5.2. ARCHITECTURE OF DIGITAL SIGNAL PROCESSOR

The TMS320C5X processor has an advanced version of Hardware architecture, with separate buses for program and data, which facilitate simultaneous access of program and data. The program bus has separate lines to transmit data and address. Similarly the data has separate lines to transmit data and address.

.

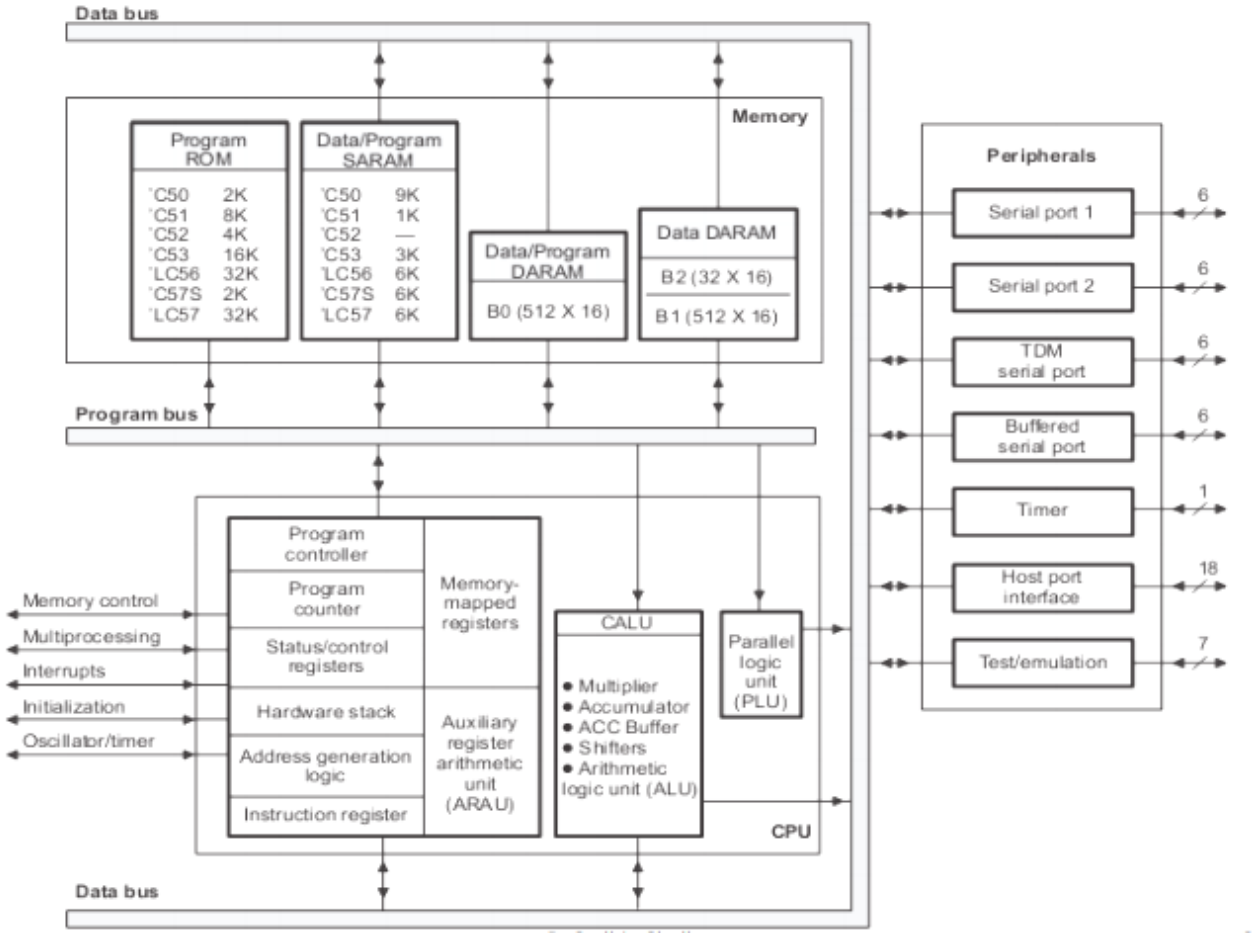

Fig.5.2.1.Internal architecture of TMS320C5X

[Source: 'Digital Signal Processing Principles, Algorithms and Applications' by J.G. Proakis and D.G. Manolakis page-674]

1.32-bit ALU/accumulator

The 32-bit ALU and accumulator implement a wide range of arithmetic and logical functions, the majority of which execute in a single cycle. The ALU is a general-purpose arithmetic/logic unit that 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, facilitating the bit manipulation ability required of a high-speed controller. One input to the ALU always is supplied by

the accumulator, and the other input can be furnished from the product register (PREG) of the multiplier, the accumulator buffer (ACCB), or the output of the scaling shifter [which has been read from data memory or from the accumulator (ACC)]. After the ALU performs the arithmetic or logical operation, the result is stored in the ACC where additional operations, such as shifting, can be performed. Data input to the ALU can be scaled by the scaling shifter.

The 32-bit ACC is split into two 16-bit segments for storage in data memory. Shifters at the output of the ACC provide a left shift of 0 to 7 places. This shift is performed while the data is being transferred to the data bus for storage. The contents of the ACC remain unchanged. When the post scaling shifter is used on the high word of the ACC (bits 31–16), the most significant bits (MSBs) are lost and the least significant bits (LSBs) are filled with bits shifted in from the low word (bits 15–0). When the post scaling shifter is used on the low word, the LSBs are filled with zeros.

2.Scaling shifters

The ’C5x provides a scaling shifter that has a 16-bit input connected to the data bus and a 32-bit output connected to the ALU. This scaling shifter produces a left shift of 0 to 16 bits on the input data. The shift count is specified by a constant embedded in the instruction word or by the value in TREG1. The LSBs of the output are filled with zeros; the MSBs may be either filled with zeros or sign extended, depending upon the value of the sign-extension mode (SXM) bit of status register ST1. The ’C5x also contains several other shifters that allow it to perform numerical scaling, bit extraction, extended-precision arithmetic, and overflow prevention. These shifters are connected to the output of the product register and the ACC.

3.Parallel logic unit

The parallel logic unit (PLU) is a second logic unit, additional to the main ALU, that executes logic operations on data without affecting the contents of the ACC. The PLU provides the bit-manipulation ability required of a high-speed controller and simplifies control/status register operations. The PLU provides a direct logic operation path to data memory space and can set, clear, test, or toggle multiple bits directly in a data memory location, a control/status register, or any register that is mapped into data memory space.

4.16 × 16-bit parallel multiplier

The ’C5x uses a 16 × 16-bit hardware multiplier that is capable of computing a signed or an unsigned 32-bit product in a single machine cycle. All multiply instructions, except the MPYU (multiply unsigned) instruction, perform a signed multiply operation in the multiplier. That is, two numbers being multiplied are treated as 2s-complement numbers, and the result is a 32-bit 2s-complement number. There are two registers associated with the multiplier: TREG0, a 16-bit temporary register that holds one of the operands for the multiplier, and PREG, the 32-bit product register that holds the product. Four product shift modes (PM) are available at the PREG’s output. These shift modes are useful for performing multiply/accumulate operations, performing fractional arithmetic, or justifying fractional products. The PM field of status register ST1 specifies the PM shift mode. The product can be shifted one bit to compensate for the extra sign bit gained in multiplying two 16-bit 2s-complement numbers (MPY).

A 4-bit shift is used in conjunction with the MPY instruction with a short immediate value (13 bits or less) to eliminate the four extra sign bits gained in multiplying a 16-bit number by a 13-bit number. Finally, the output of PREG can, instead, be right-shifted 6 bits to enable the execution of up to 128 consecutive multiply/accumulates without the possibility of overflow. The load-TREG0 (LT) instruction normally loads TREG0 to provide one operand (from the data bus), and the MPY instruction provides the second operand (also from the data bus). A multiplication also can be performed with a short or long immediate operand by using the MPY instruction with an immediate operand. A product is obtained every two cycles except when a long immediate operand is used.

5. Auxiliary registers and auxiliary-register arithmetic unit (ARAU)

The ’C5x provides a register file containing eight auxiliary registers (AR0–AR7). The auxiliary registers are used for indirect addressing of the data memory or for temporary data storage. Indirect auxiliary-register addressing allows placement of the data memory address of an instruction operand into one of the auxiliary registers. These registers are referenced with a 3-bit auxiliary register pointer (ARP) that is loaded with a value from 0 through 7, designated AR0 through AR7, respectively. The auxiliary

registers and the ARP can be loaded from data memory, the ACC, the product register, or by an immediate operand defined in the instruction. The contents of these registers can be stored in data memory or used as inputs to the central arithmetic logic unit (CALU). These registers are accessible as memory-mapped locations within the ’C5x data-memory space. The auxiliary register file (AR0–AR7) is connected to the auxiliary register arithmetic unit (ARAU). The ARAU can autoindex the current auxiliary register while the data memory location is being addressed. Indexing can be performed either by ±1 or by the contents of the INDX register. As a result, accessing tables of information does not require the CALU for address manipulation; thus, the CALU is free for other operations in parallel.

6. Memory

The ’C5x implements three separate address spaces for program memory, data memory, and I/O. Each space accommodates a total of 64K 16-bit words (see Figures 1 through 7). Within the 64K words of data space, the 256 to 32K words at the top of the address range can be defined to be external global memory in increments of powers of two, as specified by the contents of the global memory allocation register (GREG). Access to global memory is arbitrated using the global memory bus request (BR) signal. The ’C5x devices include a considerable amount of on-chip memory to aid in system performance and integration including ROM, single-access RAM (SARAM), and dual-access RAM (DARAM). The amount and types of memory available on each device are shown in Table 1. On the ’C5x, the first 96 (0–5Fh) data-memory locations are allocated for memory-mapped registers. This memory-mapped register space contains various control and status registers including those for the CPU, serial port, timer, and software wait-state generators. Additionally, the first 16 I/O port locations are mapped into this data-memory space, allowing them to be accessed either as data memory using single-word instructions or as I/O locations with two-word instructions. Two-word instructions allow access to the full 64K words of I/O space. The mask-programmable ROM is located in program memory space. Customers can arrange to have this ROM programmed with contents unique to to any particular application.

The ROM is enabled or disabled by the state of the MP/MC control input upon resetting the device or by manipulating the MP/MC bit in the PMST status register after

reset. The ROM occupies the lowest block of program memory when enabled. When disabled, these addresses are located in the device's external program-memory space. The 'C5x also has a mask-programmable option that provides security protection for the contents of on-chip ROM. When this internal option bit is programmed, no externally-originating instruction can access the on-chip ROM. This feature can be used to provide security for proprietary algorithms.

7. Interrupts and subroutines

The 'C5x implements four general-purpose interrupts, INT4–INT1, along with reset (RS) and the nonmaskable interrupt (NMI) which are available for external devices to request the attention of the processor. Internal interrupts are generated by the serial port (RINT and XINT), by the timer (TINT), and by the software-interrupt (TRAP, INTR, and NMI) instructions. Interrupts are prioritized with RS having the highest priority, followed by NMI, and INT4 having the lowest priority. Additionally, any interrupt except RS and NMI can be masked individually with a dedicated bit in the interrupt mask register (IMR) and can be cleared, set, or tested using its own dedicated bit in the interrupt flag register (IFR). The reset and NMI functions are not maskable. All interrupt vector locations are on two-word boundaries so that branch instructions can be accommodated in those locations. While normally located at program memory address 0, the interrupt vectors can be remapped to the beginning of any 2K-word page in program memory by modifying the contents of the interrupt vector pointer (IPTR) located in the PMST status register. A built-in mechanism protects multicycle instructions from interrupts. If an interrupt occurs during a multicycle instruction, the interrupt is not processed until the instruction completes execution. This mechanism applies to instructions that are repeated (using the RPT instruction) and to instructions that become multicycle because of wait states. Each time an interrupt is serviced or a subroutine is entered, the PC is pushed onto an internal hardware stack, providing a mechanism for returning to the previous context. The stack contains eight locations, allowing interrupts or subroutines to be nested up to eight levels deep.

8.Power-down modes

The 'C5x implements several power-down modes in which the 'C5x core enters a dormant state and dissipates considerably less power. A power-down mode is invoked

either by executing the IDLE/IDLE2 instructions or by driving the HOLD input low. When the HOLD signal initiates the power-down mode, on-chip peripherals continue to operate; this power-down mode is terminated when HOLD goes inactive. While the ’C5x is in a power-down mode, all internal contents are maintained; this allows operation to continue unaltered when the power-down mode is terminated. All CPU activities are halted when the IDLE instruction is executed, but the CLKOUT1 pin remains active. The peripheral circuits continue to operate, allowing peripherals such as serial ports and timers to take the CPU out of its powered-down state. A power-down mode, when initiated by an IDLE instruction, is terminated upon receipt of an interrupt.