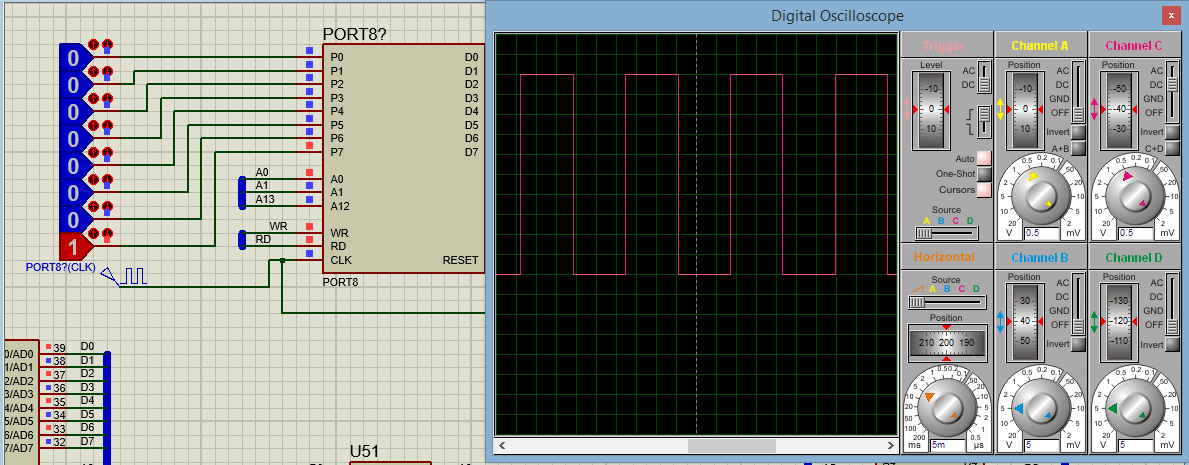
Finalmente la salida toma la señal de reloj del sistema y la salida del generador de forma de onda y los lleva a un multiplexor que es quien define la salida del puerto OCn, su modo de funcionamiento está definido por CR4 y CR5 que en ausencia de señal de ambos pone un cero en la salida y mientras que haya un valor en 1 va a dar como salida la señal PWM generada.



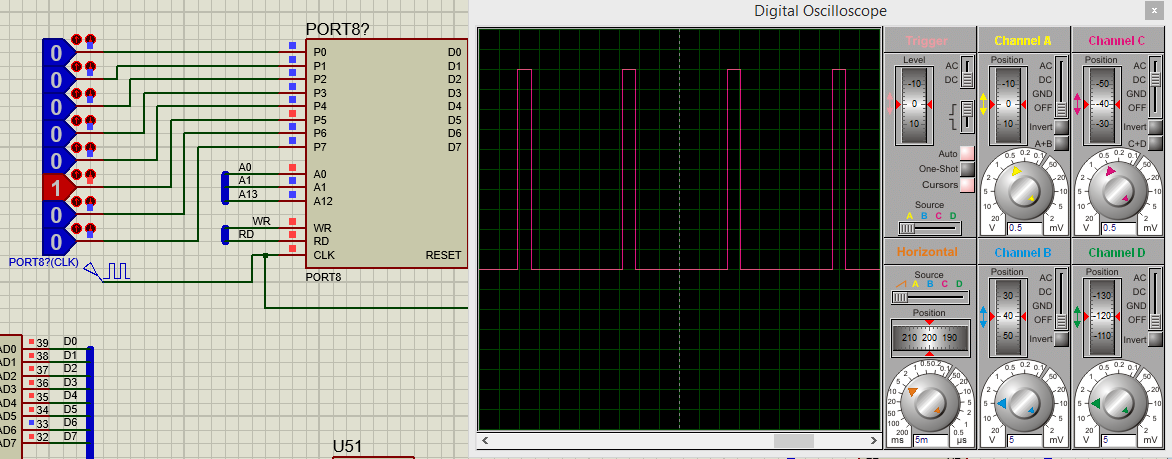
**Fig. 25** Circuito para la salida OCn

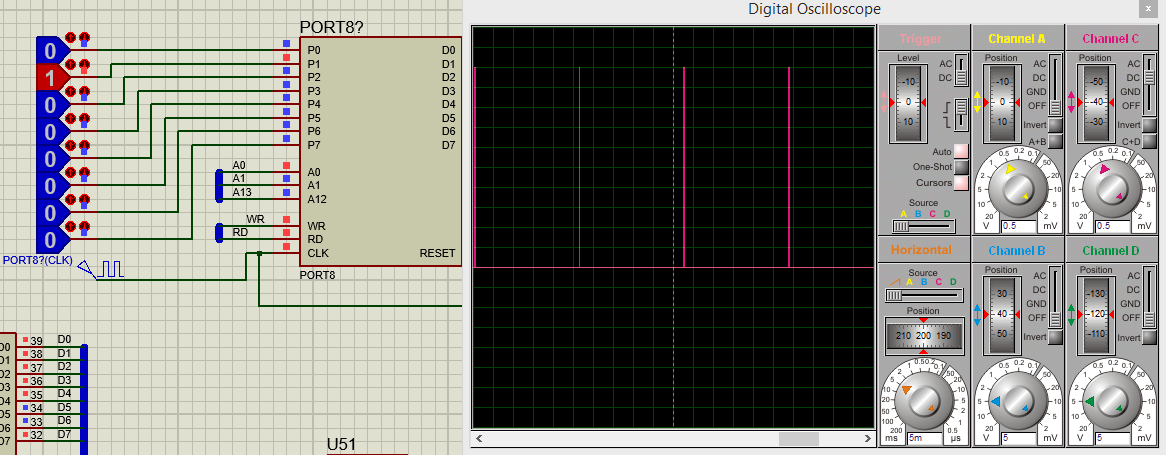
Al tener un claro entendimiento de la función que cumple cada componente descrito, anteriormente, se procede a realizar las pruebas correspondientes, sobre todo el modulo temporizador, para confirmar si este trabaja de manera correcta.

Lo que se hace es, básicamente, cargar un algoritmo a la CPU AVR que haga uso de este módulo, configurándolo para que el mismo trabaje en modo PWM, y su ancho de pulso varíe a partir de un valor que se envía, haciendo uso del puerto de entrada.



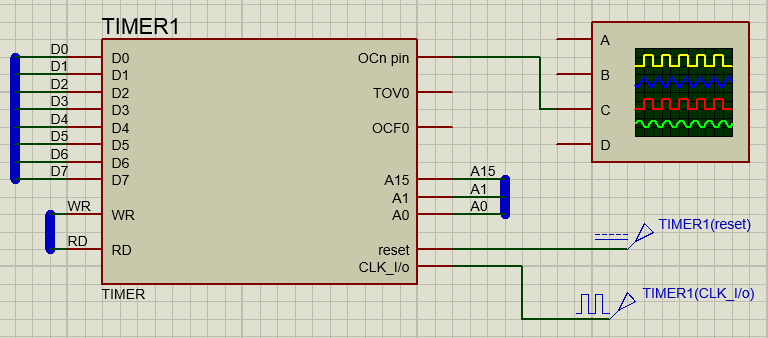
**Fig. 26** Emisión de señal PWM con el bit 7 en 1





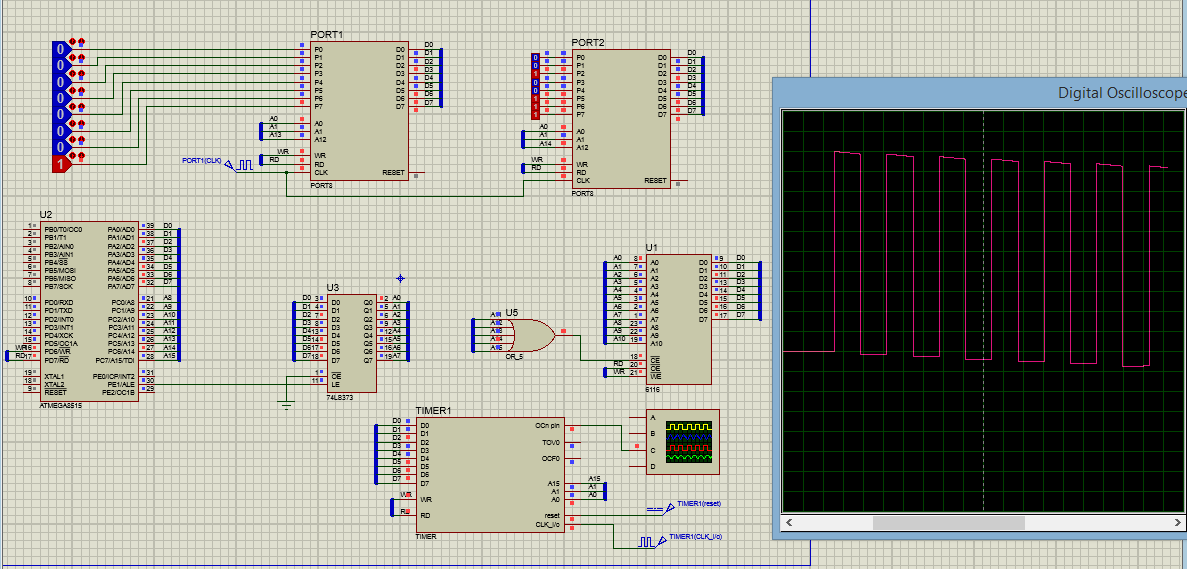
**Fig. 27** Emisión de señal PWM con el bit 5 y 1 activos

Por último se procede a realizar el encapsulado del módulo timer, en un circuito integrado, con el fin de ahorrar espacio y organizar el microcontrolador de una forma más óptima.



**Fig. 28** Integrado para el módulo timer

Posteriormente, y para dar por finalizado el proyecto, se procede a realizar la prueba del funcionamiento con el integrado observado en la figura 28, añadiendo los demás módulos del microcontrolador, como se muestra en la figura 29.



**Fig. 29** Montaje del microcontrolador con emisión de PWM

# Conclusiones

* Finalmente se logra la implementación del microcontrolador propuesto, para este proyecto, el cual ha basado su funcionamiento en el uso de una CPU AVR de 8 bits, cuya funcionalidad ha sido útil para comprobar el funcionamiento de cada módulo instaurado dentro del microcontrolador, siguiendo las instrucciones proporcionadas por la guía y observando las especificaciones dadas por el datasheet del dispositivo, para entender que configuración debe poseer cada módulo, en términos de nomenclatura, dirección, conexión y adecuaciones en general puesto que, con un conocimiento previo de estos factores, se garantiza la consecución de un dispositivo capaz de realizar operaciones básicas, de manera óptima.
* En cuanto a los módulos analizados y montados, haciendo uso del software proteus, cabe destacar que todos cumplieron con los requerimientos esperados en cuanto a su funcionamiento pues, por ejemplo, para los módulos de entrada y salida se verifica el cumplimiento de las tareas concernientes a lectura y escritura de datos, al establecer conexión entre el usuario y la máquina, y esto se ha confirmado al revisar el almacenamiento de estos, en la memoria RAM, haciendo uso del algoritmo más apropiado que permitiría la utilización y posterior revisión del funcionamiento de este módulo.
* Por otro lado ha sido posible realizar la instauración y posterior revisión del funcionamiento descrito por el temporizador de 8 bits al observar que, cuando el mismo se configuraba en el modo de señal PWM, era posible conseguir la misma, observándola mediante el osciloscopio virtual que posee proteus, y ejercer control sobre esta, en términos del acho de pulso, haciendo uso del puerto de entrada, puesto que dicho ancho depende de un valor enviado por el usuario, el cual es tomado y comparado por un contador, como se ha explicado anteriormente.

Referencias

1. A. Vera, W. Jimenez & J. Cardona “Fundamentos y Aplicaciones de los Microcontroladores AVR” 1ra edición, Armenia, 2008