2.6 Digital signal processors (DSPs)and multiprocessor structures with DSPs

The first practical single-chip Digital Signal Processors (DSP) were introduced around 1980. Because of their real-time processing capability, high throughput and intensive math-processing capability, DSPs began to replace general-purpose processors in many applications. These applications were well suited for real-time processing such as speech processing, telecommunications and high-speed control. They also pushed DSP to the forefront of technology and created one of the fastest going markets of decade.

During the past decade, while CPU performance increased from 5 MIPS in early 1980s to over 100 MIPS today, application performance developed exponentially. especially in imaging, graphics and high-end data processing. The exponentially growing requirements in conjunction with the “silicon wall” has created a situation in which application needs have vastly out-paced the ability of single processors to keep up. This condition inspired rapid development in parallel processing. Currently, 75 to 80% of all 32-bit, floating-point DSP applications use multiple processors in their design for several reasons. First, DSP algorithms are inherently suited to task partitioning and thus, to parallel processing solutions. Second, as the cost of single-chip DSPs decrease, using multiple DSPs in a system becomes increasingly cost effective. Third, the high data throughput, real-time processing capability, and intrinsic on-chip parallelism of DSPs make them especially suitable for multiprocessing systems.

Parallel processing enhancements can be divided into two broad categories: on-chip and off-chip. On-chip parallelism relies on architectural enhancements for improved performance, while off-chip parallelism incorporates additional processors.

2.6.1 On-Chip Parallel Processing

Each DSPs have some on-chip parallelism. One of the common properties of DSPs is the Harvard architecture is also an on-chip parallelism that allows s higher speed. Besides this one, architectural enhancements on DSPs can be grouped into three categories: superpipelining, superscaling and multi-CPU integration. Most of the new DSP families combine these solutions, therefore they cannot be always realized in pure form.

Superpipelining

This technique breaks the instruction pipeline into smaller pipeline stages, allowing the CPU to start executing the next instruction before completing the previous one. The processor can run multiple instructions simultaneously, with each instruction being at different stage of completion.

The main drawbacks of this technique are the increased level of control logic on the processor, difficulty in task switching. Real-time multitasking on a superpipelined processor can become impossible if pipeline grows too deep.

A good example among the DSPs for superpipelining is the TMS320C62xx processor family from the Texas Instrument. The processor has 12 item long pipeline:

Pipeline	**Pipeline Phase**		**Instruction Types Completed**
Program Fetch	PG	Program Address Generate	
	PS	Program Address Send	
	PW	Program Wait	
	PR	Program Data Receive	
Program Decode	DP	Dispatch	
	DC	Decode	
Execute	E1	Execute 1	Single-cycle
	E2	Execute 2	Multiplies
	E3	Execute 3	
	E4	Execute 4	
	E5	Execute 5	Loads
	E6	Execute 6	Branches

The result of this long pipeline is that for example after the branch instructions there are five delay slots. That means, the processor executes five following instructions before executing the branch.

The software pipelining is the technique that able to handle this long delay slots. The example in the next subsection introduces the software pipelining technique and shows how hard to read (and write) a code written for a processor that uses superpipelining. Fortunately the high level compilers and optimizers can hide these difficulties.

Superscalar and VLIW

Instead of breaking the pipeline into smaller stages, superscalar and VLIW (Very Long Instruction Word) architecture processors creates multiple pipeline, allowing the CPU to execute multiple instructions simultaneously.

However, when multiple instructions are executed simultaneously, any data dependency between the instructions (such as a conditional branch) increases the complexity of the programming. Programmers must make certain that simultaneously executed instructions don't need the same on-chip resource, or that one executing instruction doesn't need the result of another whose result is not yet available.

The TMS320C62xx processor family is also a good example for VLIW architecture. Using 8*32 long instruction word, in optimal case it can execute eight instructions simultaneously on its 2*4 execution units. The following example shows a short program written in C and in optimized assembly.

Dot-product C code :

```
int dotp(short a[], short b[] )
{
int sum, i;
sum = 0;

for( i=0; i<100; i++ )
     sum += a[i] * b[i];

return( sum );
}
```

Dot-product optimized assembly code :

```
        LDW   .D1   *A4++,A2    ; load ai & ai+1 from memory
||      LDW   .D2   *B4++,B2    ; load bi & bi+1 from memory
||      MVK   .S1   50,A1       ; set up loop counter
||      ZERO  .L1   A7          ; zero out sum0 accumulator
||      ZERO  .L2   B7          ; zero out sum1 accumulator

   [A1] SUB   .S1   A1,1,A1     ; decrement loop counter
||      LDW   .D1   *A4++,A2    ;* load ai & ai+1 from memory
||      LDW   .D2   *B4++,B2    ;* load bi & bi+1 from memory

   [A1] SUB   .S1   A1,1,A1     ;* decrement loop counter
|| [A1] B     .S2   LOOP        ; branch to loop
||      LDW   .D1   *A4++,A2    ;** load ai & ai+1 from memory
||      LDW   .D2   *B4++,B2    ;** load bi & bi+1 from memory
```

```
   [A1] SUB   .S1   A1,1,A1     ;** decrement loop counter
|| [A1] B     .S2   LOOP        ;* branch to loop
||      LDW   .D1   *A4++,A2    ;*** load ai & ai+1 from memory
||      LDW   .D2   *B4++,B2    ;*** load bi & bi+1 from memory

   [A1] SUB   .S1   A1,1,A1     ;*** decrement loop counter
|| [A1] B     .S2   LOOP        ;** branch to loop
||      LDW   .D1   *A4++,A2    ;**** load ai & ai+1 from memory
||      LDW   .D2   *B4++,B2    ;**** load bi & bi+1 from memory

        MPY   .M1X  A2,B2,A6    ; ai * bi
||      MPYH  .M2X  A2,B2,B6    ; ai+1 * bi+1
|| [A1] SUB   .S1   A1,1,A1     ;**** decrement loop counter
|| [A1] B     .S2   LOOP        ;*** branch to loop
||      LDW   .D1   *A4++,A2    ;***** ld ai & ai+1 from memory
||      LDW   .D2   *B4++,B2    ;***** ld bi & bi+1 from memory

        MPY   .M1X  A2,B2,A6    ;* ai * bi
||      MPYH  .M2X  A2,B2,B6    ;* ai+1 * bi+1
|| [A1] SUB   .S1   A1,1,A1     ;***** decrement loop counter
|| [A1] B     .S2   LOOP        ;**** branch to loop
||      LDW   .D1   *A4++,A2    ;****** ld ai & ai+1 from memory
||      LDW   .D2   *B4++,B2    ;****** ld bi & bi+1 from memory

LOOP:
        ADD   .L1   A6,A7,A7    ; sum0 += (ai * bi)
||      ADD   .L2   B6,B7,B7    ; sum1 += (ai+1 * bi+1)
||      MPY   .M1X  A2,B2,A6    ;** ai * bi
||      MPYH  .M2X  A2,B2,B6    ;** ai+1 * bi+1
|| [A1] SUB   .S1   A1,1,A1     ;****** decrement loop counter
|| [A1] B     .S2   LOOP        ;***** branch to loop
||      LDW   .D1   *A4++,A2    ;******* ld ai & ai+1 fm memory
||      LDW   .D2   *B4++,B2    ;******* ld bi & bi+1 fm memory
        ; Branch occurs here
        ADD   .L1X  A7,B7,A4    ; sum = sum0 + sum1
```

The combination of superpipelining and VLIW architectures results a very fast processor. The optimized assembly code in the example requires only 48 cycles, and the cycle time can be very short (5 ns), because of the superpipelining technique. The price of this effectiveness is the hardly readable code.

Multi-CPU Integration

This technique goes a step further than the preceding techniques and integrates multiple CPUs into a single piece of silicon. The number of processors may vary, depending on chip size, power dissipation, and pin count.

The first multiprocessor DSP on the market was the TMS320C8x family. The TMS320C8x architecture contains up to four parallel processors and one master processor. The master processor (MP) is a 32-bit RISC (reduced instruction set computer) with an integral IEEE-754 floating-point unit. The MP's general-purpose processing capabilities make it ideal for coordinating on-chip processing activities and for communicating with external devices.

Each parallel processor (PP) is an advanced 32-bit DSP. In addition to having the processing capabilities of conventional DSPs, a PP has special features to speed up operations on image data and to accelerate applications that manipulate data organized into bit fields. A PP's wide instruction word, three-operand arithmetic logic unit (ALU), and single-cycle multiplier enable it to perform a number of RISC-like operations in a single clock cycle.

Communication between the MP and PPs takes place over a high-speed crossbar network that provides simultaneous access to multiple banks of on-chip RAM. Software tools allow you to write programs in C and assembly language for the MP and PPs.

In addition to the fully programmable processors, the chip contains an intelligent DMA controller, called the transfer controller (TC). The TC manages all memory traffic by performing several services. One service takes the form of packet transfers that move data between two memory regions.

To provide on-chip storage, the TMS320C80 offers 50 KB of SRAM and the TMS320C82 offers 44 KB of SRAM. On-chip RAM, with the exception of the caches, is shared among the processors to support a variety of parallel-processing configurations.

2.6.2 Off-Chip Parallel Processing

Off-chip parallel processing is not necessarily better, it is inevitable. No single processor, no matter how it is pipelined, how it is scaled, or how many CPUs it has on board, can handle all applications. Recognizing this, manufacturers developed techniques to integrate multiple processors efficiently. Like building blocks, off-chip parallel processors connect easily to form expandable systems of virtually infinite size and variety.

Several processors employ more or less this technique. These processors also incorporate on-chip parallel processing features to achieve high individual performance. These processors offer both the high performance of the on-chip parallel processing architectural enhancements and the extra features of off-chip expansion.

Off-chip expansion is achieved by connecting multiple processors together with zero glue logic for direct processor-to-processor communication. While methods are different, the concept is the same: connect multiple processors together to create a topology or array of virtually any size to achieve the performance needed by high-end applications. The communication ports (or links) on the devices are supplemented by parallel memory buses and other support peripherals, allowing designers broad flexibility in designing their systems.

These are some benefits of off-chip parallelism:

Expandability You can easily add more processors to your system to meet performance requirements.

Flexibility You can implement a wide array of processor topologies that best fit your application needs. Unlike hardwired multi-CPU integration, off-chip processing can implement everything from 1D pipelines to 4D hypercubes.

Upgradability With processors that connect like building blocks, systems can be designed in a modular fashion, allowing extra processing power to be added at a later date to meet expanding processing needs.

One of the largest problems facing developers of multiprocessing systems is programming. Issues such as program partitioning, load balancing, and program routing present unique difficulties. Various solutions have been offered: parallel and graphical programming languages and some operating systems that support the real-time multitasking systems.

The most widely used DSP in multiprocessor systems is the TMS320C40 processor from the Texas instrument. The TMS320C4x devices are 32-bit floating-point digital signal processors optimized for parallel processing. The TMS320C4x family combines a high performance CPU and DMA controller with up to six communication ports to meet the needs of multiprocessor and I/O-intensive applications. All TMS320C4x devices are compatible with TI's multi-chip development environment. Each device contains an on-chip analysis module, which supports hardware breakpoints for parallel-processing development and debugging.

Key features of the TMS320C4x:

- Up to 40 MIPS/80 MFLOPS performance with 488-Mbytes/s I/O capability
  - IEEE floating-point conversion for ease of use
  - Register-based CPU
  - Single-cycle byte and half-word manipulation capabilities
  - Divide and square root support for improved performance
- On-chip memory includes 2K words of SRAM, 128 words of program cache, and bootloader
- Two external buses providing an address reach of up to 4 gigawords
- Two memory-mapped 32-bit timers
- 6 and 12 channel DMA
- Up to six communication ports for multiprocessor communication
- Idle mode for reduced power consumption