UNIT V - DIGITAL SIGNAL PROCESSOR

Introduction

General-purpose digital signal processors are basically high speed microprocessors with hard ware architecture and instruction set optimized for DSP operations. These processors make extensive use of parallelism, Harvard architecture, pipelining and dedicated hardware whenever possible to perform time consuming operations

There are two types of special purpose DSP processors based on hardware. (i) Hardware designed for efficient execution of specific DSP algorithms such as digital filter, FFT. (ii) Hardware designed for specific applications, for example telecommunication, digital audio.

The principal feature of Harvard architecture is that the program and the data memories lie in two separate spaces, permitting full overlap of instruction fetch and execution.

Typically these types of instructions would involve their distinct type.

- Instruction fetch
- Instruction decode
- Instruction execute

The way to implement the correlation and convolution is array multiplication Method. For getting down these operations we need the help of adders and multipliers. The combination of these accumulator and multiplier is called as multiplier accumulator.

There are two types MAC'S available

- Dedicated & integrated
- Separate multiplier and integrated unit

The pipeline technique is used to allow overall instruction executions to overlap. That is where all four phases operate in parallel. By adapting this technique, execution speed is increased.

What are four phases available in pipeline technique

The four phases are

- Fetch
- Decode
- Read

- Execution

In a non-pipeline machine, the instruction fetch, decode and execute take 30ns, 45 ns and 25 ns respectively. Determine the increase in throughput if the instruction were pipelined.

Write down the name of the addressing modes.

- Direct addressing.
- Indirect addressing.
- Bit-reversed addressing.
- Immediate addressing.
- Short immediate addressing
- Long immediateaddressing
- Circular addressing

Write the name of various part of C5X hardware.

Central arithmetic logic unit (CALU)

Parallel logic unit (PLU)

Auxiliary register arithmetic unit (ARAU)

List the various registers used with ARAU of DSP processor

- Eight auxiliary registers
- Auxiliary register pointer
- Unsigned 16 bit ALU

Write short notes about arithmetic logic unit and accumulator.

The 32-bit general-purpose ALU and ACC implement a wide range of arithmetic and logical functions, the majority of which execute in a single clock cycle. Once an operation is performed in the ALU, the result is transferred to the ACC, where additional operations, such as shifting, can occur. Data that is input to the ALU can be scaled by the prescaler.

The following steps occur in the implementation of a typical ALU instruction:

- Data is fetched from memory on the data bus,
- Data is passed through the prescaler and the ALU, where the arithmetic is performed, and
- The result is moved into the ACC.

The ALU operates on 16-bit words taken from data memory or derived from immediate instructions. In addition to the usual arithmetic instructions, the ALU can perform Boolean operations, thereby facilitating the bit manipulation ability required of high-speed controller. One input to the ALU is always supplied by the ACC. The other input can be transferred from the PREG of the multiplier, the ACCB, or the output of the prescaler. After the ALU has

performed the arithmetic or logical operation, the result is stored in the ACC

Questions & Answers:

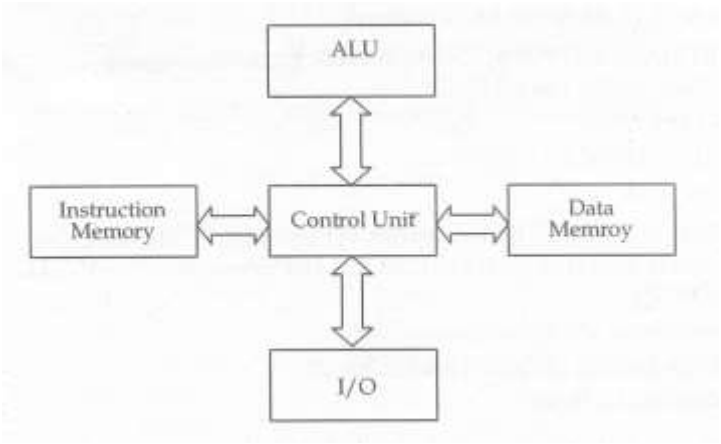
1. **Draw the block diagram of Von Neumann, Harvard architecture and modified Harvard**

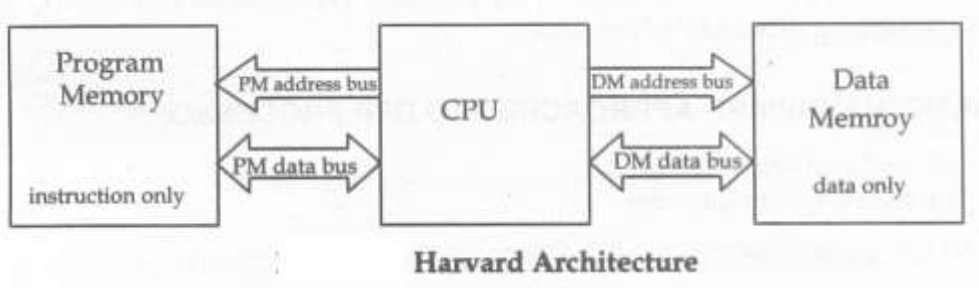
architecture and explain. (Nov/Dec 2012) (Nov/Dec 2013) (16)

Solution:

HARVARD ARCHITECTURE

The **Harvard architecture** is a computer architecture with physically separate storage and signal pathways (buses) for instructions and data. The term originated from the **Harvard Mark I** relay-based computer, which stored instructions on punched tape (24 bits wide) and data in electro-mechanical counters. These early machines had data storage entirely contained within the central processing unit, and provided no access to the instruction storage as data. Programs needed to be loaded by an operator; the processor could not initialize itself. Today, most processors implement such separate signal pathways for performance reasons, but actually implement a modified Harvard architecture, so they can support tasks like loading a program from disk storage as data and then executing it.





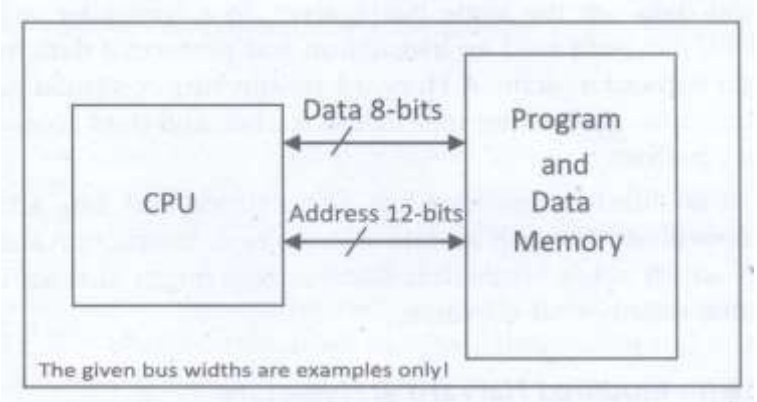
Harvard Architecture

VON NEUMANN ARCHITECTURE

The von Neumann architecture, also known as the von Neumann model and Princeton architecture, is a computer architecture based on that described in 1945 by the mathematician and physicist **John von Neumann** and others .This describes a design architecture for an electronic digital computer with parts consisting of a processing unit containing an arithmetic logic unit and processor registers, a control unit containing an instruction register and program counter, a memory to store both data and instructions, external mass storage, and input and output mechanisms. The meaning has evolved to be any stored-program computer in which an instruction fetch and a data operation cannot occur at the same time because they share a common bus. This is referred to as the von Neumann bottleneck and often limits the performance of the system.

The design of a von-Neumann architecture is simpler than the more modern Harvard architecture which is also a stored-program system but has one dedicated set of address and data buses for reading data from and writing data to memory, and another set of address and data buses for fetching instructions.

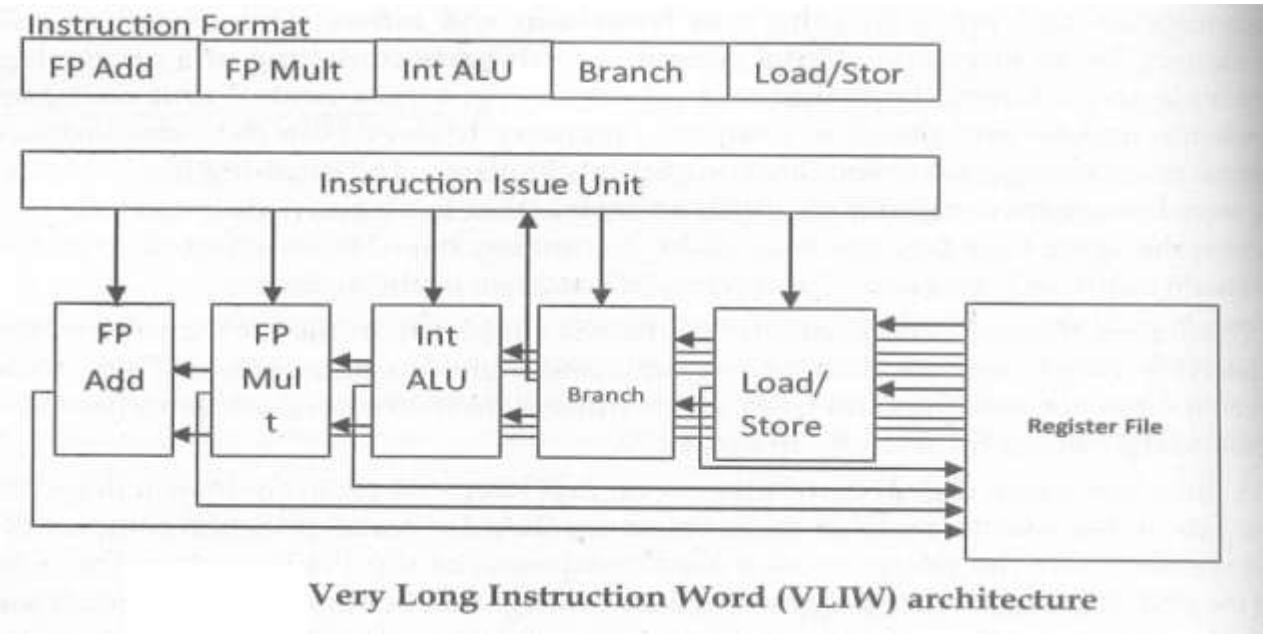
A stored-program digital computer is one that keeps its program instructions, as well as its data, in read-write, random-access memory (RAM). Stored-program computers were advancement over the program-controlled computers of the 1940s, such as the Colossus and the ENIAC, which were programmed by setting switches and inserting patch leads to route data and to control signals between various functional units. In the vast majority of modern computers, the same memory is used for both data and program instructions, and the von Neumann vs. Harvard distinction applies to the cache architecture, not the main memory.



VON NEUMANN ARCHITECTURE

VERY LONG INSTRUCTION WORD (VLIW) OR MODIFIED VON NEUMANN ARCHITECTURE

Very Long Instruction Word (VLIW) or **modified von Neumann architecture** refers to processor architectures designed to take advantage of Instruction Level Parallelism (ILP). Whereas conventional processors mostly allow programs only to specify instructions that will be executed in sequence, a VLIW processor allows programs to explicitly specify instructions that will be executed at the same time (that is in parallel). This type of processor architecture is intended to allow higher performance without the inherent complexity of some other approaches.



Very Long Instruction Word (VLIW) architecture

Contrast with von Neumann architectures

Under pure von Neumann architecture the CPU can be either reading an instruction or reading/writing data from/to the memory. Both cannot occur at the same time since the instructions and data use the same bus system. In a computer using the Harvard architecture, the CPU can both read an instruction and perform a data memory access at the same time, even without a cache. A Harvard architecture computer can thus be faster for a given circuit complexity because instruction fetches and data access do not contend for a single memory pathway.

Also, a Harvard architecture machine has distinct code and data address spaces: instruction address zero is not the same as data address zero. Instruction address zero might identify a twenty-four bit value, while data address zero might indicate an eight-bit byte that is not part of that twenty-four bit value.

Contrast with modified Harvard architecture

A modified Harvard architecture machine is very much like a Harvard architecture machine, but it relaxes the strict separation between instruction and data while still letting the CPU concurrently access two (or more) memory buses. The most common modification includes separate instruction and data caches backed by a common address space. While the CPU executes from cache, it acts as a pure Harvard machine. When accessing backing memory, it acts like a von Neumann machine (where code can be moved around like data which is a powerful technique). This modification is widespread in modern processors, such as the ARM architecture and x86 processors. It is sometimes loosely called Harvard architecture, overlooking the fact that it is actually "modified".

Another modification provides a pathway between the instruction memory (such as ROM or flash memory) and the CPU to allow words from the instruction memory to be treated as read-only data. This technique is used in some microcontrollers, including the Atmel AVR. This allows constant data, such as text strings or function tables, to be accessed without first having to be copied into data memory, preserving scarce (and power-ungry) data memory for read/write variables. Special machine language instructions are provided to read data from the instruction memory. (This is distinct from instructions which themselves embed constant data, although for individual constants the two mechanisms can substitute for each other.)

VAN-NEUMANN Architecture	Harvard Architecture
Used in conventional processors found in PCs and Servers, and embedded systems with only control functions.	Used in DSPs and other processors found in latest embedded systems and Mobile communication systems, audio, speech, image processing systems
The data and program are stored in the same memory	The data and program memories are separate
The code is executed serially and takes more clock cycles	The code is executed in parallel
There is no exclusive Multiplier	It has MAC (Multiply Accumulate)
Absence of Barrel Shifter	Barrel Shifter help in shifting and rotating operations of the data
The programs can be optimized in lesser size	The program tend to grow large in size

Comparison of Van-Neumann Architecture and Harvard Architecture

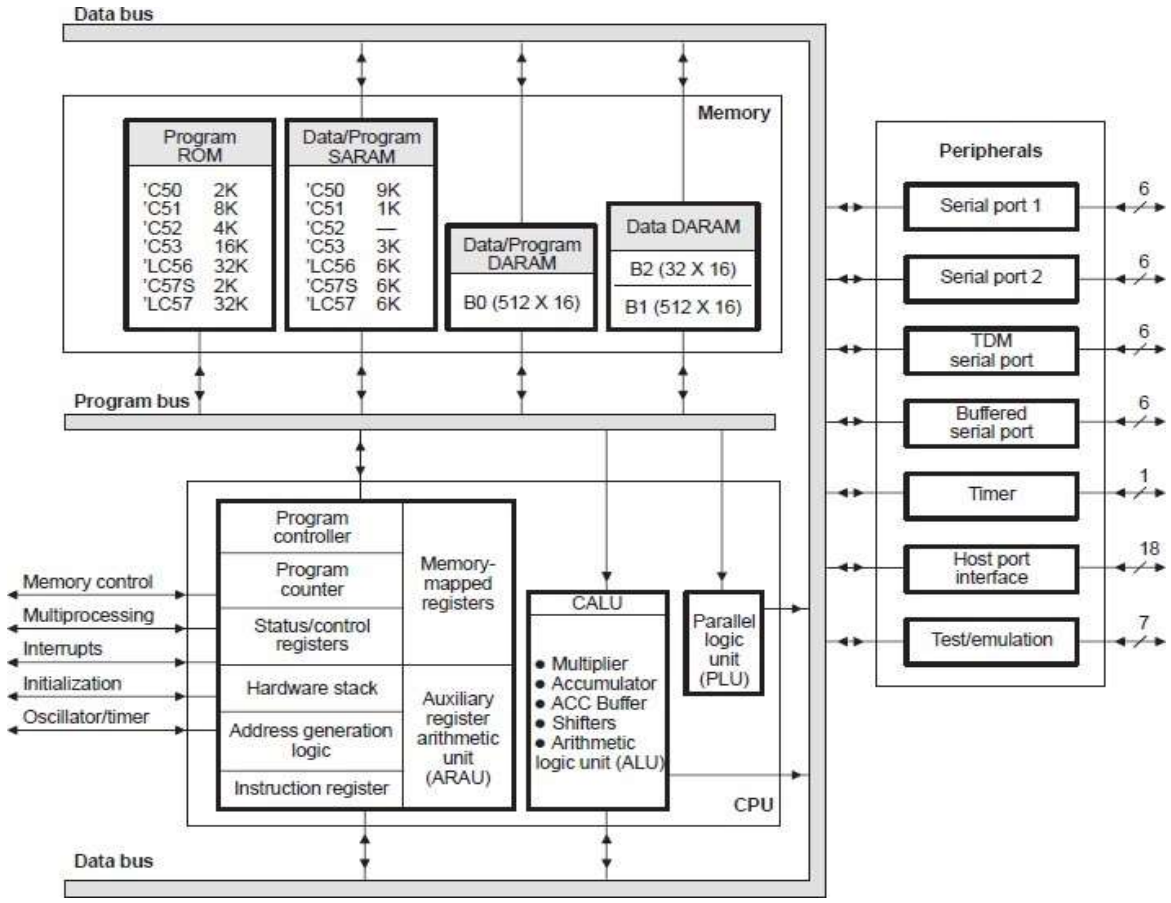
2. **Draw and explain the architecture of TMS 320C54x processor** (Apr/May 2015) (Nov/Dec 2014)(May/June 2014) (May/June 2013) (May/June 2012)

The architectural structure of the C5x which consists of the buses, on-chip memory, central processing unit (CPU)and on-chip peripherals.

The ’C5x uses an advanced, modified Harvard-type architecture based on the ’C25 architecture and maximizes processing power with separate buses for program memory and data memory. The instruction set supports data transfers between the two memory spaces. The Figure shown below is a functional block diagram of the ’C5x. All ’C5x DSPs have the same CPU structure; however, they have different on-chip memory configurations and on-chip peripherals

Continue on next page…

Functional Block Diagram



Bus Structure

Separate program and data buses allow simultaneous access to program instructions and data, providing a high degree of parallelism. For example, while data is multiplied, a previous product can be loaded into, added to, or subtracted from the accumulator and, at the same time, a new address can be generated. Such parallelism supports a powerful set of arithmetic, logic, and bit-manipulation operations that can all be performed in a single machine cycle. In addition, the ’C5x includes the control mechanisms to manage interrupts, repeated operations, and function calling.

The ’C5x architecture is built around four major buses:

- Program bus (PB)

- Program address bus (PAB)
- Data read bus (DB)
- Data read address bus (DAB)

The PAB provides addresses to program memory space for both reads and writes. The PB also carries the instruction code and immediate operands from program memory space to the CPU. The DB interconnects various elements of the CPU to data memory space. The program and data buses can work together to transfer data from on-chip data memory and internal or external program memory to the multiplier for single-cycle multiply/accumulate operations

Central Processing Unit (CPU)

The ’C5x CPU consists of these elements:

- Central arithmetic logic unit (CALU)
- Parallel logic unit (PLU)
- Auxiliary register arithmetic unit (ARAU)
- Memory-mapped registers
- Program controller

The ’C5x CPU maintains source-code compatibility with the ’C1x and ’C2x generations while achieving high performance and greater versatility. Improvements include a 32-bit accumulator buffer, additional scaling capabilities, and a host of new instructions. The instruction set exploits the additional hardware features and is flexible in a wide range of applications. Data management has been improved through the use of new block move instructions and memory-mapped register instructions.

Central Arithmetic Logic Unit (CALU)

The CPU uses the CALU to perform 2s-complement arithmetic. The CALU consists of these elements:

- 16-bit x 16-bit multiplier
- 32-bit arithmetic logic unit (ALU)
- 32-bit accumulator (ACC)
- 32-bit accumulator buffer (ACCB)
- Additional shifters at the outputs of both the accumulator and the product register (PREG)

Parallel Logic Unit (PLU)

The CPU includes an independent PLU, which operates separately from, but in parallel with, the ALU. The PLU performs Boolean operations or the bit manipulations required of high-speed controllers. The PLU can set, clear, test, or toggle bits in a status register, control register, or any data memory location.

The PLU provides a direct logic operation path to data memory values without affecting the contents of the ACC or PREG. Results of a PLU function are written back to the original data memory location.

Auxiliary Register Arithmetic Unit (ARAU)

The CPU includes an unsigned 16-bit arithmetic logic unit that calculates indirect addresses by using inputs from the auxiliary registers (ARs), index register (INDX), and auxiliary register compare register (ARCR). The ARAU can autoindex the current AR while the data memory location is being addressed and can index either by 1 or by the contents of the INDX. As a result, accessing data does not require the CALU for address manipulation; therefore, the CALU is free for other operations in parallel. For information on the ARAU.

Memory-Mapped Registers

The ’C5x has 96 registers mapped into page 0 of the data memory space. All ’C5x DSPs have 28 CPU registers and 16 input/output (I/O) port registers but have different numbers of peripheral and reserved registers. Since the memory-mapped registers are a component of the data memory space, they can be written to and read from in the same way as any other data memory location. The memory-mapped registers are used for indirect data address pointers, temporary storage, CPU status and control, or integer arithmetic processing through the ARAU.

Program Controller

The program controller contains logic circuitry that decodes the operational instructions, manages the CPU pipeline, stores the status of CPU operations, and decodes the conditional operations. Parallelism of architecture lets the ’C5x perform three concurrent memory operations in any given machine cycle: fetch an instruction, read an operand, and write an operand. The program controller consists of these elements:

- Program counter
- Status and control registers
- Hardware stack
- Address generation logic

- Instruction register

On-Chip Memory

The ’C5x architecture contains a considerable amount of on-chip memory to aid in system performance and integration:

- Program read-only memory (ROM)
- Data/program dual-access RAM (DARAM)
- Data/program single-access RAM (SARAM)

The ’C5x has a total address range of 224K words x 16 bits. The memory space is divided into four individually selectable memory segments: 64K-word program memory space, 64K-word local data memory space, 64K-word input/ output ports, and 32K-word global data memory space.

Program ROM

All ’C5x DSPs carry a 16-bit on-chip maskable programmable ROM. The ’C50 and ’C57S DSPs have boot loader code resident in the on-chip ROM, all other ’C5x DSPs offer the boot loader code as an option. This memory is used for booting program code from slower external ROM or EPROM to fast on-chip or external RAM. Once the custom program has been booted into RAM, the boot ROM space can be removed from program memory space by setting the MP/MC bit in the processor mode status register (PMST). The on-chip ROM is selected at reset by driving the MP/MC pin low. If the on-chip ROM is not selected, the ’C5x devices start execution from off-chip memory. The on-chip ROM may be configured with or without boot loader code. However, the on-chip ROM is intended for your specific program. Once the program is in its final form, you can submit the ROM code to Texas Instruments for implementation into your device.

Data/Program Dual-Access RAM

All ’C5x DSPs carry a 1056-word x 16-bit on-chip dual-access RAM (DARAM). The DARAM is divided into three individually selectable memory blocks:

512-word data or program DARAM block B0, 512-word data DARAM block B1, and 32-word data DARAM block B2. The DARAM is primarily intended to store data values but, when needed, can be used to store programs as well. DARAM blocks B1 and B2 are always configured as data memory; however, DARAM block B0 can be configured by software as data or program memory. The DARAM can be configured in one of two ways:

- All 1056 words × 16 bits configured as data memory
- 544 words × 16 bits configured as data memory and 512 words × 16 bits configured as program memory

DARAM improves the operational speed of the ’C5x CPU. The CPU operates with a 4-deep pipeline. In this pipeline, the CPU reads data on the third stage and writes data on the fourth stage. Hence, for a given instruction sequence, the second instruction could be reading data at the same time the first instruction is writing data. The dual data buses (DB and DAB) allow the CPU to read from and write to DARAM in the same machine cycle.

Data/Program Single-Access RAM

All ’C5x DSPs except the ’C52 carry a 16-bit on-chip single-access RAM (SARAM) of various sizes (see Table 1–1). Code can be booted from an offchip ROM and then executed at full speed, once it is loaded into the on-chip SARAM. The SARAM can be configured by software in one of three ways:

- All SARAM configured as data memory
- All SARAM configured as program memory
- SARAM configured as both data memory and program memory

The SARAM is divided into 1K- and/or 2K-word blocks contiguous in address memory space. All ’C5x CPUs support parallel accesses to these SARAM blocks. However, one SARAM block can be accessed only once per machine cycle. In other words, the CPU can read from or write to one SARAM block while accessing another SARAM block. When the CPU requests multipleaccesses, the SARAM schedules the accesses by providing a not-ready condition to the CPU and executing the multiple accesses one cycle at a time. SARAM supports more flexible address mapping than DARAM because SARAM can be mapped to both program and data memory space simultaneously.However, because of simultaneous program and data mapping, an instruction fetch and data fetch that could be performed in one machine cycle with DARAM may take two machine cycles with SARAM.

On-Chip Memory Protection

The ’C5x DSPs have a maskable option that protects the contents of on-chip memories. When the related bit is set, no externally originating instruction can access the on-chip memory spaces.

On-Chip Peripherals

All ’C5x DSPs have the same CPU structure; however, they have different onchip peripherals connected to their CPUs. The ’C5x DSP on-chip peripherals available are:

- Clock generator
- Hardware timer
- Software-programmable wait-state generators
- Parallel I/O ports
- Host port interface (HPI)
- Serial port
- Buffered serial port (BSP)
- Time-division multiplexed (TDM) serial port
- User-maskable interrupts

Clock Generator

The clock generator consists of an internal oscillator and a phase-locked loop (PLL) circuit. The clock generator can be driven internally by a crystal resonator circuit or driven externally by a clock source. The PLL circuit can generate an internal CPU clock by multiplying the clock source by a specific factor, so

you can use a clock source with a lower frequency than that of the CPU.

Hardware Timer

A 16-bit hardware timer with a 4-bit prescaler is available. This programmable timer clocks at a rate that is between 1/2 and 1/32 of the machine cycle rate (CLKOUT1), depending upon the timer’s divide-down ratio. The timer can be stopped, restarted, reset, or disabled by specific status bits.

Software-Programmable Wait-State Generators

Software-programmable wait-state logic is incorporated in ’C5x DSPs allowing wait-state generation without any external hardware for interfacing with slower off-chip memory and I/O devices. This feature consists of multiple waitstate generating circuits. Each circuit is user-programmable to operate in different wait states for off-chip memory accesses.

Parallel I/O Ports

A total of 64K I/O ports are available, sixteen of these ports are memory-mapped in data memory space. Each of the I/O ports can be addressed by the IN or the OUT instruction. The memory-mapped I/O ports can be accessed with any instruction that reads from or writes to data memory. The IS signal indicates a read or write operation through an I/O port. The ’C5x can easily interface with external I/O devices through the I/O ports while requiring minimal off-chip address decoding circuits.

Host Port Interface (HPI)

The HPI available on the ’C57S and ’LC57 is an 8-bit parallel I/O port that provides an interface to a host processor. Information is exchanged between the DSP and the host processor through on-chip memory that is accessible to both the host processor and the ’C57.

Number of Serial/Parallel Ports Available in Different ’C5x Package Types

TMS320 Device	Package ID†	High-Speed Serial Port	TDM Serial Port	Buffered Serial Port	Host Port (Parallel)
’C50/’LC50	PQ	1	1	–	–
’C51/’LC51	PQ/PZ	1	1	–	–
’C52/’LC52	PJ/PZ	1	–	–	–
’C53/’LC53	PQ	1	1	–	–
’C53S/’LC53S	PZ	2	–	–	–
’LC56	PZ	1	–	1	–
’C57S/’LC57S	PGE	1	–	1	1
’LC57	PBK	1	–	1	1

† PGE is a 20 × 20 × 1.4 mm thin quad flat-pack (TQFP) package
PJ is a 14 × 20 × 2.7 mm quad flat-pack (QFP) package
PQ is a 20 × 20 × 3.8 mm bumpered quad flat-pack (BQFP) package
PZ and PBK are a 14 × 14 × 1.4 mm thin quad flat-pack (TQFP) package

Serial Port

Three different kinds of serial ports are available: a general-purpose serial port, a time-division multiplexed (TDM) serial port, and a buffered serial port (BSP). Each ’C5x contains at least one general-purpose, high-speed synchronous, full-duplexed serial port interface that provides direct communication with serial devices such as codecs, serial analog-to-digital (A/D) converters, and other serial systems.

The serial port is capable of operating at up to onefourth the machine cycle rate (CLKOUT1). The serial port transmitter and receiver are double-buffered and individually controlled by maskable external interrupt signals. Data is framed either as bytes or as words.

Buffered Serial Port (BSP)

The BSP available on the ’C56 and ’C57 devices is a full-duplexed, doublebuffered serial port and an autobuffering unit (ABU). The BSP provides flexibility on the data stream length. The ABU supports high-speed data transfer and reduces interrupt latencies.

TDM Serial Port

The TDM serial port available on the ’C50, ’C51, and ’C53 devices is a fullduplexed serial port that can be configured by software either for synchronous operations or for time-division multiplexed operations. The TDM serial port is commonly used in multiprocessor applications.

User-Maskable Interrupts

Four external interrupt lines (INT1–INT4) and five internal interrupts, a timer interrupt and four serial port interrupts, are user maskable. When an interrupt service routine (ISR) is executed, the contents of the program counter are saved on an 8-level hardware stack, and the contents of eleven specific CPU registers are automatically saved (shadowed) on a 1-level-deep stack. When a return from interrupt instruction is executed, the CPU registers’ contents are restored.

CENTRAL PROCESSING UNIT (CPU)

The TMS320C5x DSP central processing unit (CPU) can perform high-speed arithmetic within a short instruction

cycle by means of its highly parallel architecture, which consists of the following elements:

- Program controller
- Central arithmetic logic unit (CALU)
- Parallel logic unit (PLU)
- Auxiliary register arithmetic unit (ARAU)
- Memory-mapped registers

Program Counter (PC)

The ’C5x has a 16-bit program counter (PC) which contains the address of internal or external program memory used to fetch instructions. The PC addresses program memory, either on-chip or off-chip, via the program address bus (PAB). Through the PAB, an instruction is loaded into the instruction register (IREG). Then the PC is ready to start the next instruction fetch cycle. Refer to Figure 4–1 for a functional block diagram of the program control elements.The PC is loaded in a number of ways. Table shows what address is loaded into the PC, depending on the code operation performed.

Applications of DSP Processor

Digital Signal Processors (DSPs) find applications in a wide range of fields due to their ability to efficiently process digital signals in real-time. Some common applications of DSP processors include:

Audio Processing:

DSP processors are extensively used in audio applications such as digital audio effects (reverb, equalization, compression), speech recognition, noise cancellation, audio synthesis, and audio compression algorithms like MP3 and AAC.

Image and Video Processing:

DSP processors are employed in image and video processing tasks such as image enhancement, object detection and tracking, video compression (e.g., MPEG), image recognition, and digital watermarking.

Wireless Communication:

DSP processors are fundamental in wireless communication systems including cellular networks (2G, 3G, 4G, and 5G), Wi-Fi, Bluetooth, Zigbee, and satellite communication. They are used for tasks such as modulation/demodulation, error correction coding, channel equalization, and beamforming.

Radar and Sonar Systems:

DSP processors play a critical role in radar and sonar systems for target detection, tracking, signal processing, and analysis. They are used in applications ranging from military radar systems to weather monitoring and automotive radar for collision detection.

Medical Imaging and Biomedical Signal Processing:

DSP processors are utilized in medical imaging modalities like MRI, CT scans, ultrasound, and PET scans for signal acquisition, processing, and reconstruction. They are also employed in biomedical signal

processing tasks such as ECG (electrocardiography), EEG (electroencephalography), and EMG (electromyography) for diagnosis and monitoring of various medical conditions.

Automotive Systems:

DSP processors are increasingly integrated into automotive systems for applications like engine control, anti-lock braking systems (ABS), active suspension systems, collision avoidance systems, in-car entertainment (e.g., audio processing), and advanced driver-assistance systems (ADAS) such as adaptive cruise control and lane departure warning.

Industrial Automation and Control:

DSP processors are used in industrial automation and control systems for tasks such as motor control, robotics, process control, power electronics, and monitoring of sensors and actuators in manufacturing environments.

Consumer Electronics:

DSP processors are present in various consumer electronics devices such as smartphones, digital cameras, camcorders, gaming consoles, smart TVs, and home theater systems for audio/video processing, image processing, and communication functionalities.

Aerospace and Defense:

DSP processors are employed in aerospace and defense applications for tasks like radar signal processing, communication systems, guidance systems, surveillance, electronic warfare, and satellite communication.

These applications demonstrate the versatility and importance of DSP processors in modern technology, where real-time processing of digital signals is crucial for achieving desired functionalities and performance requirements.

Q Explain Multirate signal processing

Multirate signal processing refers to the manipulation and analysis of signals at different rates or sampling frequencies. This technique is commonly used in digital signal processing (DSP) applications where signals need to be processed efficiently while taking into account the trade-offs between computational complexity, memory requirements, and desired performance.

There are several key concepts and techniques associated with multirate signal processing:

Sampling Rate Conversion:

This involves changing the sampling rate of a signal. Two common operations in sampling rate conversion are upsampling (increasing the sampling rate) and downsampling (decreasing the sampling rate). Upsampling involves inserting zeros between samples, followed by low-pass filtering to eliminate spectral images. Downsampling involves discarding samples, typically after low-pass filtering to prevent aliasing.

Filter Banks:

Multirate systems often use filter banks to process signals efficiently. A filter bank consists of multiple filters arranged in parallel or in series. In multirate signal processing, filter banks are used for operations such as decomposition (splitting a signal into multiple frequency bands) and reconstruction (combining multiple frequency bands to reconstruct the original signal).

Polyphase Representation:

This is a technique used to implement efficient filter banks. In polyphase representation, the impulse response of a filter is partitioned into multiple phases, allowing for more efficient computation. Polyphase representation is particularly useful in multirate filter banks where the filters operate at different rates.

Decimation and Interpolation:

Decimation refers to the process of reducing the sampling rate of a signal by a factor of L, while interpolation refers to increasing the sampling rate of a signal by a factor of L. Decimation and interpolation are often used in conjunction with filtering to achieve sampling rate conversion.

Multistage Processing:

In some cases, multirate signal processing algorithms can be decomposed into multiple stages, each operating at a different sampling rate. Multistage processing can help reduce computational complexity and memory requirements.

Multirate signal processing finds applications in various fields, including telecommunications, audio processing, image processing, and digital audio and video compression. It allows for efficient processing of signals while meeting the requirements of different applications in terms of computational resources and performance