Digital Signal Processing

Digital Signal Processing

Digital Signal Processing (DSP) refers to methods of filtering and analysing time-varying signals based on the assumption that the signal amplitudes can be represented by a finite set of integers corresponding to the amplitude of the signal at a finite number of points in time. For example, figure 1 compares a 1 kHz sine wave with its representation by 6-bit samples taken at rate of $20\ \mathrm{ms}^{-1}$. To avoid **aliasing** (see figure 2) it is essential to make sure that the system satisfies the **Nyquist Criterion**, *i.e.* If the sampling rate is n the sampled signal must contain no components of frequency greater than $1/(2n)$.

Figure 3 illustrates the components of a generic DSP system. The great advantage of these systems is that their function is easily specified by software. There is an enormous literature on DSP algorithms many of which are of great importance in their own right (*e.g.* the Fast Fourier Transform). Performance or these systems is usually limited by the performance (*i.e.* speed, resolution and linearity) of the analogue-to-digital converter. However, when using a DSP you should never forget two facts:

1. If information was not present in the sampled signal to start with, no amount of digital manipulation will extract it.
2. Real signals come with noise.

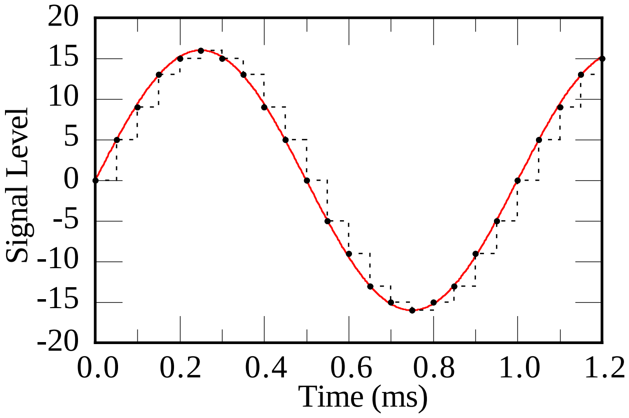

Figure 1. Sampled Signal

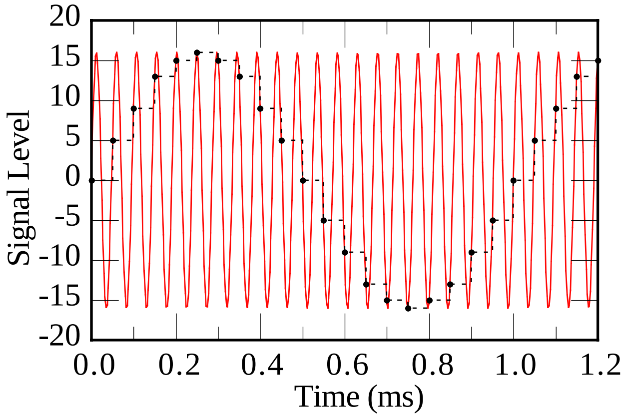

Figure 2. Aliased sampling

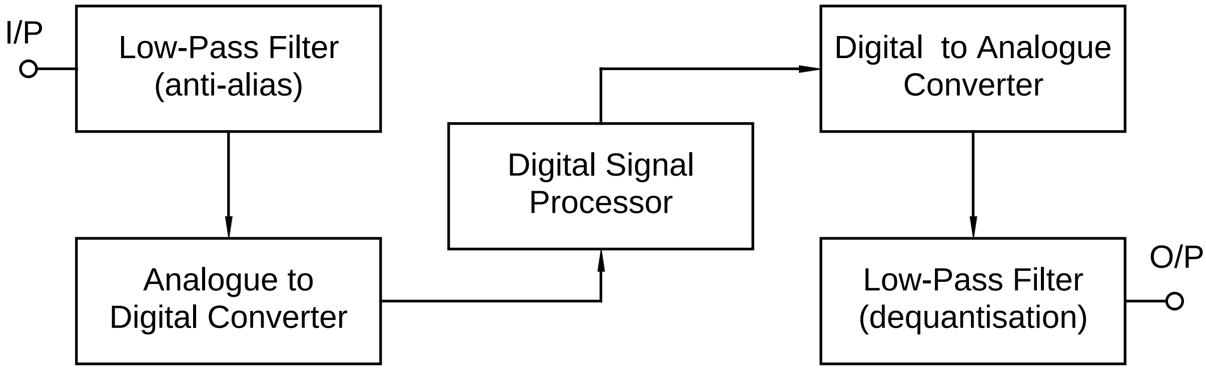

Figure 3. Generic Digital Signal Processor System

Using the DSPlay System

The DSPLAY softare allows you to construct signal-processing electronics by specifying a block diagram, known as a ‘flowgram’. You then run the system to assess the effect of the processing in terms of the relation between the input and output.

Functions are selected from menus by typing an initial letter – to cancel use the **ESC**ape key. Use the arrow keys to move around the workspace. **I**nsert will add a new signal-processing block. **E**dit + **P**arameters will allow you to specify the function of the new block. **I**nsert applied to an existing block will allow you to position interconnections. Use **I**nsert again to connect to the destination point.

Use **L**oad to open an existing flowgram. To run a flowgram, press **H**ome + **R**un. Then you can use **D**isplay + **P**lot_buffer on individual blocks to examine waveforms and see what’s going on. **B**uffer mode allows you to position cursors on the waveform display.