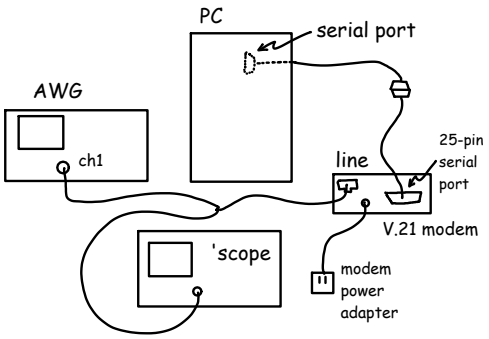
V.21 FSK Modulator

Introduction

In this lab you will write a program to create an FSK-modulated data signal. You will modify the program you wrote earlier to generate an RS-232 serial data waveform. But instead of a bipolar NRZ waveform your program will now generate a 300 bps FSK waveform that complies with the ITU-T V.21 standard.

To test your program you will output the generated FSK waveform from the AWG to a telephone-line modem. The modem will demodulate the FSK signal and the demodulated data will be output to your PC via a serial port. If you did everything right a terminal emulator running on the PC will display your name.



V.21

A copy of the V.21 standard is available on the course web site or from http://www.itu.int/rec/T-REC-V/en/.

Frequencies

V.21 is a full-duplex protocol using FDD which means a different set of frequencies are used in each direction. The V.21 standard says that the lower pair of frequencies, 1080±100 Hz, is to be used by the modem ("station") placing ("originating") the call and a higher pair of frequencies, 1750±100 Hz, by the station answering the call. Use the set of frequencies assigned for the station originating the call: 980 Hz for a mark ('1') and 1180 Hz for a space ('0').

Signal Levels

The V.21 modem specification allows a maximum output of 0 dBm and says that typical connections have attenuations of from 5 to 30 dB. You should set the level at the modem input to -10 dBm.

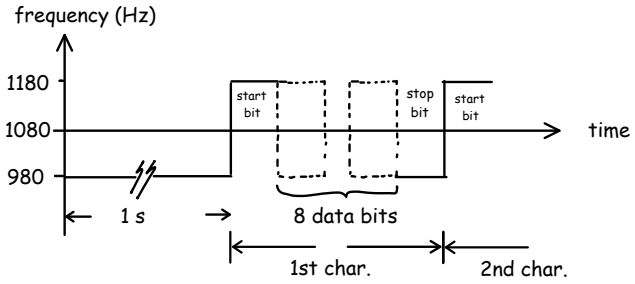
Software

Modify the C program you wrote for the RS-232 lab or modify the code in `lab8incomplete.c` on the course web site so that:

- it computes and outputs samples of a modulated V.21 FSK signal instead of an RS-232 waveform.
- the data rate is 300 bps and the sample rate 9600 Hz
- your waveform begins with at least one second (300 bits) of mark ('1') to allow the modem to detect the carrier (as per section 8.2.2 of the V.21 spec)

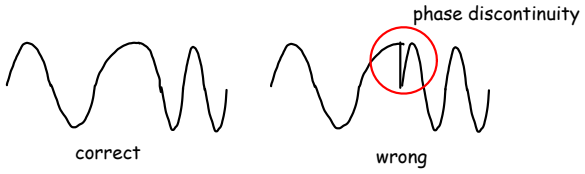
An FSK modulator software varies the carrier frequency so that the frequency (980 or 1180 Hz) depends on the voltage that would have appeared on the RS-232 line. All other aspects of the character format such as the need for start and stop bits, the bit order and the bit duration are the same.

So, although the diagram below appears to show a voltage waveform, it actually shows *frequency* (not voltage) as a function of time. There should be one second of 980 Hz tone followed by 3.3 ms of 1180 Hz tone for the start bit, then 3.3 ms of either 980 Hz or 1180 Hz tone for the LS data bit, etc:



There are various algorithms that could be used to compute the sample values for an FSK waveform.

Whichever algorithm you choose, you should generate a signal that does not have phase discontinuities between bits. In other words, it should be "phase continuous." The diagram below shows an example where the phase is continuous between a bit at a low frequency and one at a higher frequency as well as an example where there is a phase discontinuity:



Here is the pseudo-code for one method that produces a phase-continuous FSK signal:

1. initialize a 'phase' variable to zero
2. for every sample:
   (a) increment this phase variable by the phase increment that corresponds to the frequency being generated at the current time (980 or 1180) – see below for how this increment can be calculated.
   (b) compute the cosine of the phase variable and scale it so that values between -1 and 1 are converted to integers in the range 0-4095 (the range accepted by the AWG).
   (c) write this sample value to the AWG (.RAF) file as before.

You will have to write multiple samples for each bit because the sample rate (9600) is higher than the bit rate (300).

If this algorithm were implemented in hardware it would be called a Numerically Controlled Oscillator (NCO) because it is a device that computes the output of an oscillator numerically instead of using an analog oscillator circuit.

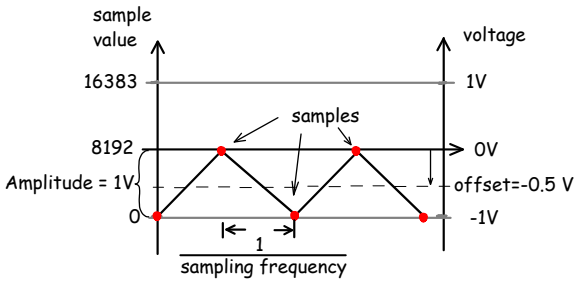
You will have to determine the appropriate phase increment per sample[1]. You will have to increment the phase by one of two different phase increments depending on whether the mark or space frequency is being generated. This in turn depends on whether you are outputting a mark or space, (i.e. a 0 or 1).

You can use a `float` value to keep track of the current phase and use the function `cos()` to compute the cosine of the phase. `#include` the file `math.h` to include the function declaration for `cos()`. You will have to convert the value from `float` to `short` before writing it to the file. In the sample code the phase variable's value needs to be retained across calls to `writebit()`. To do this, the phase variable must either be declared *static* or outside of a function.

We need to create a file with sample values to load into the AWG. The AWG can read binary files in the `.RAF` format documented on page 2-75 of the DG1000Z user manual.

An RAF file consists of a sequence of 16-bit sample values in little-endian byte order. Each sample value must be an unsigned integer between 0 and 16383 ($2^{14} - 1$). These integers are converted by the AWG into voltages using the Vpp and offset settings configured into the AWG through the front panel.

For example, if the AWG was configured for a peak-to-peak voltage of 1 V, an offset of -0.5 V, and the RAF file contained values alternating between 0 and 8192 then the output would be a triangle (or square) wave with levels of -1V and 0V:



The following C code creates a `.RAF` file by scaling values in the array `x` so they fall in the range 0 to 16383 and writing the scaled values to a file as 16-bit

[1]Hint: Frequency is defined as phase difference per time increment. The time increment in this case is the time between samples.

integers. Note that this code will only output the bytes in the right order when run on a computer where integers are stored with the LS Byte first.

`xmin` and `xmax` are minimum and maximum values in the array `x` (-1 and +1 if they are the values returned from the `cos()` function).

```
FILE *f ;
...
f = fopen ( "waveform.RAF","wb" ) ;
...
short w ;
...
for ( i=0 ; i<N ; i++ ) {
   w = (x[i]-xmin)/(xmax-xmin)*16384 ;
   fwrite( &w, 2, 1, f ) ;
}
```

An incomplete solution is available in the file `lab8incomplete.c` on the course web site. You can use this as a reference when writing your own code but you don't have to.

Procedure

Generate the Waveform

Compile and run your program to generate the `.RAF` file containing the FSK signal using the same software (Notepad++ and tcc) and instructions as in the RS-232 waveform lab (e.g. `tcc -run lab8.c`).

Import the `.RAF` file into Audacity as signed 16-bit samples with a sample rate of 9600 Hz. Check the waveform amplitude (it should take up half of the positive portion of the waveform display). Check the waveform shape (it should look like a sine wave).

Configure AWG and Load Waveform

- Press [Utility] [Set To Default] [OK] to reset the AWG to it's default settings.
- Press [Arb] to select Arbitrary waveform output.
- Press [Arb Mode] until SRate (sample rate mode) is selected. Note: to access this and some of the following buttons you may need to press [▽] (gray key on the bottom) to toggle between two sets of menus.
- Plug your USB drive into the USB socket on the front of the AWG. You should see the message USB device detected and the USB icon (⇢) will appear on the display.
- Press [Select Waveform] [Stored Wforms].
- Press [File Type] [Arb File]
- Press [Browser] until Dir is selected.
- Use the knob to highlight drive `D:` (the USB drive).
- Press [Browser] until File is selected.
- Use the knob to highlight your `.RAF` file.
- Press [Read] and you should see the message Completed! Arb Wave data have been read.
- Press [Arb] to return to the Arb menu.
- Press [Srate] and enter the sample rate on the keyboard.
- Press [Ampl] until Ampl is highlighted.
- Use the keypad to enter the peak-to-peak voltage you computed in the pre-lab and the units.
- Press [Output1] to turn on the Channel 1 output.

Check the frequency and voltages of your waveforms using the scope. You can capture a complete waveform (a total of about 2 seconds, depending on the length of your string) and then use the horizontal time and position controls to "zoom in" on different portions of the waveform. You can use the cursor and measurement functions to check that the various waveform parameters are correct.

Connect the AWG output to the modem's *line* (not phone) port as shown in the diagram above. You will be supplied a cable that has an RJ-11 plug on one side and bare wires on the other. If there are four wires, use the red and green ones. Polarity does not matter.

Connect the 9-pin RS-232 cable from the back of the PC to the 9-to-25 pin cable on the bench. Connect

the 25-pin end of that cable to the RS-232 port on the modem.

Connect the modem's power adapter to the modem and 120VAC power.

Run Teraterm as in the previous lab except that the baud rate must be set to 300 bps.

Type the following commands in the Teraterm window. These commands will be sent to the modem and configure it for V.21 modulation and then have it "pick up" the phone line and begin receiving data:

- type `ATZ` (followed by Enter) to reset the modem. If the modem receives the command it will respond with "OK". The modem sets the bit rate of the RS-232 interface to whatever rate is used to send this command.
- type `AT&FLNB15` to: set the factory default settings (`F`), reduce the modem's speaker level (`L`), disable speed negotiation (`N`), and enable V.21 modulation (`B15`),
- type `ATA` to have the modem go "off-hook" and connect to the phone line input

After a few seconds the modem should detect carrier, print CONNECT, start demodulating the FSK-modulated data and sending it to the PC over the RS-232 serial interface. You should see your name displayed on the terminal.

Show the display to the instructor to get credit for completing the lab and take screen captures of the Teraterm output and the 'scope screen (including voltage and frequency measurements) for your report.

A known-good test waveform is available in the file `lab8test.RAF` on the course web site. If you have a problem you can use this waveform to determine if the problem is with your waveform or somewhere else (AWG configuration, Teraterm settings, signal levels, modem, ...).

Pre-Lab

Hand in the answers to the following questions and a printout of your code before the start of the lab:

1. What is the power of a sine wave of amplitude A? What peak-to-peak voltage corresponds to a power of -10 dBm if the modem's input impedance is 600 ohms?
2. A sinusoid's frequency is defined by the rate at which the phase changes with time (e.g. a 1 kHz signal changes phase 1000 x 360 degrees every second). If the frequency is the V.21 originate mark frequency and the sample rate is 9600 Hz, what is the phase increment per sample? How about for the space frequency?
3. How many samples per bit are required for a 300 bps waveform sampled at 9600 Hz?
4. What will happen if Teraterm is configured to read more data bits than are generated by your software? Fewer?

Modify your RS-232 code (or modify the supplied incomplete code) as described above so that it generates the samples of an FSK signal with the appropriate frequencies and baud rate.

You should try to have a working program before the lab because you will probably not have enough time to figure out how to modify the code *and* complete the procedure given above during the lab period.

Lab Report

Submit a report in PDF format to the appropriate dropbox on the course web site. The report should include the usual identification information (name, ID, section, lab, title, date), a copy of your code including any additional changes you made in the lab, a screen capture showing the Teraterm window with your name displayed and a screen capture of the 'scope showing voltage and a frequency measurements on the screen.