Unit 7 Interfacing Memory & Parallel I/O Peripherals to DSP Devices

7.1 Introduction: A typical DSP system has DSP with external memory, input devices and output devices. Since the manufacturers of memory and I/O devices are not same as that of manufacturers of DSP and also since there are variety of memory and I/O devices available, the signals generated by DSP may not suit memory and I/O devices to be connected to DSP. Thus, there is a need for interfacing devices the purpose of it being to use DSP signals to generate the appropriate signals for setting up communication with the memory. DSP with interface is shown in fig. 7.1.

![Diagram showing DSP system with interfacing](image)

Fig. 7.1: DSP system with interfacing

7.2 Memory Space Organization: Memory Space in TMS320C54xx has 192K words of 16 bits each. Memory is divided into Program Memory, Data Memory and I/O Space, each are of 64K words. The actual memory and type of memory depends on particular DSP device of the family. If the memory available on a DSP is not sufficient for an application, it can be interfaced to an external memory as depicted in fig. 7.2. The On-Chip Memory are faster than External Memory. There are no interfacing requirements. Because they are on-chip, power consumption is less and size is small. It exhibits better performance by DSP because of better data flow within pipeline. The purpose of such memory is to hold Program / Code / Instructions, to hold constant data such as filter coefficients / filter order, also to hold trigonometric tables / kernels of transforms employed in an algorithm. Not only constants are stored in such memory, they are also used to hold variable data and intermediate results so that the processor need not refer to external memory for the purpose.
External memory is off-chip. They are slower memory. External Interfacing is required to establish the communication between the memory and the DSP. They can be with large memory space. The purpose is being to store variable data and as scratch pad memory.

Program memory can be ROM, Dual Access RAM (DARAM), Single Access RAM (SARAM), or a combination of all these. The program memory can be extended externally to 8192K words. That is, 128 pages of 64K words each. The arrangement of memory and DSP in the case of Single Access RAM (SARAM) and Dual Access RAM (DARAM) is shown in fig. 7.3. One set of address bus and data bus is available in the case of SARAM and two sets of address bus and data bus is available in the case of DARAM. The DSP can thus access two memory locations simultaneously.

Fig. 7.2: Internal memory and interfacing of external memory

Fig. 7.3: SARAM & DARAM
There are 3 bits available in memory mapped register, PMST for the purpose of on-chip memory mapping. They are microprocessor / microcomputer mode. If this bit is 0, the on-chip ROM is enabled and addressable and if this bit is 1 the on-chip ROM not available. The bit can be manipulated by software / set to the value on this pin at system reset. Second bit is OVLY. It implies RAM Overlay. It enables on-chip DARAM data memory blocks to be mapped into program space. If this bit is 0, on-chip RAM is addressable in data space but not in Program Space and if it is 1, on-chip RAM is mapped into Program & Data Space. The third bit is DROM. It enables on-chip DARAM 4-7 to be mapped into data space. If this bit is 0, on-chip DARAM 4-7 is not mapped into data space and if this bit is 1, on-chip DARAM 4-7 is mapped into Data Space. On-chip data memory is partitioned into several regions as shown in table 7.1. Data memory can be on-chip / off-chip.

Table 7.1: Data memory 64 K

<table>
<thead>
<tr>
<th>Memory Mapped Registers</th>
<th>0000-005F 96 locations</th>
<th>0060-007F 32 locations</th>
<th>0080-7FFF On-chip DARAM 0-3 32Kx16bit</th>
<th>8000-FFFF 32K locations On-chip DARAM 4-7 for Data</th>
</tr>
</thead>
</table>

The on-chip memory of TMS320C54xx can be both program & data memory. It enhances speed of program execution by using parallelism. That is, multiple data access capability is provided for concurrent memory operations. The number of operations in single memory access is 3 reads & one write.

The external memory to DSP can be interfaced with 16 -23 bit Address Bus, 16 bit Data Bus. Interfacing Signals are generated by the DSP to refer to external memory. The signals required by the memory are typically chip Select, Output Enable and Write Enable. For example, TMS320C5416 has 16K ROM, 64K DARAM and 64K SARAM. Extended external Program Memory is interfaced with 23 address lines i.e., 8192K
locations. The external memory thus interfaced is divided into 128 pages, with 64K words per page.

7.3: External Bus Interfacing Signals: In DSP there are 16 external bus interfacing signals. The signal is characterized as single bit i.e., single line or multiple bits i.e., Multiple lines / bus. It can be synchronous / asynchronous with clock. The signal can be active low / active high. It can be output / input Signal. The signal carrying line / lines Can be unidirectional / bidirectional Signal. The characteristics of the signal depend on the purpose it serves. The signals available in TMS320C54xx are listed in table 7.2 (a) & table 7.2 (b).

<table>
<thead>
<tr>
<th>Table 7.2 (a) External Bus Interfacing Signals</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>6</td>
</tr>
<tr>
<td>7</td>
</tr>
<tr>
<td>8</td>
</tr>
</tbody>
</table>

In external bus interfacing signals, address bus and data bus are multi-lines bus. Address bus is unidirectional and carries address of the location refereed. Data bus is bidirectional and carries data to or from DSP. When data lines are not in use, they are tri-stated. Data Space Select, Program Space Select, I/O Space Select are meant for data space, program space or I/O space selection. These interfacing signals are all active low. They are active during the entire operation of data memory / program memory / I/O space reference. Read/Write Signal determines if the DSP is reading the external device or writing.
Read/Write Signal is low when DSP is writing and high when DSP is reading. Strobe Interfacing Signals, Memory Strobe and I/O Strobe both are active low. They remain low during the entire read & write operations of memory and I/O operations respectively. External Bus Interfacing Signals from 1-8 are all are unidirectional except Data Bus which is bidirectional. Address Lines are outgoing signals and all other control signals are also outgoing signals.

<table>
<thead>
<tr>
<th>Table 7.2 (b) External Bus Interfacing Signals</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
</tr>
<tr>
<td>10</td>
</tr>
<tr>
<td>11</td>
</tr>
<tr>
<td>12</td>
</tr>
<tr>
<td>13</td>
</tr>
<tr>
<td>14</td>
</tr>
<tr>
<td>15</td>
</tr>
<tr>
<td>16</td>
</tr>
</tbody>
</table>

Data Ready signal is used when a slow device is to be interfaced. Hold Request and Hold Acknowledge are used in conjunction with DMA controller. There are two Interrupt related signals: Interrupt Request and Interrupt Acknowledge. Both are active low. Interrupt Request typically for data exchange. For example, between ADC / another Processor. TMS320C5416 has 14 hardware interrupts for the purpose of User interrupt, McBSP, DMA and timer. The External Flag is active high, asynchronous and outgoing control signal. It initiates an action or informs about the completion of a transaction to the peripheral device. Branch Control Input is a active low, asynchronous, incoming control signal. A low on this signal makes the DSP to respond or attend to the peripheral device. It informs about the completion of a transaction to the DSP.
Problem P7.1: What is the range of addresses that can be decoded if A19 is pulled low in a processor with 20 address lines?

Solution: 20 address lines of processor are A0-A19. With A19 pulled low, effectively there are 19 address lines. Thus, number of addressable locations = $2^{19} = 80000h$. The range addresses that can be decoded = 00000 - 7FFFFh

7.4 The Memory Interface: The memory is organized as several locations of certain number of bits. The number of locations decides the address bus width and memory capacity. The number of bits per locations decides the data bus width and hence word length. Each location has unique address. The demand of an application may be such that memory capacity required is more than that available in a memory IC. That means there are insufficient words in memory IC. Or the word length required may be more than that is available in a memory IC. Thus, there may be insufficient word length. In both the cases, more number of memory ICs are required.

Typical signals in a memory device are address bus to carry address of referred memory location. Data bus carries data to or from referred memory location. Chip Select Signal selects one or more memory ICs among many memory ICs in the system. Write Enable enables writing of data available on data bus to a memory location. Output Enable signal enables the availability of data from a memory location onto the data bus. The address bus is unidirectional, carries address into the memory IC. Data bus is bidirectional. Chip Select, Write Enable and Output Enable control signals are active high or low and they carry signals into the memory ICs. The task of the memory interface is to use DSP signals and generate the appropriate signals for setting up communication with the memory. The logical spacing of interface is shown in fig. 7.4.
The timing sequence of memory access is shown in fig. 7.5. There are two read operations, both referring to program memory. Read Signal is high and Program Memory Select is low. There is one Write operation referring to external data memory. Data Memory Select is low and Write Signal low. Read and write are to memory device and hence memory strobe is low. Internal program memory reads take one clock cycle and External data memory access require two clock cycles.

**Fig. 7.4 Memory Interface for TMS320C5416**

**Fig. 7.5 Timing Sequence for External Memory Access**
To Interface Slower off-chip Memory & I/O wait states are required. User accessible Memory Mapped Software Wait State Register (SWWSR) is used for the purpose. There is no need for external hardware to insert wait states. 3 bit Field in SWWSR are meant for indicating number of wait states. 000 implies no wait state and 111 implies seven wait states. However, wait states are applicable to program & data memory of 2 pages each of 32K words and to I/O space of one page of 64K. Further to interface devices requiring more than 7 wait states, hardware READY Signal is used. External Device sends signal on READY line to the DSP to indicate its readiness for transaction.

**Problem P7.2:** Up to what limit can the program memory be extended in a processor with 20 address lines? How must the extended memory be organized to for addressing by a C54xx Processor?

Solution: With 20 address lines, the program memory can be extended to $2^{20}$ locations=100000h. Memory may be organized into several pages of 64K words each. Number of pages = $2^{20} / 64K = 2^{20} / (2^6 \times 2^{10})= 16$ pages

**Problem P7.3:** 256K x 16 bit RAM is interfaced to the DSP Processor with A17 address line of the memory device connected to Program Space Select line. Determine the total Program memory and Data Memory Addressable.

Solution: 64K words of Memory is for the purpose of data (not more than that). As Program Space select line is connected to A17, it makes number of addressable locations = $2^{17} = 128K$ words. Remaining 256K-(64K+128K)=64K is unused.

**Problem P7.4:** Is it possible to connect a memory device without Decode (interfacing) Circuit? If so, connect 8k x 16 SRAM to TMS320C54xx. What are the merits & demerits of the solution?

Solution: The interfacing circuit is shown in fig. P7.4. 8K x 16 SRAM to be interfaced. The address lines are A19 A18 ...[A12 A11 A10 A9 A8 A7 A6 A5 A4 A3 A2 A1 A0] Address lines required: A0-A12. Address range is 0000-1FFFh.
Effects of ‘No decode’ interface are

- Fast memory Access
- ENTIRE Address space is used by the Device that is connected
- Memory responds to 0000-1FFFh and also to all combinations of address bits A13-A19 (In the example quoted)
- Program space select & data space select lines are not used
- SRAM is thus indistinguishable as a program or data Memory

**Problem P7.5**: Design an interface to connect a 64K x 16 flash memory to a TMS320C54xx device. The Processor address bus to be used is A0-A15. The flash memory has the signals as shown in fig. P7.5.

Solution: Address lines from A0-A15 are used to address 64K locations. All the data lines, D0-D15 are used to carry data word. Data Space Select line is connected to chip enable of memory so that whenever DSP refers to data memory, this flash memory is enabled. When DSP refers to memory and it is a write operation, both memory strobe and read/write signals will be low. They are combined in using OR gate and used as write enable for memory. Memory read is performed by combining memory strobe and XF signals.
Problem P7.6: Design a data memory system with address range 000800h-000FFFh for a C5416. Use 2K x 8 RAM memory chips.

Solution: Memory capacity: 2K. Memory word length: 8 bits < that of the processor. Word length needs to be extended to 16 bits. 2 memory ICs of same capacity are required. One IC is used to hold LS byte of the data and the other IC to hold MS byte. Both ICs are selected simultaneously and are given all other control signals, together. Data bus of DSP is split into two halves, LS and MS byte and they are connected to respective ICs. Address bus of DSP from A0-A10 are connected to both the ICs. Whenever DSP issues an address and issues appropriate read / write control signals, both the ICs respond simultaneously. The remaining lines of address bus, A11-A22 are used for the purpose of decoding. The arrangement of ICs for extension of memory word length is shown in fig. P7.6.
**Problem P7.7:** Interface an 8K x 16 ROM to the C5416 DSP in the address range 7FE000h-7FFFFFh.
**Problem P7.8:** Design a circuit to interface 4K x 16 and a 2K x 16 memory chip to realize program memory space for the TMS320C54xx processor in the address Ranges 03F000h-03FFFFh and 05F800h-05FFFFh, respectively.

Solution: 4K x 16 with address range 03F000h-03FFFFh. Address lines: 12 in number: A11-A0. Remaining Address lines: 11 in number: A22-A12 out of which A22-A18 all zeros, A17-A12 all ones.

2K x 16 with address range 05F800h-05FFFFh. Address lines: 11 in number: A10-A0. Remaining Address lines: 12 in number: A22-A11 out of which A22-A19 and A17 all zeros, A16-A11 and A18 all ones. Connect the memory and design the decoding logic accordingly.

**Problem P7.9:** Design a circuit to interface 64K words of the program memory space from 0FFFFFh-0F0000h for the TMS320C5416 processor using 16K x 16 memory chips.

Solution: 64K required > the memory capacity of IC. The memory capacity needs to be extended. 4 Memory ICs are required. Address lines for 16K: 14 in number: A13-A0. Remaining Address lines: 9 in number: A22-A14. A22-A20 all zeros, A19-A16 all ones, Address on A15-A14 vary from 00 to 11. (4 combinations).

![Diagram of memory interfacing](image-url)
7.5 Parallel I/O Interface: I/O devices are interfaced to DSP using unconditional I/O mode, programmed I/O mode or interrupt I/O mode. Unconditional I/O does not require any handshaking signals. DSP assumes the readiness of the I/O and transfers the data with its own speed. Programmed I/O requires handshaking signals. DSP waits for the readiness of the I/O readiness signal which is one of the handshaking signals. After the completion of transaction DSP conveys the same to the I/O through another handshaking signal. Interrupt I/O also requires handshaking signals. DSP is interrupted by the I/O indicating the readiness of the I/O. DSP acknowledges the interrupt, attends to the interrupt. Thus, DSP need not wait for the I/O to respond. It can engage itself in execution as long as there is no interrupt.

7.6: Programmed I/O interface: The timing diagram in the case of programmed I/O is shown in fig. 7.6. I/O strobe and I/O space select are issued by the DSP. Two clock cycles each are required for I/O read and I/O write operations.

An example of interfacing ADC to DSP in programmed I/O mode is shown in fig. 7.7. ADC has a start of conversion (SOC) signal which initiates the conversion. In programmed I/O mode, external flag signal is issued by DSP to start the conversion.
ADC issues end of conversion (EOC) after completion of conversion. DSP receives Branch input control by ADC when ADC completes the conversion. The DSP issues address of the ADC, I/O strobe and read / write signal as high to read the data. An address decoder does the translation of this information into active low read signal to ADC. The data is supplied on data bus by ADC and DSP reads the same. After reading, DSP issues start of conversion once again after the elapse of sample interval. Note that there are no address lines for ADC. The decoded address selects the ADC. During conversion, DSP waits checking branch input control signal status for zero. The flow chart of the activities in programmed I/O is shown in fig. 7.8.

Fig. 7.7: ADC in Programmed I/O mode

Fig. 7.8: Programmed I/O mode
7.7 Interrupt I/O: This mode of interfacing I/O devices also requires handshaking signals. DSP is interrupted by the I/O whenever it is ready. DSP Acknowledges the interrupt, after testing certain conditions, attends to the interrupt. DSP need not wait for the I/O to respond. It can engage itself in execution.

There are a variety of interrupts. One of the classifications is maskable and non-maskable. If maskable, DSP can ignore when that interrupt is masked. Another classification is vectored and non-vectored. If vectored, Interrupt Service subroutine (ISR) is in specific location. In Software Interrupt, instruction is written in the program. In Hardware interrupt, a hardware pin, on the DSP IC will receive an interrupt by the external device. Hardware interrupt is also referred to as external interrupt and software interrupt is referred to as internal interrupt. Internal interrupt may also be due to execution of certain instruction can causing interrupt. In TMS320C54xx there are total of 30 interrupts. Reset, Non-maskable, Timer Interrupt, HPI, one each, 14 Software Interrupts, 4 External user Interrupts, 6 McBSP related Interrupts and 2 DMA related Interrupts. Host Port Interface (HPI) is a 8 bit parallel port. It is possible to interface to a Host Processor using HPI. Information exchange is through on-chip memory of DSP which is also accessible Host processor.

Registers used in managing interrupts are Interrupt flag Register (IFR) and Interrupt Mask Register (IMR). IFR maintains pending external & internal interrupts. One in any bit position implies pending interrupt. Once an interrupt is received, the corresponding bit is set. IMR is used to mask or unmask an interrupt. One implies that the corresponding interrupt is unmasked. Both these registers are Memory Mapped Registers. One flag, Global enable bit (INTM), in ST1 register is used to enable or disable all interrupts globally. If INTM is zero, all unmasked interrupts are enabled. If it is one, all maskable interrupts are disabled.

When an interrupt is received by the DSP, it checks if the interrupt is maskable. If the interrupt is non-maskable, DSP issues the interrupt acknowledgement and thus serves the interrupt. If the interrupt is hardware interrupt, global enable bit is set so that no other interrupts are entertained by the DSP. If the interrupt is maskable, status of the INTM is
checked. If INTM is 1, DSP does not respond to the interrupt and it continues with program execution. If the INTM is 0, bit in IMR register corresponding to the interrupt is checked. If that bit is 0, implying that the interrupt is masked, DSP does not respond to the interrupt and continues with its program execution. If the interrupt is unmasked, then DSP issues interrupt acknowledgement. Before branching to the interrupt service routine, DSP saves the PC onto the stack. The same will be reloaded after attending the interrupt so as to return to the program that has been interrupted. The response of DSP to an Interrupt is shown in flow chart in fig. 7.9.

![Flowchart](image)

**Fig. 7.9: Response of DSP to interrupt**

Problem P7.10: Interface the TMS320C54xx to a 10 bit ADC and an 8 bit DAC. The sampled signal read from the ADC is to be written to the DAC after adjusting its size (to 8 bits). The start of conversion is to be initiated by the TOUT signal of the timer available on DSP.
Solution: Initialize processor with respect to desired speed, internal registers: PMST, BSCR, SWWSR. Initialize internal timer for sampling period of ADC. Apply analog input signal. Send start conversion through TOUT. Continue with any other program execution. ADC interrupts DSP after conversion on INT1. DSP reads 10 bit data and converts to 8 bit by shifting to right. It then processes the sample and sends this data to DAC. DAC converts the data back to analog. The corresponding program is as follows.

```assembly
buffer: .bss sample,1 ;data buffer
    ;text
_c_int00:
    STM #0X0500,SP  ;initialize Stack Pointer
    SSBX INTM      ; disable all interrupts
    CALL init_DSP  ; initialize DSP processor
    CALL init_timer ;initialize timer
    STM #0XFFFF,IFR ;pending interrupts cleared
    ORM #0002H,IMR ;INT 1 unmasked
    RSBX INTM      ; enable all interrupts
    ; Initialize DSP Processor ;init_DSP
    PMST_VAL       .set  00A0h ; MC & OVLY, interrupt vector is set
    BSCR_VAL       .set  0000h ; bank switching reset, 64K only
    SWWSR_VAL      .set  2000h ; s/w wait, 2 clock wait states
```

Fig. P7.10: Interfacing DSP to ADC & DAC
.text

init_DSP:
    LD #0,DP ; data page initialized
    STM #0,CLKMD
    STM #0,CLKMD
    STM #0X4007,CLKMD ; processor speed set
    STM #PMST_VAL, PMST ; initialize PMST
    STM #BSCR_VAL, BSCR ; initialize BSCR
    STM #SWWSR_VAL, SWWSR ; initialize SWWSR
    SSBX OVM ; identify overflow
    SSBX SXM ; sign extension set
    RET ; return from subroutine
    NOP
    NOP

; Initialize Timer
    .set 9999 ; FSAMPLING / FCPU
    .set 0000 ; start timer

.init_timer:
    STM PRD_VAL, PRD ; initialize timer period register
    STM TCR_VAL, TCR ; initialize timer count register
    RET ; return from subroutine
    NOP
    NOP

; ISR for ADC Read & DAC Write
    .set 05h ; port address of ADC
    .set 07h ; port address of DAC

.text

    PORTR ADC_DATA_IN_addr, sample ; read data from ADC
    LD sample,-2,A ; 10 bit data to 8 bit
    STL A, sample ; store in buffer
    PORTW sample, DAC_DATA_OUT_addr ; write to DAC
    RET ; return from subroutine
    NOP
    NOP

7.8: Direct Memory Access (DMA) operation: In any application, there is data transfer between DSP and memory and also DSP and I/O device, as shown in fig. 7.10. However, there may be need for transfer of large amount of data between two memory regions or between memory and I/O. DSP can be involved in such transfer, as shown in fig. 7.11. Since amount of data is large, it will engage DSP in data transfer task for along time.
DSP thus will not get utilized for the purpose it is meant for, i.e., data manipulation. The intervention of DSP has to be avoided for two reasons: to utilize DSP for useful signal processing task and to increase the speed of transfer by direct data transfer between memory or memory and I/O. The direct data transfer is referred to as direct memory access (DMA). The arrangement expected is shown in fig. 7.12. DMA controller helps in data transfer instead of DSP.

![Fig. 7.10: Interface between DSP and external devices](image)

![Fig. 7.11: Data transfer with intervention by DSP](image)

![Fig. 7.12: data transfer without intervention by DSP](image)
In DMA, data transfer can be between memory and peripherals which are either internal or external devices. DMA controller manages DMA operation. Thus DSP is relieved of the task of data transfer. Because of direct transfer, speed of transfer is high. In TMS320C54xx, there are up to 6 independent programmable DMA channels. Each channel is between certain source & destination. One channel at a time can be used for data transfer and not all six simultaneously. These channels can be prioritized. The speed of transfer measured in terms of number of clock cycles for one DMA transfer depends on several factors such as source and destination location, external interface conditions, number of active DMA channels, wait states and bank switching time. The time for data transfer between two internal memory is 4 cycles for each word.

Requirements of maintaining a channel are source & Destination address for a channel, separately for each channel. Data transfer is in the form of block, with each block having frames of 16 / 32 bits. Block size, frame size, data are programmable. Along with these, mode of transfer and assignment of priorities to different channels are also to be maintained for the purpose of data transfer.

There are five, channel context registers for each DMA channel. They are Source Address Register (DMSRC), Destination Address Register (DMDST), Element Count Register (DMCTR), Sync select & Frame Count register (DMSFC), Transfer Mode Control Register (DMMCR). There are four reload registers. The context register DMSRC & DMDST are source & destination address holders. DMCTR is for holding number of data elements in a frame. DMSFC is to convey sync event to use to trigger DMA transfer, word size for transfer and for holding frame count. DMMCR Controls transfer mode by specifying source and destination spaces as program memory, data memory or I/O space. Source address reload & Destination address reload are useful in reloading source address and destination address. Similarly, count reload and frame count reload are used in reloading count and frame count. Additional registers for DMA that are common to all channels are Source Program page address, DMSRCP, Destination Program page address, DMDSTP, Element index address register, Frame index address register.
Number of memory mapped registers for DMA are $6 \times (5+4)$ and some common registers for all channels, amounting to total of 62 registers required. However, only 3 (+1 for priority related) are available. They are DMA Priority & Enable Control Register (DMPREC), DMA sub bank Address Register (DMSA), DMA sub bank Data Register with auto increment (DMSDI) and DMA sub bank Data Register (DMSDN). To access each of the DMA Registers Register sub addressing Technique is employed. The schematic of the arrangement is shown in fig. 7.13. A set of DMA registers of all channels (62) are made available in set of memory locations called sub bank. This avoids the need for 62 memory mapped registers. Contents of either DMSDI or DMSDN indicate the code (1’s & 0’s) to be written for a DMA register and contents of DMSA refers to the unique sub address of DMA register to be accessed. Mux routes either DMSDI or DMSDN to the sub bank. The memory location to be written

![Diagram of register subaddress technique](image)

**Fig. 7.13: Register Subaddress Technique**

DMSDI is used when an automatic increment of the sub address is required after each access. Thus it can be used to configure the entire set of registers. DMSDN is used when
single DMA register access is required. The following examples bring out clearly the method of accessing the DMA registers and transfer of data in DMA mode.

**Problem P7.11:** Write code to show how the DMA channel 2 source address register can be initialized with the value 1111h. (single register to be written)

Solution:
DMSA .set 55h ; subbank address register address
DMSDN .set 57h ; subbank Access register address
DMSRC2 .set 0Ah ; address of DMA channel 2, source address register

STM DMSRC2, DMSA ; DMSA = addr of DMSRC2
STM #1111H, DMSDN ; write 1111h to DMSRC2

**Problem P7.12:** Write a TMS320C54xx code to show how the DMA channel 5 context registers can be initialized. Choose arbitrary values to be written to the registers. (all five registers to be written)

Solution:
DMSA .set 55h ; subbank address register addr
DMSDI .set 56h ; subbank access register addr
DMSRC5 .set 19h ; sub address of DMSRC5 to
DMDST5 .set 1Ah ; DMMCR5, consecutive
DMCTR5 .set 1Bh ; locations
DMSFC5 .set 1Ch
DMMCR5 .set 1Dh

STM DMSRC5, DMSA ; DMSA = first location address
STM #1111h, DMSDI ; write 1111h to DMSRC5
STM #2222h, DMSDI ; write 2222h to DMDST5
STM #3333h, DMSDI ; write 3333h to DMCTR5
STM #4444h, DMSDI ; write 4444h to DMSFC5
STM #5555h, DMSDI ; write 5555h to DMMCR5

; DMSDI is automatically incremented to point to next location i.e., register

**Problem P7.13:** Write a TMS320C54xx code to transfer a block of data from the program memory to the data memory. Following are the specifications.

Solution:
Source address: 6000h in Program space
Destination addr: 8000h in Data space
Transfer size: 800h single (16 bit) words
Channel use: DMA channel #1
Solution:
; Channel to be used channel #1
DMSA .set 55h ;subbank address register addr
DMSDI .set 56h ;subbank access register addr
DMSRC1 .set 19h ;sub address of DMSRC1 to
DMDST1 .set 1Ah ;DMMCR1, consecutive
DMCTR1 .set 1Bh ;locations
DMSFC1 .set 1Ch
DMMCR1 .set 1Dh
STM DMSRCP, DMSA ;Set source program page
STM #0h, DMSDN ;as page 0
STM DMSRC1, DMSA ;points to first location
STM #6000h, DMSDI ;source address = 6000h
STM #8000h, DMSDI ;destination address = 8000h
STM #(800-1)h, DMSDI ;count =800h
STM #4444h, DMSDI ;depends on given
STM #5555h, DMSDI ;specification, arbitrarily set in this example

Problem P7.14: For a TMS320C54xx operating at a clock frequency of 100MHz, how many 16 bit data elements can be transferred between two internal memory locations per second in DMA mode?

Solution: Clock cycle period=1/(100x10^6)=0.01x10^-6 s (given)
Number of clock cycles per transfer of single 16 bit data element = 4 cycles (known)
Time for transferring single data element = 4 x 0.01x10^-6 s
Number of data elements transferred per second= 1/(4 x 0.01x10^-6)=25x10^6 data elements.