Data Acquisition Module
NOTES

The windows audio input driver is set to record either one or two channels of data in either 8- or 16-bit precision, either two's complement or offset binary representation, with or without “mu-law” encoding, and minimally at sampling rates of 8 kHz, 11.025 kHz, 22.050 kHz, and 44.1 kHz. Drivers for some specific hardware support additional features, particularly a larger number of and/or greater range of sampling rates. After setting the sampling parameters for the driver, one or more buffers are allocated and placed in a FIFO queue for the driver. When started, the driver fills each queued buffer and signals its completion. Samples received when there are no queued buffers are lost.

Additional buffers may be added to the queue after sampling has started.

In our routine, samples are collected using using a double buffering scheme. We are emptying and/or processing one buffer while the other fills. To avoid missed samples, the filled buffer must be processed and returned to the driver’s buffer queue before the other (filling) buffer is completed.

There are a number of signaling methods that can be used by the driver to indicate completion of a buffer. The method we use is to send a message (MM_WIM_DATA) to the window from which RecStart was called. When the calling window receives an MM_WIM_DATA message, it calls the BufServ function to process the buffer and return it to the driver queue.
NOTES
This function is called each time the A/D driver returns a buffer. The circular buffer must be large enough to hold all data from the input buffer plus enough extra to hold the window length minus frame size number of samples, that is:

\[
\text{CirBufSize} = \text{InBufSize} + \text{WindowSampleCount} - \text{FrameSampleCount};
\]

As soon as samples are transferred to the circular buffer, the input buffer should be returned to the driver queue. This minimizes the chance of missing some samples if, for some reason we get delayed in the process of extracting frames of data and queuing them up for anybody who wants them.
NOTES

This is an expanded view of an earlier slide to illustrate the difference between analysis windows and data frames. The structure of the speech signal (in terms of the source characteristics and the pattern of poles and zeros--or more generally its spectral structure) is a constantly changing function of time. Acoustic analyses that are intended to extract spectral information must sample the continuous changes in spectrum just as the acoustic waveform must be sampled. The rate at which spectral information is sampled is called the frame rate and each sample is referred to as a frame. A frame rate of between 100 and 200 Frames Per Second (FPS) is considered high enough for speech. Note that aliasing is a potential problem for this type of sampling too.

To get a stable estimate of spectral characteristics, it is necessary to analyze waveform data within a region of the signal centered around the time associated with a frame. This region is called a window. The window length--or width if you prefer to think of it that way--determines the precision with which spectral information can be determined. Longer windows provide finer spectral resolution. However, analysis of the data within a window provides an average spectrum for the window. If there are changes in the spectral structure of the signal within the scope of a window, those changes are distorted by the averaging. A sweeping narrow band feature will look like a stationary broader band feature, with its apparent bandwidth a function of its rate of change. Consequently, windows lengths are generally as short as possible while still providing adequate spectral resolution. Depending upon the type and purpose of the analysis, typical window lengths range from 5 to 40 msec.
Discrete time-frequency tradeoff

$$\Delta F = 1/(N \times \Delta T) = S / N$$

Where:

- $\Delta F$ is the frequency bin spacing in Hertz
- $\Delta T$ is the sample spacing in seconds
- $N$ is the number of samples in the window
- $S$ is the sampling rate in Hertz.

NOTES
This provides a more specific description of the relationship between analysis window length and frequency resolution. In the digital domain, one can determine magnitude estimates for a discrete number of frequency components (or bins) from a discrete number of waveform samples. The specific relationship is a function of both the number of samples analyzed and the sampling rate as shown above. Note also that frequency bins are determined only for frequencies up to and including the Nyquist frequency.

To illustrate, a 6 msec window with 10 kHz sampling rate would nominally provide 30 frequency components spaced 166.67 Hz over the range from 0 to 5 kHz. However, when using a tapered window such as a Hanning window (see slide on window types), the effective window length is about half the nominal window length, leading to an effective resolution of more like 333.33 Hz per bin. Note that since there are still 30 bins between 0 and 5 kHz, the broader effective bandwidth implies that there is overlap in the frequency range associated with each bin.
DAM Requirements

- **Sampling Parameters**
  - Sampling rate
  - Sample format
- **Framing & Windowing Parameters**
  - Frame rate
  - Window length
  - Window type
- **Segmentation Parameters**
  - Silence amplitude threshold
  - Silence duration threshold

**NOTES**

Set-able parameters (ones we may want to adjust on the fly) are listed here. Sampling rate and format, frame rate, window length are pretty obvious. Window type is the form (if any) of the tapering function used to make the windowed samples in a frame approach 0 amplitude at the edges of the window. There are times when having no tapering function (a rectangular window) is desirable, however, in most cases, we will want to use a window like the *Hamming* window or *Hanning* window to taper the samples (see next slide).

An additional important requirement of the DAM is that it recognize when there is no speech to sample and go into a wait mode until speech appears to be present again. Obviously, the sampling should be continuous over brief silences such as the silence in stop consonants, and possible even between phrases or sentences if there is not substantial silence delimiting them.
Window Types

Hanning: \[ w_i = 0.5 - 0.5 \cos(2\pi i / I) \] \( 0 \leq i < I \)

Hamming: \[ w_i = .54 - .46 \cos(2\pi i / I) \] \( 0 \leq i < I \)

Rectangular: \[ w_i = 1.0 \] \( 0 \leq i < I \)

NOTES
This provide specific definitions of three common window types as mentioned in the previous slide. The waveform I/O library contains functions to return window coefficients for Hamming and Hanning windows.
### Issues

- Global information structures
  - Frame parameters
  - Status (e.g., silence detected)
  - Queue structures
- Frame structure
  - Include time?
  - Sample format?
- Testing?

**NOTES**

These are matters for class discussion and agreement prior to beginning to program the DAM.

We will need to determine the overall structure of the program vis a vis communication between threads, data structures needed to handle queuing of frames between threads, how frames will be structured (what information ties to each frame and which information can be stored as global state variables.

Finally, we will need to decide how the program will be tested to ensure that it is functioning correctly.