Detector de nivel de agua y comunicación vía ethernet con FPGA

Marko Peshevski
Escola Politècnica Superior d'Enginyeria de Vilanova i la Geltrú
Universitat Politècnica de Catalunya
marko.peshevski@estudiant.upc.edu

***Resumen*—En este trabajo se describe la implementación y utilización de un sensor de nivel de agua económico sobre un núcleo MicroBlaze en una placa de evaluación de la FPGA Spartan 6 de Xilinx. Para poder realizar la lectura de este sensor se utiliza un conversor analógico – digital de Microchip (MCP3002). Todo ello se muestra en una página web a través de un servidor que también corre sobre el mismo núcleo. La comunicación con la red se hace a través de Ethernet puesto que la placa de evaluación utilizada lo incorpora.**

1. Introducción

En este trabajo se describe el desarrollo del software necesario sobre un núcleo MicroBlaze de Xilinx para hacer la lectura de un sensor de nivel de agua de bajo coste y mostrar los datos en una página web que también es servida por el mismo núcleo. Para este propósito se ha utilizado una placa de evaluación de la FPGA Spartan 6 de Xilinx (véase Figura 1). Esta placa en concreto es el modelo Spartan 6 LX9 MicroBoard de Avnet [1].

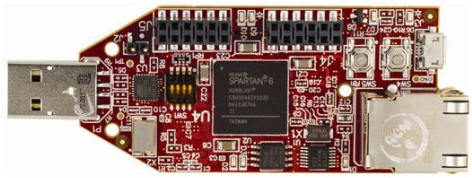

Figura 1. Placa de evaluación de la FPGA utilizada

2. Objetivo

El principal objetivo de este trabajo es conseguir mostrar los datos leídos del conversor analógico – digital en una página web accesible por cualquier navegador relativamente moderno[1], y que esta visualización tenga la menor latencia posible.

3. Sensor de nivel de agua

El sensor de nivel de agua utilizado en este trabajo es uno de muy bajo coste. En la Figura 2 se puede ver una imagen de este sensor. La placa de circuito impreso del sensor en cuestión tiene un transistor NPN con su base conectada directamente a unas pistas en circuito abierto. Paralelas a las pistas en circuito abierto hay otras que están conectadas a la tensión de alimentación, a través de una resistencia limitadora de corriente de $100\,\Omega$. Cuando el agua entra en contacto con las pistas cierra el circuito, polarizando la base, y con ello amplificando la corriente de colector-emisor. Debido a que el sensor puede estar más o menos hundido en agua, la longitud de pista hundida modificará la resistencia que habrá en serie con la base del transistor. Variando esta resistencia se amplificará en mayor o menor medida la corriente colector-emisor, y con ello se puede saber el nivel de agua del depósito en el que estuviera hundido dicho sensor.

Figura 2. Sensor de nivel de agua utilizado para desarrollar este trabajo

4. Conversor analógico – digital

El conversor usado para leer el nivel de tensión del sensor de nivel de agua es el conversor de 10 bits MCP3002 de Microchip[2]. Este conversor se comunica con el núcleo MicroBlaze a través de un canal serie SPI. Para obtener datos del mismo basta con una escritura de tan sólo 4 bits, a partir de los cuales el conversor ya responde con la medida de 10 bits. Éstos 4 bits, según el datasheet son:

- Bit de start: inicio de comunicación, debe ser siempre de nivel alto;
- Bit SGL/DIFF: selecciona si se quiere utilizar el conversor para medidas pseudo-diferenciales o respecto al punto común (masa);

1. Que tenga soporte para WebSockets.

2. Se puede encontrar más información al respecto en: http://www.microchip.com/wwwproducts/en/MCP3002

- Bit ODD/SIGN: selecciona el orden de las entradas no inversora e inversora en el modo pseudo-diferencial, y cuál de los dos canales disponibles se quiere leer en el modo de medida respecto al punto común;
- Bit MSBF: selecciona si se quiere recibir el bit más significativo primero. Si este bit estuviera a nivel bajo el conversor primero devuelve los datos *dos veces*, primero en orden decreciente (Most Significant Bit First) y luego en orden creciente (Least Significant Bit First).

En la Figura 3 se puede ver resumido el protocolo que hace falta para comunicarse con el conversor.

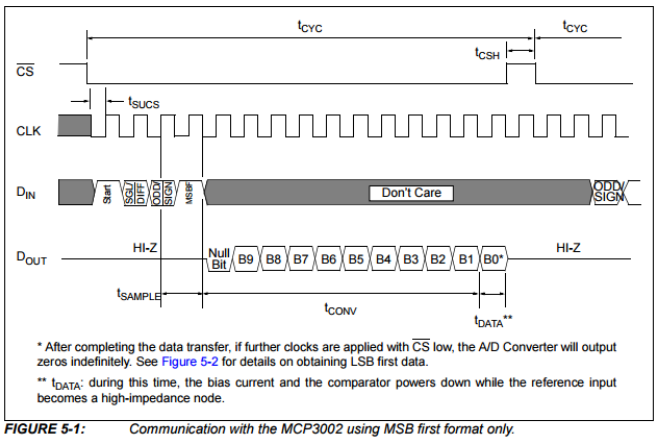

Figura 3. Resumen del protocolo SPI utilizado para leer del conversor analógico – digital. Extraída del datasheet del fabricante

5. Interfaz Ethernet

Para desarrollar todo el apartado de comunicación vía Ethernet y servidor web en este trabajo se ha partido de un ejemplo proporcionado por el fabricante de la placa de evaluación, Avnet [2]. En este ejemplo se utiliza la interfaz Ethernet a través de un núcleo de propiedad intelectual (IP Core) de Xilinx llamado *AXI_Ethernetlite*. Este IP Core es el encargado de gestionar la comunicación a través de la interfaz ethernet, y desde el código del núcleo MicroBlaze se utiliza la misma desde un nivel relativamente alto. Este IP Core gestiona la pila de protocolo del modelo OSI hasta el nivel de enlace de datos (donde se hallan los protocolos de acceso al medio), siendo el resto de niveles (por encima del nivel de red, incluído), responsabilidad del usuario del IP Core.

6. Librería TCP/IP: lwIP

Para gestionar los niveles superiores del modelo OSI se utiliza la librería de protocolo TCP/IP llamada lwIP (lightweight IP). Esta es una librería de código abierto especialmente diseñada para sistemas empotrados. Esta es la librería que, en el código del núcleo MicroBlaze, se encarga de gestionar las conexiones TCP, de comunicarse con el nodo lejano que intenta establecer una conexión con el servidor, etcétera. Parte del código del núcleo que utiliza esta librería se reaprovecha del ejemplo del fabricante ([2]), aunque con modificaciones mayores para adaptarlo a la aplicación del trabajo.

7. WebSockets

Para conseguir el propósito de mínima latencia posible en la visualización de los datos en el navegador que accede a la página web servida por la FPGA se necesita comunicación bidireccional. Dicha comunicación no es fácilmente realizable con un servidor web de protocolo HTTP. Si se quisiera utilizar un protocolo HTTP para este fín lo que cabría hacer es que el cliente (navegador web) hiciera peticiones de tipo GET y POST para que el servidor le responda con los datos. Esto es tremendamente lento debido a que con el protocolo HTTP cada vez que se termina una petición o respuesta a unos datos se cierra la conexión. Una posible solución a este problema es utilizar WebSockets. Un WebSocket es, *grosso modo*, una conexión HTTP elevada a otro protocolo (WebSocket) que permite comunicación bidireccional en cualquier momento, sin que cliente o servidor tengan que hacer una petición. Para el propósito del trabajo esto es ideal puesto que una vez establecida la comunicación el servidor puede mandar información al cliente tan a menudo como sea necesario. Se puede consultar toda la especificación del protocolo WebSockets en [3].

En la Figura 4 se puede ver un diagrama resumiendo cómo funciona el protocolo WebSocket.

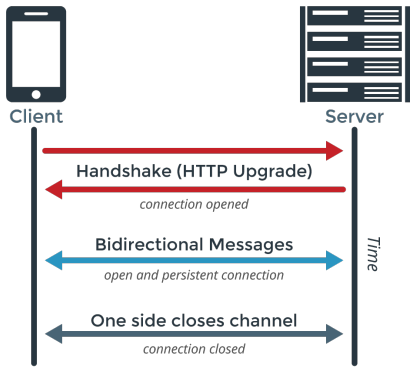

Figura 4. Diagrama resumen del protocolo WebSocket

8. Lado cliente

Tras todo el resto la última parte que compone este trabajo es la visualización de los datos en la página web. Para ésta se ha utilizado un archivo HTML sencillo para crear un botón que sirva para abrir la conexión WebSocket (elevar el protoclo HTTP), y un gráfico utilizando librerías

de JavaScript ajenas [4]. En este gráfico se muestra directamente el valor de 10 bits $[0, 1023]$ de salida del conversor analógico – digital. Además de este gráfico (que sólo se dibuja una vez se tienen datos, es decir, una vez abierta la comunicación WebSocket), hace falta un pequeño script en lenguaje JavaScript para gestionar la comunicación.

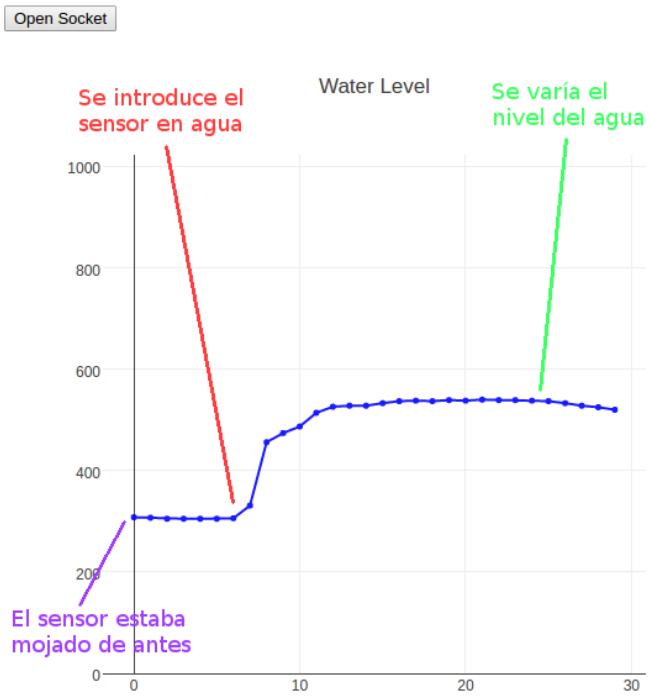

Figura 5. Web diseñada para este trabajo, mostrando brevemente el significado de los datos visualizados

9. Conclusiones

Con el desarrollo descrito anteriormente se han conseguido los objetivos propuestos para este trabajo: mostrar el nivel de agua de un depósito leído con un sensor de muy bajo coste en una página web con la menor latencia posible, utilizando para ello un núcleo MicroBlaze sobre una FPGA Spartan 6 de Xilinx.

10. Trabajo futuro

Por cuestiones lógicas de tiempo y simplicidad, muchas cosas han quedado por desarrollar en este trabajo. Entre ellas, algunas de las más relevantes serían:

- Implementar el resto de funcionalidad del protocolo WebSocket: ser capaz de gestionar varias conexiones, abrir/cerrar conexiones, etcétera.
- Linealizar el sensor y calibrarlo de alguna manera. Puede ser necesario utilizar algún amplificador del nivel de tensión en la salida del sensor.
- Hacer que lo que se muestra en la web no sea solamente la lectura del conversor analógico – digital sino algo con más sentido.

Referencias

[1] Xilinx Inc., *Avnet Spartan-6 LX9 MicroBoard*, https://www.xilinx.com/products/boards-and-kits/1-3i2dfk.html.

[2] Avnet Inc., *LwIP Applications Software 201 for the Spartan-6 LX9 MicroBoard*, https://forums.xilinx.com/xlnx/attachments/xlnx/EMBEDDED/9425/2/AvtS6LX9MicroBoard_SW201_lwIP_Apps_14_4_01.pdf.

[3] I. Fette y A. Melnikov, *The WebSocket Protocol*, https://www.rfc-editor.org/rfc/rfc6455.txt.

[4] Plotly, *What is plotly.js?*, https://plot.ly/javascript/.

[5] M. Peshevski, *SDAV_mini-projecte*, repositorio Git con toda la información sobre este trabajo, https://github.com/markopesevski/SDAV_mini-projecte.