Designing a smart TV remote using a 2.4GHz RF radio

RamaSai Krishna Vakkantula, Cypress

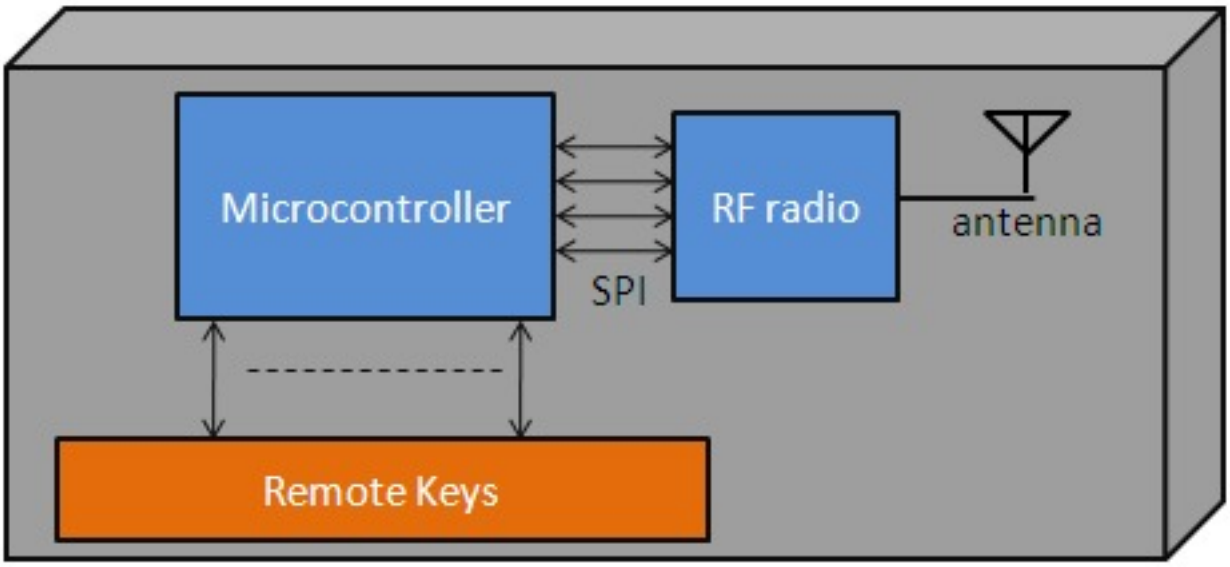
Today's television manufacturers are designing televisions that are smart and have the ability to connect to the Internet so that users can browse web content and socialize. Usually, these smart TVs are mounted on the wall and all other entertainment equipment is located behind the closed doors of an entertainment center. Traditional IR-based remote controllers, however, must be in line-of-sight to the receiving device for commands to be received.

To address this issue, remote controller manufacturers are turning to RF technology. RF remote controllers operate over the 2.4GHz Industrial, Scientific and Medical (ISM) band. The ISM band is a portion of the radio spectrum reserved internationally for the use of radio frequency energy for industrial, scientific, and medical purposes other than communications. Despite the intent of the original allocations, in recent years the fastest-growing uses of this band have been for short-range, low-power communication systems.

This article explores the design architecture of a remote for smart TVs using a 2.4GHz RF radio as well as the architecture of a receiving dongle present on the TV and how to implement a reliable protocol between the two. It will also describe how the user experience can be enhanced by adding speaker and microphone functionality to the remote.

RF-based architecture

The architecture for an RF-based remote controller is shown in the Figure 1. Remote keys are arranged in a matrix and connected to the general-purpose pins of a microcontroller. The RF radio is interfaced to the microcontroller over an SPI interface with four signals: SCK (clock), MOSI (Master Out Slave In), MISO (Master In Slave Out), and SS (Slave Select).



The microcontroller continuously polls for the key presses by scanning the key matrix and transmits commands to the receiver dongle via the RF radio.

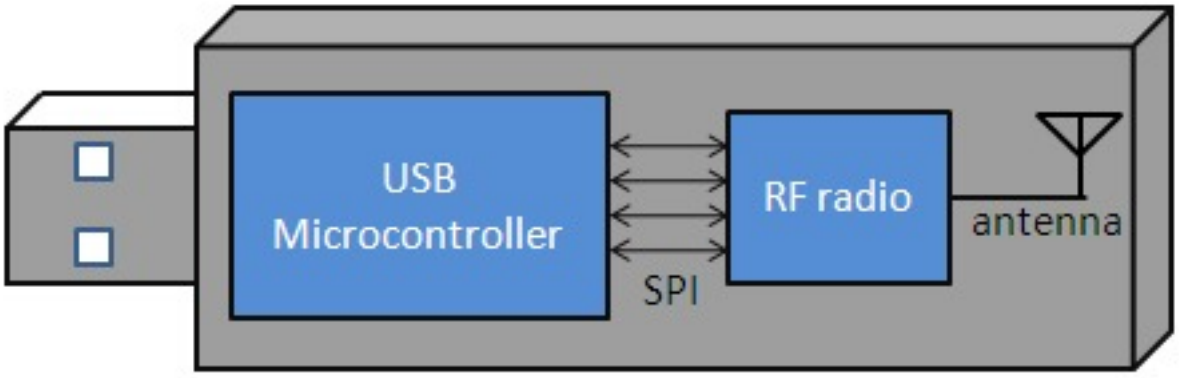
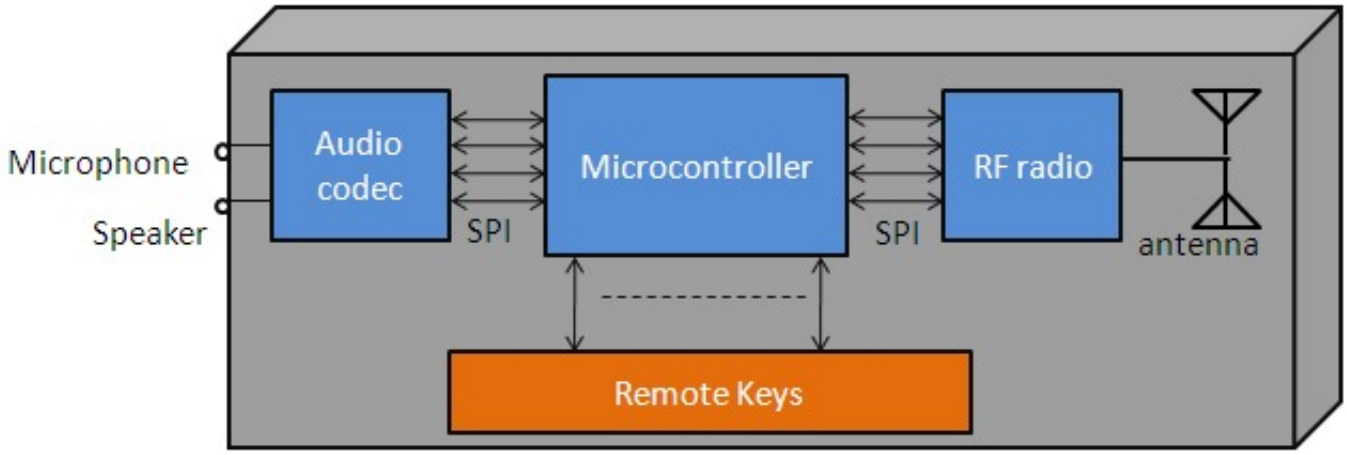


Figure 2 shows the RF receiver dongle architecture comprising a USB microcontroller connected to an RF radio via an SPI interface. A full-speed USB microcontroller should be sufficient since the bandwidth required by the remote controller is minimal. This dongle could be integrated into the smart TV by connecting it to an available USB port and enumerates as a HID device. In this way, no special firmware drivers are needed on the TV side since this is a standard class driver.

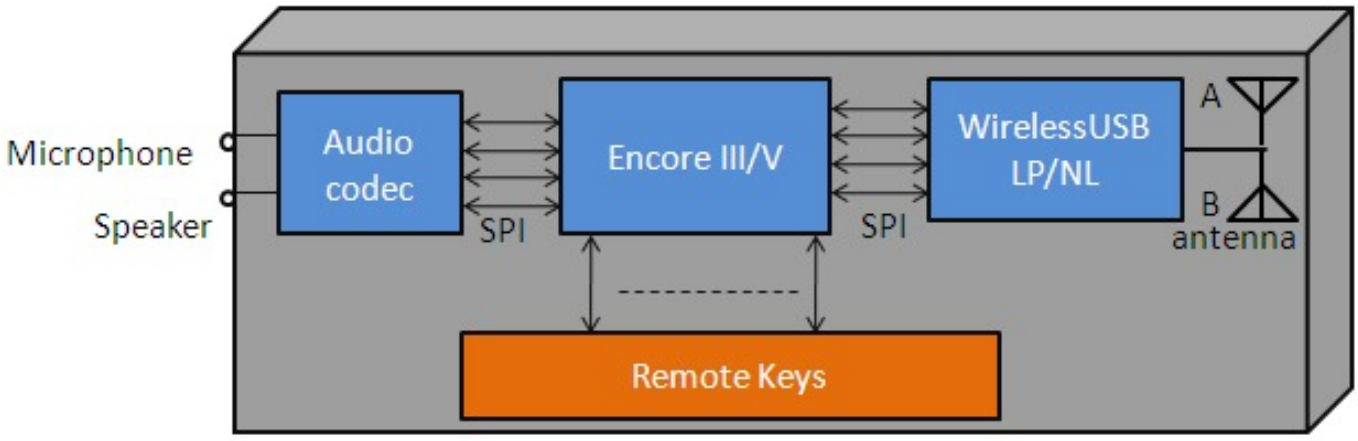
The RF radio on the dongle receives key information whenever there is a key press on the remote. The USB microcontroller continuously polls the RF radio registers to check if there is any data to be read and sends that data to TV over an interrupt endpoint.

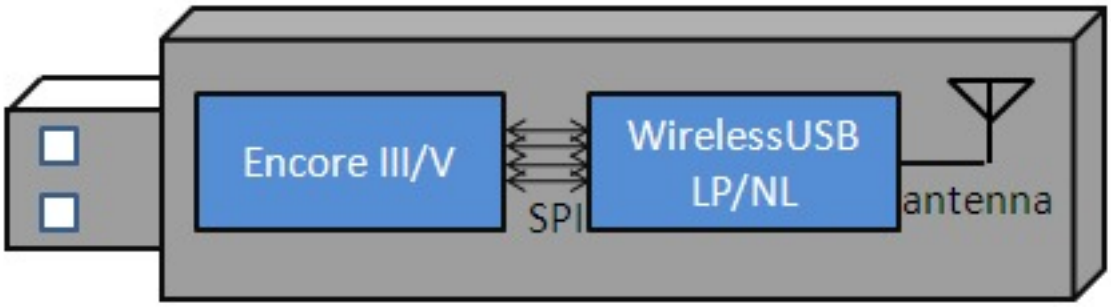
Adding Voice over RF (VoRF)



Users are not typically expected to sit close to the TV. To support online conversation capabilities, the remote controller could have a microphone and a speaker (or headphone functionality). The architecture of such a remote controller is shown in Figure 3.

An extra audio codec is connected to the microcontroller over a SPI interface and converts the input audio data to PCM samples before sending them to the microcontroller. The microcontroller converts these PCM samples to ADPCM samples (to save RF bandwidth) and sends them to the receiver dongle via the RF radio. The USB microcontroller present in the receiver dongle converts these ADPCM samples to PCM samples before sending them to the USB host in the TV. To support VoRF, the dongle needs to enumerate as a composite device with one interface as audio class and the other one as HID device. Isochronous endpoints are used to transmit the audio in both directions.





Today’s system-on-chip microcontrollers integrate the capabilities required to implement RF-based remote controllers with a simple, low-cost architecture.

Design challenges and recommendations:
ADPCM to save RF bandwidth: The USB host transfers voice as PCM samples which would requires 256 Kbps. By converting PCM samples to ADPCM samples before transmitting over air, voice bandwidth can be reduced to 64 Kbps (i.e., the 16-bit sample used for PCM is compressed to a 4-bit sample stream using the same sample rate).

Two antennas to increase the reliability of voice data transfer: RF channels can have multipath fading similar to constructive and destructive waves of water. If only one antenna path is available, it is easy to place two antennas in such a position as to create a multipath null. Then you move either the bridge or remote antenna about 3 cm to get a good signal again. For wireless applications such as keyboards and mice, multipath null is not a big problem. A keyboard or mouse is usually within a few feet of the receiving dongle and users do not notice 10 ms or 20 ms delay, so messages can be retried many times if needed. Users also are used to pressing a key again if it occasionally does not work the first time.

However, voice is different, given that a person may sometimes be stationary and sometimes moving, and nulls become obvious interruptions in the audio stream. To mitigate multipath nulls, the RF path (or the environment) must change with a second antenna either at the remote or bridge. Because the ~128 Kbps channel can contain a redundant copy of the 64 Kbps voice stream, you can send packets using Remote Antenna A and duplicate packets using Remote Antenna B to provide two different RF paths to the receiving dongle.

Similarly, the bridge can send two copies of the same packet and the far side (remote controller) can use Remote Antenna A to receive the first packet and switch to Remote Antenna B to receive the duplicate packet. Now to lose a packet, both paths must be in a multipath null, which happens much less frequent than if only a single antenna path is used.

No ACK for voice: Because audio is latency sensitive, the standard wireless "ACK" is not used. This eliminates retransmission of audio packets that will introduce too much variable latency as well as reduce use of the RF channel usage.

ACK for key presses: There should be acknowledgements coming from the receiving device whenever the remote controller transmits key presses. This creates issues since “ACKs" needs to disabled while transmitting voice data and enabled while transmitting HID data. This situation can be avoided by keeping "ACKs" disabled all the time and checking the reliability of HID data by doing a loopback between the remote controller and receiver dongle.

Source URL (retrieved on *03/10/2014 - 8:07am*):
http://www.ecnmag.com/articles/2013/01/designing-smart-tv-remote-using-24ghz-rf-radio