Documentación N°3

Diana Marcela Álzate B, Andrés Felipe Veloz Peralta, Juan Sebastián Sánchez Vega. Programa de Ingeniería Electrónica, Facultad de Ingeniería Universidad del Quindío - Colombia

En este adelanto hizo lo siguiente:

1. Se modificaron algunas conexiones del pin y del puerto:

* En el pin se modificaron algunas conexiones del 74LS373, al igual que los nombres de algunos pines del RESET.
* En el puerto se cambió la forma de alimentación del VCC debido que estaba alimentado con un logicstate, el cual generaba algunas inconsistencias la hora de probar dicho puerto, esta alimentación fue modificada por un TERMINAL POWER con un voltaje de 5V.
* Luego cargamos el ATMEGA con el código creado por el docente Luis Miguel Capacho, el cual modificamos para verificar que el puerto creado funciona correctamente tanto de lectura como escritura.

El código usado fue el siguiente:

#include "procesador.h"

int main()

{

uint8\_t value; //se crea una variable de 8 bits llamada value

CPUInit(); // esta función inicializa las variables en los rangos establecidos y da un estado.

CPUWrite(0x2000, 0xFF);//Es una función creada para escribir la información existente en 0x2000 en la dirección de memoria 0xFF

while(1) // ciclo que se repite siempre y cuando no pase de 1

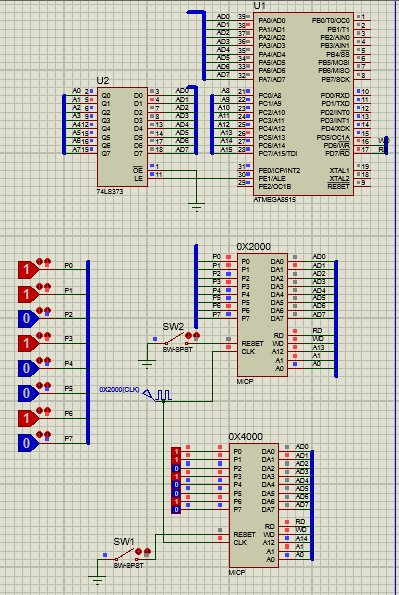
{

value = CPURead(0x2002); // Llama la función CPURead para leer la información existente en la dirección 0x2002, retorna cada valor del bit y la almacena en la variable VALUE.

CPUWrite(0x4001, value); // Se llama la function CPUWrite para escribir cada uno de los valores de VALUE en la direccion 0x4000.

}

}



**Imagen N°1 prueba de los puertos con el código de programación probarAtmega**

1. Luego creamos la configuración con la memoria RAM para verificar que este circuito guarda de manera temporal los datos generados y procesados por el ATmega y los puertos a la memoria RAM. Para la verificación de este funcionamiento se utilizó el código de programación descrito a continuación.

**#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;**

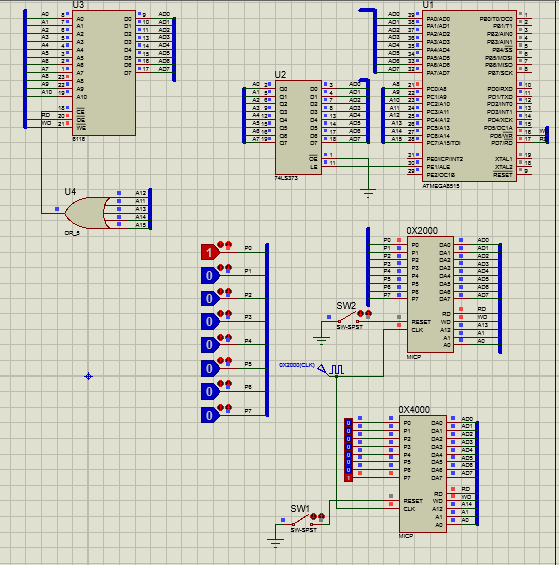
**CPUWrite(PORT\_B, value);**

**break;**

**}**

**}**

**El programa prueba el comportamiento de la RAM otorgando un valor inicial digitado por el usuario y reescribiéndolo en la salida y almacenándolo de diferentes comportamientos de la salida del atmega.**

****

**Imagen N°2 prueba de los puertos con el código de programación probarRam**