**MEMORIA RAM**

**ESNEYDER MORCILLO OCAMPO**

**JHOAN CAMILO ROMAN HURTADO**

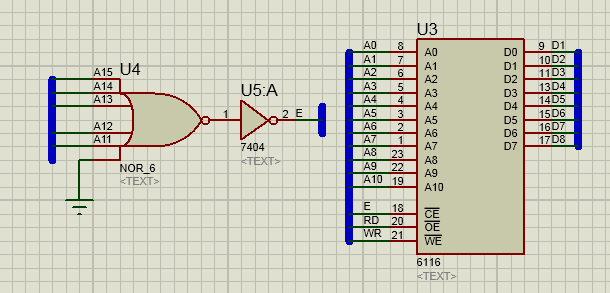
**JUAN CAMILO VELASQUEZ**

**UNIVERSIDAD DEL QUINDIO**

**FACULTAD DE INGENIERIA**

**INGENIERIA ELECTRONICA**

**2016**



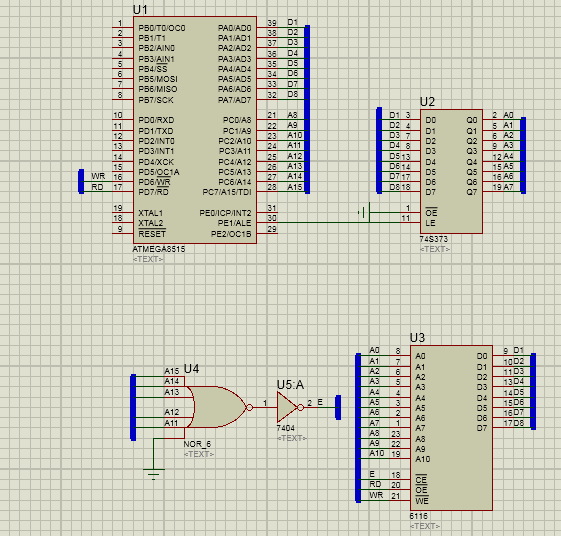
*Fig. 1 memoria RAM*

Entiéndase por memoria RAM la siguiente definición web: 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.

Teniendo el concepto de ‘RAM’ claro procedemos a enfatizar en el circuito propuesto realizado que se encuentra en la fig.1. El circuito anterior es la conexión de un circuito que permite detectar las señales del microcontrolador y la memoria RAM como tal, el funcionamiento se basa de la siguiente forma: para acceder a las diferentes posiciones de almacenamiento no hay que ser rigurosos en el orden puesto que esta memoria es comprendida como una matriz en donde cada posición puede guardar un bit. Teniendo en cuenta lo anterior la ubicación de un bit específico está determinado según la posición (fila o columna) donde se encuentre; postergando el dispositivo que haga uso de esta memoria reconoce cada dirección de matriz mediante números, a lo cual fue llamado dirección de memoria; con esta se puede inferir en donde está la posición del bit almacenado.

El circuito implementado fue entrelazado con el microcontrolador y quedo evidenciado en la siguiente figura:



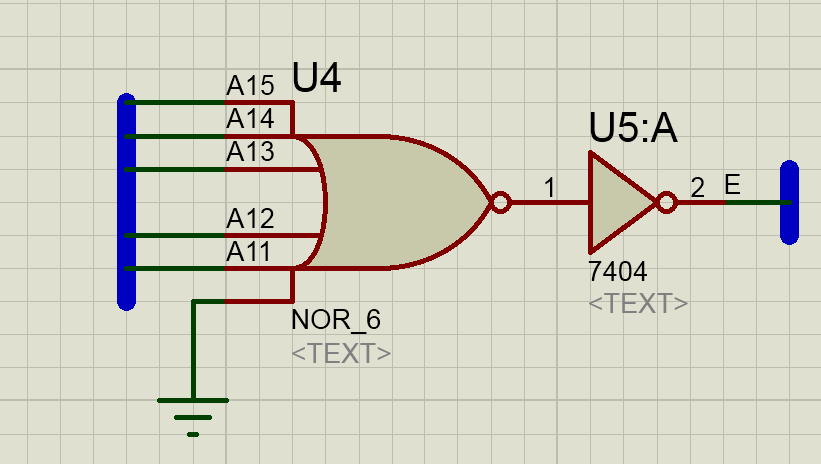
*Fig. 2 RAM conectada al Atmega 8515*

Para la interconexión de la memoria RAM al Atmega 8515 se manejaron 11 direcciones de memoria que son las comprendidas desde el A0 hasta el A10, las direcciones que se excluyen son la 0x2000 y la 0x4000 con las respectivas subdirecciones.

El cumplimiento de CE que se activa en 1, es el de activar una de las 11 direcciones preestablecidas, para que congenie con el microcontrolador, para lograr la activación se hace uso de un circuito aparte que es el de la fig. 3, el cual se explicara cómo funciona más abajo.

El OE y el WE son proporcionados por el Atmega y sus alter ego son el RD y el WR que son para lectura y escritura respectivamente.

La memoria RAM funciona siempre que está activa, cuando es apagada o desactivada borra todo tipo de almacenamiento en su interior.



*Fig. 3 compuerta NOR que maneja CE*

La fig. 3 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.

Finalizando, se probó la memoria usando un código en c el cual va a definir el funcionamiento del circuito de acuerdo al valor que se ingrese por el puerto de entrada además de verificar la correcta conexión y el correcto funcionamiento de la memoria RAM, el código fue el siguiente:

#include "procesador.h"

#include "ports.h"

void secuencias(uint8\_t sel);

int main(){

uint8\_t sel;

CPUInit();

CPUWrite(0x100, 0x01);

CPUWrite(0x201, 0x80);

CPUWrite(DDR\_A, 0);

CPUWrite(DDR\_B, 0xFF);

while(1){

sel = CPURead(PIN\_A);

secuencias(sel);

\_delay\_ms(5);

}

}

void secuencias(uint8\_t sel)

{

uint8\_t value;

switch(sel)

{

case 1:

value = CPURead(0x100);

if(value == 0)

value = 1;

CPUWrite(PORT\_B, value);

value <<= 1;

CPUWrite(0x100,value);

break;

case 2:

value = CPURead(0x201);

if(value == 0)

value = 0x80;

CPUWrite(PORT\_B, value);

value >>= 1;

CPUWrite(0x201,value);

break;

case 4:

value = CPURead(PORT\_B);

value++;

CPUWrite(PORT\_B, value);

break;

case 8:

value = CPURead(PORT\_B);

value--;

CPUWrite(PORT\_B, value);

break;

default:

value = CPURead(PORT\_B);

value ^= 0xFF;

CPUWrite(PORT\_B, value);

break;

}

}

Este código es el que se ha venido trabajando y tiene una breve modificación para reflejar el funcionamiento de la memoria RAM mediante unos módulos que se ingresan al puerto, entones lo que hace el código es que cuando yo ingrese un 1 al puerto este debe mostrarme en la memoria RAM una secuencia que es la siguiente:

0000000**1**

000000**1**0

00000**1**00

0000**1**000

000**1**0000

00**1**00000

0**1**000000

**1**0000000

Y cuando yo ingrese un 2 la secuencia prevista anteriormente debe devolverse en sentido contrario denotando la veracidad y la coherencia del funcionamiento de la memoria RAM

Cuando se compila el código se obtiene un archivo .hex que es el que se carga en el procesador para dar el funcionamiento general del sistema.