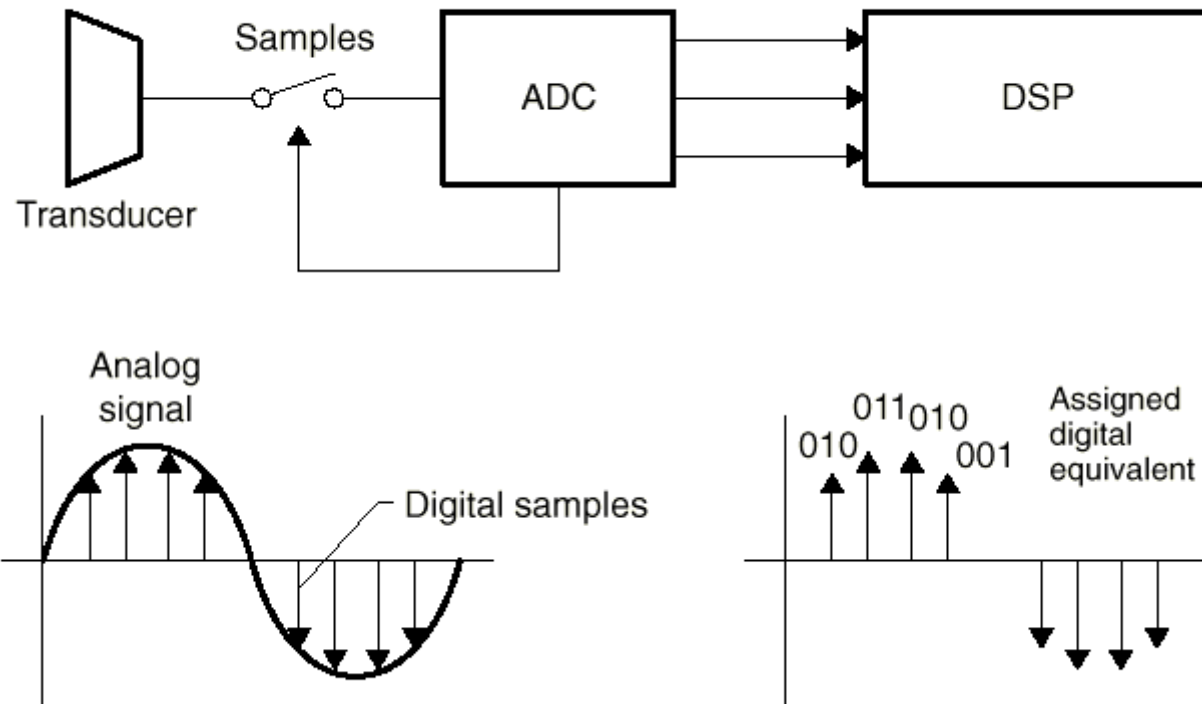


Introduction to DSP controller

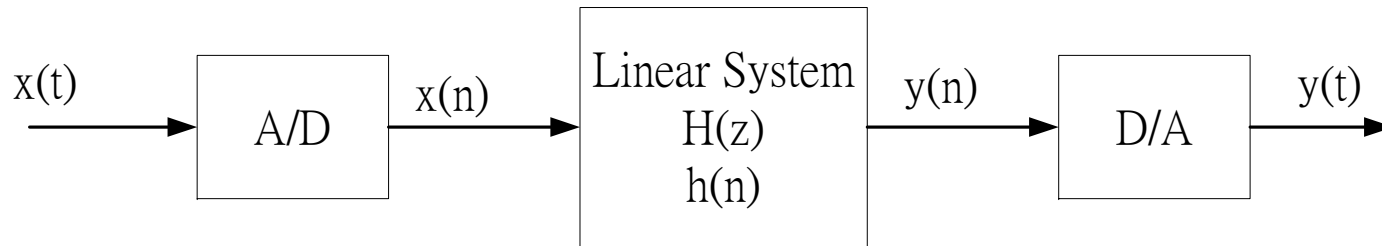
-- TMS320F24x--

What is DSP ?



- Digital signal processing is concerned with the representation of signals by sequences of numbers, and the transformation or processing of such signal representations by numerical computation procedures.

Linear Time-Invariant System



轉移函數

$$H(z) = \frac{Y(z)}{X(z)} = \frac{\sum_{k=0}^M b(k) \times z^{-k}}{1 + \sum_{k=1}^N a(k) \times z^{-k}}$$

差分方程式

$$y(n) = - \sum_{k=1}^N a(k) \times y(n-k) + \sum_{k=0}^M b(k) \times x(n-k)$$

取樣定理

- **Aliasing** does not occur when the sampling frequency ($1/T$) is **greater than** or **equal to twice** the highest frequency component in the waveform being sampled.

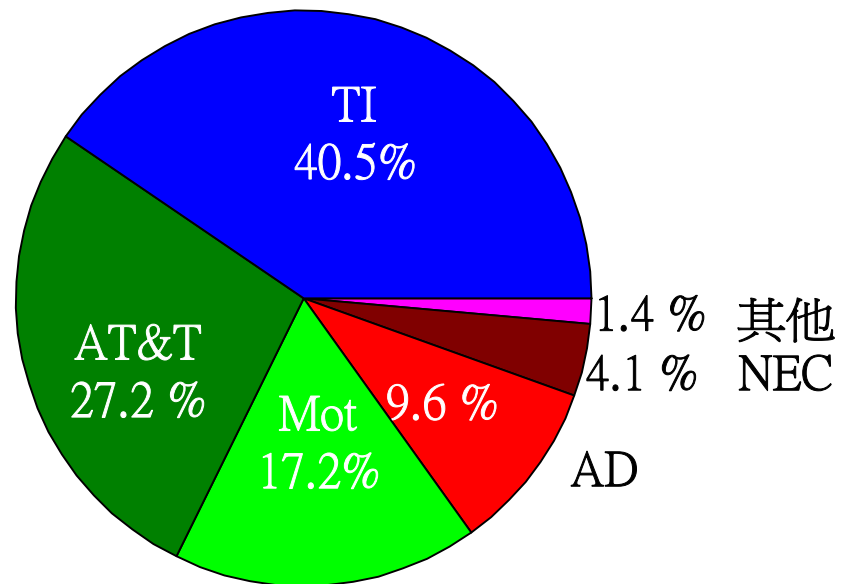
例如

人類聲音大約 4kHz

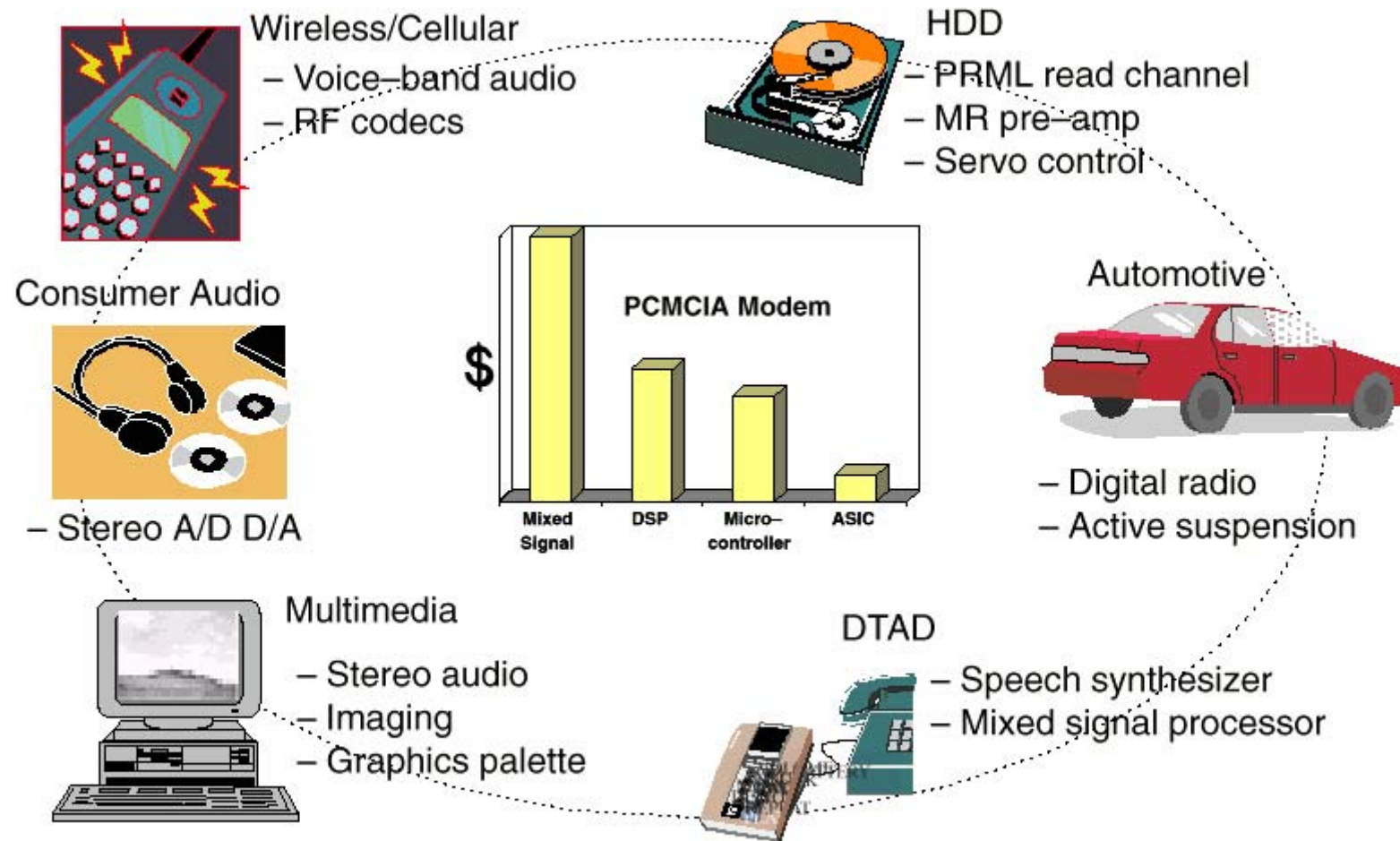
則需要至少 8kHz 取樣頻率

- A **lowpass** filter required to limit the signal bandwidth before the sampling performed by the A/D.

1997 DSP 晶片主要供應廠商

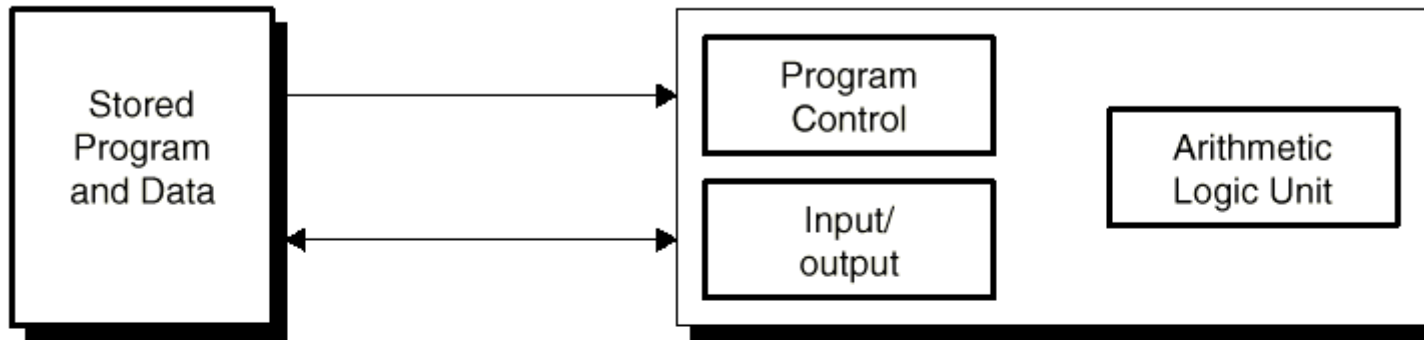


全球一般用途DSP應用分析

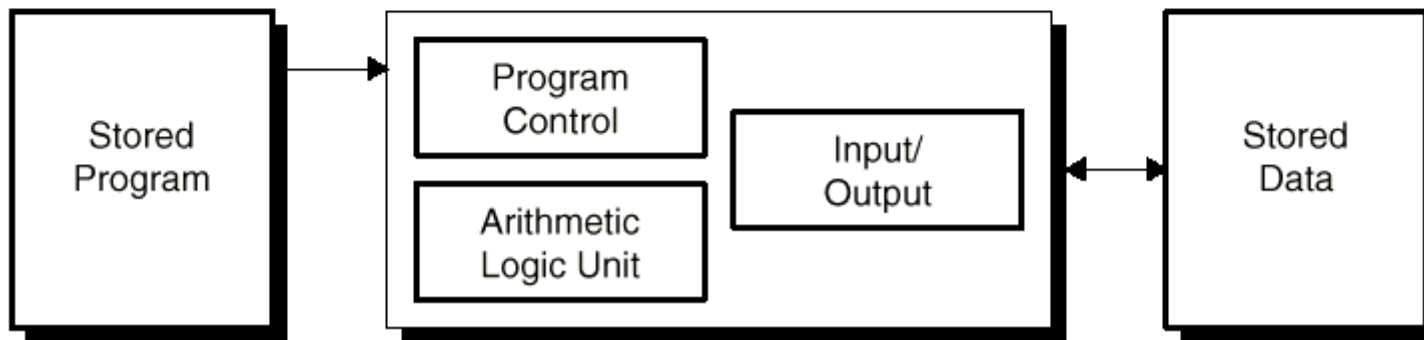


DSP Architecture

Von Neumann Architecture

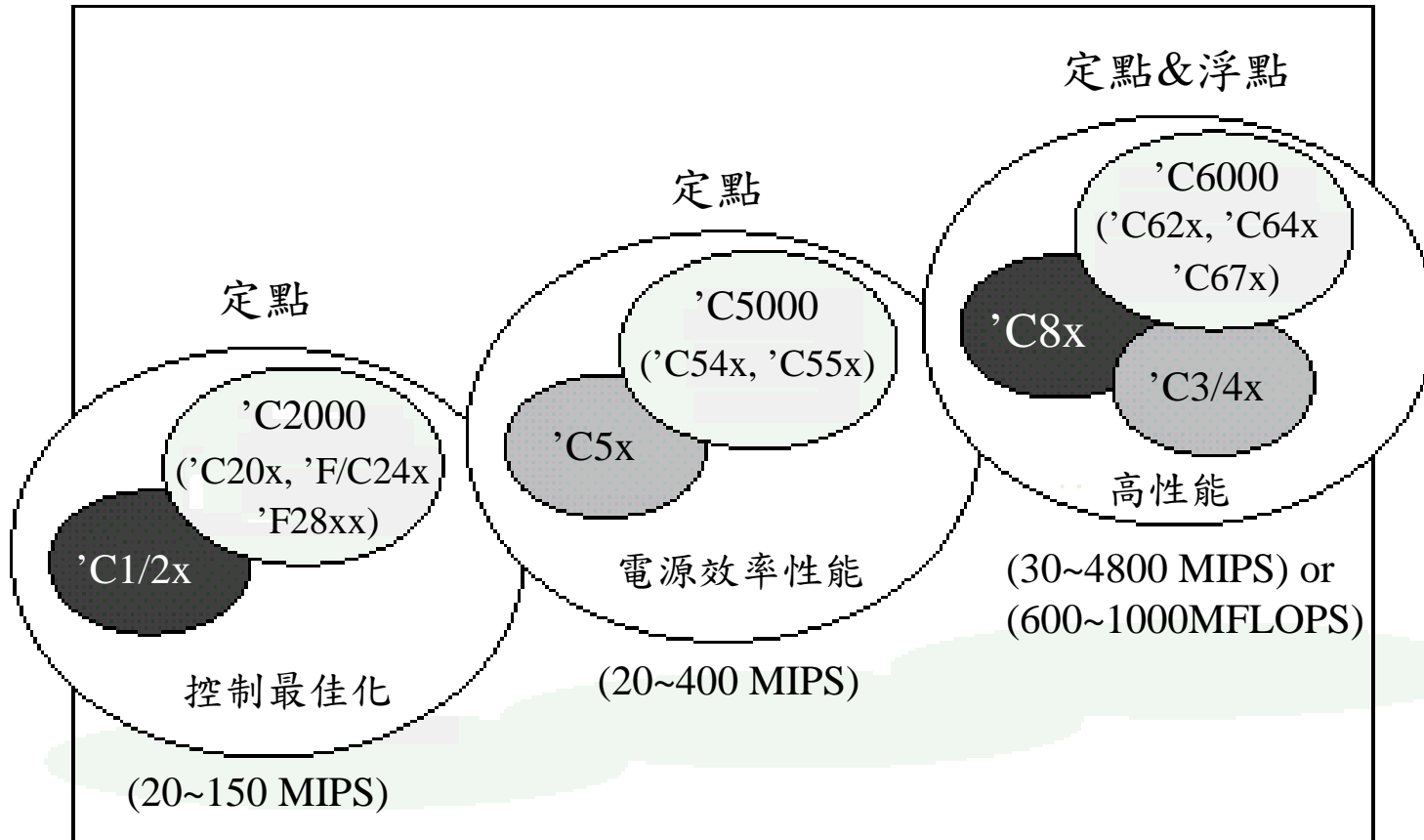


Harvard Architecture

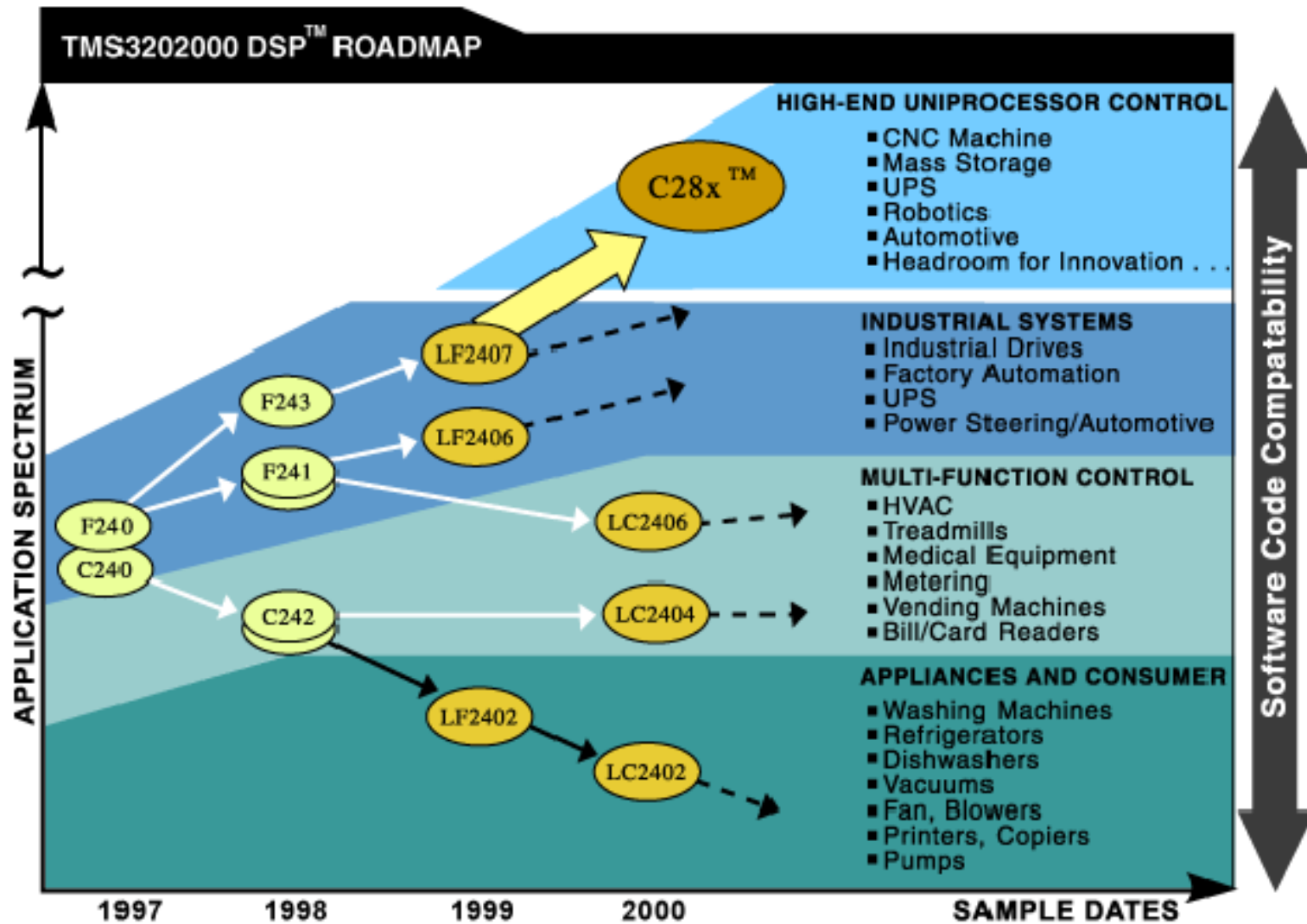


- **Von Neumann architecture** – 已使用40年。
- 架構非常簡單。
- 所有的程式及資料都可擺在相同之記憶體映射空間。
- 此架構之缺點為所有程式與資料空間共用一個匯流排。因此任何一個時間僅能有一個資料空間或程式空間進出。
- **Harvard architecture**-將資料記憶體空間或程式記憶體空間分開。有兩個匯流排讓各個位址空間使用，因此可確保資料及程式可並行處理以增加處理速度。
- 架構複雜、成本較高。
- 兩個記憶體空間需要兩倍的位址，也就是需要兩倍的資料腳位 (data pins) 。
- **Modified Harvard architecture**– 晶片外部僅有一個匯流排(因此可減化腳數)，而晶片內部有程式及資料匯流排。

TMS320 DSP families



'F/C24X roadmap



選用CPU因素

- 三個P決定因素：
 - P - Performance (功能，記憶體，週邊)
 - P - Professional (支援力，發展工具)
 - P - Price

16 - BIT Fixed-Point DSP

**TMS320C24X
INTRODUCTION**

▶ TMS320F24x OVERVIEW

- 架構: **MODIFY HARVARD TYPE
SEPARATE PROGRAM & DATA BUS**
- 執行速度: **20MIPS & 30MIPS
OPTIMIZES SPEED BY IMPLEMENTING
FUNCTIONS IN HARDWARE
*OTHER PROCESSORS IMPLEMENT
THROUGH SOFTWARE OR MICROCODE**
- **TMS320F24x 家族: TMS320C240, TMS320C242
TMS320F240, TMS320F241, TMS320F243
TMS320C2402, TMS320C2404, TMS320C2406
TMS320LF2402, TMS320LF2406, TMS320LF2407
TMS320F2810, TMS320F2812**

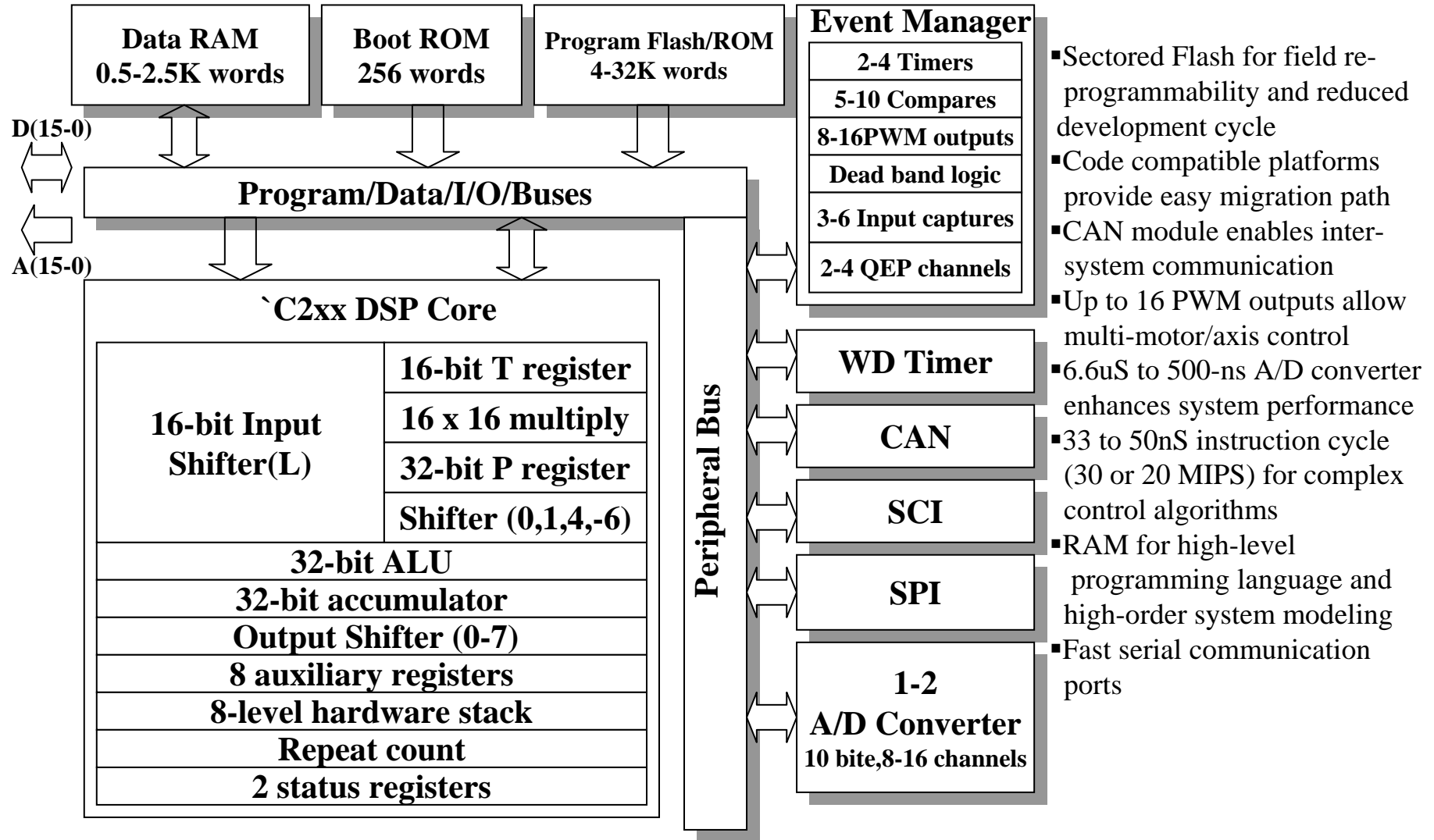
表 1-1 TMS320C2000 各元件之功能說明

名稱	功能	描述
TMS320F2812	3.3V (Core: 1.8V), 150MHz,128K FLASH, 12-BIT A/D (200ns), CAN, EVA &EVB,SPI,SCI	16 位元定點 DSP (具 Flash EEPROM)
TMS320LF2407	3.3V, 32K FLASH, EMIF, 10-BIT A/D (500ns), CAN, 30MIPS, EVA &EVB,SPI,SCI	16 位元定點 DSP (具 Flash EEPROM)
TMS320LF2406	3.3V, 32K FLASH, CAN, 10-BIT A/D(500ns) ,30MIPS, EVA &EVB,SPI,SCI	16 位元定點 DSP (具 Flash EEPROM)
TMS320LF2402	3.3V, 8K FLASH,10-BIT A/D(500ns), 30MIPS, EVA,SCI	16 位元定點 DSP (具 Flash EEPROM)
TMS320LC2406	3.3V, 32K ROM,10-BIT A/D(500ns), 30MIPS, EVA &EVB,SPI,SCI	16 位元定點 DSP
TMS320LC2404	3.3V, 16K ROM,10-BIT A/D(500ns), 30MIPS, EVA &EVB,SPI,SCI	16 位元定點 DSP
TMS320LC2402	3.3V, 4K ROM,10-BIT A/D(500ns), 30MIPS EVA,SCI	16 位元定點 DSP
TMS320F243	5V, 8K FLASH, EMIF, 10-BIT A/D(850ns), CAN, 20MIPS, EVA,SPI,SCI	16 位元定點 DSP (具 Flash EEPROM)
TMS320F241	5V, 8K FLASH, 10-BIT A/D(850ns), CAN, 20MIPS, EVA,SPI,SCI	16 位元定點 DSP (具 Flash EEPROM)
TMS320F240	5V, 16K FLASH, 10-BIT A/D(6us), 20MIPS EVA,SPI,SCI	16 位元定點 DSP (具 Flash EEPROM)
TMS320C242	5V, 4K ROM, 10-BIT A/D(850ns), 20MIPS EVA, SCI	16 位元定點 DSP
TMS320C240	5V, 16K ROM, EMIF, 10-BIT A/D(6us), 20MIPS, EVA,SPI,SCI	16 位元定點 DSP

TMS320F24x之應用場合

- 工業馬達驅動器
 - 汽車系統，如電子動力方向盤、自動鎖住剎車器及溫度控制。
 - HVAC 吹風機/壓縮機馬達控制之應用。
 - 印表機、複印機及其它辦公室產品。
 - 光碟機驅動器、磁帶機驅動器及其它大量資料儲存產品。
 - 機器人及CNC等機器
-

TMS320F/C24X DSP controller features



The Features of TMS320F24x

TMS320C2xx core CPU :

- 32位元中央算術邏輯單元(CALU)。
- 32位元累積器(ACC)。
- 具32位元乘算能力之16位元x16位元乘法器。
- 三個比例移位器。
- 八個16位元輔助暫存器與一個算術單元可作為資料記憶體間接定址。

Memory :

- 544字組x16位元晶片內(on-chip)之資料/程式DARAM。
- 'F/C240晶片：16K字組x16位元晶片內程式ROM或快閃記憶體。
- F243晶片：8K字組x16位元晶片內程式快閃記憶體(flash EEPROM)。
- 224K字組x16位元的定址記憶體空間(64K字組程式空間、64K字組資料空間、64K字組I/O空間、32K字組全域資料空間)。
- 外部記憶體介面模組，其具有軟體設定之等待狀態產生器、16位元位址匯流排、16位元資料匯流排。
- 支援硬體設定之等待狀態。

Program control :

- 4層管線式操作(Pipeline operation)。
 - 8層硬體堆疊器。
 - 6個外部中斷：電源驅動保護中斷、重置、NMI及三個可遮罩式中斷。
-

The Features of TMS320F24x

□ Instruction set :

- TMS320家族晶片中，'C2x、'C2xx、'C5x定點DSP之原始碼可共用。
- 單一指令重覆運算。
- 單週期多/累積指令。
- 具記憶體區塊移動功能，可用於程式/資料管理。
- 指標定址能力。
- 具位元反向指標定址能力，可用於以2的幕次方之快速傅立業轉換。

□ Power :

- ◆ Static CMOS technology
- ◆ Four power-down modes to reduce power consumption

□ Emulation: IEEE Standard 1149.1

- ### □ Speed: 每一個指令週期為50 ns (20 MIPS)，且幾乎大部分指令僅需要一個指令週期即可完成。
-

The Features of TMS320F24x

□ 事件管理者模組:

- 'F/C240 晶片：有12個比較/脈波寬度調變(PWM)頻道。
- 'F243 晶片：有8個脈波寬度調變(PWM)頻道。
- 'F/C240 晶片：有3個具有6種模式之16位元一般用途計時器。
- 'F243 晶片：只有2個具有4種模式之16位元一般用途計時器。
- 3個可具停滯帶(Dead band)之16位元完全比較單元。
- 'F/C240 晶片有4個但是 'F243 晶片只有3個捕捉單元(Capture unit)，其中兩個可作為90度相位差方形光編碼器脈波(QEP)介面。

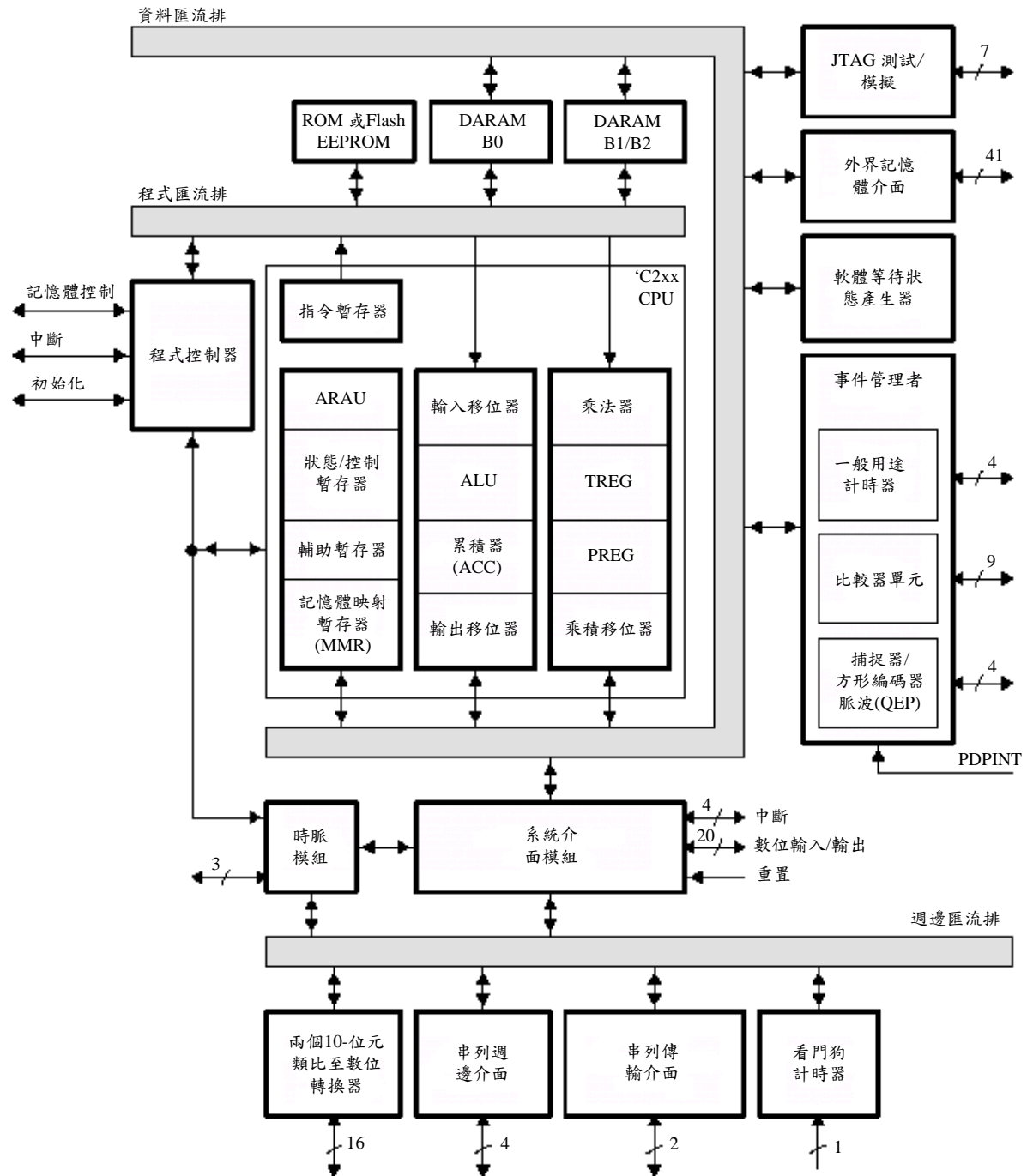
□ 雙10位元類比/數位轉換器(ADC):

- 'F/C240 晶片：具有16個ADC轉換通道。ADC轉換時間約為6us。
 - 'F243 晶片：具有8個ADC轉換通道。ADC轉換時間約為1us。
-

The Features of TMS320F24x

- 28 個獨立、可程式、雙工之I/O接腳。
 - 相位鎖住迴路(PLL)為基礎之時脈模組。
 - 具有即時中斷(RTI)之看門狗(WD)計時器。
 - 串列傳輸介面(SCI)。
 - 串列週邊介面(SPI)。
 - 'F243 晶片具有CAN(Controller Area Network)介面。
-

TMS320x24 Function locl



TMS320 Device Nomenclature

TMS 320 (B) F 240 PGE (L)

字首

TMX = 實驗室元件
TMP = 雛形機元件
TMS = 商品化元件

家族

320 = TMS320 家族

開機載

入選擇

製造技術

C = CMOS

E = CMOS EPROM

F = Flash EEPROM

LC = Low-voltage CMOS (3.3V)

LF = Low-voltage Flash EPROM (3.3V)

VC = Low-voltage CMOS (3V)

溫度範圍

(預設：0°C至70°C)

L = 0°C至70°C

A = -40°C至85°C

S = -40°C至125°C

Q = -40°C至125°C

包裝型式

PAG = 64-pin plastic TQFP

PGE = 144-pin plastic QFP

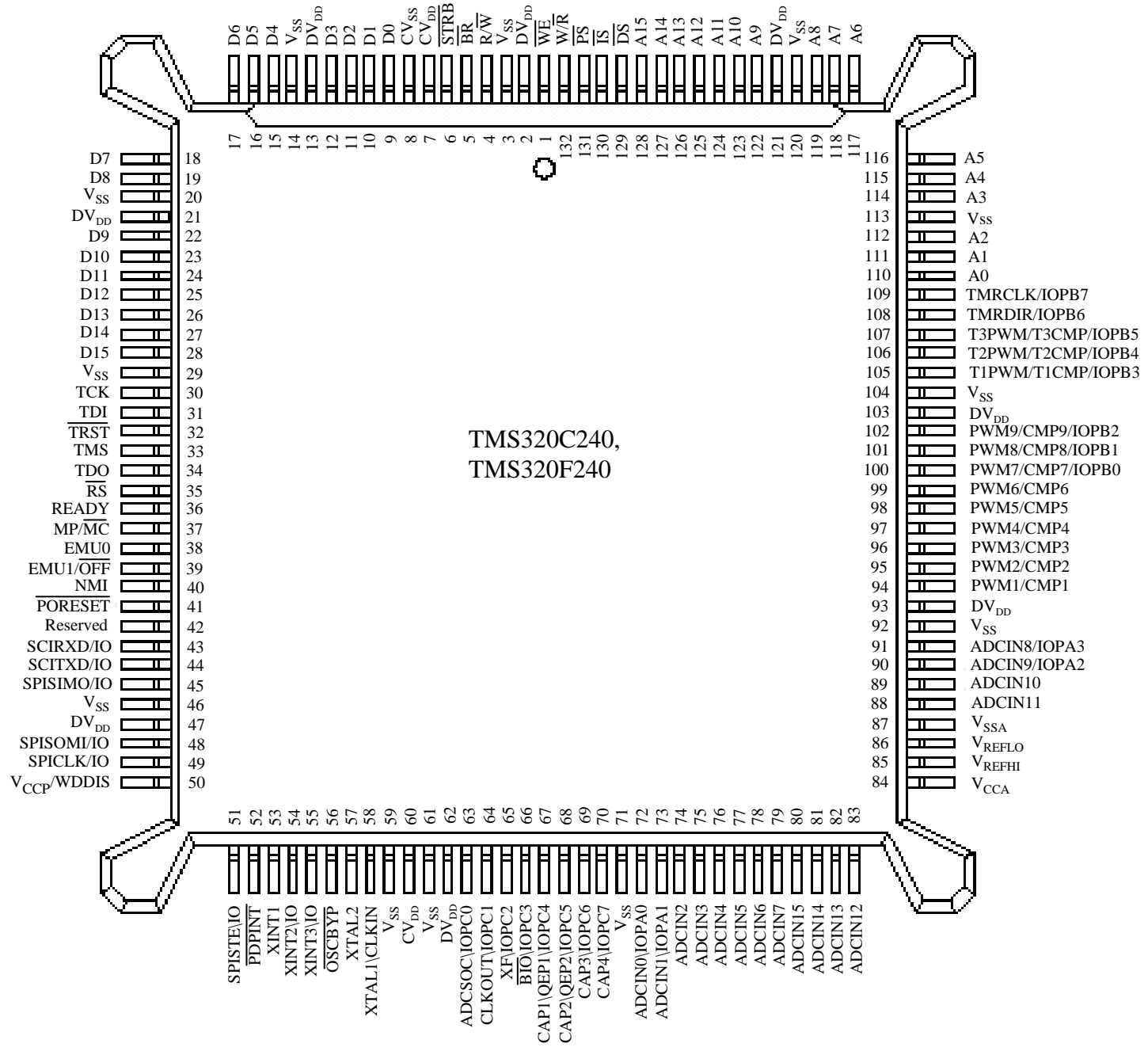
PZ = 100-pin plastic TQFP

元件

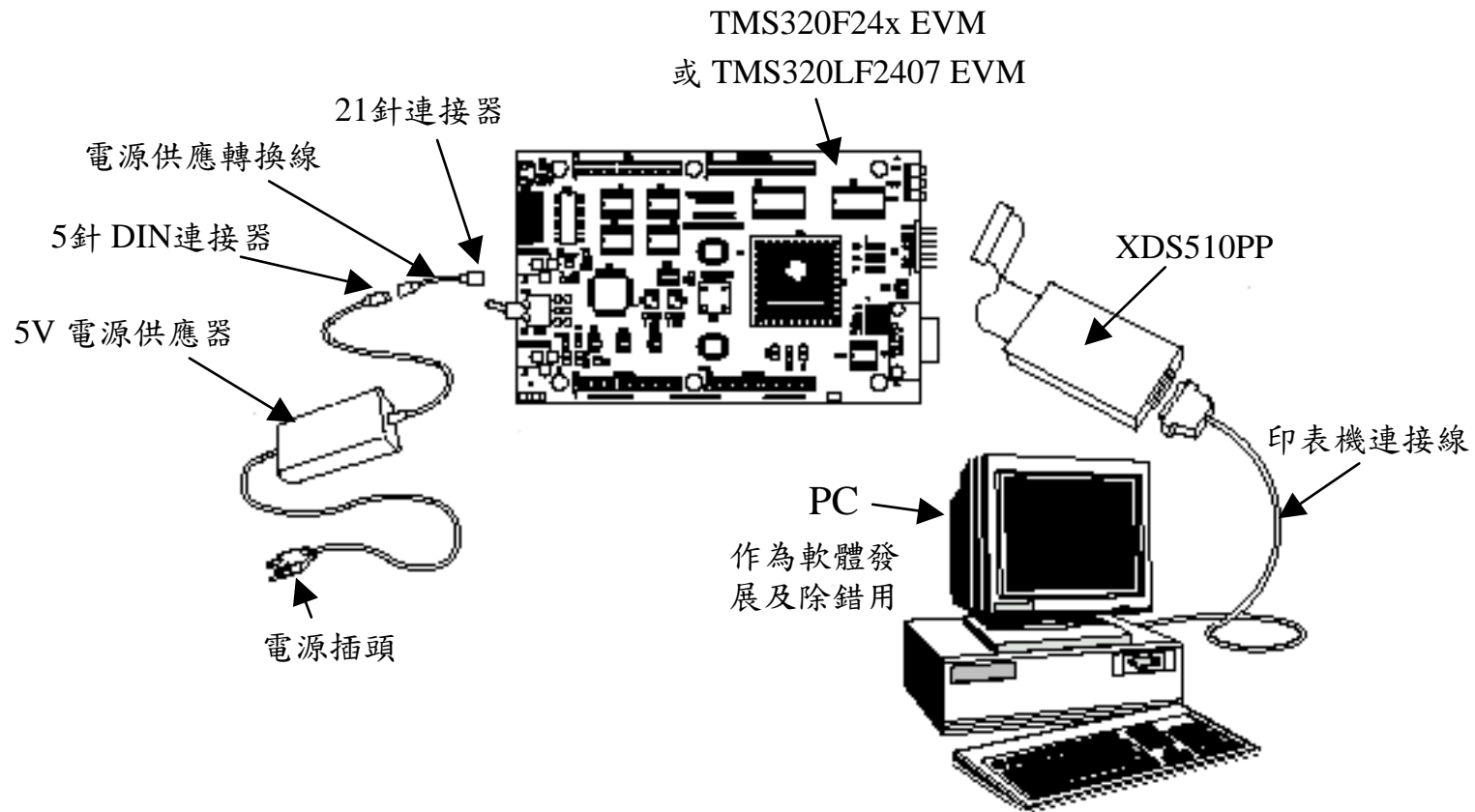
'24x DSP : 240, 241, 242, 243

'24xx DSP : 2402, 2404, 2406,
2407

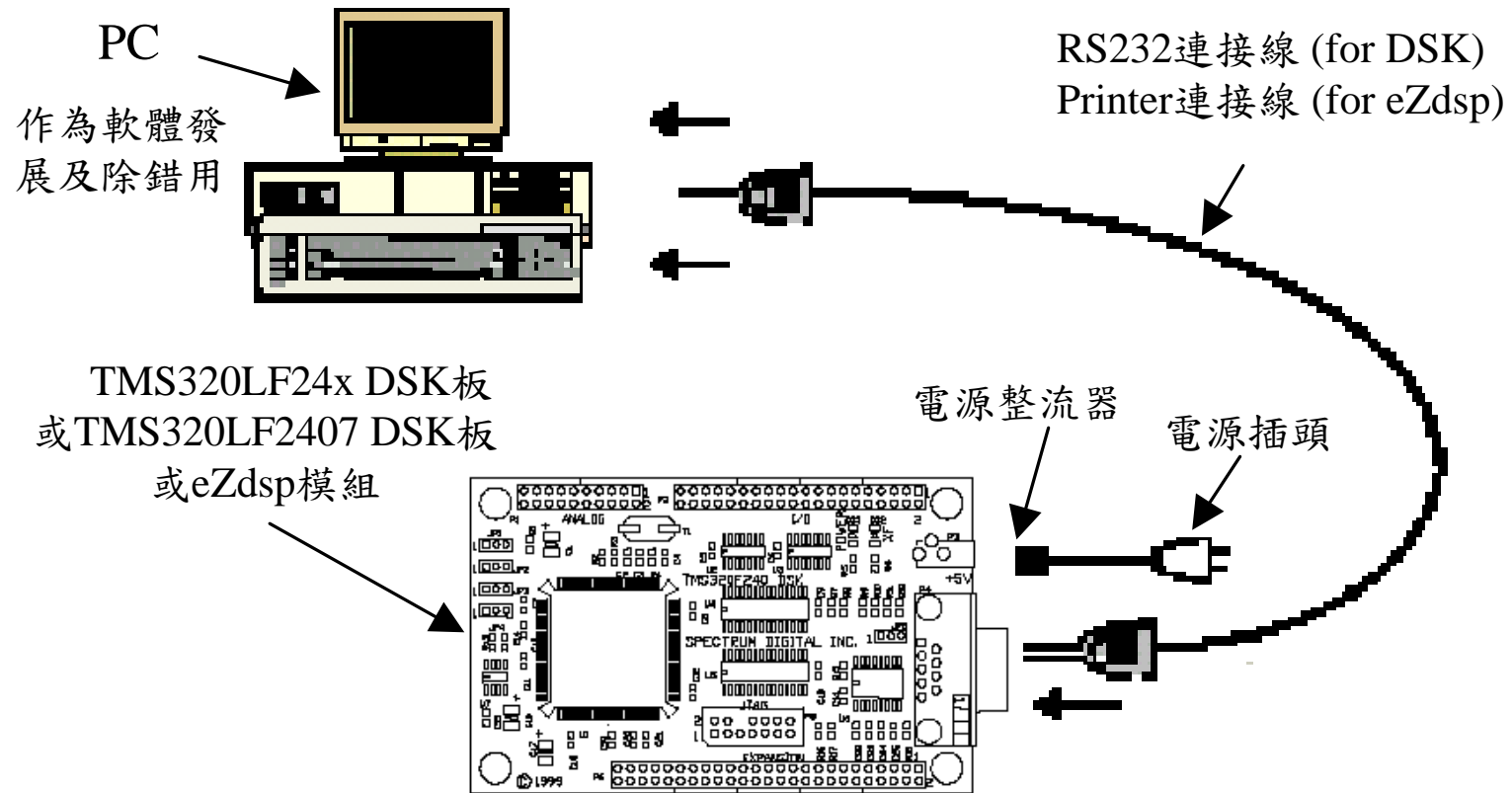
'28xx DSP : 2810, 2812



TMS320F24xx (含EV板及XDS510PP模擬器)



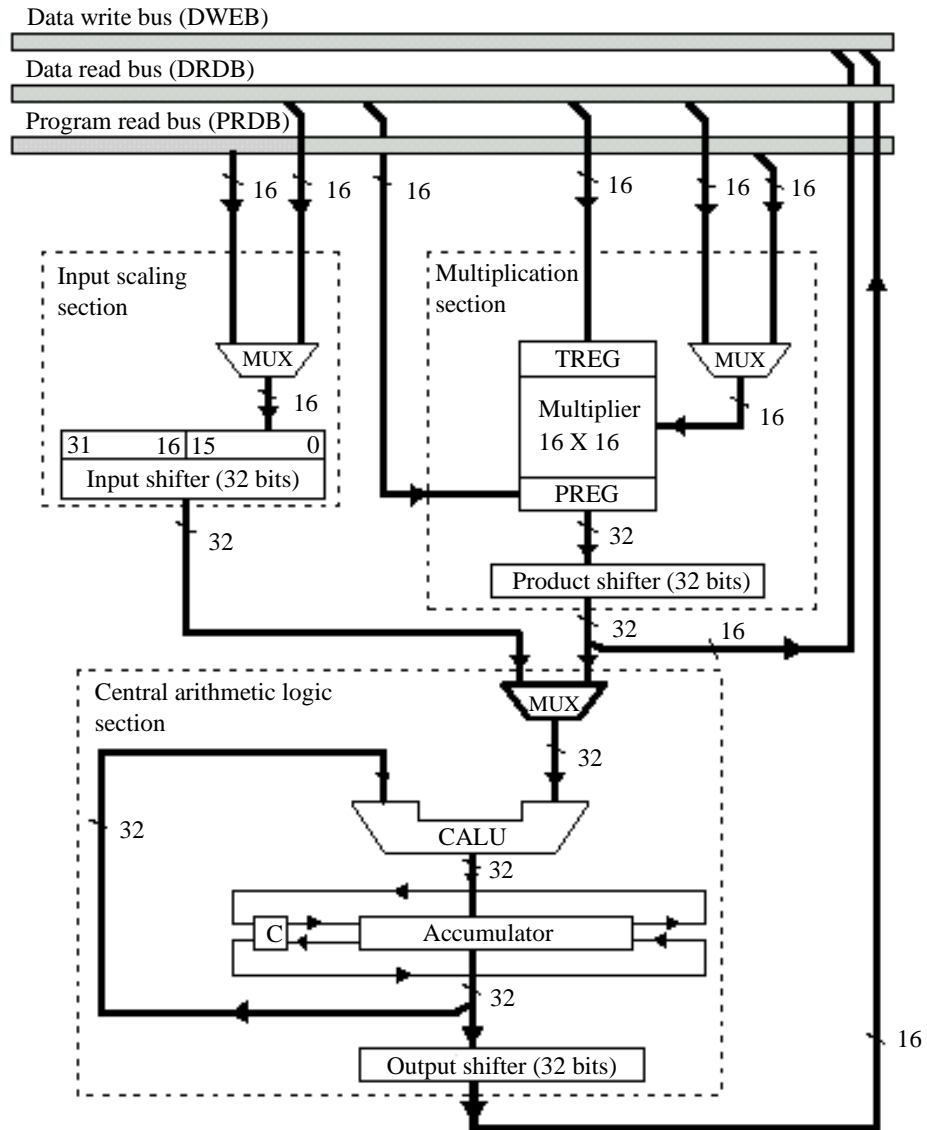
TMS320F24xx DSK(初學板) 及eZdsp發展系統



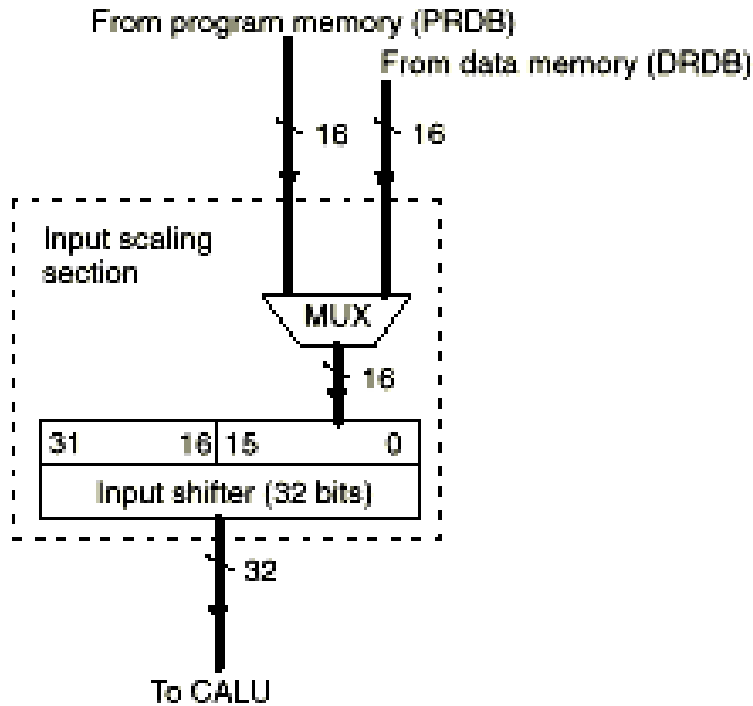
Central Processing Unit (CPU)

- Input Scaling Section
 - Multiplication Section
 - Central arithmetic logic unit (CALU) Section
 - Auxiliary register arithmetic unit (ARAU)
 - Status register ST0 and ST1
-

Central Processing Unit (CPU)



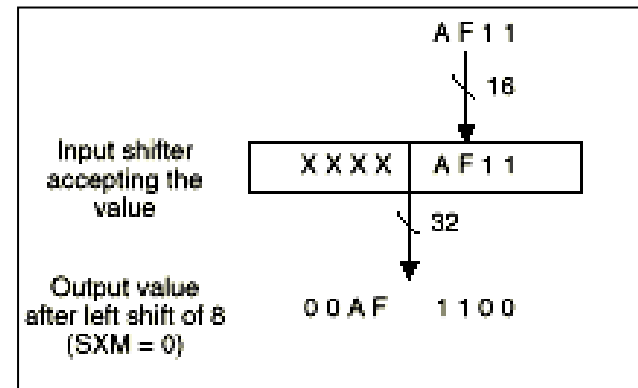
Input Scaling Section



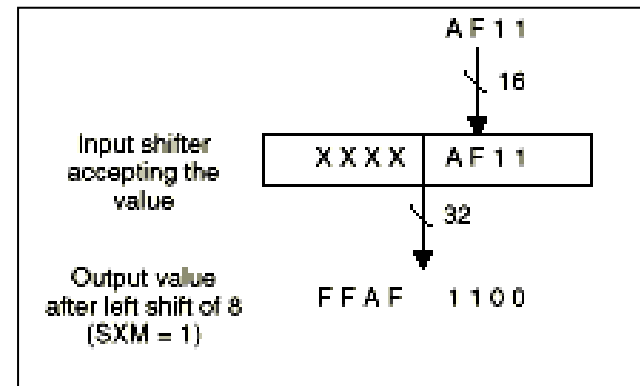
Sign-extension mode

bit(SXM)

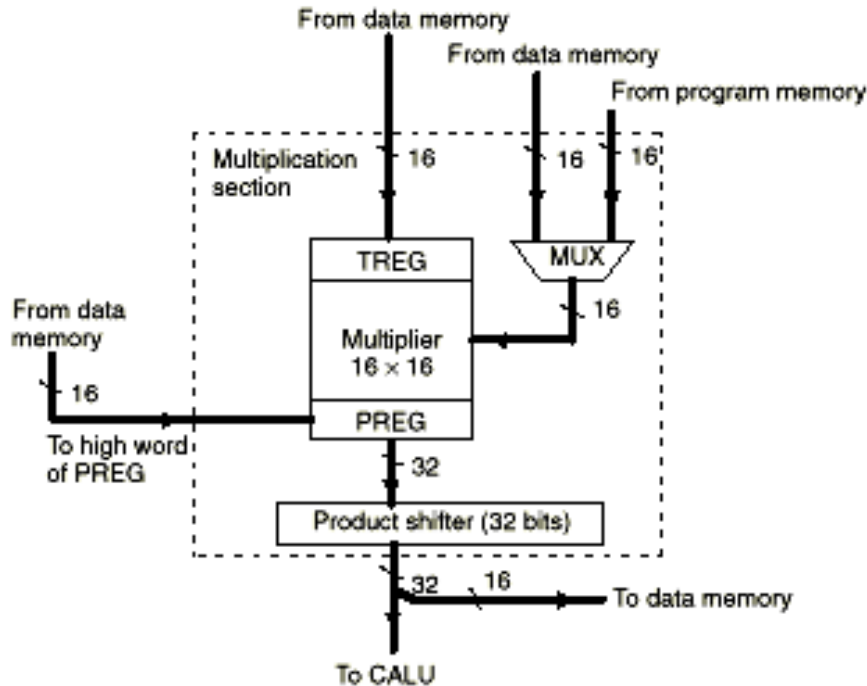
★ Operation of the input shift for SXM=0



★ Operation of the input shift for SXM=1



Multiplication Section



- 16-bit x16-bit hardware multiplier that can produce a signed or unsigned 32-bit product in a single machine cycle
- 16-bit temporary register (TREG)
- 32-bit product register (PREG)

Sum of Product Example

Compute $Y = A * X1 + B * X2 + C * X3 + D * X4$

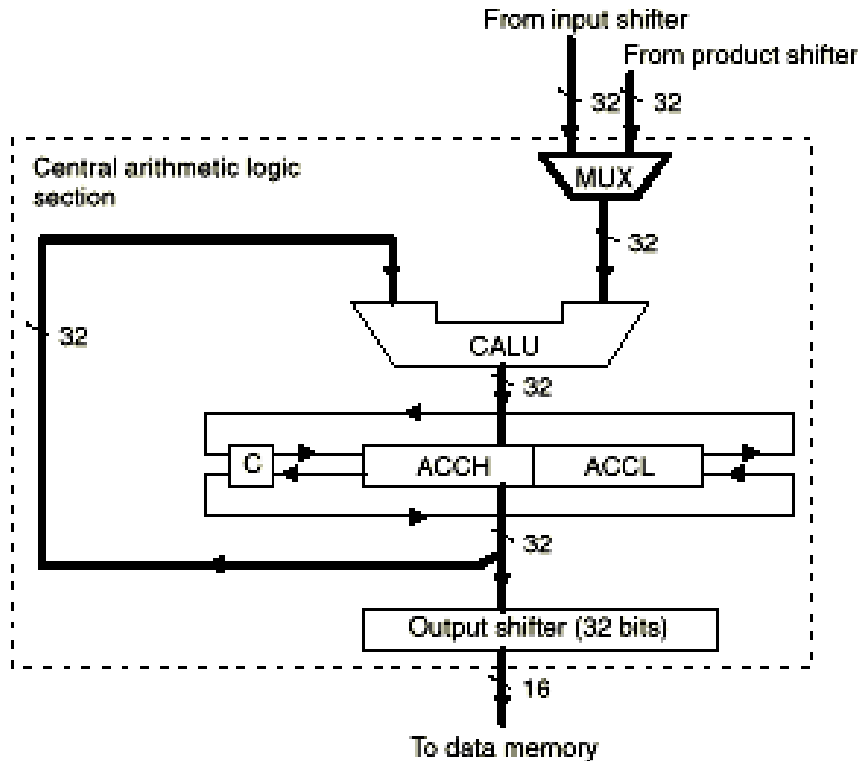
```

ZAC          ; ACC=0
LT   X1     ; T=X1
MPY  A      ; P=A*X1
APAC          ; ACC=A*X1
LT   X2     ; T=X2
MPY  B      ; P=B*X2
APAC          ; ACC=A*X1+B*X2
LT   X3     ; T=X3
MPY  C      ; P=C*X3
APAC          ; ACC=A*X1+B*X2+C*X3
LT   X4     ; T=X4
MPY  D      ; P=D*X4
APAC          ; ACC=A*X1+B*X2+C*X3+D*X4
SACH  Y1    ; Store 32_bit result
SACL  Y2    ; at locations Y1,Y2
    
```

● Product-scaling Shifter

PM	Shift	Comments†
00	No shift	Product sent to CALU or data write bus (DWEB) with no shift
01	Left 1	Removes the extra sign bit generated in a 2s-complement multiply to produce a Q31 product
10c	Left 4	Removes the extra four sign bits generated in a 16-bit x 13-bit 2s-complement multiply to produce a Q31 product when multiplying by a 13-bit constant
11	Right 6	Scales the product to allow up to 128 product accumulations without overflowing the accumulator. The right shift is always sign extended, regardless of the value of the sign-extension mode bit (SXM) of status register ST1.

Central arithmetic logic unit (CALU)



CALU

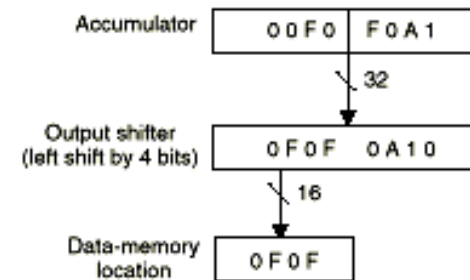
- 32-bit addition
- 32-bit subtraction
- Boolean logic operations
- Bit testing, shifting, and rotating

ACC

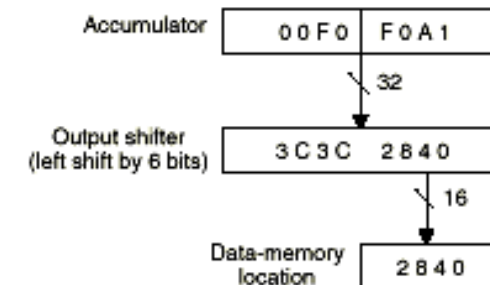
- 32-bit Accumulator
- Status bit :
Carry bit (C), Overflow mode bit(OVM)
Overflow flag bit (OV)
Test/Control bit (TC)

Output Data-Scaling Shifter

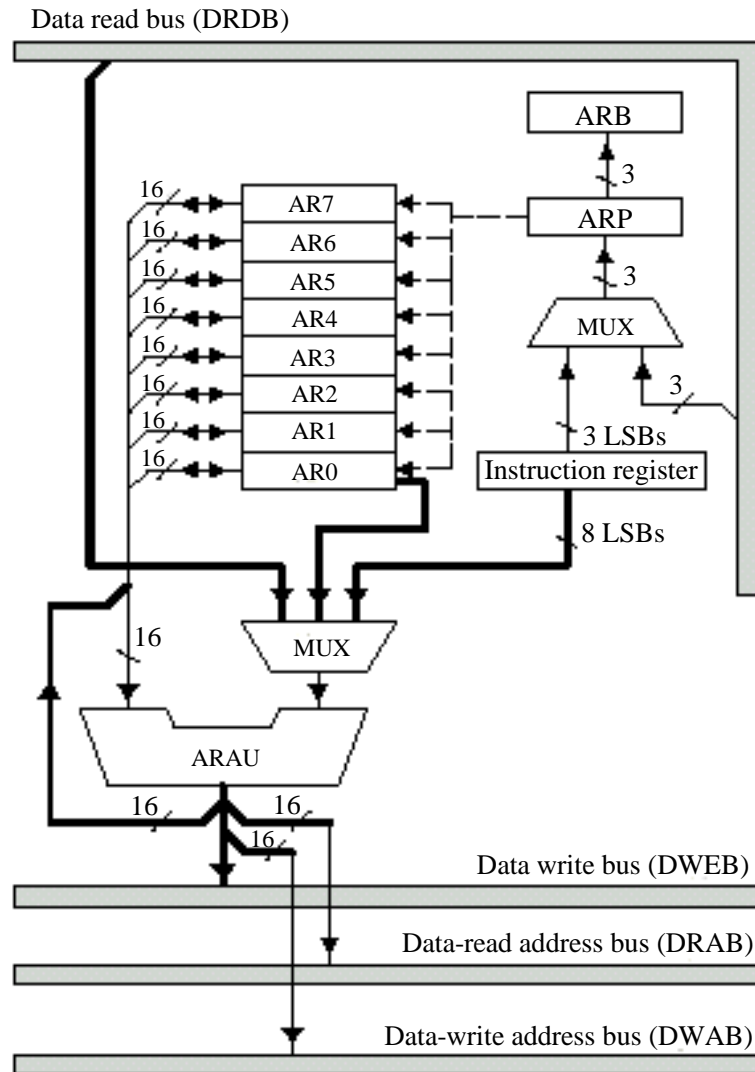
- Shifting and storing the high word of Accumulator



- Shifting and storing the low word of Accumulator



輔助暫存器算術單元(ARAU)

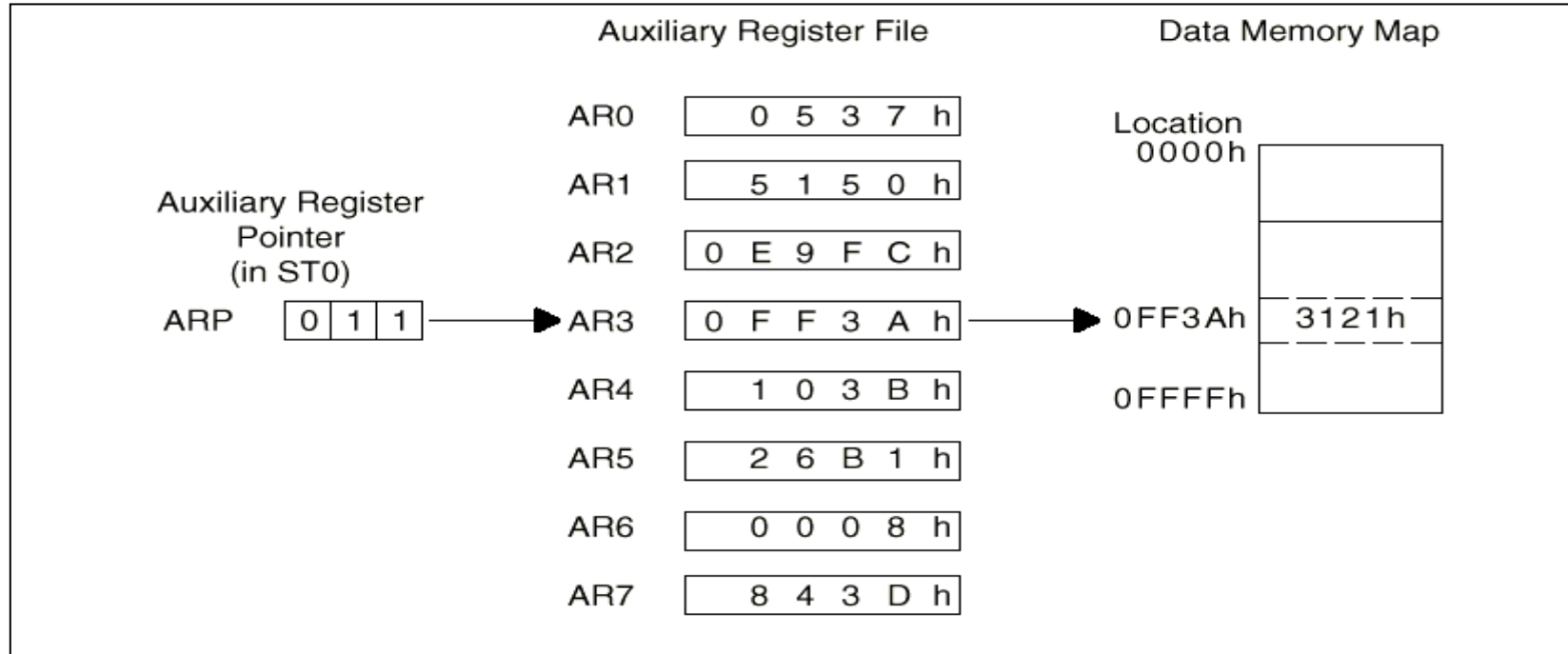


- ARAU主要功能為執行八個輔助暫存器(AR7至AR0)之算術運算，並可與CALU並行運算。
- ARAU內之八個輔助暫存器(AR7至AR0)在間接定址時提供強而有彈性之功能。所有64K資料記憶體空間皆可由輔助暫存器之內含值完全定址。

輔助暫存器算術單元(ARAU)

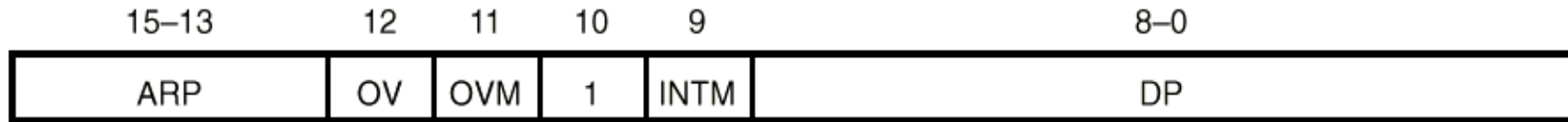
- 將輔助暫存器之內含值增、減1或某一指標量(藉由任一具有間接定址功能之指令)。
 - 將輔助暫存器之內含值加一個常數值(ADRK指令)或減一個常數值(SBRK指令)。
 - 將目前AR之內含值與AR0之內含值相比較，然後將結果存入狀態暫存器ST1之測試/控制旗標(TC)內(CMPR指令)。
-
- 藉著使用CMPR指令，輔助暫存器可以支援有條件分支(Branches)、呼叫(Call)及返回(Return)之功能。CMPR指令可執行將目前AR之內含值與AR0之內含值作比較，再將其結果置入狀態暫存器ST1之測試/控制旗標(TC)位元。 ex: **CMPR CM**
 - 將輔助暫存器當作暫時儲存處。可藉著使用LAR指令將常數值或資料記憶體之內含值載入輔助暫存器，或使用SAR指令將輔助暫存器之內含值存入資料記憶體內。
 - 使用輔助暫存器作為軟體記數器(Software Counter)，是需求可將其值增量或減量。
-

間接輔助暫存器定址



- Ex: **LACC** * ;ST0'(13-15-bit) ARP=3
- The ARAU updates the ARs during the decode phase (second stage) of the pipeline, while the CALU writes during the execution phase (fourth stage). Therefore, the two instructions that immediately follow the CALU write to an AR should not use the same AR for address generation.

狀態暫存器0 (ST0)



名稱

功能描述

- ARP** 輔助暫存器指標(Auxiliary register pointer)。在間接定址模式中，以ARP之3位元來決定哪一個輔助暫存器(AR)被選擇到。除了LST（載入狀態暫存器）指令之執行外，當輔助暫存器指標(ARP)被載入，則先前ARP值將會複製至ARB內。ARP可由MAR指令、LST指令及間接定址時記憶體參考指定來修改。當LST（載入狀態暫存器）指令在執行期間，若ARB被載入，則相同之數值也會複製至ARP內。
- OV** 溢位旗標位元(Overflow flag bit)。此位元可指示CALU是否有溢位發生。一旦CALU溢位發生時，OV位元將保持被設定為1，直到系統重置、有條件分支及LST指令才可將OV位元清除為0。
- OVM** 溢位模式位元(Overflow mode bit)。OVM決定CALU在溢位發生時如何處理。
SETC及CLRC指令可設定及清除此位元。LST指令也可用來修改OVM。
OVM = 0 溢位結果存於累積器內
OVM = 1 根據溢位發生狀況，累積器被填入最大正值或最大負值

名稱	功能描述
INTM	<p>中斷模式位元(Interrupt mode bit)。此位元使可遮罩中斷致能或禁能。INTM可由 SETC INTM及CLRC INTM指令來設定或清除。INTM不能影響不可遮罩中斷及軟體中斷。INTM不受LST（載入狀態暫存器）指令影響。當中斷發生時(除了TRAP指令之例子)，INTM被設定為1。系統重置時，此位元為1。</p> <p>INTM = 0 所有不被遮罩之中斷將被致能 INTM = 1 所有可遮罩之中斷將被禁能</p>
DP	<p>資料頁指標(Data page pointer)。當指令使用直接定址時，9位元之DP值結合指令字組7個LSBs將形成16位元資料記憶體位址。LST指令及LDP(載入DP)指令可修改DP值。</p>

狀態暫存器1 (ST1)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARB		CNF		TC	SXM	C	1†	1†	1†	1†	XF	1†	1†	PM	
R/W-x		R/W-0		R/W-x	R/W-1	R/W-1					R/W-1			R/W-00	

名稱

功能描述

ARB 輔助暫存器緩衝器(Auxiliary register pointer buffer)。不管何時，只要輔助暫存器指標(ARP)被載入，則先前ARP值將會複製至ARB內。除外，當LST(載入狀態暫存器)指令在執行期間，若ARB被載入，則相同之數值也會複製至ARP內。

CNF 晶片內DARAM規劃位元(On-chip DARAM configuration bit)。此位元決定DARAM區塊(B0區塊)被規劃為資料空間或程式空間。此位元可由SETC CNF，CLRC CNF及LST指令來修正。系統重置時，此位元為0。

CNF = 0 將DARAM區塊(B0區塊)規劃為資料空間

CNF = 1 將DARAM區塊(B0區塊)規劃為程式空間

TC 測試/控制旗標位元(Test/control flag bit)。下列情況TC位元將被設為1。

●若以BIT或BITT指令作位元測試時為1

●以CMPR指令執行目前AR及AR0間之比較測試結果為真

●以NORM指令測試累積器內含值兩個MSBs位元之互斥或閘為真
依據TC之狀態，程式可執行有條件分支、呼叫及返回指令。

名稱

功能描述

- SXM** **符號擴展模式位元**(Sign-extension mode bit)。SXM不會影響某些指令之基本操作。例如ADDS指令不管SXM為何值，符號擴展皆將被抑制。此位元可由SETC SXM指令來設定為1，或由CLRC SXM指令來清除為0，也可由LST指令來載入。系統重置時，SXM被設定為1。
- SXM = 0 此模式抑制符號擴展
- SXM = 1 此模式時，在輸入移位器之資料值傳至CALU前被左移位者，將被符號擴展
- C** **進位位元**(Carry bit)。假如加法運算產生進位，此位元會被設定為1；假如減法運算產生借位，此位元會被清除為0。否則，除了具16位元移位之ADD及SUB指令外，在加法運算後將被清除為0或在減法運算後將被設定為1。單一位元之移位、旋轉及SETC、CLRC與LST指令也會影響此位元。依據C之狀態，程式可執行有條件分支、呼叫及返回指令。系統重置時，此位元為1。
- XF** **XF腳位狀態位元**(XF pin status bit)。此位元決定XF腳位之狀態。XF腳位為一般目的使用之輸出腳位。系統重置時，XF被設定為1。
- PM** **乘積移位模式**(Product shift mode)。PM主要決定PREG內含值傳至CALU或資料記憶體前之移位數。需注意的是，PREG之內含值保持不變，過程中僅將此值複製到乘積移位器，然後再進行移位動作。PM可由SPM指令或LST指令載入。系統重置時，PM位元被清除。
- PM = 00 PREG之輸出傳至CALU或資料記憶體時並無移位
- PM = 01 PREG之輸出傳至CALU或資料記憶體前左移一位
- PM = 10 PREG之輸出傳至CALU或資料記憶體前左移四位
- PM = 11 此模式產生右移四位，並且符號擴展
-

Sum of Product Example

```
* sumofpdl.asm
    .title " program for sum of product (method 1)"

    .include "..\\include\\pathway.inc"
    .entry
    .ds 0300h
A    .word 2
B    .word 3
C    .word -3
D    .word 2
X1   .word -1
X2   .word 2
X3   .word 2
X4   .word 5
Y1   .word 0
Y2   .word 0

    .ps 0fe00h        ; starting address for this section is
                    ; 0fe00h in Program Space
B    0000h           ; (00h) Hardware Reset
B    Phantom_ISR    ; (02h) Interrupt Level 1
B    Phantom_ISR    ; (04h) Interrupt Level 2
B    Phantom_ISR    ; (06h) Interrupt Level 3
B    Phantom_ISR    ; (08h) Interrupt Level 4
B    Phantom_ISR    ; (0Ah) Interrupt Level 5

    .ps 0fe50h
    .entry
start
SETC  INTM           ; INTM = 1, disable global interrupts
LDP   #0e8h          ; DP -> 0x7400 - 0x747f (Event Manager)
SPLK  #0ffffh, IFR   ; clear all pending interrupts
SPLK  #0010h, IMR    ; enable Level 5 interrupts
                    ; so monitor can communicate with host

CLRC  INTM           ; enable interrupts
SETC  SXM            ; enable sign extension

loop  LDP    #06      ; setting data page
      ZAC    ; ACC=0
      LT     X1      ; T=X1
      MPY   A        ; P=A*X1
      LTA   X2      ; ACC=A*X1      ;T=X2
      MPY   B        ; P=B*X2
      LTA   X3      ; ACC=A*X1+B*X2  ;T=X3
      MPY   C        ; P=C*X3
      LTA   X4      ; ACC=A*X1+B*X2+C*X3 ;T=X4
      MPY   D        ; P=D*X4
      APAC  ; ACC=A*X1+B*X2+C*X3+D*X4
      SACH  Y1      ;Store 32_bit result
      SACL  Y2      ;at locations Y1,Y2
      B     loop

Phantom_ISR:
    B     Phantom_ISR
    .end

;產生執行檔
;compiler batch file (sumofpdl.bat)
;@echo off
;rem
;rem    Command Line Options:
;rem        -203 Use TMS320C2xx instruction set
;rem        -k   Produce .dsk Object Format
;rem        -l   Produce a List File (.lst)
;rem
;tasm -203 -k -l sumofpdl.asm
;
; After compiling, it will get two file, sumofpdl.dsk & sumofpdl.lst
;
;執行檔      列表檔
```

```

Dis-Assembly
FE4D 8B00 NOP
FE4E 8B00 NOP
FE4F 8B00 NOP
FE50      start
FE50 BE41 SETC   INTM
FE51 BCE8 LDP    #e8h
FE52 AE06 SPLK  #0ffffh,X3
FE54 AE04 SPLK  #10h,X1
FE56 BE40 CLRC  INTM
FE57 BE47 SETC   SXM
FE58      loop ; setting data page
FE58 BC06 LDP    #6h
FE59 B900 LACL  #0h
FE5A 7304 LT     X1
FE5B 5400 MPY   A
FE5C 7005 LTA   X2
FE5D 5401 MPY   B
FE5E 7006 LTA   X3
FE5F 5402 MPY   C
FE60 7007 LTA   X4
FE61 5403 MPY   D
FE62 BE04 APAC
FE63 9808 SACH  Y1
FE64 9009 SACL  Y2
FE65 7980 B     0fe58h,*
FE67      Phantom_ISR
FE67 7980 B     0fe67h,*

```

Data Memory (16-Bit Signed Int)

0300:	A						
0300:	2						
0301:	B						
0301:	3						
0302:	C						
0302:	-3						
0303:	D						
0303:	2						
0304:	X1						
0304:	-1						
0305:	X2						
0305:	2						
0306:	X3						
0306:	2						
0307:	X4						
0307:	5						
0308:	Y1						
0308:	0						
0309:	Y2						
0309:	8	16	6	6	6	0	0
0310:	-559	96	-655	-655	1023	-16385	-1025
0317:	-12553	-15344	0	4096	4	-1	-1
031E:	-9	-16395	0	4106	263	0	-4098
0325:	-1	-1	-1	-24576	0	16396	0

C2xx Registers

ACC = 0000001B	ARO = FFFD	PC = FE64	
PREG = 0000000A	AR1 = 0000	SK0 = 1100	
TREG = 0005	AR2 = 0006	SK1 = 0400	
	AR3 = 0100	SK2 = 1000	
ARP = ARO	AR4 = 2008	SK3 = 0000	
DP = 0300	AR5 = FFFF	SK4 = FFE7	
STO = 0606	AR6 = 7FFF	SK5 = F3FD	
ST1 = 15FC	AR7 = FFFF	SK6 = FFFF	
IMR = 0010	OV = 0	C = 0	TC = 0
IFR = 0010	OVM = 0	PM = 0	XF = 1
GREG = 0000	INTM = 1	SXM = 1	CNF = 1