January 2020

COMMUNICATION BUS FOR EMBEDDED SENSOR SYSTEMS

HP INC

Communication Bus for Embedded Sensor Systems

Abstract: A communications bus using features of both I2C and SPI topologies enables communications between a master and multiple slaves at high data rates over a span of several feet.

This disclosure relates to the field of electronic interfaces.

A technique is disclosed that provides a reliable communication bus system for embedded sensor systems to communicate between a master device and several slave devices (such as sensors) at high data rates, and which is flexible (expansible) to support as many as 127 slave devices over a span of as much as several feet.

Embedded systems usually use the popular and ubiquitous Inter-Integrated Circuit ("I2C") bus and Serial Peripheral Interface ("SPI") bus communication topologies to communicate with other devices, such as sensors or other processors. While the I2C bus system is very flexible and only requires 2 signal lines to communicate with up to 127 devices, it is relatively slow and can in general not extend too far beyond a printed circuit board due to it's open drain driver architecture. A SPI bus on the other hand, supports much higher data speeds up to 100MHz, but it is not as flexible as it requires a dedicated chip select (CS) line for each slave. Therefore, SPI doesn't easily allow a flexible interface to attach an unknown amount of devices at a later time.

According to the present disclosure, features of both the I2C and SPI bus topologies are combined to create a new communication bus that possesses both of their advantages and overcomes both of their disadvantages.

The SPI bus is primarily used for data exchange. The I2C bus is used to virtualize the chip select (CS) signal. When the master initiates communication to a particular slave, it lowers its CS signal to indicate that it wants to communication with it. Here the technique completely foregoes the dedicated CS signals and instead runs an I2C bus and a CS Strobe signal in parallel to the SPI bus to each slave device. Each slave device requires a very small and simple microcontroller (MCU) with a unique bus address. In order to perform SPI data communication, the master first communicates with the slave device's MCU and sends an "activation" command. Next, the master uses regular SPI communication and the single CS Strobe signal to communicate with the slave device. The CS Strobe signal is connected to the MCUs of all of the slaves, but only the MCU that received the activation signal will pass it on to the respective slave device's CS input signal. The MCUs therefore perform the function of a digital AND gate or pass switch and will only pass the signal on if they have been activated.

This method allows fast data throughput since all of the major communication occurs over the faster SPI interface. Before and after each SPI communication cycle with a particular slave, I2C communication takes place to "activate" and "deactivate " each slave. This occurs at very low speeds (i.e. 100kHz), which has minimal throughput impact as there is only very little data to transfer, while running I2C at lower speeds for these functions makes the interface more robust and allows it to operate over longer cable or trace lengths.

In alternative examples, instead of using I2C, any other bus or communication topology could be used. For example, a UART (serial port), a "bit-banged" design, or any other readily available simple communication media could replace the I2C port. This makes the techinque really flexible. As most microcontrollers have only a limited number of SPI, I2C, and UART ports, the particular implementation can be chosen based on which types of ports are available.

The disclosed technique advantageously combines I2C and SPI busses to create a new combined bus that can operate at high bus speeds while being very flexible. The number of slaves to be attached does not need to be pre-determined. This technique can be integrated using any off-the-shelf microcontroller as most of them have at least one I2C, one SPI, and one UART port.

Disclosed by Adrian Rothenbuhler, HP Inc.