Universidad Técnica Federico Santa María
Departamento de Electrónica
Redes de Computadores I

PROYECTO

Diseño e Implementación de detección de errores mediante CRC y algoritmo que supervisa la comunicación entre dispositivos.

Integrante : Darwin Cardemil
Rol : 2891002-9
Integrante : Camilo Díaz
Rol : 2704606-1
Profesor :Agustín González
Fecha : 22/07/2012

Resumen

Este documento presenta dos soluciones implementadas, una para detectar errores en la transmisión de datos, a través de un código detector de errores (CRC) y otra para supervisar pérdidas de información en la comunicación entre dispositivos (Maestro - Esclavo) en un medio alambrado. Para este último caso, se diseñó un algoritmo basado en el protocolo TCP/IP, que supervisa y verifica la correcta transmisión de las instrucciones.

Introducción

En la práctica se requieren códigos detectores de errores para verificar la transmisión de datos. Esto nos permite detectar los errores, y si los hay, solicitar retransmisión.

Por otra parte en la comunicación entre dispositivos se necesita un protocolo que permita verificar la transmisión y recepción de datos, ya que en casos donde se exige confiabilidad, se debe disponer de una comunicación robusta para cumplir tales exigencias.

En la teoría existen diferentes alternativas para solucionar estas problemáticas, las cuales tienen ventajas y desventajas comparativas, que están directamente relacionadas con la capacidad técnica disponible y la dificultad de implementación.

Diagrama general

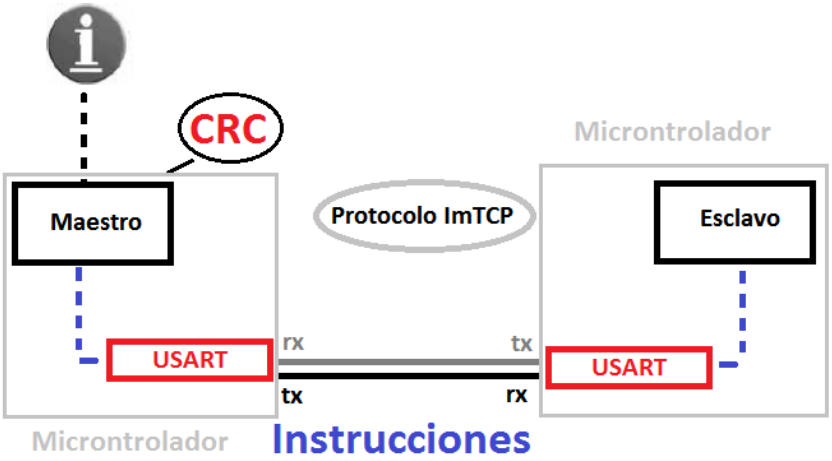

Figura 1: Diagrama general

En el diagrama anterior se muestra un esquema general del proyecto, donde se detallan los bloques más importantes acorde a la asignatura.

Tanto el maestro como esclavo están construidos en microcontroladores, donde en el maestro se incluye el código CRC para verificar la transmisión de datos desde el ibutton, siendo éste un elemento pasivo (sólo lectura). Además se graba el código que permite verificar la comunicación con el esclavo mediante un protocolo que imita al TCP/IP. Mientras que en el esclavo se adjunta sólo el código de comunicación nombrado anteriormente. Se destaca que la comunicación entre maestro y esclavo se realiza mediante el uso del periférico USART que poseen los microcontroladores, usando comunicación asincrónica.

Ibutton

El ibutton es un microchip encapsulado con una cubierta de acero inoxidable, lo cual lo hace apto para ser llevado por el usuario como llavero, anillo u otros objetos personales y así transportar la información a cualquier lugar.

En el mercado existen distintos ibutton, para diferentes aplicaciones, en este caso se utiliza el ibutton que en su interior almacena un número de serie único en el mundo, siendo ideal para aplicaciones de control de acceso, identificación, etc.

La información es almacena en una memoria ROM de 64 bits, la cual se accede a través del protocolo 1-Wire, que requiere un único cable de datos y un retorno de tierra solamente.

Operación

A la memoria ROM interna del ibutton se accede a través de una sola línea de datos. El número de serie de 48-bit, 8-bit de código de la familia y el CRC de 8 bits, se recuperan mediante el protocolo 1-Wire. Este protocolo define las transacciones de bus en términos de estado de bus durante intervalos de tiempo especificados. Todos los datos se leen y escriben el bit menos significativo en primer lugar.

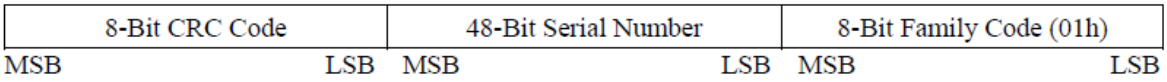

Figura 2: Campo de datos del ibutton.

El bus 1-Wire es un sistema que tiene, valga la redundancia, un sistema de bus maestro y uno o más esclavos. En todos los casos, el ibutton es un dispositivo esclavo. El maestro del bus es típicamente un microcontrolador.

Se debe tener en cuenta el hardware para la comunicación con el ibutton, la cual debe satisfacer ciertos requisitos, los cuales se pueden ver en el datasheet del ibutton, por ejemplo ibutton DS1990A.

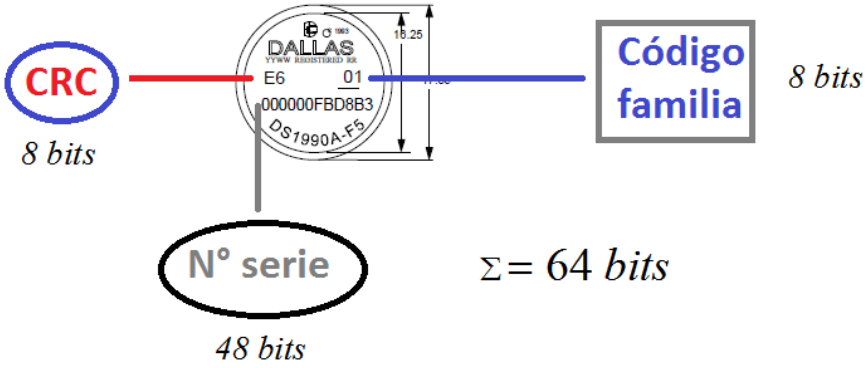

Figura 3: Descripción de los campos del Ibutton.

Detección de errores

El uso de códigos detectores de errores, permite detectar las fallas de transmisión que pueden ser producidos por varios factores como por ejemplo: ruido térmico, interferencia electromagnética, etc. Para detectar estos errores, se incluyen bits adicionales en la transmisión de datos, que pasan a ser de redundancia.

Código CRC

El código CRC o de comprobación de redundancia cíclica que tiene como finalidad agregar bits redundantes al final de la transmisión, lo más pequeña posible, tal que sea capaz de detectar la mayor cantidad de errores. El cálculo del CRC se basa en el uso de un polinomio generador de grado r, el cual debe saber tanto el transmisor como el receptor. Los n bits que se transmitirán se toman como los coeficientes de un polinomio de grado n-1, por ejemplo si se desea transmitir los datos: 1101 estos datos se pueden expresar en un polinomio de grado 3, de la forma: $x^3+x^2+0*x^1+x^0= x^3+x^2+x^0$. Luego a estos bits a transmitir se agregan r bits, para que el polinomio final sea divisible por el polinomio generador, sin generar resto.

El polinomio generador en este caso está dado en su hoja técnica, el cual es: $x^8+x^5+x^4+1$.

En el receptor se debe seguir los siguientes pasos:

- Se divide el polinomio de los datos recibidos por el polinomio generador.
- Se comprueba si se genera resto.
- Si no hay resto, no se han producido errores en la transmisión.
- Si hay resto, se han producido errores, por lo que se pide retransmisión.

En el transmisor se deben seguir los siguientes pasos:

- Obtener el polinomio equivalente al mensaje.
- Multiplicar por el grado del polinomio generador (equivale agregar tantos ceros como sea el grado del polinomio generador).
- Dividir el polinomio del mensaje por el generador y obtener el resto.
- Concatenar el resto al mensaje original y transmitir.

Comunicación entre maestro y esclavo

La comunicación entre maestro y esclavo se realiza mediante un periférico USART propio de los microcontrolador, con una comunicación asincrónica. El algoritmo diseñado para verificar la transmisión y recepción de los datos, se basa en el protocolo TCP/IP. En este caso se envían instrucciones básicas que son codificadas con dígitos decimales, las cuales son comprobadas por el esclavo y si pertenecen a la tabla adjunta en memoria, envían un acuse de recibo (OK). En el caso que llegue un dato que no está dentro de la tabla de instrucciones permitidas se envía una instrucción al maestro que identifica un error en la recepción.

El maestro envía las instrucciones permitidas (tabla adjunta en memoria) y espera acuse de recibo para enviar una nueva instrucción. Si no llega un acuse de recibo (OK) después de unos segundos, interpreta una pérdida de paquete y envía la última instrucción.

Esta última acción se repite cuando llega un dato errado, tomando nuevamente la decisión de reenviar la última instrucción enviada.

Figura 4: Tablas de maestro y esclavo.

Conclusiones

La detección de errores es fundamental en la transmisión de datos, ya que permiten detectar los errores antes de procesarlos. De la misma forma en la comunicación entre dispositivos, es necesario tener un protocolo de comunicación que admita verificar tanto la transmisión como la recepción de los datos, para cumplir con la confiabilidad. Las distintas alternativas en protocolos y en códigos detectores de errores, está directamente relacionado con los recursos disponibles y la dificultad de implementación.

Referencias [1]: datasheet DS1990A.

Referencias [2]: CRC web.

Referencias [3]: datasheet PIC 16F648A.