Digital signal processing is the mathematical manipulation of digitally represented signals. DSP processors are microprocessors designed to perform efficient, high-speed, low-cost digital signal processing. DSP is nowadays used in many diverse signal processing applications such as digital filtering, convolution, spectrum analysis, speech processing, speaker recognition, image processing, image compression, automatic control and many more.

Digital signal processor systems are built from several elements:

- A digital microprocessor fast enough to carry out the mathematical operations;
- Fast memory to store the program and data samples;
- A/D and D/A converters to get real signals into and out of the microprocessor;
- The microprocessor program.

Most DSP operations are based on multiply-accumulate (MAC) cycles. For example, the FIR digital filter is mathematically expressed as

$$\sum_{k} h[k] \cdot x[k]$$

where $x$ is a vector of input data, and $h$ is a vector of filter coefficients. The input samples are multiplied by the corresponding coefficients and the results are summed. Thus, the main operation here is a dot product, consisting of multiplication and addition. These operations are common to most digital signal processing applications.

In early days of digital signal processing, ordinary microprocessors like the 8085, Z80, 6801 and others were used in real DSP applications. Multiplication was performed in software by a series of shift and add operations, each of which consumed one or more clock cycles. The microprocessor clock-speed was low and the instructions were primitive, thus causing considerable delay in each processing loop. The maximum clock frequency was not higher than the audio frequencies and as such the processing capabilities were rather limited. Later, the speed of DSP processing has increased considerably by the development of fast hardware multiplier chips that could be interfaced to the general purpose microprocessors. The overall performance was still poor, and a typical implementation required several external chips, including memory, multiplier, A/D converter and D/A converter, resulting in high power consumption and design complexity.

The first commercially available DSP chip, the TMS32010, was introduced by Texas Instruments (TI) in 1992. This chip incorporated special architecture and multiplication hardware to yield faster performance. Since then there has been major advances in the development of DSP chips, and today several DSP processor vendors (Texas Instruments, Analog Devices, Motorola, Lucent Technologies etc) are offering high-performance DSP processors.

The processors for embedded digital signal processing can be classified into three groups:

- **DSP processors**: These are specialized microprocessors for signal processing applications. For example, Texas Instruments’s TMS320C55xx series, or Analog Devices’s ADSP-2106x series. These processors usually have multiple operations per instruction.

- **DSP-enhanced processors (DSPEP)**: These are efficient general-purpose microcontrollers with added DSP features. Among them are the Microchip dsPIC series, Analog Devices’s Blackfin etc. These processors can have more than one operation per instruction.

- **General purpose processors (GPP)**: These are fast general purpose microcontrollers that can be used in DSP applications, as well as in general microcontroller based applications. Examples include ARM, ARM7 etc.
These processors usually have one operation per instruction.

**CHOOSING A DSP PROCESSOR**

The choice of a DSP processor depends very much on the type of application and the tools available for development. For low-cost and low-speed applications the user can select a GPP. Most GPPs incorporate on-board memories, multiplier modules and A/D converters, making them suitable candidates for low-cost digital signal processing applications. For higher performance one can select a DSPEP processor. These processors have modified architectures supporting DSP operations (e.g. fast multiplier modules) in addition to usual microcontroller functions. For very high speed signal processing applications the only choice is to use a DSP processor.

Table 1 gives a list of some of the commercially available DSP processors. The points that are important in the choice of a DSP processor include:

**Arithmetic format:** Most DSPs use fixed-point arithmetic, where numbers are represented as integers or as fractions. Some DSPs use floating-point arithmetic, where numbers are represented by a mantissa and an exponent. Floating-point DSPs usually use a 32-bit data width. A higher data width gives higher numeric precision, but the size of the data width has direct impact on the cost of the chip as a higher data width increases the chip complexity and size of external memory, resulting in an overall higher cost.

**Speed:** The execution speed of a DSP chip is very important as it is a measure of its suitability for a particular application. Traditionally, processor speeds are measured using the MIPS (millions of instructions per second) or MOPS (millions of operations per second). It is very important to realize that using MIPS or MOPS in DSP applications can give very misleading results. This is because some DSP processors can perform much more work in a given instruction than others. For example, some DSP processors can perform all the required MAC operations by a single instruction consisting of several instructions to perform the same operations.

Another used performance index is the MACS (multiply-accumulates per second). This index is also misleading since DSP applications involve operations other than multiplication and addition and therefore MACS is not a reliable measure of the performance index.

One of the most commonly used DSP performance indices is the Berkeley Design Technology’s benchmarking, known as the BDTImark2000, which is based on the calculation of the execution time, memory usage and energy consumption for each DSP chip tested.

Typical DSP applications used in the benchmark are: FIR and IIR digital filter blocks, vector dot products, vector addition, vector maximum, FFT and so on.

**THE MATLAB FUNCTION FIR CAN BE USED TO OBTAIN THE COEFFICIENTS OF AN FIR FILTER EASILY. TO DESIGN AN FIR FILTER WITH ORDER 10, FS = 20KHZ AND FC = 2.5KHZ WE HAVE TO USE THE FOLLOWING MATLAB STATEMENT:**

```matlab
>> h = fir(10, 0.25)
```

where the second parameter is “fc / (fs / 2)”, i.e. 2.5 / (20/2) = 0.25. The fir function uses the Hamming Window by default. The filter parameters displayed by the above statement are given below:

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>h[0]</td>
<td>0.0039</td>
<td>0</td>
<td>0.0321</td>
<td>0.1167</td>
<td>0.2207</td>
<td>0.2687</td>
</tr>
<tr>
<td>h[10]</td>
<td>0.0039</td>
<td>0</td>
<td>0.0321</td>
<td>0.1167</td>
<td>0.2207</td>
<td>0.2687</td>
</tr>
</tbody>
</table>

The filter frequency and phase responses can be plotted using the MATLAB function:

```matlab
>> freqz(h)
```

which displays the filter response shown in Figure 5. Notice that the response is plotted with normalized frequency where the cut-off frequency is at 0.25.
Development time: The ease of development is also important in DSP-based applications. Most DSP manufacturers offer DSP development kits together with suitable assemblers or high-level language compilers. Using such kits shorten the project development time considerably.

Memory: Fast memory is a requirement in all DSP applications. Most DSP chips are based on multiport memories which permit multiple memory accesses per instruction cycle. Harvard architecture and its derivatives are used in DSP chips with separate bus for the memory in order to increase the memory throughput. Some DSP chips use up to 11-stage pipeline with multi-cycle latencies, and several level cache memory systems. The size of the total memory is also important and is application dependent.

Power consumption: Many embedded DSP applications are portable and as such low power consumption is usually a fundamental requirement. Some DSP chips operate with reduced voltages to lower the power consumption. Most DSP chips feature idle or sleep modes that help to reduce the power consumption when the chip is not busy. In some DSP chips the unused peripherals can be turned off to lower the power consumption.

Cost and system complexity: The cost is also an important factor when choosing a DSP for an application. System complexity depends upon the number of external chips required and the total pin count used. For example, external A/D and D/A, or external memory requirements are key factors that affect overall cost of the system.

DIGITAL FILTERING

Figure 1 shows the block diagram of the digital filtering process. Digital filtering has specific characteristics that we need to pay special attention to. The analog input signal must satisfy certain requirements. Furthermore, on converting an output digital signal into analog form, it is necessary to perform additional signal processing in order to obtain the appropriate result.

If an input signal contains frequency components higher than half the sampling frequency (fs/2), it will cause distortion to the original spectrum. This is the reason why it is first necessary to perform filtering of the input signal using a low-pass filter to eliminate high-frequency components from input frequency spectrum. This filter is called anti-aliasing filter as it prevents aliasing. The signal is then converted into digital, the filtering action is carried out via the algorithm implemented on the digital processor, and the signal is converted back into analog form. In some DSP applications the A/D and D/A converters are incorporated on the DSP chip. But usually these converters are added externally to the DSP chip. Notice that the output signal is sometimes filtered to remove any high frequency components.

Types of Digital Filters

The two major types of digital filters are the finite impulse response (FIR) and the infinite impulse response (IIR). Both types have some advantages and disadvantages that should be carefully considered when designing a filter. Besides, it is necessary to take into account all fundamental characteristics of a signal to be filtered as these are very important when deciding which filter to use. A filter has two important characteristics: the frequency response.
and the phase response. Most types of filters can be designed with excellent frequency response characteristics. The phase response on the other hand depends on the type of filter designed and it can either be linear or non-linear. Speech signal, for example, can be processed with non-linear phase characteristic as the phase characteristic of a speech signal is not of the essence and as such can be neglected, which results in the possibility to use much wider range of filter systems for its processing.

There are also signals for which a linear phase characteristic is required. A typical example is signals obtained from various sensors in industry. Here, it is important that a filter has linear phase characteristic to prevent losing information. FIR filters have linear phase characteristics, but higher filter order is required, making the design complex. These filters are inherently stable. IIR filters on the other hand have non-linear phase characteristics, but a lower filter order is required, making them less complex, and the resulting filter has the potential to become unstable.

**FIR Filters**

FIR filters are digital filters with finite impulse response. They are also known as ‘non-recursive digital filters’ as they do not have feedback and the filter output depends only on the present and pass inputs. The output samples of the FIR filter are computed as:

\[ y[n] = \sum_{k=0}^{N-1} h[k] \cdot x[n - k] \]

where \( x[k] \) are the filter input samples, \( h[k] \) are the filter coefficients and \( y[n] \) are the filter output samples. The FIR filter transfer function can be expressed as:

\[ H(z) = \frac{Y(z)}{X(z)} = \sum_{n=0}^{N-1} a[n] \cdot z^{-n} \]

There are several types of FIR filter realizations. Here, we will only look at the direct realisation which is the most commonly used FIR filter realisation. Figure 2 shows a block diagram of the FIR realisation. As can be seen from the diagram the filtering action consists of delay lines, multiplications and summations.

In practical applications two circular buffers are used in the filter program to hold the input samples and the filter coefficients. The contents of these buffers are multiplied and added as in Equation 1 by manipulating the buffer pointers. The total sum is the filter output and is sent to the D/A converter.

FIR filters are normally designed with a suitable windowing function used to shape the response. There are many windowing functions, such as Rectangular, Triangular, Hann, Hamming, Bohman, Blackman and so on. The windowing theory is well established and readers are recommended to look at books on DSP for further information (e.g. Continuous Discrete Time Signals and Systems, by Mandal and Asif).

The filter coefficients can be calculated analytically from the filter impulse response. There are many freeware and commercially available computer programs that can be used to obtain these coefficients quickly and reliably. For example, ScopeFIR (www.iowegian.com),

---

**Figure 5: Frequency and phase responses of the filter**

---

**Design Using a DSP Development Kit**

IN THIS SECTION WE WILL BE DESIGNING AND IMPLEMENTING OUR FIR DIGITAL FILTER USING THE LOW-COST DSPICPRO4 DSP DEVELOPMENT BOARD (WWW.MIKROE.COM), PROGRAMMED USING THE MIKROC PRO DSPIC30/33 C LANGUAGE.

dSPICPRO4 is a low-cost DSP development board (see Figure 6) that can be used in low-cost, low-speed DSP applications, and especially in teaching the practical aspects of designing DSP systems. The board supports 64 and 80-pin dsPIC30F MCUs. The board includes all the hardware needed for developing a DSP project.

The brief specifications of this development board are:

- dsPIC30F6014A DSP microcontroller
- USB programmer with mikroICD debugger
- 67 LEDs and 67 push-button switches
- Real-time clock chip
- MMC/SD card slot
- RS232 and RS485 serial ports
- LCD, GLCD and touch-screen interface
- Ethernet controller chip
- CAN interface module
- 12-bit D/A converter (A/D converter is on the MCU)
- Voltage reference for A/D and D/A converters

The filter software was developed using the mikroC PRO for dsPIC compiler. This is a C compiler developed specifically for DSP applications, using the dsPIC series of microcontrollers. The compiler supports a large number of built-in library functions for interfaces such as RS232/RS485, I2C, Ethernet, CAN, USB, SD card, LCD, GLCD, touch-screen and many more. In addition, the compiler has a filter design tool that can be used to generate the filter coefficients for FIR (or IIR) filters, and it also generates a template code that can be used as the basis for the actual filtering process.

**Figure 6: dsPICPRO4 development board**
IIR Filters

FIR filters are digital filters with infinite finite impulse response. They are also known as ‘recursive digital filters’ as they have feedback and the filter output depends not only on the present and pass inputs, but also on outputs. Some commonly used IIR digital filters are Butterworth, Chebyshev, Bessel and so on. The transfer function of an IIR filter can be expressed as:

\[
H(z) = \frac{Y(z)}{X(z)} = \frac{\sum_{k=0}^{N} b[k] z^{-k}}{1 + \sum_{k=4}^{N} a[k] z^{-k}}
\]

where \( N \) is the order of the filter, \( b[k] \) and \( a[k] \) are the filter coefficients. The filter output samples can be calculated from the expression:

\[
y[n] = \sum_{k=0}^{N} b[k] x[n - k] - \sum_{k=4}^{N} a[k] y[n - k]
\]

There are several IIR realizations. One of the commonly used ones is the cascade realization shown in Figure 3 for the general first order and Figure 4 for the second order sections. Higher order filters are constructed by cascading these basic sections. The filter coefficients can be calculated by initially designing an analog filter and then converting it into discrete form using a bilinear transformation. Alternatively, there are many programs that can be used to obtain the coefficients \( a[k] \) and \( b[k] \).

EXAMPLE DIGITAL FILTERING

In this section we will be designing a low-pass FIR digital filter as an example. First, the filter will be designed and simulated using the MATLAB package. Then, the filter will be implemented in real-time on a DSP development kit.

The specifications of the filter to be designed are:

- Sampling frequency, \( f_s = 20\text{kHz} \)
- Cut-off frequency, \( f_c = 2.5\text{kHz} \)
- Filter order, \( N = 10 \)
- Window used = Hamming

...