FUUNCIONAMIENTO MICROCONTROLADOR AVR

Arcos Cerón Jorge Alberto, Arcos Araujo Brayan

Hemos realizado la implementación de un Puerto de 8 bits, cuyo funcionamiento está a cargo de una configuración entre un Procesador y un registro Latch, cuyo procesador rige sus acciones de acuerdo a unas líneas de código que el usuario le proporciona, de acuerdo a sus necesidades.

Nuestro Objetivo es diseñar dichas líneas de código para que el procesador junto con el registro Latch, configuren nuestro Puerto y obtengamos de él, la respuesta esperada.

Partiendo de nuestro Puerto conectado a la configuración de Procesador y registro Latch (ver Figura 1).

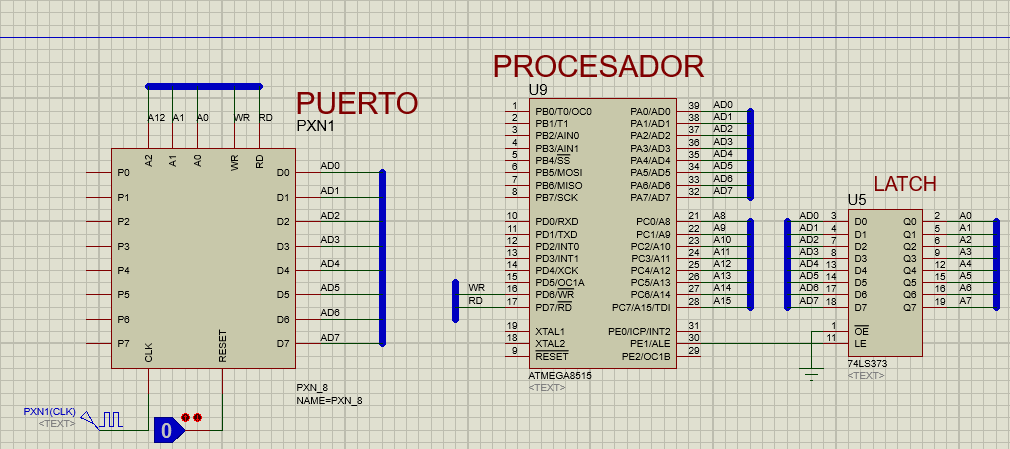


Figura 1, Puerto + Procesador

Creamos un arreglo que consta de Dos de nuestros Puertos conectados simultáneamente al mismo Procesador y registro Latch (ver Figura 2).

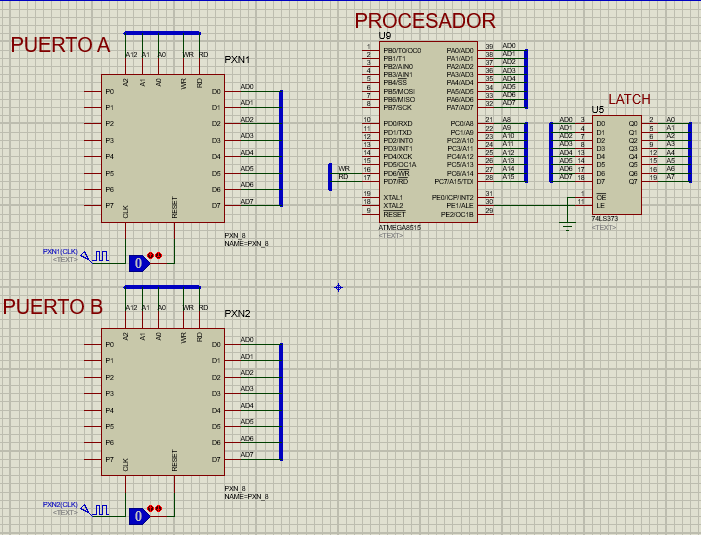


Figura 2, Arreglo de Dos Puertos A,B

Para el cual Implementamos unas líneas de código, usado la herramienta CodeBlocks, con el fin de configurar el arreglo de Dos de nuestros Puertos, de tal manera que uno de ellos sea nuestra entrada de datos (PUERTO A) y el otro nuestra salida de datos (PUERTO B).(Ver Figura 3).

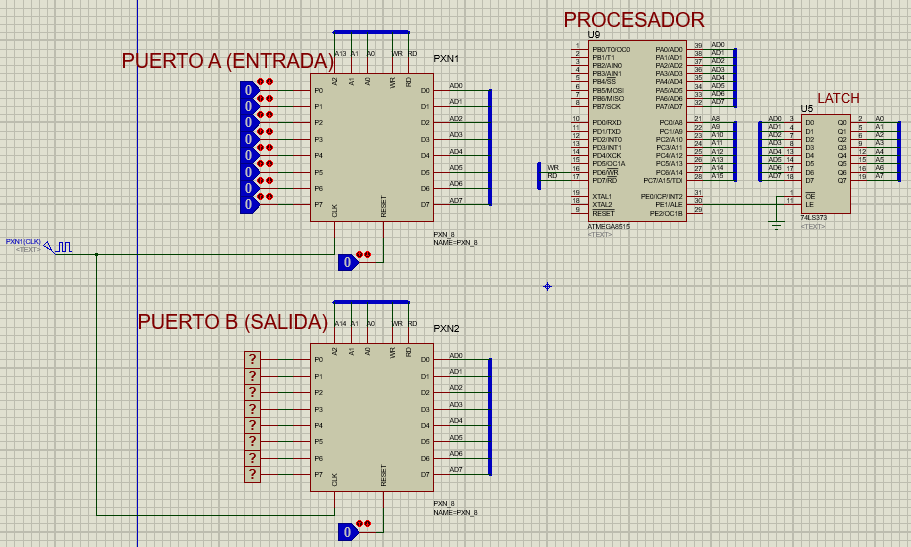


Figura 3, Configuración Arreglo

El funcionamiento de nuestro codigo, consiste principalmente en diferentes comportamientos para nuestro puerto de salida B, de acuerdo con lo que se ingrese al puerto A.

* En nuestro main, configuramos el Puerto A como entrada, colocando Ceros (0) en el registro DDR correspondiente. Y el Puerto B como salida al colocar Unos (1) en su registro DDR.

Ademas leemos por el Registro PIN del Puerto A el dato ingresado y lo almacenamos en un vector de 8 bits denominado ‘sel’, el que enviamos a la funcion ‘secuencias’.(Ver Figura 4).

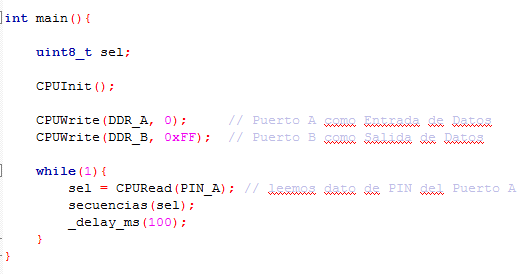


Figura 4, Configuracion Puertos A,B

* Nuestra funcion ‘secuencias’ resibe el dato Ingresado al Puerto A y lo engresa a un proceso de comparacion .

Para el Caso 1 (si se ingresa en Puerto A un ‘10000000’)

La funcion lee el valor existente en el puerto B y lo almacena en un vector ‘value’ de 8 bits, al que compara. Si este valor es Diferente de Uno (‘00000001’), asigna a ‘value’ un Uno. Si el valor existente en el puerto B es un Uno, asigna a ‘value’ un Uno corrido una posicion hacia la izquierda (‘00000010’), lo que corresponderia a tener en el pin (P1) un 1. posteriormente escribe en el puerto B el valor de ‘value’. (Ver Figura 5).

Por lo que para este caso, miraremos un comportamiento alternado de 1 y 0 en los pines P0, P1, del Puerto B. ya que la funcion hace el recorrido indefinidamente.(Ver Figura 6).

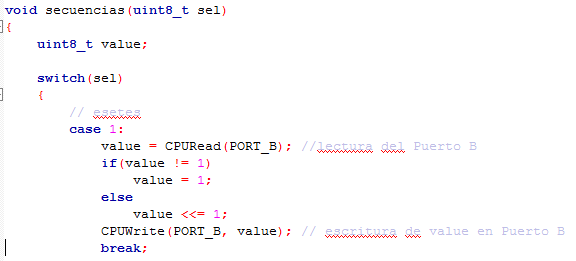


Figura 5, Codigo Caso 1

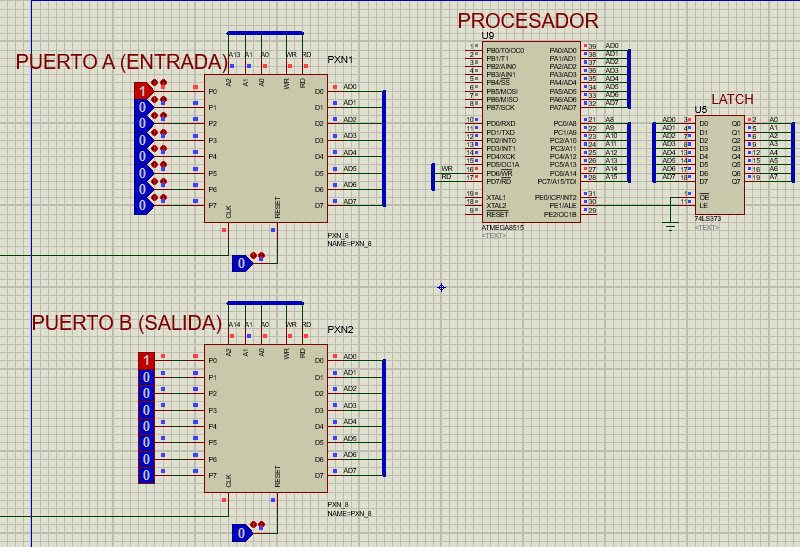
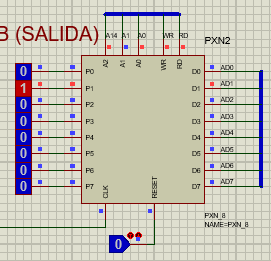
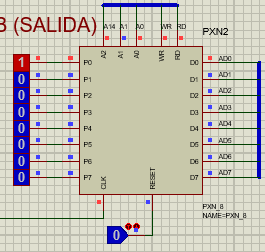


Figura 6, Comportamiento Caso 1

* Para el Caso 2, ( si se ingresa en Puerto A un ’01000000’)

Nuestra funcion compara el valor existente en la direccion (0x80) del Puerto B, con el valor de ‘value’, si el valor existente en el Puerto B es diferente al de ‘value’ la funcion refleja el valor del del Puerto B en el vector ‘value’.

Si el valor del Puerto B es igual al de ‘value’, la funcion le asigna a ‘value’ un Uno pero corrido una posicion hacia la derecha, lo que correnponderia a ‘000000001’, (ver Figura 7).

Puesto que notaremos un comportamiento similar al del caso 1 pero entre los pines P7 y P6 del Puerto B, (ver Figura 8).

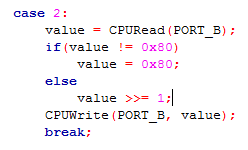


Figura 8, Codigo Caso 2.

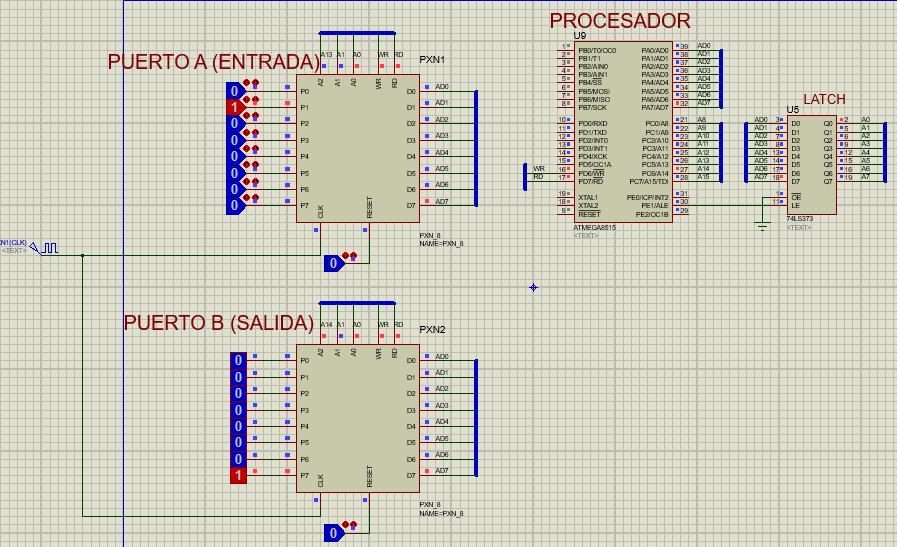
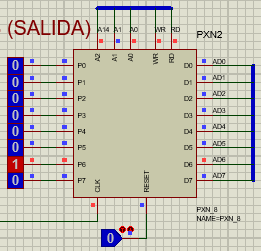
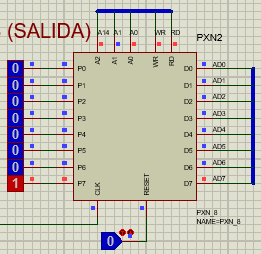


Figura 8, Comportamiento Caso 2.

* Para el Caso 4 ( si en Puerto A ingresa ‘00100000’)

Nuestra funcion ‘secuencias’ asigna al vector ‘value’ el valor que se lee en el Puerto B atrasvez del comando CPURead, Al que ingrementa progresivamente en un unidad y luego escribe mediante el comando CPUWrite en el mismo Puerto B. (ver Figura 9).

Esto sucede para cada ciclo recorrido, en el que CPUWrite, sobreescribe el dato anterior por el actual dato, en otras palabras se ejecuta una sumatoria en numeros Binarios. (ver Figura10).

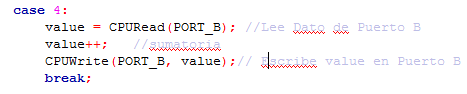
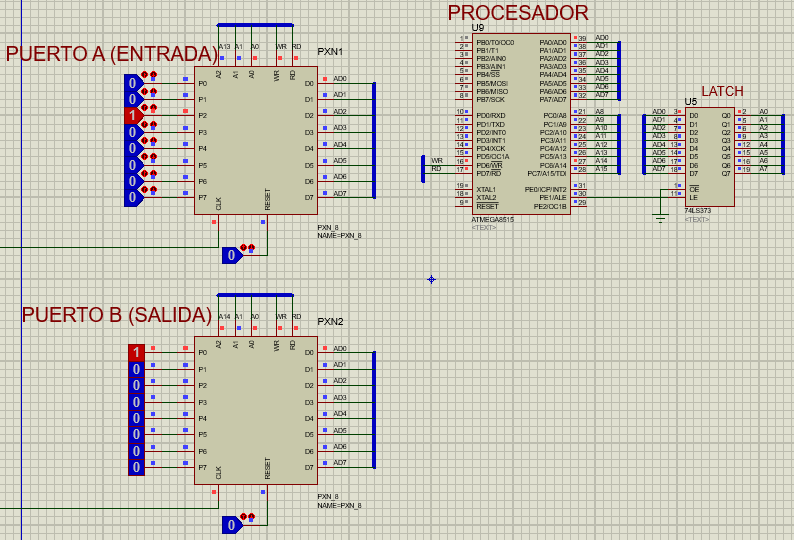
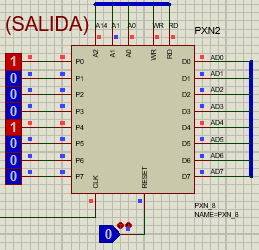
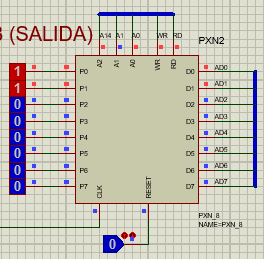


Figura 9, Codigo Caso 4.

Figura 10, Comportamiento Caso 4.

* Para el Caso 8 ( si ingresa en Puerto A un 8 (‘00001000’)

‘sencuacias’ lee el dato existente en el Puerto B y se lo asigna al vector ‘value’, al que despues le aplica un decremento unitario para posteriormente escribirlo denuevo en el Puerto B,(ver Figura 11).

Eso se hace para cada ciclo indefinidamente por lo que obtendriamos como salida en nuestro Puerto B una Resta de numeros Binarios, (ver Figura 12).

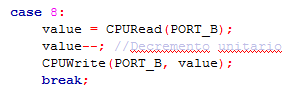


Figura 11, Codigo Caso 8

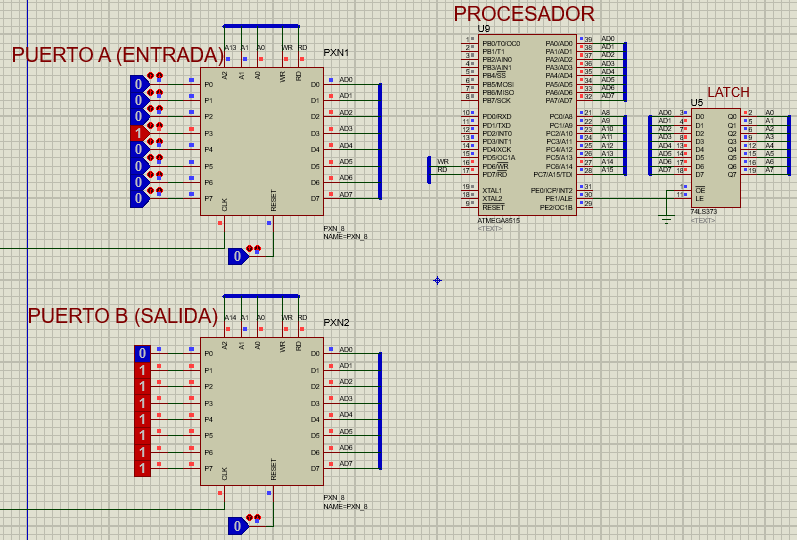


Figura 12, Comportamiento Caso 8 ( Resta )

En nuestro codigo tambien esta estupilado el comportamiento para la salida, cuando por el Puerto A ingrese un valor diferente al expuesto en los casos anterioes (1,2,4,8), a esa configuracion la denominamos Configuracion por Defecto.

Dicho comportamiento se rige por el codigo de la Figura 13.

En el que leemos el dato existente en el Puerto B y despues de almacenarlo en el vector ‘value’, este vector se somete a una operación de XOR contra un vector de igual tamaño (8) lleno de Unos (‘FF’=’11111111’) . por lo que obtendremos como resultado de esta operación un Cero (0) para cada posicion del vector ‘value’o para cada salida del Puerto B, en la que haya un Uno (1) y un Uno (1) para cada posicion del vector ‘value’ o para cada salida del Puerto B, que contenga un Cero(0).

En otras palabras cambie el valor de cada Pin de salida por su Complementario.esto para cada ciclo por lo que obtenemos como resultado una intermitencia de Unos y Ceros, según los valores iniciales del Puerto B. (ver Figura 14).

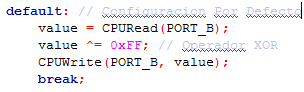
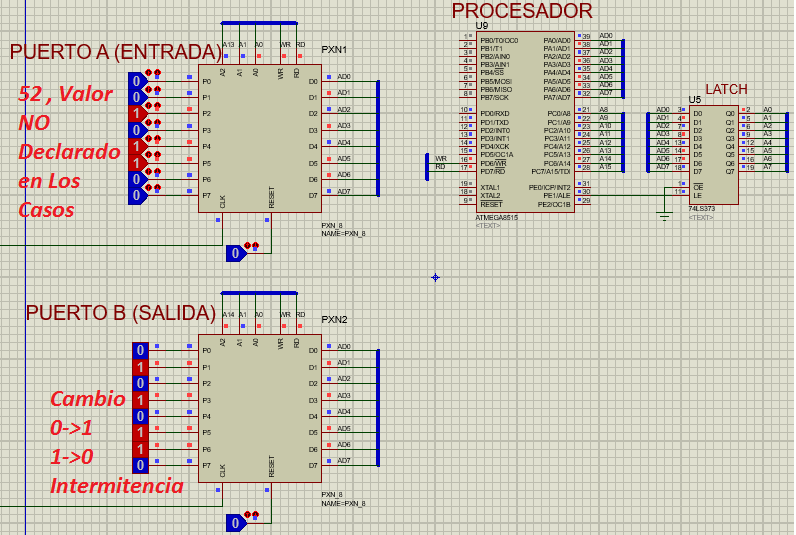
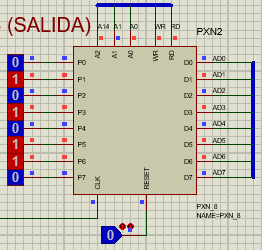
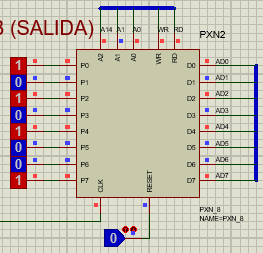
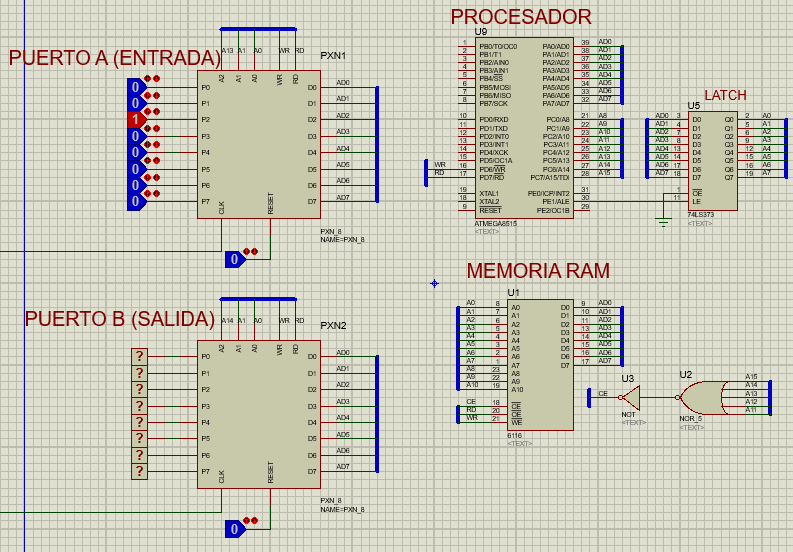


Figura 13, Configuracion Por Defecto

Figura 14, Funcionamiento Por Defecto

**INCLUSION DE LA MEMORIA RAM**

De acuerdo a los avances en el curso, logramos incluir en nuestro sistema una Memoria RAM, para mejorar en Eficacia el Funcionamiento de nuestro Procesador.



Por el momento hemos podido probar el funcionamiento de esta Memoria al acoplarla en nuestro Sistema, para ello modificamos el código anterior de tal forma que el dato que leemos del registro PIN del Puerto A, el que escribimos con ayuda del comando CPUWrite en la dirección 0x0300, que según los términos de diseño pertenece al rango de direcciones reservadas para la memoria RAM. Posteriormente y con la ayuda del comando CPURead leemos este mismo valor almacenado en la dirección 0x0300 asignarlo al vector ‘sel’ para enviarlo como en nuestro Primer Código a la función ‘secuencias’, (ver Figura 15). Que sigue los mismos principios de funcionamiento iniciales.

Con esto comprobamos la lectura y almacenamiento correctos desde y a la memoria RAM. (ver Figura 16).

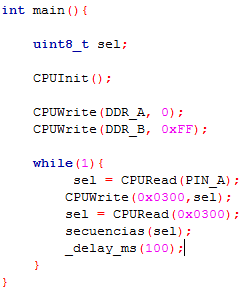


Figura 15, Código Funcionamiento Memoria RAM

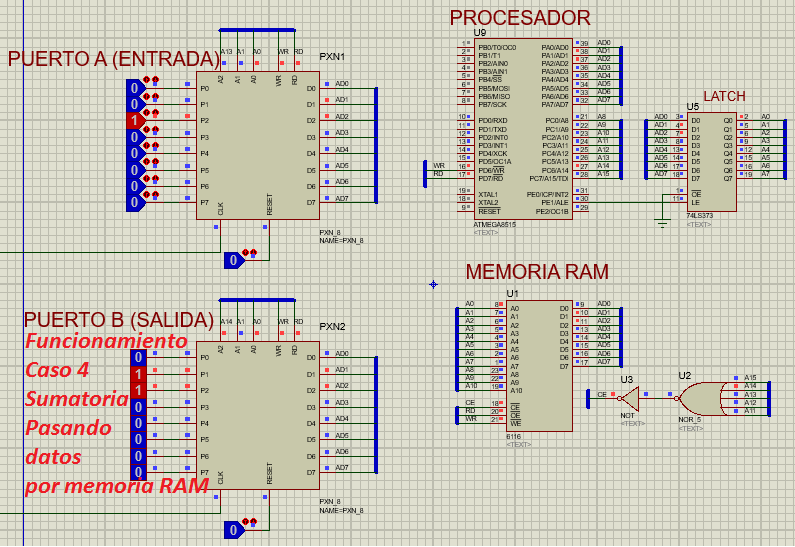


Figura 16, Funcionamiento Con Memoria RAM