

3. DIGITAL SIGNAL-PROCESSING REQUIREMENTS AND LIMITATIONS

Besides ADC's, digital signal processing is another key element in radios using digitization of the RF or IF. The amount of time required for signal processing is of critical importance in radio receiver applications. The required processing time is a function of the received signal bandwidth, the speed of the processor, and the number and complexity of the algorithms required to perform the needed radio receiver operations. These operations are application-specific and may include some or all of the following: downconversion, filtering, multiple access processing, demultiplexing, frequency despread, demodulation, synchronization, channel decoding, decryption, and source decoding [22,23]. Because of the wide variety of algorithms possible in radio receiver applications, digital signal-processing limitations are more difficult to discuss than ADC's.

The intent of this section is to discuss the requirements and limitations of digital signal processing. It is not intended to provide a detailed presentation of the wide variety of digital signal-processing techniques and algorithms that are available. Many books and papers are available to provide detailed information on digital signal-processing techniques and algorithms. One fundamental book on digital signal processing is [24].

3.1 Processors

Many different processors are available to provide digital signal processing. These processors vary substantially in speed of operation, physical size, and cost. Speed is usually a critical requirement in selecting a processor. Other factors including dynamic range, arithmetic precision, cost, and size are also important considerations when choosing a processor.

A common method used to increase total processing speed beyond that of a single processor is to employ multiple processors operating in parallel. Assuming a given processor, by putting more and more of these processors together and operating them in parallel, higher and higher processing speeds can be achieved. This, of course, also increases power consumption, size, and cost.

Many radio receiver applications require processors with small physical size and relatively low cost. For these cases, single chip processors are the preferred choice. Single chip processors can be general purpose microprocessors (such as the Intel 80486), digital signal processors (such as the Texas Instruments TMS320C40), or specialized integrated circuits for dedicated processing tasks (such as the Harris HSP50016 Digital Downconverter). Some specialized radio receiver applications may not be bound by stringent physical size and cost limitations. Therefore, processors of all types are considered in this report, ranging from single chip general purpose microprocessors to supercomputers.

Computations in digital signal processing can be performed using fixed-point arithmetic or floating-point arithmetic, although many of the computations require floating-point arithmetic. The advantage of floating-point arithmetic over fixed-point arithmetic is that it permits the use

of numbers with a much greater dynamic range. This is important in many digital signal-processing operations.

In fixed-point arithmetic, the position of the decimal point in the register where each operand is stored always is assumed to be the same. In floating-point arithmetic, each operand is represented by a number stored in a register representing a fraction or integer. A number stored in a second register specifies the position of the decimal point of the number stored in the first register. Some processors do not have floating-point hardware and require floating-point operation to be implemented in software. Software implementation of floating-point arithmetic is typically much slower than hardware implementation.

Because floating-point operations are so important in digital signal processing, the speed of processors is often specified in terms of millions of floating-point operations per second (MFLOPS). This parameter allows comparison of the processing speed of different processors and also allows determination of the time required to execute certain algorithms.

Many different benchmarks (such as the SPEC benchmarks, Whetstone, Dhrystone, and Linpack) are used to compare speeds between processors. Each benchmark provides a number indicating the relative speed of processing based on testing varying tasks. Results from the application of a benchmark to different processors can be compared. However, results between different benchmarks, in general, should not be compared. While these benchmarks are useful for comparing processor performance, the parameter chosen to compare processing speeds between processors in this report is the theoretical peak MFLOPS. This parameter was chosen due to its ease of availability for virtually all floating-point processors, its lack of dependence on specific benchmarking algorithms, and its relevancy to dedicated applications used in implementation of a radio receiver. The theoretical peak MFLOPS parameter gives the maximum possible speed of performance for the processor. It is found by computing the number of floating-point additions and multiplications (using the processor's full precision) that can be performed during a given time interval [25].

Some examples of the processing speed of various types of processors ranging from single chip processors to supercomputers are presented in Table 3. This table only gives a sampling of the range of capabilities that exist in digital signal processing. Many other processors, with varying capabilities, either exist or have been proposed. In addition, new developments with increasing capabilities are announced all the time. An extensive listing of processing speeds for many different computers is found in [25].

In certain situations, especially in the high-throughput case, overall processing performance is not limited by the processor speed but by the maximum data transfer rates of the peripheral components such as memory or I/O (input/output) ports. The inclusion of these factors in platform evaluation should not be ignored when choosing a processor.

Table 3. Examples of Processing Technology

Processing Speed*	Number of Processors	Platform	Manufacturer and Model
50 MFLOPS	1	DSP Chip	Texas Instruments TMS320C40
120 MFLOPS	1	DSP Chip	Analog Devices ADSP-21060/62
400 MFLOPS	8	VME Board	Pentek 4285
800 MFLOPS	16	Computer Workstation	SUN Sparc 2000
6.48 GFLOPS	4	Supercomputer	Convex C4/XA-4
32 GFLOPS	4	Supercomputer	Hitachi S-3800/480
184 GFLOPS	3680	Massively Parallel Computer	Intel Paragon XPS140
236 GFLOPS	140	Massively Parallel Computer	National Aerospace Laboratory Numerical Wind Tunnel (Japan)

*Theoretical peak processing speed.

3.2 Real-Time Operation

For most radio receiver applications, real-time operation is important. In many types of processing, such as computing Fast Fourier Transforms (FFT's), the data is partitioned into blocks of a finite length. Processing is performed on the entire block of data. In this block type processing, assuming a single processor, real-time operation essentially means that all processing on a given block of data (including any required data transfers) is completed before all of the next block of data to be processed is captured. This concept is illustrated in Figure 12 [26].

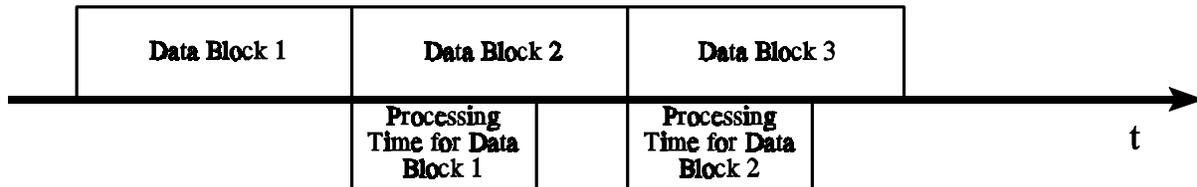


Figure 12. Real-time processing for block data using a single processor.

If the processing time (including any required data transfers) is longer than the time required to capture all of the next block of data (again assuming a single processor), data collection must be stopped until the processing is completed. At that time data collection may resume. Under these conditions some of the input data is missed and is not processed. This is an example of processing that does not take place in real time. Depending upon the application and the amount of data lost, this may not be acceptable.

This problem can be alleviated by using two or more processors operating cooperatively. This general technique is called multiprocessing and is used frequently. To illustrate how multiprocessing can be used to speed up overall data throughput, consider the previous example of data partitioned into data blocks but with two processors available instead of one. As shown in Figure 13, processor 1 operates on one block of data and processor 2 operates on the next block of data. The processors continue to operate on alternating data blocks. The processed data output is obtained by switching back and forth between the outputs of the two processors. This is sometimes called a “ping-pong” technique. Using this technique, the processing time of each processor can take longer than the time to capture the next block of data and still provide real-time output. The processing time cannot exceed the time to capture the next two blocks of data, however. This technique can be extended to more than two processors to achieve even faster overall data throughput.

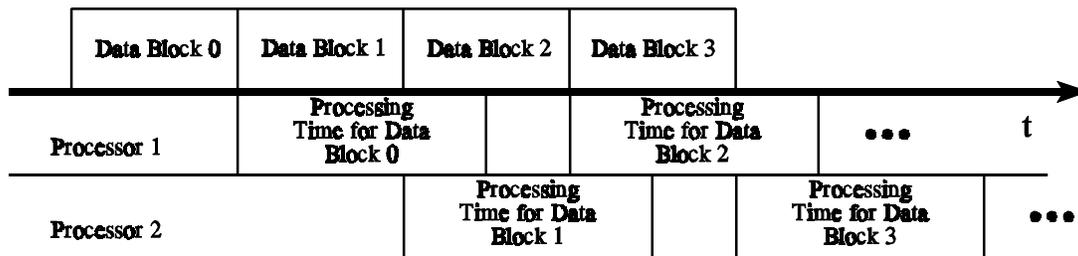


Figure 13. Real-time processing for block data using two processors.

3.3 Algorithms

Providing a general discussion on algorithms used for implementing radio receiver functions is difficult. This is due to the wide variety of types of receivers as well as the various ways of implementing the required receiver operations for each type of receiver. In short, algorithms are highly application-specific. The details of the many potential algorithms used in radio receivers are beyond the scope of this report. It is beneficial, however, to look at an example algorithm to observe the methodology in determining algorithm complexity vs. the potential for real-time operation. This type of assessment is crucial for radio receivers that use digitization at the RF or IF.

The FFT is an example of an algorithm frequently used in digital signal-processing and radio receiver applications. The FFT transforms time-domain samples of received signals into a set of frequency-domain samples, allowing operations on the received signals to be performed

directly in the frequency domain. These received signals are typically bandpass signals when digitization occurs at the RF or IF. These bandpass signals may be digitized using either sampling at twice the maximum frequency, bandpass sampling, or oversampling. The FFT also can be applied to signals that have been downconverted to baseband but the signal must be split into co-phase and quadrature-phase components before digitization unless coherent downconversion has been used.

Regardless of the sampling method employed, the resolution of the transformed signal in the frequency domain is a function of both the time spacing between the samples of the signal Δt in the time domain and the number of samples N used in the computation of the FFT. The frequency spacing between samples in the frequency domain is then given as

$$\Delta f = \frac{1}{N \Delta t} \text{ (Hz) .}$$

The maximum frequency of the spectrum is then

$$f_{\max} = \frac{N}{2} \Delta f = \frac{1}{2 \Delta t}$$

since there are $N/2$ samples in the FFT computed from N real-valued time-domain samples. (Actually, there are $(N/2) + 1$ samples in the FFT ranging from DC to f_{\max} if both DC and f_{\max} are included.) For a fixed N -point FFT (i.e., an FFT computed from N real-valued time-domain samples), the frequency spacing between the frequency domain samples Δf must be changed in order to change the maximum frequency of the spectrum. That requires changing the time spacing Δt between the time-domain samples. By decreasing Δt and holding N constant, the time duration of a block of N samples is reduced and the maximum frequency of the spectrum is thereby increased. Therefore, for fixed values of N , the higher the maximum frequency desired, the shorter the duration of the block of N samples must be. With a fixed number of samples N , a given processor, and a given FFT algorithm², the processing time to compute an N -point FFT is fixed. For real-time operation, this computation of the FFT (including any other required processing such as windowing and any required data transfers) must be performed within the time taken to capture all N samples of the current data block assuming that a single processor is used. A parameter called real-time bandwidth then can be defined as the maximum frequency that can be processed in real time.

To achieve real-time processing, careful consideration of processor speed, the signal bandwidth (data rate), the number of computations required in implementing the signal-processing algorithms, and the speed of any necessary data transfers is required. An example showing how to estimate the amount of processing power required for real-time analysis is given below. For

²There are many different FFT algorithms available requiring different numbers of floating-point operations. $N \log_2 N$ commonly is used as an approximation for the number of floating-point operations required in computing an FFT.

this example, the simplified case of looking at the time required to compute an FFT is investigated. While this example shows the methodology used to determine a required processing speed, the processing required for radio receiver applications normally would involve much more than computing a single FFT. In this simplified example, it is assumed that an input signal is sampled at a fixed rate and that the only processing performed on the sampled input signal is the FFT. No other processing (such as windowing or averaging) is performed. Data transfer time is also neglected.

Assuming a bandlimited input signal with a maximum frequency of 5 MHz, the $2f_{\max}$ sampling rate would be 10 Msamples/s. For this sampling rate, the time between samples Δt is 100 ns. Assume that FFT's are computed from blocks of $N=1024$ samples. Therefore, it takes $N\Delta t = 102.4 \mu\text{s}$ to capture a block of data. The number of floating-point operations (actually multiplications) required² to compute an N-point FFT is estimated as $N \log_2 N$. Therefore, roughly 10,240 floating-point operations are required for the 1024-point FFT. In order to achieve real-time processing in the single processor case, the FFT must be computed within the time period required to capture a block of data ($102.4 \mu\text{s}$). The minimum required processing speed is then found by

$$\frac{\text{Number of floating point operations required}}{\text{Time to capture a block of data}} = \text{Minimum processor speed (FLOPS)}.$$

In this simplified case, the minimum processing speed is 100 MFLOPS. One can then compare the required processing speed to the processing speeds available for different types of processors such as those listed in Table 3.