Microcontrolador con CPU AVR

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

***Resumen***— *en el presente artículo está plasmado el diseño de un microcontrolador con CPU AVR implementado en proteus, tanto en su parte física como en su funcionamiento interno por así decirlo.*

**Índice de Términos**— **Índice de Términos**— puertos, AVR, temporizador.

1. INTRODUCCION

Se construyó desde la base un microcontrolador de 8 bits el cual está basado en la arquitectura de un AVR, dicho modelaje se construyó en el simulador proteus y su sistema programado en el simulador codeblocks.

Este diseño se realizó por medio de diagramas de bloques para su comprensión y posterior verificación de cada una de las etapas que lo componen.

1. MARCO TEORICO

* Memoria para Programas: Cada programa que se desarrolla para los [AVR,](http://www.atmel.com/products/AVR/) se almacena en una región de la memoria no volátil (es decir, permanece al apagar el dispositivo), además es programable con el procedimiento de carga (flash). La primera sección de esta región es la sección de carga (flash) de la aplicación y es donde se almacena el programa que se escribe para el AVR. La segunda sección se llama: 'Boot Flash Section', o sección de carga del inicio y se puede configurar para que funcione una vez que el dispositivo (sistema ensamblado), se prende o se enciende.

La sección de carga del inicio, es muy útil si se programa con una aplicación que toma datos del serial y los pasa a la región de memoria de aplicación. Esta aplicación se denomina*'bootloader'* y permite que el micro-controlador pueda ser programado desde un puerto serial común y corriente, en lugar de utilizar circuitos de programación costosos y complejos. [1]

* Código hexadecimal: Todo el código de programación que se desarrolla y se escribe, se liga, se ensambla y compila a código hexadecimal. Este código es una serie de números hexadecimales, que son interpretados como instrucciones por el micro-controlador. La ventaja de trabajar con lo [AVR,](http://www.atmel.com/products/AVR/) es que se pueden programar utilizando el lenguaje de programación **'C'**. [1]
* Temporizador (reloj): Microcontroladores como los AVR, utilizan un reloj con pulsos a intervalos constantes y con un paso regular. Una variedad de diferentes tipos de relojes y temporizadores, además de velocidades, están disponibles para cada micro-controlador. Para aplicaciones en tiempo-real, velocidades entre 14.746CPS (ciclos por segundo o Hertz), y 16MHz., son suficientes para generar operaciones y procesar instrucciones. Los AVRtambién incluyen un circuito o aditamentos para conectar un cristal oscilador que regula la velocidad del procesador. Nótese que la velocidad es mucho menor a la de un computador personal, por lo que es importante tener en cuenta al momento de programar el microcontrolador. Cada micro-controlador se diferencian en el número de pulsos de reloj o en otros términos, en la cantidad de pulsos (ticks), que toma para ejecutar una instrucción. Por esto en los AVR se utiliza el término de 'MIPS' o número de millones de instrucciones por segundo. Una de las cualidades de los AVR, es que cada instrucción toma un ciclo de reloj y por lo tanto corre alrededor de (1.0 MIPS/MHz.). Otros micro-controladores corren a (16MHz.), y por ende a menos MIPS. [1]
* Memoria para datos y almacenamiento de información: Parte de la memoria de los AVR es volátil, del tipo RAM y está organizada en registros de 8-bits.
* Registros: Toda la información en el micro-controlador, desde la memoria de programas, la información del temporizador, hasta el estado de los pines en los puertos de entrada y salida, se almacena en registros de memoria. Los registros son como cajones en un gabinete. Se usan cajones que pueden guardar por ejemplo ocho (8) tarjetas y en donde cada tarjeta almacena un número binario de un bit (i.e. un cero (0) o un uno (1)). Cada cajón posee una dirección asignada para poder ser encontrado por el micro-controlador.
* Algunos registros, como por ejemplo los de RAM, se utilizan para almacenar datos en general. Otros tienen funciones específicas para controlar los convertidores análogo-digitales, contabilizar duraciones en tiempo o para asignar o conseguir valores de los pines de entrada. Los registros en RAM pueden ser leídos o escritos. Otros registros pueden ser para acceder o leer únicamente.
* Bits y bytes: Un byte, se compone de 8-bits con 256 valores solamente. Toda la información en el micro-controladores almacenada en pedazos de tamaño de un byte. Para facilitar la comprensión y operación con números binarios en el formato de unos y ceros, cada byte de información se representa con un número hexadecimal de dos dígitos. Por ejemplo el número (1111 0011) en binario, equivale a (243) en decimales y a ( F3 ) en hexadecimal. Es costumbre adicionar el prefijo '0x' para indicar que el número está en base 16 (hexadecimal), es decir (0xF3). Las direcciones de lugares en la memoria normalmente se dan en hexadecimal pero con el prefijo '$' para indicar que son el número de una dirección en lugar de un valor (e.g. $03DF). [1]
* Registros de Entrada y Salida: Para recibir o mandar (leer o escribir) a los pines de entrada y salida del micro-controlador, se necesita conocer un poco más sobre la arquitectura y diseño en este tipo de dispositivos. La razón por la que 32 pines de entrada y salida en un ATMega, están divididos en cuatro (4) puertos cada uno de ocho (8) pines, es porque ésta configuración permite que el estado de los pines pueda estar representado por cuatro (4) bytes, llamados PORTA, PORTB, PORTC, y PORTD respectivamente. Cada pin físico corresponde a un bit lógico en el puerto. Por ejemplo el valor del PIN-3, reside en el lugar 3 del PORTD. [1]
* Asignando y Variando Entradas y Salidas: El termino para asignar el valor de ``vivo'', (i.e. “high'”), es el (1). Esto equivale a encendido o activo. A veces también se denomina como ``High Byte''. El término para desactivar, borrar o apagar (i.e. ``low''), tiene un valor de (0). También se denomina como ``Low Byte''. Se pueden asignar valores a los registros solo un byte a la vez.
* Registros DDR: No todos los registros de entrada y salida en los AVR son pines físicos. Ya que los pines pueden estar configurados o para entrada y salida, el micro-controlador, necesita un lugar para almacenar el direccionamiento de cada bit. Para esto se utilizan los registros DDR (data direction registers). Como en los otros registros del micro-controlador, los DDR se configuran con ceros (000) y unos (111), pero en este caso el (0), indica una entrada y el (1), una salida. Este tipo de registros funcionan como un vigilante para controlar cuales son los permisos que tiene cada proceso para transformar datos en los cajones de la memoria bit-por-bit. Por ejemplo si al asignar un valor de (0xF0) (aka. 1111 0000) al registro DDRA, esto indicaría que los bits 7-4 en el puerto PORTA, son de salida y los bits 3-0 serían configurados como entradas en el mismo puerto.
* Ejecución de Programas: Código hexadecimal (Hex Code), es lo que se almacena en la región de programas (flash program memory) del micro-controlador. Al correr el programa con el código hexadecimal, las instrucciones son leídas por un contador. Este procedimiento a la vez carga la siguiente instrucción a un registro especial de instrucciones. Los operandos de cada instrucción son subsecuente-mente transferidos a la unidad de lógica aritmética (ALU), mientras que la instrucción está siendo decodificada y posteriormente ejecutada por la unidad ALU. [1]

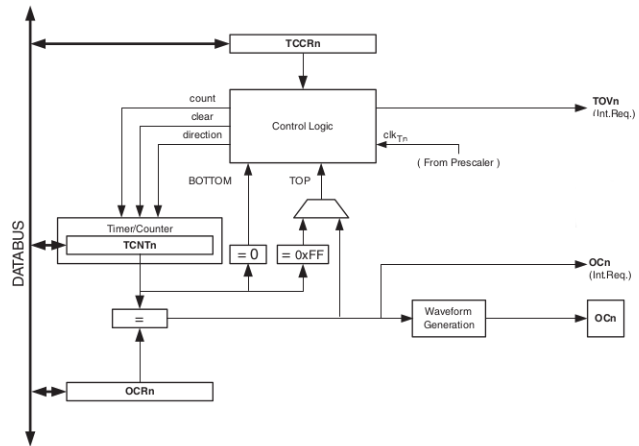
1. ANALISIS Y PROCEDIMIENTO

Se realizó la implementación de los puertos digitales de entrada/salida, esto por medio del simulador proteus, inicialmente se creó el circuito para un pin digital entrada/salida, luego se probó su correcto funcionamiento para posteriormente crear su respectivo componente, el cual se replicó 8 veces para crear dicho puerto de entrada/salida.

Luego se implementó la memoria flash y la memoria RAM por medio del circuito de la imagen 1. Como se puede observar en la figura este circuito consta de dos puertos uno configurado como entrada y el otro configurado como salida, para realizar esta configuración y comprobar el adecuado funcionamiento de estas memorias, se usa un código de programación contenido en el ATMega8515 que para este microcontrolador sería la CPU del sistema.

**Imagen 1 Implementación de los pines digitales, memoria y CPU del microcontrolador AVR**

Finalmente se implementa el timer de 8 bits de acuerdo al diagrama de bloques de la imagen 2. En donde:



**Imagen 2 diagrama de bloques temporizador de 8 bits.**

* TCCRn Registro de Control del Temporizador. Permite modificar el funcionamiento del temporizador.

**Imagen 3 Registro TCCRn.**

* OCRn es el registro encargado del almacenamiento de los datos de un valor de comparación.

**Imagen 4 Registro OCRn.**

* TCNTn registro encargado del almacenamiento de la cuenta actual del temporizador



**Imagen 5 Registro TCNTn.**

* Preescalador es el encargado de contar las banderas de tiempo actual y sobrepaso.



**Imagen 6 preescalador.**

* Control logic la lógica de control define una seria de pasos, con cada paso el control logic genera salidas específicas. Estás salidas son alimentadas el multiplexor de ruta de datos y registro de entrada de escritura. Cada etapa de la secuencia está diseñada para ser realizado en un solo ciclo de reloj.



**Imagen 7 Control logic.**

1. CONCLUSIONES

* Conocimos y entendimos el funcionamiento que posee el microcontrolador con CPU AVR por medio de diagrama de bloques.
* Utilizamos la herramienta proteus para entender el comportamiento y visualizar claramente el funcionamiento de cada uno de los bloques que posee en microcontrolador con CPU AVR.
* Utilizamos los puertos creados y probados con anterioridad para conectarlos a una memoria RAM que a su vez junto con el ATMEGA y un TIMER previamente diseñado e implementado, con todo esto unido se visualiza la señal que entrega configurándolo desde la programación en c cada uno de los 8 bits ocupados.
* La herramienta de proteus que permite encapsular circuitos por así decirlo es muy útil para crear circuitos muy robustos.

1. REFERENCIAS

|  |  |
| --- | --- |
| [1] | J. Reyes, «Microcontroladores, Sistemas Ensamblados,» CCRMA, Stanford University., [En línea]. Available: https://ccrma.stanford.edu/~juanig/articles/pidht/pidtoot/Microcontroladores\_Sistemas.html. [Último acceso: 27 marzo 2016]. |
| [2] | A. tecnologia, «areatecnologia,» [En línea]. Available: http://www.areatecnologia.com/TUTORIALES/EL%20TRANSISTOR.htm. [Último acceso: 3 marzo 2016]. |
| [3] | M. S. R. ,. G. C. C. J. Savant, «Diseño electronico,» de *Diseño electronico*, Pretice Hall, p. 252. |
| [4] | A. d. electronica, «Apuntesdeelectronica,» [En línea]. Available: https://apuntesdeelectronica.files.wordpress.com/2011/10/amplificacion-amplif-de-sec3b1al-pequec3b1a.pdf. [Último acceso: 04 marzo 2016]. |