[[1]](#footnote-1)

“Implementación De Un Microcontrolador Usando La Herramienta De Simulación Proteus”

Roman H. Jhoan Camilo, Morcillo O. Esneyder, Velásquez G. Juan Camilo.

*Resumen*— la importancia de saber que es un microcontrolador, que le corresponde hacer y qué elementos lo conforman ha sido una característica de estudio por parte del equipo de trabajo, se han tomado las medidas necesarias para evidenciar la verosimilitud de lo que es un microcontrolador real, evidencias tales como la correcta virtualización de un microcontrolador con sus partes más relevantes y su proceso de simulación para convalidar su autenticidad. Las partes involucradas en el proceso de virtualización son los puertos de entrada y salida, la memoria RAM y el temporizador, de los cuales se hizo una reseña para aclarar los aspectos más importantes de cada uno.

*Palabras clave*— memoria RAM, Simulación, Virtualización, Microcontrolador, Temporizador, Puerto E/S, Registros, Bus de datos.

# INTRODUCCIÓN

L

os microcontroladores forman parte fundamental del diario vivir desde el teléfono móvil hasta las sofisticadas computadoras, día a día se han roto las fronteras que limitaban la creación de objetos que en una época no muy lejana se creían no más que ciencia ficción o una falacia teórica. La ruptura de estas fronteras ha podido permitir la adecuación de dispositivos muy rápidos, más eficaces y mucho más pequeños lo que ha logrado en su mayor plenitud desarrollar una premisa que es facilitarle la vida al hombre, permitiéndole así sentir más su entorno y ser un ser más fructífero a nivel laboral, económico, social, cultural, etc.

Para lograr todo lo anterior se tiene que estudiar a fondo la raíz de las cosas, el componente principal de todo el sistema es el microcontrolador y este a su vez para funcionar correctamente tiene unos componentes que posibilitan su uso entre ellos destacan la memoria RAM que es una ‘matriz de almacenamiento’, el timer que es un temporizador y por supuesto el medio de comunicación que son los puertos, todo este sistema complementa una única funcionalidad. En el laboratorio se ha podido implementar virtualmente la elaboración de estos 3 componentes y se ha evidenciado a través de pruebas y lecturas el papel crucial que cumplen para dar ese funcionamiento a lo que se denominó como microcontrolador.

# OBJETIVOS

OBJETIVO GENERAL

Implementar mediante simulación en el software proteus un microcontrolador de 8 bits AVR de Atmel

.

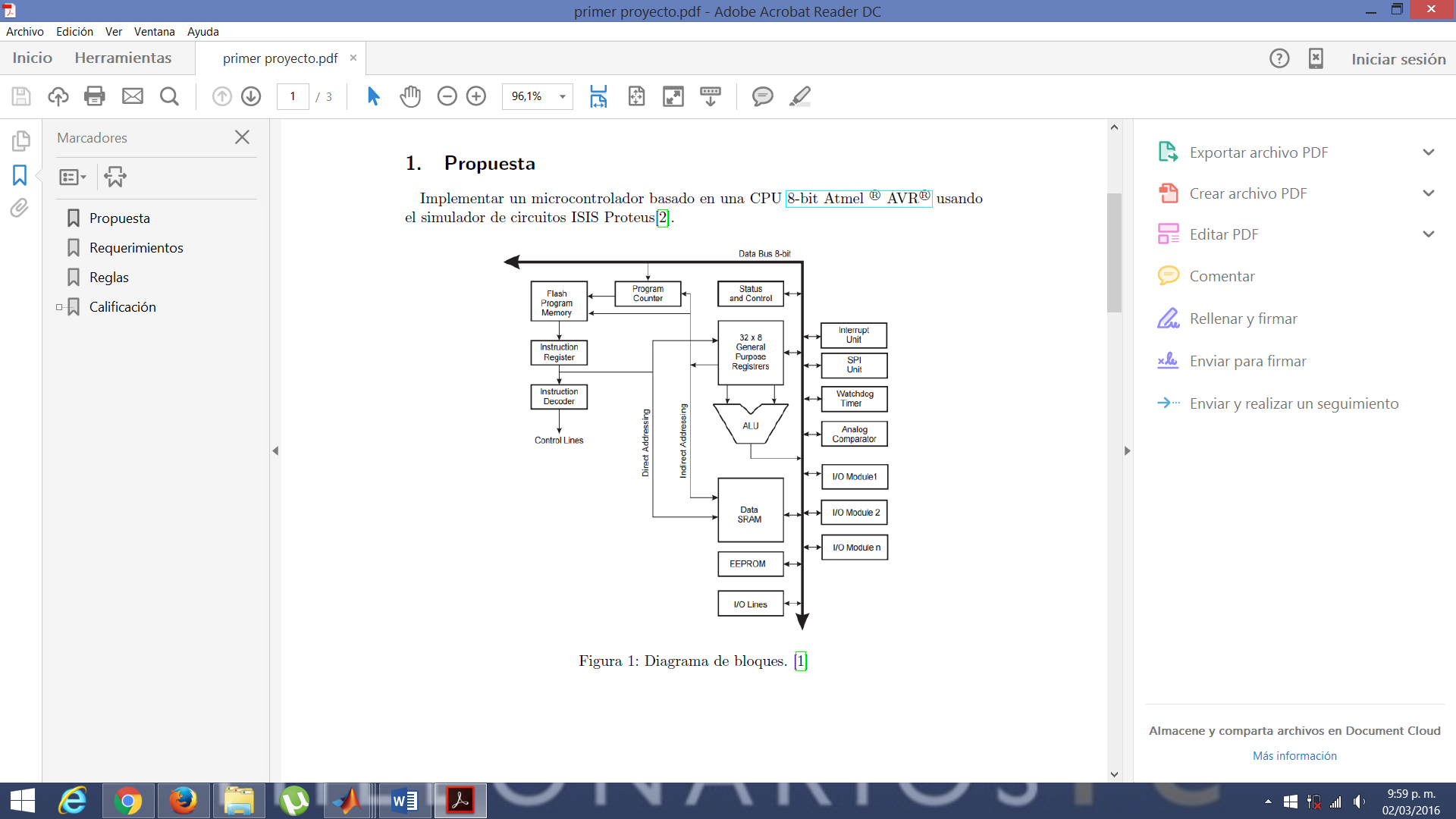
OBJETIVOS ESPECÍFICOS

* Utilizar circuitos específicos e implementar un circuito integrado que corresponde al puerto de entrada/salida.
* Utilizar un circuito integrado definido en el proteus para realizar la conexión de una memoria RAM.
* Mediante los distintos tipos de registros que contiene el timer y sus respectivos circuitos de funcionalidad, implementar un circuito integrado que los contenga y se utilice como temporizador del microcontrolador
* Interconectar todos los elementos mediante bus o cable ordenado para afianzar el correcto funcionamiento del microcontrolador

# MARCO TEÓRICO

Un microcontrolador (abreviado μC, UC o MCU) es un circuito integrado programable, capaz de ejecutar las órdenes grabadas en su memoria. Está compuesto de varios bloques funcionales, los cuales cumplen una tarea específica. Un microcontrolador incluye en su interior las tres principales unidades funcionales de una computadora: unidad central de procesamiento, memoria y periféricos de entrada/salida.

El microcontrolador en general precisa del siguiente diagrama de bloques:



En este diagrama se denota las partes primordiales que contiene el microcontrolador, pero yendo más a fondo se incluirá las definiciones de las características a implementar.

CPU

La unidad central de procesamiento o unidad de procesamiento central (conocida por las siglas CPU, del inglés: central processing unit), es el hardware dentro de una computadora u otros dispositivos programables, que interpreta las instrucciones de un programa informático mediante la realización de las operaciones básicas aritméticas, lógicas y de entrada/salida del sistema. El término en sí mismo y su acrónimo han estado en uso en la industria de la Informática por lo menos desde el principio de los años 1960. La forma, el diseño de CPU y la implementación de las CPU ha cambiado drásticamente desde los primeros ejemplos, pero su operación fundamental sigue siendo la misma.

Memoria RAM

La memoria de acceso aleatorio (Random Access Memory, RAM) se utiliza como memoria de trabajo de computadoras para el sistema operativo, los programas y la mayor parte del software.

En la RAM se cargan todas las instrucciones que ejecuta la unidad central de procesamiento (procesador) y otras unidades del computador.

Se denominan «de acceso aleatorio» porque se puede leer o escribir en una posición de memoria con un tiempo de espera igual para cualquier posición, no siendo necesario seguir un orden para acceder (acceso secuencial) a la información de la manera más rápida posible.

Puerto E/S

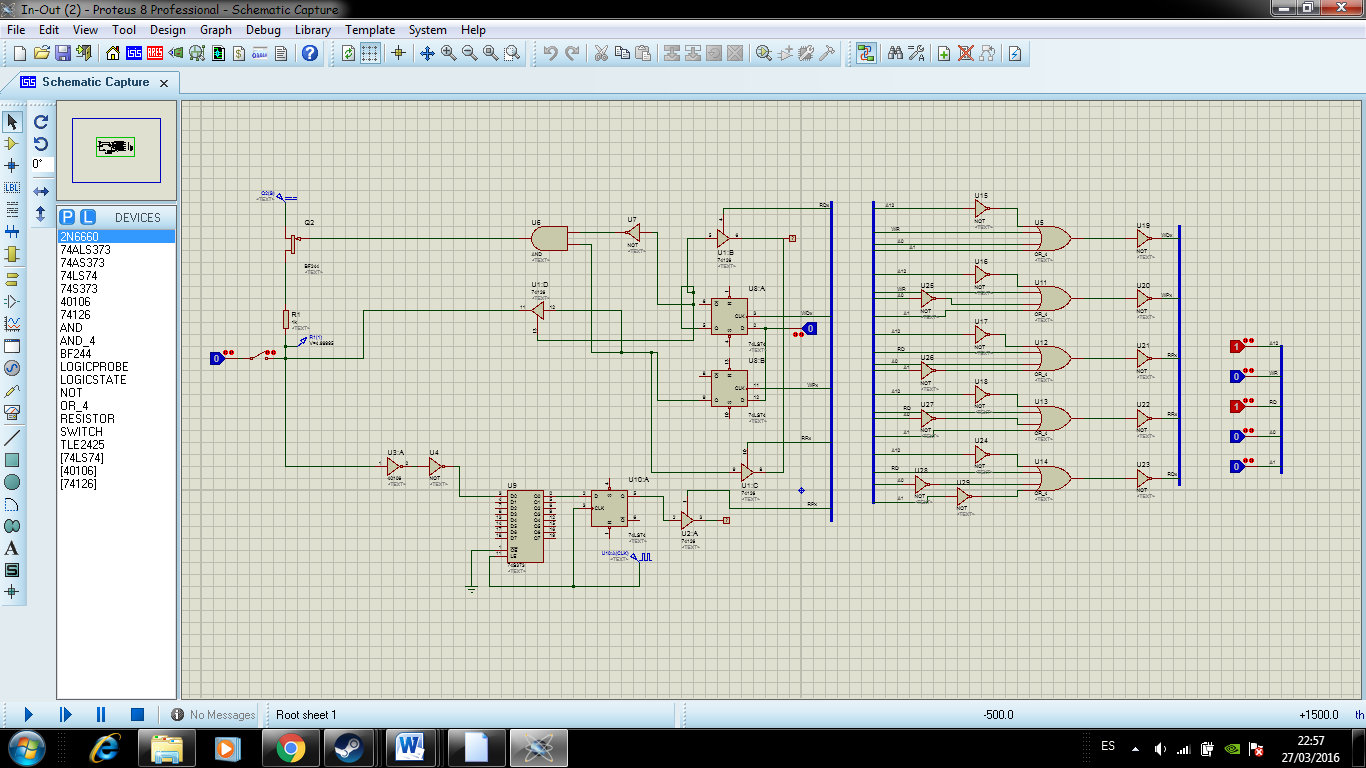
un periférico de Entrada/Salida (E/S), en inglés: Input/Output (I/O), es un dispositivo que permite la comunicación entre un sistema de procesamiento de información, tal como la computadora y el mundo exterior, y posiblemente un humano u otro sistema de procesamiento de información.

Los periféricos de Entrada/Salida son utilizados por una persona (o sistema) para comunicarse con computadoras.

# PROCEDIMIENTO

El circuito general se dividió en 3 partes: el puerto, la memoria RAM y el timer.

Para el puerto se implementó 8 veces el siguiente circuito:



*Fig.1 circuito referenciado a un pin*

En este circuito se tuvieron en cuenta los parámetros preestablecidos en la clase dando lugar al correcto funcionamiento del pin (la fig.1 aparte de ser el circuito del pin también fue la prueba del mismo), este circuito tiene unas partes que son definidas algunas de estas son:

Un triestate que actúa como un switch haciendo que este se comporte como entrada o salida

El Schmitt Trigger que tiene como función principal rectificar la señal conservando el nivel lógico

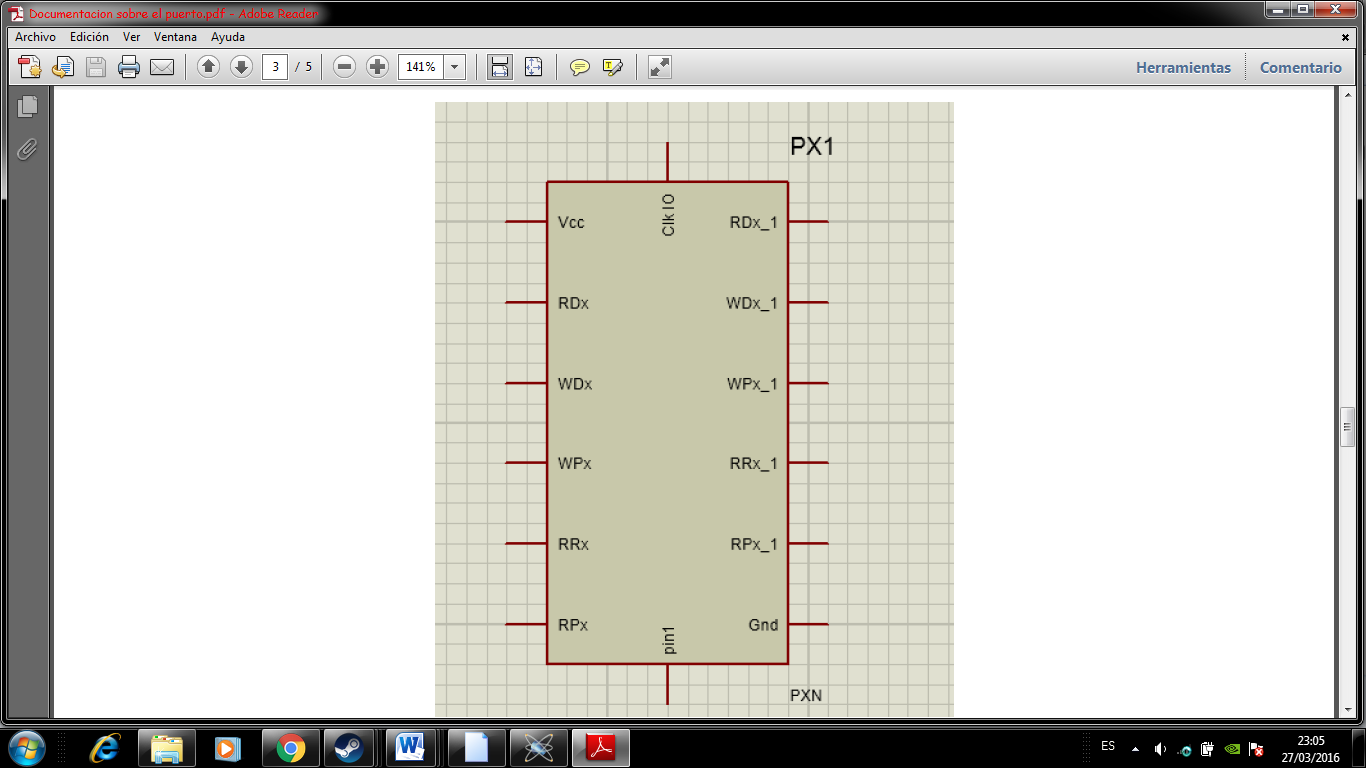
Estos pines RRx, RPx, RDx, WDx y WPx, controlan el manejo de los registros dando lugar a una lectura o escritura

Pxn es el pin resultante el cual será bidireccional, manejando la lectura o escritura dependiendo de su configuración.

Los flip flops tienen como objetivo pasar cada dato recibido en cada ciclo de reloj siendo manejado el susodicho por el microcontrolador

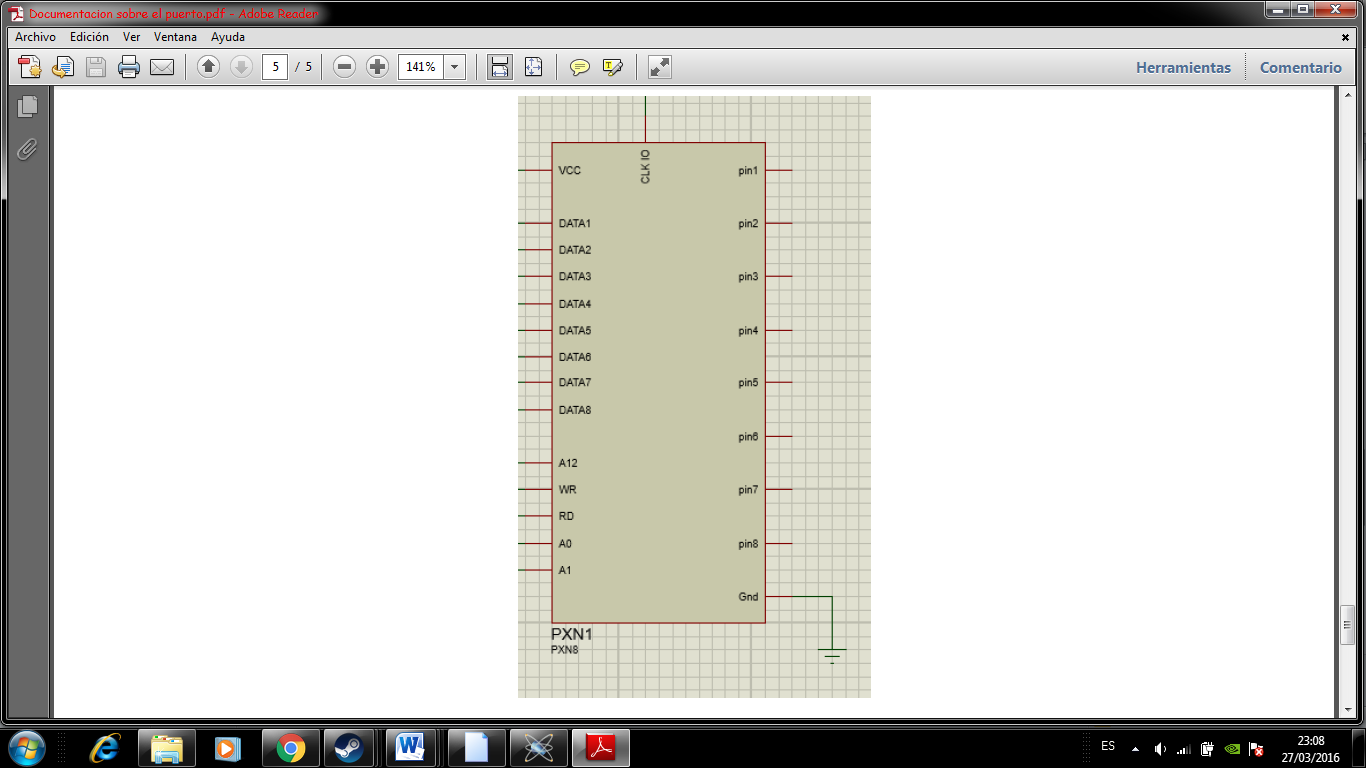
El Latch y el flip flop son independientes al procesador pues su señal de reloj no es controlada por este.

Ya sabiendo más acerca del uso que se le da a cada pin se procedió a montar un integrado general (fig.2)



*Fig.2 integrado de un solo pin*

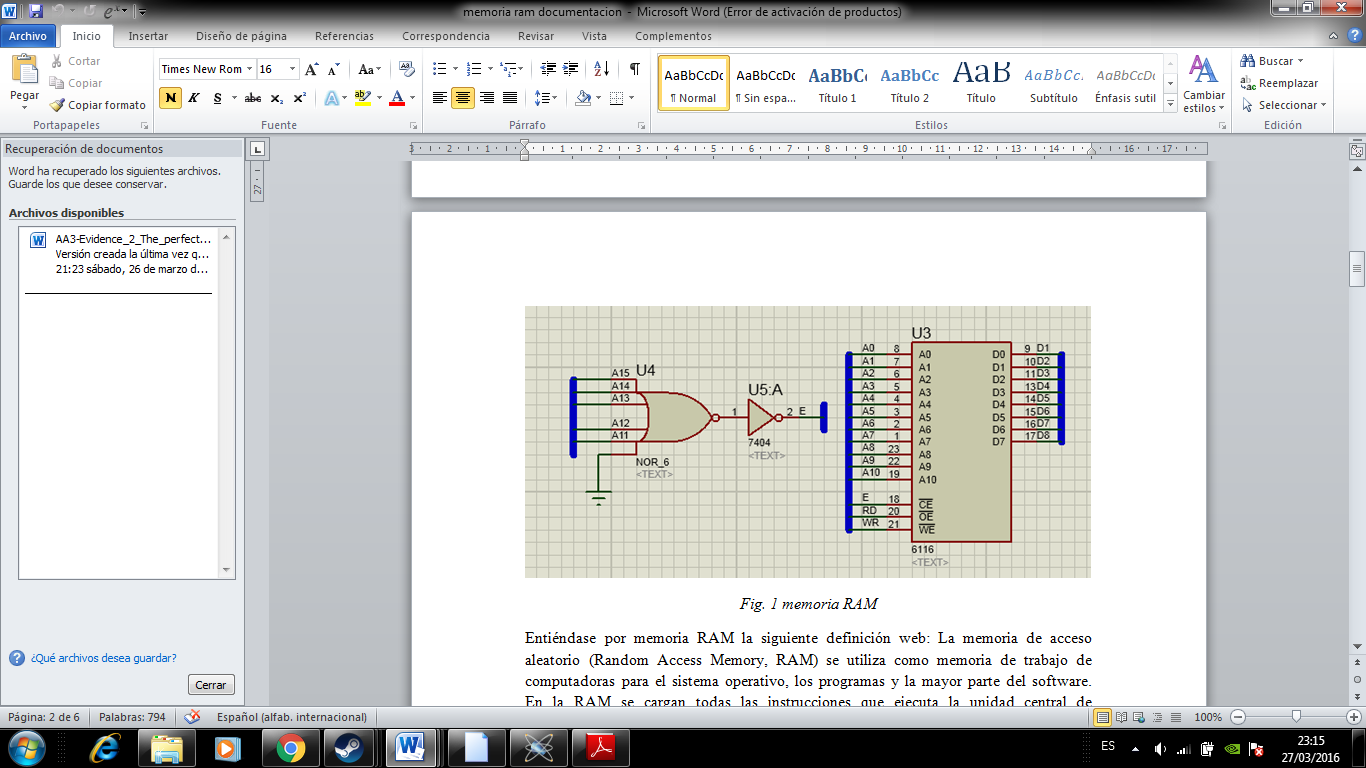
Cuando se tiene bien diseñado el circuito de la fig.1 se procedió a dar lugar a un integrado (véase fig.2) el cual fue multiplicado 8 veces para volverlo a integrar y sacar lo que se conoció como puerto el cual quedo de la siguiente manera:



*Fig.3 puerto digital*

En la fig.3 se distingue la creación del integrado ya listo y denominado como puerto los datas al ser bidireccionales pueden ser entradas o salidas, los pines comparten esta misma configuración.

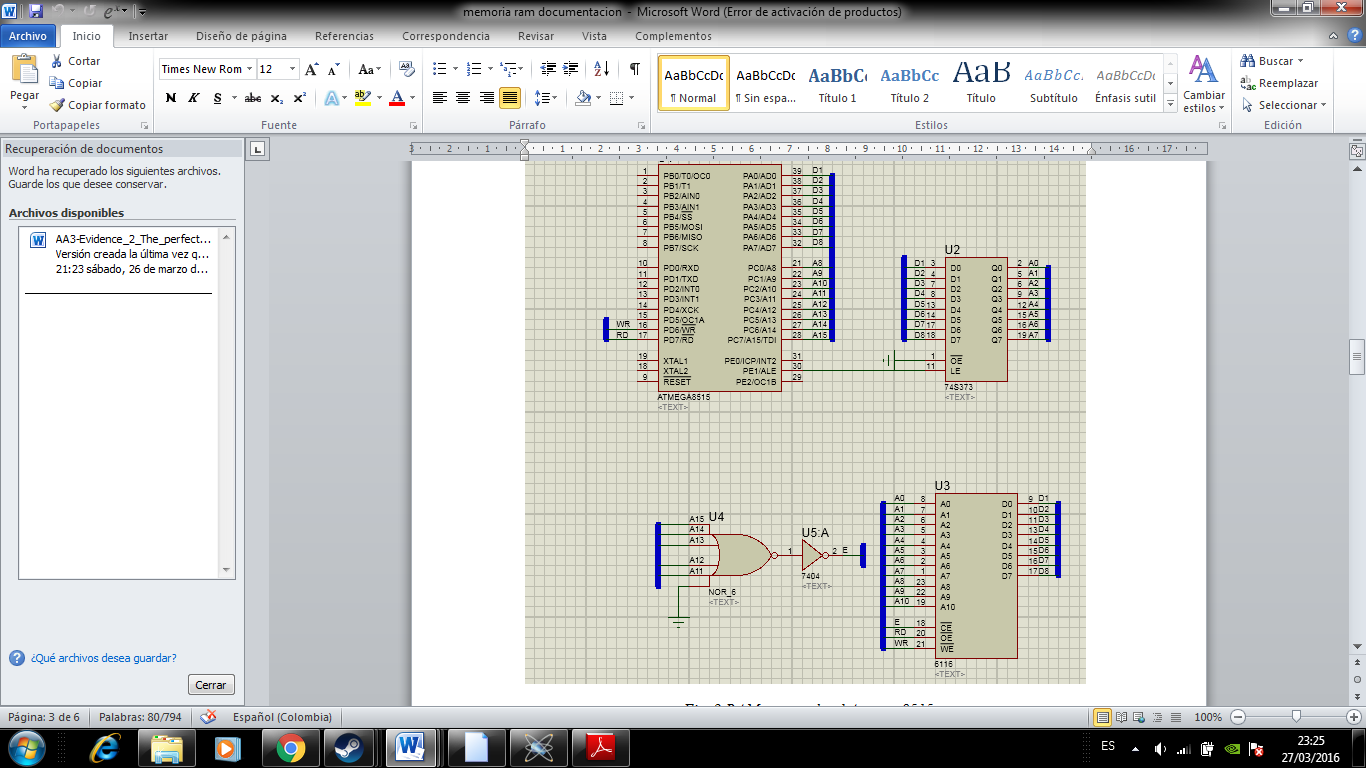
Para la memoria RAM se usó un integrado que ya venía incluido en el software proteus, este tiene como referencia 6116, para la interconexión se usaron 11 direcciones que diferían de las usadas en el puerto (0x2000 y 0x4000), se escogieron 11 porque la RAM almacena de 0 a 10, por esto se añadió una compuerta NOR para que la RAM solo funcione cuando se le indique, esta operación es realizada por CE el cual se activa en 1 y modifica el uso (lectura o escritura).



*Fig.4 memoria RAM*

El circuito que está en la parte izquierda de la fig.4 representa el circuito de operación que es encargado de evaluar los comandos del Atmega para verificar y determinar el correcto funcionamiento de la memoria para saber si esta procesa o no procesa la información, la compuerta está determinada por una NOR de 6 entradas de las cuales se usan 5 (una de ellas es enviada a tierra), estas entradas son el conjunto acogido desde la A11 hasta la A15. Para funcionar debe salir al final del circuito un 1 lógico.

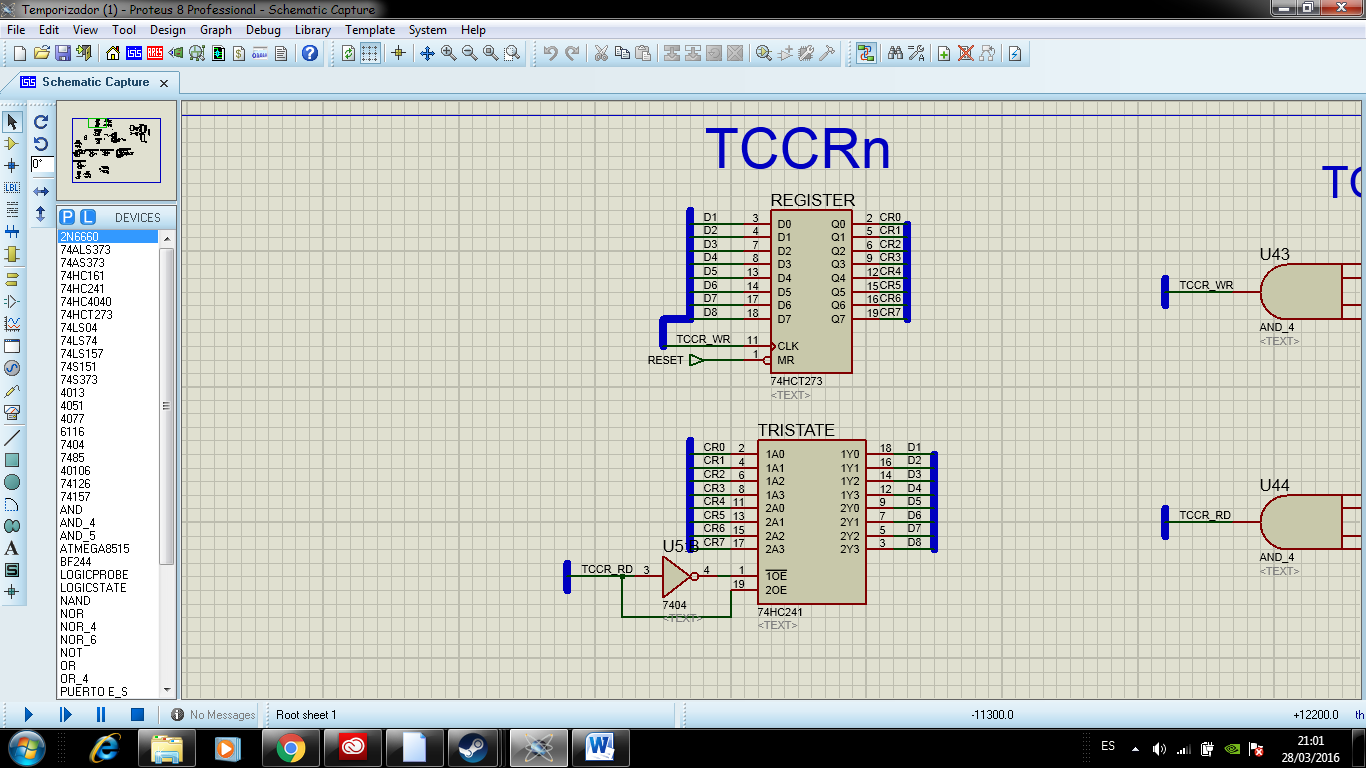
La memoria RAM al final cuando es conectada al Atmega queda de la siguiente forma:



*Fig.5 memoria conectada al Atmega.*

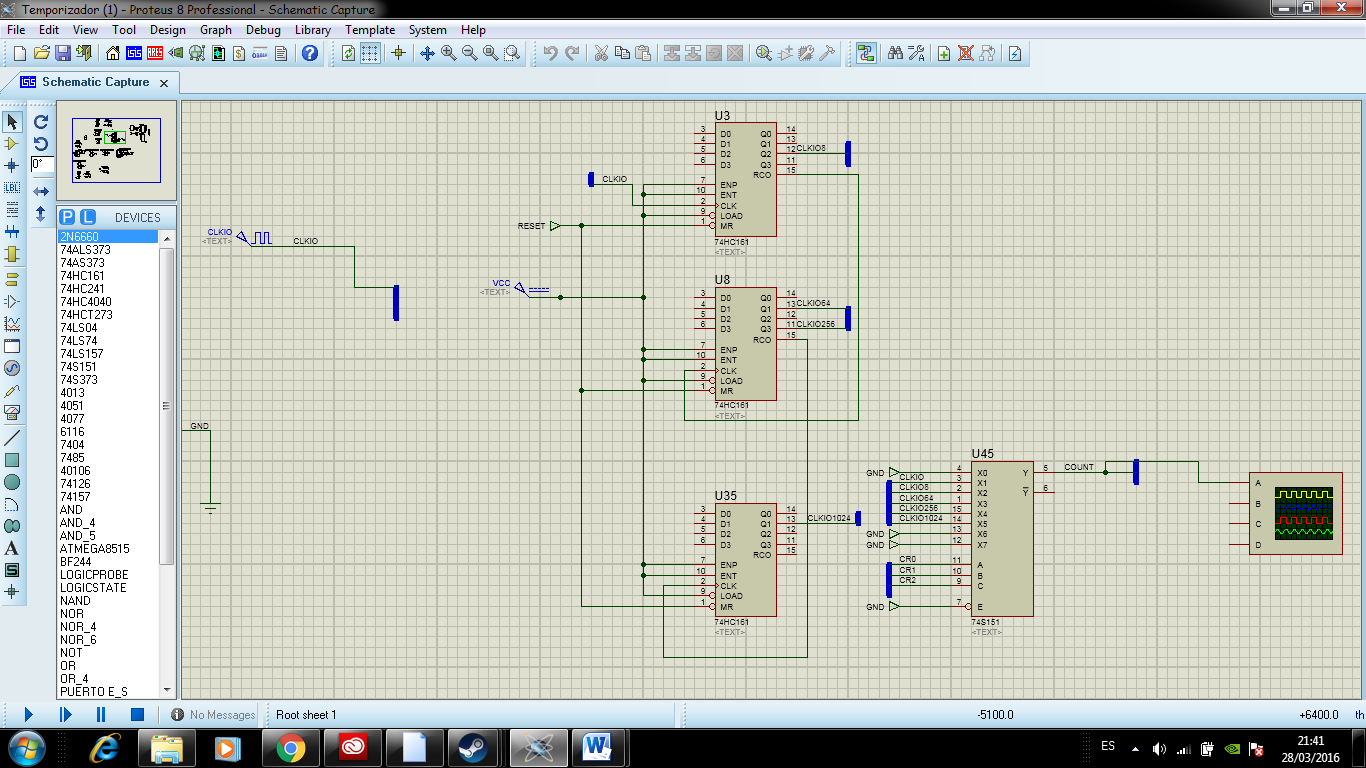
Esto sería lo que se hizo en la memoria RAM y ahora se explicara cómo se hizo el timer el cual contenía varios registros

Para empezar se tiene el registro que proporciona los registros OCR, este registro es el encargado de implementar el tipo de operación que se dará lugar en el timer, además de dar el manejo propicio del pwm y de controlar el tipo de prescalado, este circuito maneja 6 registros distintos y tiene un 7 que no es de utilidad para el timer, en la fig.6 se puede ver el circuito equivalente a el TCCRn, el cual cumple su función a cabalidad y proporciona los 6 valores usables de los 7.



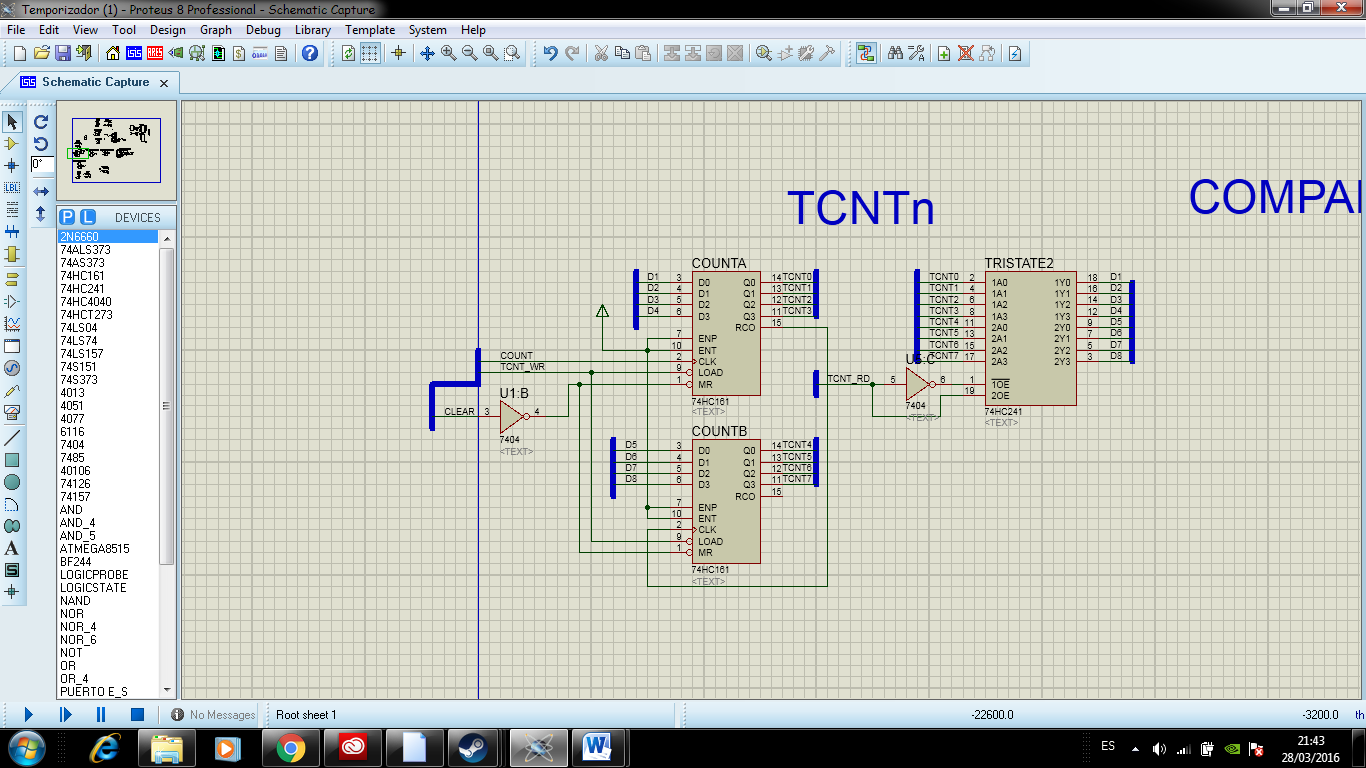
*Fig.6 TCCRn*

El timer en su diagrama de bloques precisaba un preescalador, que este a su vez tiene la función de dividir la frecuencia 2^n (2 elevado a la n) dependiendo del valor que salía del OCR, este circuito se muestra en la fig.7 dada a continuación:



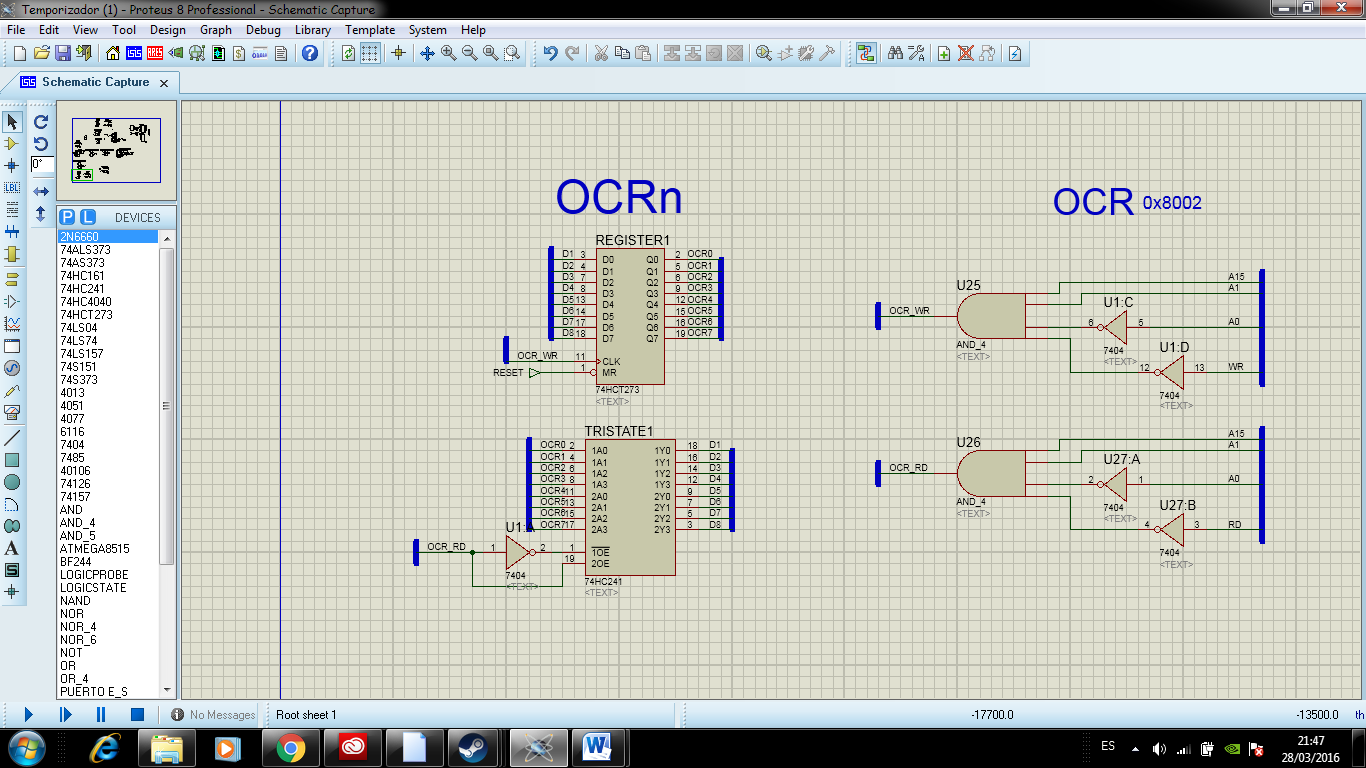
*Fig.7 preescalador*

El TCNTn contiene los datos actuales del temporizador haciendo un registro de ellos, este funciona con un contador, el circuito es evidenciado en la fig.8.



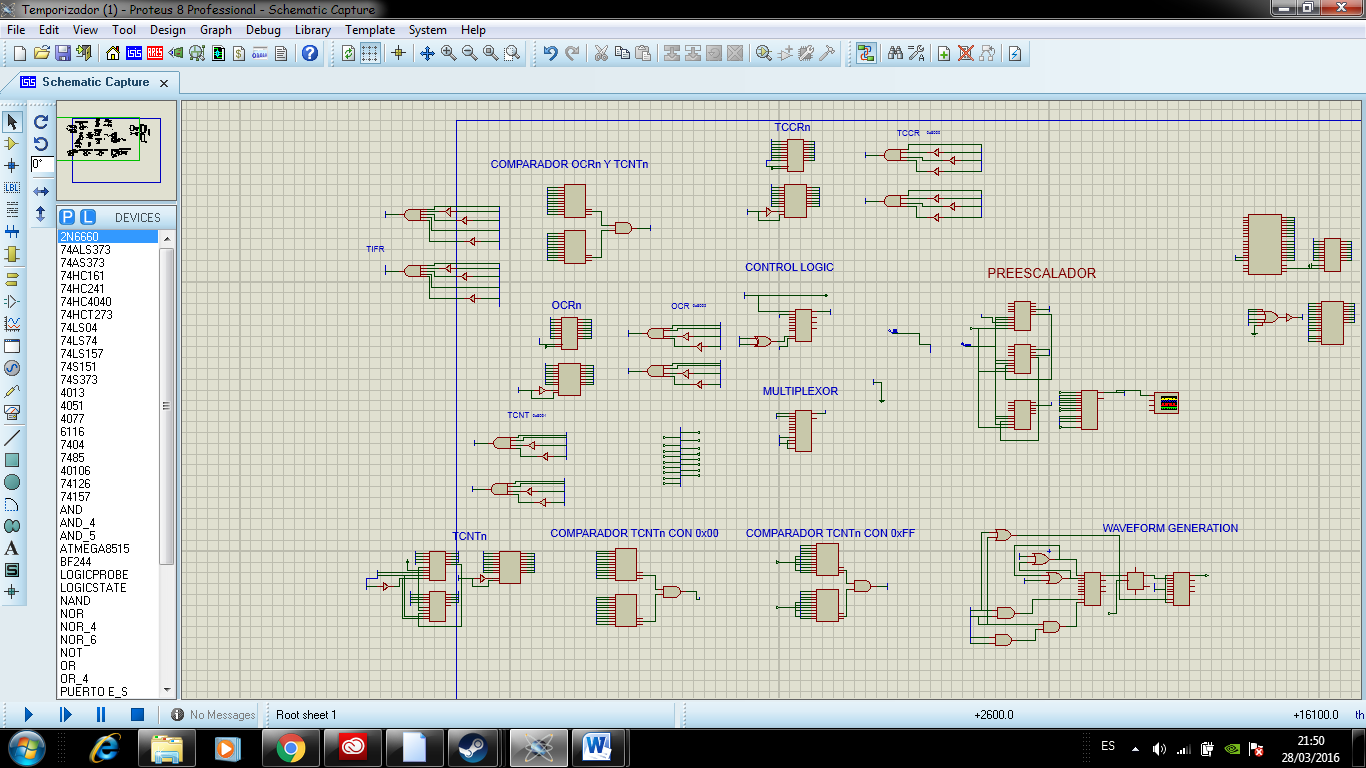
*Fig.8 TCNTn*

El OCRn Tiene como función principal determinar el ancho de pulso mediante comparaciones con valores registrados provenientes del temporizador, el circuito que manipula este registro es el de la fig.9.



*Fig.9 OCRn*

Como tal esos fueron los registros más importantes a la hora de realizar el timer pero como tal el timer preciso de comparadores, control lógico, generador de ondas y alimentaciones. La gráfica del circuito general se encuentra en la fig.10



*Fig.10 timer completo*

Para comprobar el funcionamiento del timer se llevaron a cabo una serie de pruebas con el código proporcionado por el profesor del horario teórico, el código es el siguiente:

#include "procesador.h"

#include "map.h"

int main(){

uint8\_t pulse;

CPUInit();

CPUWrite(DDR\_A, 0);

CPUWrite(DDR\_B, 0xFF);

CPUWrite(TCCR\_0, 0b01101101);

while(1){

pulse = CPURead(PIN\_A);

CPUWrite(OCR\_0, pulse);

CPUWrite(PORT\_B, CPURead(TCNT\_0));

\_delay\_ms(30);

}

}

Este código lo que hace es configurar los registros del TCCR habilitando una serie de ejecuciones.

# CONCLUSIONES

* Se adquirió el conocimiento básico del comportamiento de un microcontrolador cuando es sometido a diferentes lineamientos de código en lenguaje de bajo nivel.
* Se aprendió sobre las partes fundamentales que componen un sistema complejo llamado microcontrolador y como a través de estas hay comunicación con el entorno.
* Se concibió la forma de montar un sistema complejo con circuitos relativamente admisibles teniendo claro el concepto preestablecido que rige la idoneidad de cada componente.
* Se comprendió y se aprendió los diferentes puntos teóricos que encaminan una completa comprensión de los registros circundantes al timer y de cómo estos funcionan y se complementan
* Mediante comprobación lógica y montaje grafico se expuso claridad con respecto a los 3 componentes virtualizados en el proyecto (puerto E/S, memoria RAM y Temporizador)
* Se asimilo de forma completa lo exuberante que puede ser un microcontrolador a nivel aplicativo y de cómo este mejora la calidad de vida del hombre.

# REFERENCIAS

[1] Datasheet AVR 8 bits, Atmel, [online] disponible en: <http://www.atmel.com/images/atmel-2486-8-bit-avr-microcontroller-atmega8_l_datasheet.pdf>

[2] ATMEL AVR, Anónimo, [online] disponible en: <https://aprendiendo-microcontroladores.wikispaces.com/ATMEL+AVR>

[3] Definición de Memoria RAM, Anónimo, [online] disponible en: <http://www.definicionabc.com/tecnologia/memoria-ram.php>

[4] Periférico de entrada/salida, Anónimo, [online] disponible en: <https://es.wikipedia.org/wiki/Perif%C3%A9rico_de_entrada/salida>

1. [↑](#footnote-ref-1)