Software Defined Modem Using Modern Micro-Controllers

Joshua Jerred

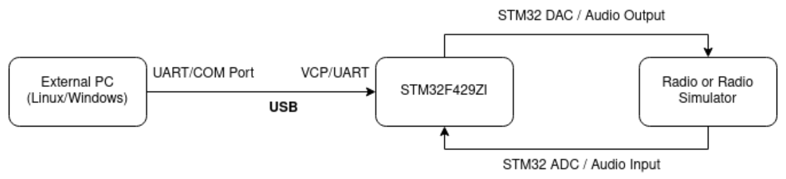
1 Introduction

Modern embedded hardware is far more capable than it has been in the past. Newer devices, such as the STM32, have powerful ARM cores matched up with other hardware inside of small single chip MCU packages. With peripherals like ADCs and DACs, along side DMA, audio can be cleanly generated in real time. This can be used to generate modulated data streams, with one common piece of hardware.

2 Background

3 Hardware

The idea of this project is that a generic microcontroller (MCU) is capable of acting as a software defined modem. Given this, the hardware will be kept simple. A single MCU will be used to act as the software modem. The modem will be connected to an external host/client, this client will control the modem. The modem has a single audio output and audio input, this can connected to various types of physical mediums to transfer the data that is modulated in the audio from one modem to another.



3.1 MCU Requirements/Considerations

When looking for the appropriate MCU for this project there were a few requirements and considerations that needed to be taken into account.

ADC A single ADC (Analog-to-Digital Converter) is required to sample the input audio. It needs suitable resolution and speed to capture the modulated audio stream.

DAC A single DAC (Digital-to-Analog Converter) is required to output modulated audio from the microcontroller.

DMA DMA (Direct Memory Access) is required to transfer data to/from the ADC/DAC automatically without CPU intervention. Although this may not be strictly required, it will free up a considerable amount of CPU time.

Hardware Timers Accurate hardware timers are required for properly timing the audio waveform generation, and the modulation.

USB/UART A USB/UART connection is required to connect an external MCU to a host device so that the host device can send/receive data to/from the modem.

Cost There are many devices that are capable of completing the task, but the cost of the device is a major consideration. The device should be as cheap and as generic as possible to allow for easy replication of this project on other hardware.

4 Software Implementation

4.1 Waveform Generation

Any method of audio modulation requires that a waveform be generated. This waveform is then modified to represent unique symbols. Therefore, a MCU must be able to generate a simple sine wave. With AFSK being the primary modulation mode for this project, a 1200 Hz wave is generated.

The goal was to have the MCU generate a non-changing sine wave without a large impact to CPU usage. Using the DAC, alongside DMA, and a hardware timer. The DAC will output a constant voltage given a constant 12-bit value. Utilizing an array of sine wave values, the DMA will transfer the values to the DAC at a constant rate set by a hardware timer. A single cycle of the sine wave is represented by 128 12-bit values. Given this information, and a hardware clock, the timer can be configured to trigger at a rate that will generate a sine wave at a given rate.

$$F_{trigger} = \frac{C_{APB1}}{(PSC + 1) * (ARR + 1)}$$

where:

C_{APB1} is the clock frequency of the APB1 bus (90 MHz)

PSC is the prescaler value

ARR is the auto-reload value

$F_{trigger}$ is the frequency of the trigger (1200 Hz)

$$F_{output} = \frac{F_{trigger}}{N_{samples}}$$

4.2 Data Modulation

$$F_{trigger} = \frac{C_{APB1}}{(PSC + 1) * (ARR + 1)}$$

where:

C_{APB1}	is the clock frequency of the APB1 bus
PSC	is the prescaler value
ARR	is the auto-reload value
$F_{trigger}$	is the frequency of the trigger

$$F_{output} = \frac{F_{trigger}}{N_{samples}}$$

- Sine Wave Generation
-