Today’s growth in markets for consumer electronics, wireless electronics, and hand-
held computing requires cost-efficient solutions that supply high performance com-
puting, energy efficiency, and programmability. General-purpose processors are
poorly suited to meet the requirements of energy efficiency and competitive cost.
ASICs are unable to provide sufficient programmability. As the result, a variety
of Application Specific Instruction-set Processors (ASIP) is emerging to meet the
requirement.

To compete with ASIC, an ASIP requires better performance/power efficiency
than a general purpose processor. To achieve this target, an ASIP must require more
parallelism in various levels, such as data level parallelism (DLP), instruction level
parallelism (ILP), and thread level parallelism (TLP) that will be depicted in this
chapter.

4.1 Specific Instruction-Set

The earliest specific instruction-set example is floating-point. Floating-point oper-
ation hardware is much more complex than integer operation. Early designed pro-
cessors can only do integer operations, and implement floating-point operation by
software emulation. Most scientific algorithms require floating-point, but emulation
implementation is too slow for them. Thus, scientific requirement drives processor
to integrate floating-point operations.

Most multimedia applications use fixed-point operations. Due to the limitations
of human eye and ear sensitivity, some precision loss on image pixels and audio
samples is acceptable. For example in a DCT algorithm, using 12-bit fixed-point to
represent a cosine value is good enough for most image quality requirements. Fixed-
point operation can be simply an integer arithmetic operation and a shift operation.
In typical integer addition and multiplication operations, since the most-significant
bits (MSB) are truncated when the result has an overflow, the following shift op-
eration will get a wrong value. Thus for multimedia applications, the fixed-point
operation applied should be able to preserve MSB.

Multiply-Accumulate (MAC) is a key operation in Finite Impulse Response
(FIR) filter function. Some DSP processors implement MAC with automatic looping
and index increase, which can process $\Sigma(a[i] \times b[i])$ as a single operation. Since MAC is composed of multiplication and addition operations, it is always the longest path in an ALU. In a RISC architecture that requires all instructions to be executed in one cycle, MAC becomes the bottleneck. Recent design utilizes VLIW wide-issue capability to implement MAC and remove the long critical path.

Division operation requires subtraction and shift operations for each divider bit, thus the path delay for a 32-bit division is extremely high. Many general purpose processors use floating-point unit to compute integer division. Rare fixed-point embedded processor supports division operation. In that case, a compiler is used to convert division into a loop with subtraction and shift operations to reduce hardware cost.

Saturation arithmetic is useful for multimedia applications. When two image pixels or audio samples are mixed, their intensions are added. By typical integer addition, mixed white pixel will become light gray when its MSB is truncated. To avoid the wrong result, software should check all pixels and keep the mixed intensity as a maximum white value when overflow occurs, which is very heavy work. The saturation arithmetic instructions are thus implemented in hardware to saturate the overflow/underflow result to an upper/lower bound to reduce the error.

Permutation operations are helpful in many algorithms. Datatype conversion is the basic permutation operation in all processor. Reverse and butterfly ordering is widely used in FIR and FFT. Many symmetric-key cryptographic algorithms such as DES and AES are based on complex permutation. The selection of permutation instructions to implement is very different between embedded processors.

### 4.2 Data Level Parallelism

Vector supercomputer was developed in the 1960s to increase the scientific computation speed. Since scientific program codes contain many one-dimensional vector and two-dimensional matrix operations, using a vector processor can perform these operations simultaneously to improve performance. A vector processor is also called a single-instruction multiple-data (SIMD) machine because it can apply one instruction on many data elements. Such kind of parallelism is often called data level parallelism (DLP).

#### 4.2.1 SIMD

Two main vector processing techniques will be introduced in this section. One uses processor array; ILLIAC-IV [38] is a representative example; Another uses automatic looping; a good example is Cray-1 [39].

ILLIAC-IV has 256 processing elements (PEs), which are partitioned into four groups; each group contains sixty-four PEs and one control unit (CU). The sixty-four PEs are structured as an $8 \times 8$ array as shown in Fig. 4.1. Each PE can communicate