UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

ARQUITECTURAS DE DPS's FAMILIA TMS320C50.

ING. LARRY ESCOBAR SALGUERO.

PRESENTACIÓN

La Facultad de Ingeniería ha decidido realizar una serie de ediciones provisionales de obras recientemente elaboradas por académicos de la institución, como material de apoyo para sus clases, de manera que puedan ser aprovechadas de inmediato por alumnos y profesores. Tal es el caso de la obra *Arquitecturas de DSP's familia TMS320 y el TMS320C50*, elaborada por Larry Hipólito Escobar Salguero.

Se invita a los estudiantes y profesores a que comuniquen a los autores las observaciones y sugerencias que mejoren el contenido de la obra, con el fin de que se incorporen en una futura edición definitiva.

ARQUITECTURAS DE DSP's

FAMILIA TMS320 Y EL TMS320C50

Por: Larry Escobar Salguero.

Facultad de Ingeniería

UNAM

Junio del 2000

TEMARIO

INTRODUCCION

Las presentes notas constituyen una breve introducción a las arquitecturas de Procesadores de Señales Digitales (DSP) de la familia TMS320 de la compañía Texas Instruments y en particular al DSP TMS320C50 de aritmética en punto entero.

El objetivo de este trabajo es aportar a la comunidad de la Facultad de Ingeniería UNAM un material de apoyo que de los elementos mínimos necesario de los DSPs y además motive a los alumnos de la materia Procesamiento Digital de Señales, alumnos de servicio social, tesistas y todo aquel interesado en esta área.

Es de hacer notar que este documento es una condensación de los manuales de usuario y de aplicaciones editados por Texas Instruments, así como un pequeño aporte de autor en su elaboración y uso de estos dispositivos, algunos ejemplos y aplicaciones con DSPs se pueden encontrar en el Manual del Laboratorio de Procesamiento Digital de Señales del mismo autor y editado por la FI, UNAM.

Es importante señalar que para entender estas notas y recibir el curso correspondiente es necesarios tener claro los conceptos de análisis de señales y sistemas, procesamiento digital de señales, filtros digitales, microprocesadores y lenguaje ensamblador.

Larry Escobar.
Profesor Asociado.
Facultad de Ingeniería, UNAM.
Junio del 2000.

EL PROCESAMIENTO DIGITAL DE SEÑALES

Y SUS APLICACIONES

Por: Larry Escobar Salguero.

Facultad de Ingeniería

UNAM

Junio del 2000

EL PROCESAMIENTO DIGITAL DE SEÑALES Y SUS APLICACIONES

El ser humano en sus actividades diarias realiza una gran variedad de tareas como reconocer, clasificar, comparar, agrupar, abstraer, comprimir, almacenar, cuantificar, inferir, predecir etc, todo este tipo de tareas las realiza sobre información que percibe del mundo real representada en imágenes, sonidos, fenómenos naturales, ondas electromagnéticas y acústicas que en general pueden ser caracterizadas como señales.

En la actualidad con el avance de la tecnología, la computadora se ha convertido en una herramienta fundamental en la simulación y el Procesamiento Digital de Señales (PDS) siendo una alternativa real para la solución de problemas de medición, control, filtrado, análisis espectral, comunicaciones etc.

Entre las áreas de interés que conforman el PDS están el análisis de señales y sistemas, el análisis y síntesis de filtros digitales, la identificación de sistemas, la ingeniería de software, la arquitectura de microcomputadoras y el diseño con circuitos de muy alta escala de integración (VLSI). El estudio y entendimiento de cada una de estas diferentes áreas, forman un conjunto de principios fundamentales que constituyen la disciplina del Procesamiento Digital de Señales.

Los principios fundamentales del PDS están dados por la teoría de señales y sistemas, las tareas del PDS se realizan eficiente y sistemáticamente a través de sistemas que combinan la tecnología de los circuitos VLSI con una gran variedad de algoritmos matemáticos y con el software necesario para su implementación.

Uno de los objetivos de un sistema de PDS es proveer una mejor aproximación del análisis o estimación contenido de la información. El análisis de señales es el proceso de definir y cuantificar las características de una señal para una aplicación.

El PDS se puede concebir como un conjunto de operaciones básicas aplicadas sobre una señal de entrada para obtener una señal de salida, formalmente estas operaciones se describen como transformaciones matemáticas de una señal a otra por una regla predefinida.

Para su estudio el PDS se puede dividir en:

1) **La parte conceptual**, que involucra los fundamentos del PDS y la generación de nuevos algoritmos.

2) **La parte algorítmica**, que trata con la interpretación y la utilización de los algoritmos existentes, es decir, la proposición de soluciones a problemas concretos.

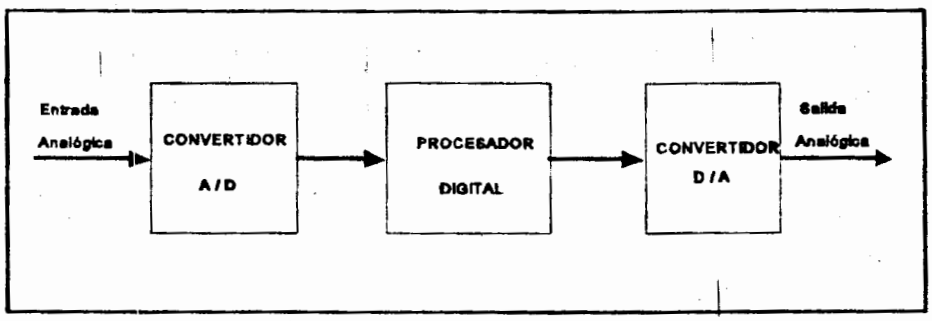
3) **La implantación de los algoritmos.**

- En esta parte se puede presentar alternativas de solución como la simulación de los algoritmos a través de lenguajes de alto nivel, la validación y verificación de los algoritmos.

- El traslado de los algoritmos a arquitecturas de procesamiento específicas, es decir la utilización de un lenguaje ensamblador para la instalación de una aplicación concreta. Esta etapa involucra la verificación algorítmica-software-hardware de la solución al problema, lo que constituiría un prototipo final.

ELEMENTOS BASICOS DE UN SISTEMA DE PROCESAMIENTO DIGITAL DE SEÑALES:

Para efectuar el PDS es necesario una interface entre la señal analógica y un procesador digital, y a la salida la señal procesada digitalmente debe convertirse a una señal analógica. El Procesador digital puede ser desde una computadora digital hasta un pequeño microprocesador cuyo programa efectúa las operaciones deseadas sobre la señal de entrada.



Sistema Básico de PDS

Con el advenimiento de los Procesadores de Señales Digitales (DSP), en la actualidad en la mayoría de aplicaciones orientadas a señales, el bloque del procesador digital está constituido por un DSP, por lo que en este trabajo se pretende conocer principalmente la arquitectura del DSP TMS320C50.

VENTAJAS DE PROCESAMIENTO DIGITAL SOBRE EL ANALOGICO:

- La programación de un sistema digital permite más flexibilidad en la reconfiguración del sistema, simplemente un cambio del programa permite cambiar el algoritmo o la aplicación. En un sistema analógico su reconfiguración implica un rediseño del hardware.
- El PDS provee un mejor control de los requerimientos de precisión. Debido a la tolerancia de los circuitos analógicos es muy difícil el control de precisión.
- Una señal digital es muy fácil almacenarla sin deterioro o pérdida de fidelidad, las señales almacenadas son fácilmente transportables para ser analizadas y/o procesadas remotamente.
- Una limitación del PDS es la velocidad de operación de un convertidor A/D. Las señales con un gran ancho de banda requieren velocidades de muestreo y conversión muy altas, sin embargo, con el avance de la tecnología actual esta limitición tiende a desaparecer.

CARACTERISTICAS DEL PROCESAMIENTO DIGITAL DE SEÑALES

El PDS cubre una gran cantidad de aplicaciones, las cuales tienen algunas características en común:

- Algoritmos matemáticos intensivos.
- Operaciones en tiempo real.
- Implementación de muestreo de datos.
- Sistemas flexibles.
- Movimiento de bloques de datos.
- Operaciones de multiplicación acumulación rápida.
- Direccionamiento eficiente de tablas.
- Operaciones en paralelo.
- Tamaño adecuado de palabra.
- Memoria RAM interna de alta velocidad.

APLICACIONES DE LOS DSP's[1]

Los DSP han sido exitosamente aplicados en voz y audio en productos de tiempo real tales como modems, canceladores de eco, codificación de voz y audio, síntesis de voz, reconocimiento de voz, digitalización de audio, síntesis de música, procesamiento de música y múltiples ampliaciones en telecomunicaciones. Además se han utilizado en aplicaciones en tiempo real tales como gráficos, imágenes médicas, simulación y procesamiento de imágenes.

[1] DSP Digital Signal Processig

Algunas aplicaciones y funciones que realizan los DSPs

Filtros digitales	**Procesamiento de señales**
De respuesta finita al impulso (FIR) De respuesta infinita al impulso (IIR) Lattice-Lader Correlación Transformada coseno Windowing Filtrado adaptable Igualación de canal Eliminación de ruido Cancelación de eco	Compresión Expansión Energía Procesamiento homomórfico Ley μ Ley A Conversión
Procesamiento de datos	**Modulación**
Encriptado Scrambling Codificación Decodificación	Amplitud Frecuencia Fase QAM
Procesamiento numérico	**Análisis Espectral**
Operaciones matriciales Funciones trascendentales Funciones no lineales generación de números aleatorios Aproximaciones numéricas	Transformada rápida de Fourier (FFT) Transformada discreta de Fourier (DFT) Modelo moving average (MA) Modelo autorregresivo (AR)

INTRODUCCION A LOS PROCESADORES DE SEÑALES DIGITALES

Inicialmente el procesamiento digital de señales se realizaba en grandes computadoras (maiframes) debido a sus velocidades de procesamiento. Posteriormente el procesamiento se hizo en arreglos de procesadores convirtiéndose en una herramienta flexible y veloz.

Los procesadores de señales digitales (DSP) son microprocesadores especializados para efectuar operaciones numéricas en tiempo real, en aplicaciones donde se requieren numerosas cálculos aritméticos en un tiempo limitado. Por sus ampliaciones específicas los DSPs tienen arquitecturas que son significante diferentes a los microprocesadores tradicionales.

En su arquitectura los DSPs incorporan unidades de multiplicación acumulación donde el tiempo de un ciclo de instrucción puede se igual al tiempo de un ciclo de la aritmética en hardware. En la actualidad existen DSPs que efectúan multiplicaciones de 32 bits en punto flotante en 60 u 80 ns, esto es de 25 a 30 millones de operaciones de punto flotante por segundo (MFLOPS). Los actuales DSPs utilizan extensivamente operaciones en pipeline, memorias independientes y unidades trabajando en paralelo.

Los DSPs lograron alcanzar desempeños que sólo se habían logrado con arreglos de procesadores. En 1982 la compañía Texas Instruments (TI) lanza al mercado su primer DSP el TMS32010, posteriormente introdujo varias generaciones de esta familia estando en la actualidad en la cuarta generación con el TMS320C40 y la más avanzada generación TMS320C80 con un hardware de alto paralelismo. En 1996-98 se dió a conocer la familia TMS320C6xx con arquitecturas de ocho unidades funcionales trabajando en paralelo, que pueden ejecutar ocho instrucciones por ciclo y alcanzar desempeños de hasta 5,000 MIPS (millones de instrucciones por segundo) y hasta un GFLOP. Estos últimos procesadores trabajan con la tecnología de palabra de instrucción muy larga (VLIW).

ALGUNAS CARACTERISTICAS GENERALES DE LOS DSPs

Los DSPs disponen de una arquitectura y un conjunto de instrucciones orientadas al Procesamiento Digital de Señales. La arquitectura de un DSP normalmente contiene buses separados para el direccionamiento simultáneo de dos operandos, un multiplicador, corrimientos, una unidad de direccionamiento, puertos seriales y temporizadores. Entre las instrucciones de grandes posibilidades están el movimiento de bloques, multiplicaciones en un ciclo de instrucción, multiplicación acumulación para la realización de operaciones de convolución.

Evolución de los DSPs de Texas Instruments

Año	DSP	Instrucción MAC (ns)	Bits en punto Fijo	Bits en punto Flotante	Ejecución de FFT 1024 (ms)
1982	TMS320C10	390	16/32		
1985	TMS320C20	195	16/32		14.2
1985	TMS320C25	100	16/32		5.7
1987	TMS320C30	60	24/32	32/40	3.8
1990	TMS320C40	50	32/64	32/40	1.55
1993	TMS320C50	35	16/32		
1995	TMS320C54xx	20	16/40		
1997	TMS320C6xx	5	32/64	32/64	

ARQUITECTURA BASICA DE LA FAMILIA

TMS320 DE TEXAS INSTRUMENTS

Por: Larry Escobar Salguero.

Facultad de Ingeniería

UNAM

Junio del 2000

ARQUITECTURA BASICA DE LA FAMILIA TMS320 de TEXAS INSTRUMENTS

La familia de procesadores TMS320 de 16/32 bits de Texas Instruments, empleada para el Procesamiento Digital de Señales (PDS) combina la flexibilidad de un controlador de alta velocidad con la capacidad numérica de un arreglo procesador. Estas características hacen que esta familia ofrezca alta capacidad de ejecución y una arquitectura de gran funcionalidad para solucionar diversos problemas en telecomunicaciones, computación, comercio, industria, en aplicaciones militares, entre otras.

La combinación de la arquitectura tipo Harvard de la familia TMS320 (separación de los buses de datos y datos de programa) y su conjunto de instrucciones especiales para el DSP proveen una alta velocidad de ejecución y gran flexibilidad. De esta forma se produce una familia de procesadores capaces de ejecutar 10 MIPS (Millones de instrucciones por segundo), ya que se implementan funciones en hardware, que en comparación a otros procesadores, recurren al software o microcódigos para generarlas.

Un DSP para cumplir con sus propósitos de realizar algoritmos para el procesamiento digital de señales debe de realizar operaciones aritméticas muy rápidamente y manejar algoritmos con matemáticas intensivas para su realización en tiempo real. La familia de procesadores TMS320 tiene algunos conceptos de diseño básicos que los caracteriza y que son comunes en sus diversas generaciones de DSPs:

- Arquitectura Harvard.
- Extensivo Pipeline.
- Un multiplicador dedicado en hardware.
- Instrucciones especiales para el PDS.
- Ciclos de instrucción rápidos.

PRIMERA GENERACION DE LA FAMILIA TMS320

El TMS32010 es el primer miembro de la familia TMS320 de Texas Instruments (TI), es un microprocesador capaz de efectuar multiplicaciones de 16 x 16 bits en un simple ciclo de instrucción de 200 ns. Contiene 144 palabras disponibles en memoria interna y hasta cuatro mil palabras de memoria externa para memoria de programa que pueden ejecutarse a velocidad máxima. También está disponible en la versión de microcomputadora con 1.5 K palabras de memoria interna ROM de programa y hasta 2.5 K palabras de memoria para programas externos, para un total de cuatro K palabras. Está fabricado con tecnología CMOS logrando una baja disipación de potencia (165 mW).

En esta familia existen algunas alternativas con características variadas dependiendo de la aplicación concreta del usuario.

El **TMS320C10-14**, es una versión del TMS320C10 a 14 Mhz, ofrece una alternativa para aplicaciones de DSP cuando no se requiere operación a máxima frecuencia, puede ejecutar 3.5 MIPS a un ciclo de instrucción de 280 ns.

El **TMS320C10-25**, es una versión del TMS320C10 a 25 Mhz, tiene un ciclo de instrucción de 160 ns. Es de baja potencia y alta velocidad, conveniente para aplicaciones de alto desempeño en el procesamiento de señales digitales (PSD).

El **TMS320C14** y el **TMS320E14** son DSP's con un ciclo de instrucción menor de 160 ns, 256 palabras en RAM interna y cuatro K palabras en ROM interna para programa (el 'C14) o EPROM ('E14) respectivamente.

El **TMS320C15** y el **TMS320E15** es compatible en código y pin a pin con el TMS32010. Cada uno ofrece una expansión de memoria RAM interna de 256 palabras y cuatro K palabras en ROM interna ('C15) o EPROM ('E15). Ambos están en tecnología CMOS y en la versión de 160 ns de ciclo de instrucción.

El **TMS320C17** y el **TMS320E17** son microcontroladores, cada uno de 256 palabras de RAM interna y cuatro K palabras en ROM o EPROM. El **TMS320C17/E17** ofrece un puerto serial de doble canal capaz de efectuar una comunicación serie full-duplex y una interfaz directa a codecs. Además provee una conexión directa con microcomputadoras de 4/8 bits y microprocesadores de 16/32 bits. El hardware interno permite comprimir expandir (compand) datos en formato ley μ y ley A).

CARACTERISTICAS PRINCIPALES DEL TMS32010

- Bus de datos bidireccional de 16 bits.
- Acumulador de la unidad aritmético lógica (ALU) de 32 bits.
- Multiplicaciones de 16 x 16 bits con un producto de 32 bits.
- Corrimiento de 0 16 bits.
- Generador interno de reloj.
- Ocho canales de entrada y ocho de salida.
- Puede ejecutar 5 millones de instrucciones por segundo (MIPS).
- Dos registros índices para direccionamiento indirecto.
- Ciclo de instrucción de 200 ns.

ARQUITECTURA

La arquitectura del TMS320C1x (TMS10) incrementa su desempeño al permitir la búsqueda del programa y solapar operaciones de datos. El TMS10 efectúa multiplicaciones en un ciclo de instrucción, su flexibilidad mejora con el conjunto de instrucciones que soporta instrucciones de propósito general o de aplicaciones de PDS.

La memoria programa y datos están en dos buses separados, permitiendo un total solapamiento de la búsqueda de instrucción y la ejecución. Las modificaciones de la arquitectura Harvard de la familia TMS320 permite la transferencia entre los espacios de dato y programa incrementando así la flexibilidad.

La Unidad Aritmética Lógica **(ALU)** y el acumulador soportan doble precisión en aritmética de dos complemento con palabras de 16 bits. **El acumulador** (ACC) de 32 bits almacena la salida de la unidad ALU y además es una entrada a la ALU. El acumulador está dividido en parte alta (bit 31 al 16) y parte baja (bit 15 al 0) para instrucciones de almacenamiento de palabras en la parte baja o alta del acumulador.

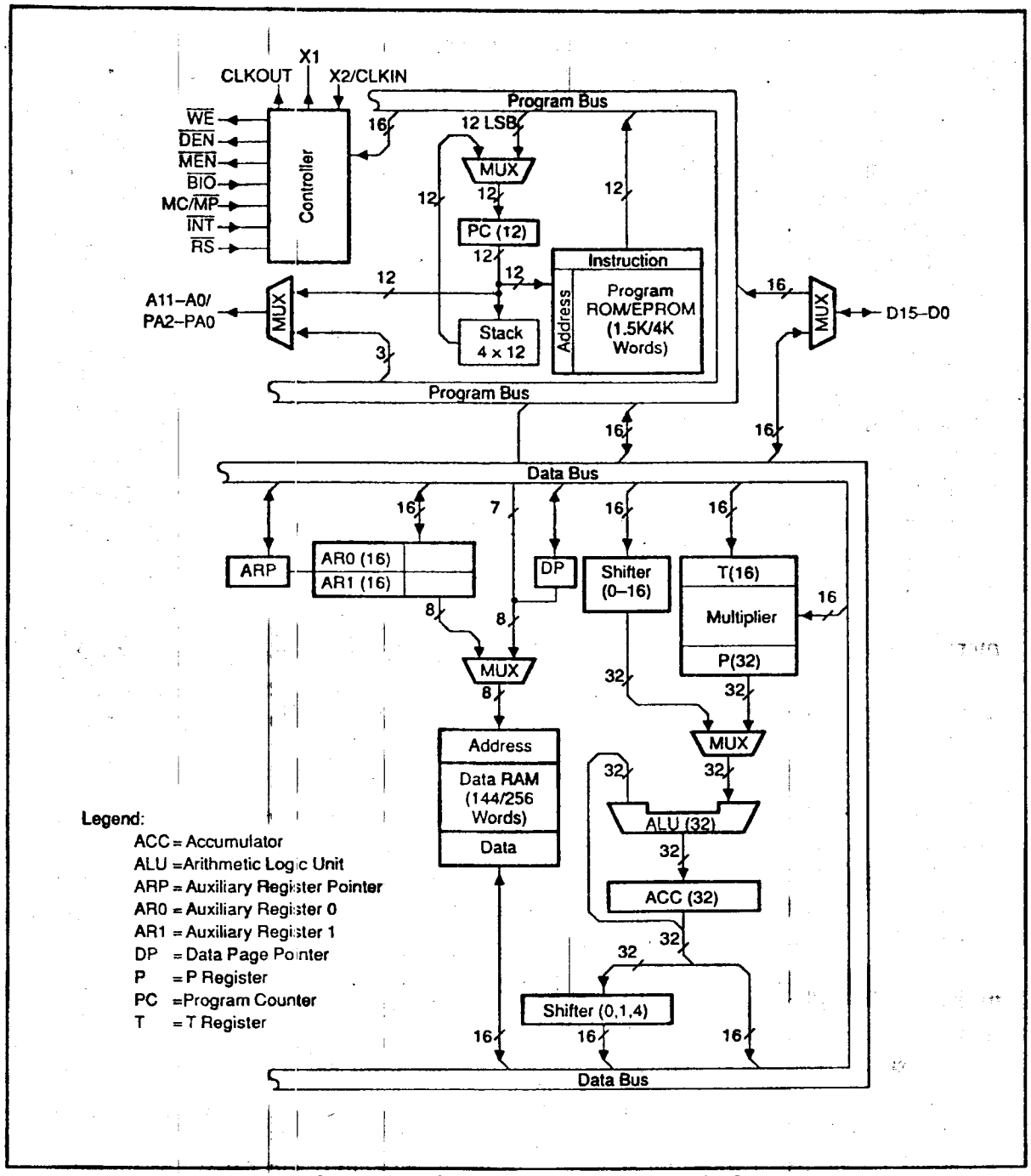
El multiplicador efectúa multiplicaciones en un ciclo de instrucción de 16 x 16 bits en dos complemento con resultado de 32 bits. El multiplicador consiste de tres elementos:

Registro **T**, de 16 bits para almacenamiento temporal del multiplicando.
Registro **P**, que almacena el producto de 32 bits.
Arreglo multiplicador.

Cada uno de los valores a multiplicar vienen de memoria dato o son derivados inmediatamente de la palabra de instrucción MPYK (multiplicación inmediata).

Existen dos posibilidades de **corrimiento** para datos:

- La ALU puede efectuar corrimientos de 0 a 16 bits a la izquierda para datos de memoria al cargarse a la ALU.
- El acumulador puede efectuar corrimientos de derecha a izquierda en 0,1 y cuatro posiciones al guardar su contenido en memoria. Estos corrimiento son útiles para el escalamiento y la extracción de bits.



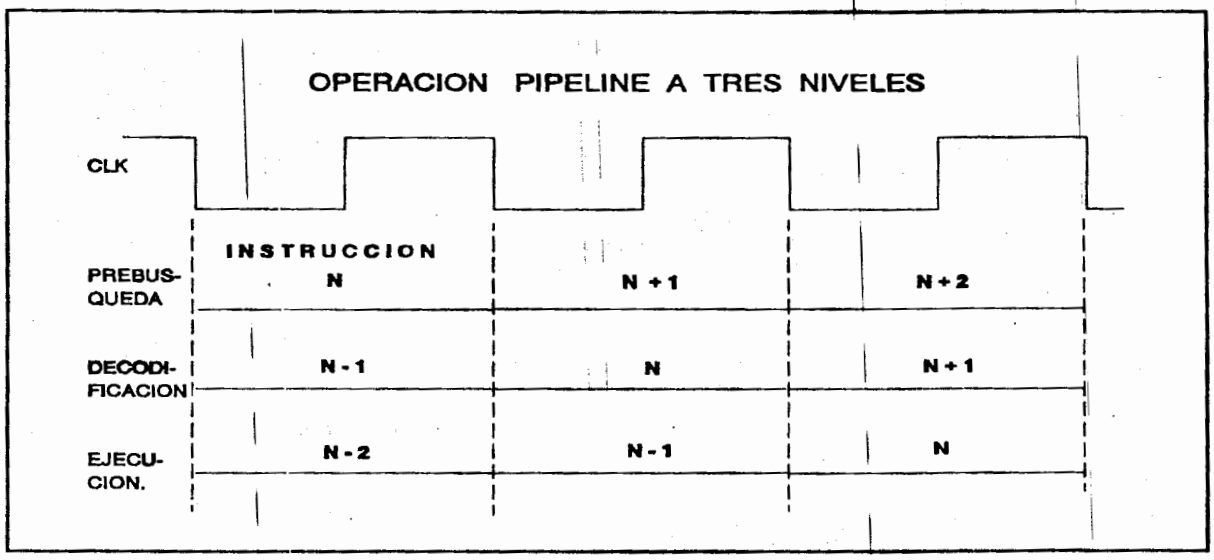
Arquitectura del TMS32C010 y TMS320C15

Contiene **cuatro niveles de pila** en hardware para salvar el contenido del contador de programa durante las interrupciones y llamadas de subrutinas, utilizando las instrucciones PUSH y POP. Los puertos de I/O son direccionados por los tres bits menos significativos (LSB) de las líneas de dirección.

LA OPERACION PIPELINE

Una instrucción de Pipeline consiste en una secuencia de operaciones externas de bus que ocurren durante la ejecución interna de la instrucción. El pipeline de las operaciones prebúsqueda, decodificación y ejecución es esencialmente transparente al usuario, excepto en algunos casos donde el pipeline debe ser interrumpido con instrucciones tales como saltos dentro del programa (B, BNZ,etc.). En la operación de pipeline, las operaciones prebúsqueda, decodificación y ejecución son independientes.

Las arquitecturas de los DSPs tienen algunos rasgos en común: mientras una instrucción es buscada, el operando de la instrucción previa es buscado, esto puede verse como un flujo donde una instrucción es buscada, una decodificada y otra instrucción es ejecutada, es decir una forma de pipeline de tres estados.



En un programa típico para DSP una instrucción buscará dos operandos de memoria, los multiplicará, sumará el producto anterior al acumulador, escribirá el resultado en memoria e incrementará el contenido de los registros de dirección; en forma secuencial llevará varios ciclos de tiempo, mientras que en pipeline se optimizará el tiempo de las operaciones mencionadas.
En esta arquitectura, las operaciones aritméticas son transferidas a estados sucesivos, donde unidades separadas de hardware proveen los cálculos a cada estado. Las computadoras que operan bajo este concepto son conocidas de procesamiento vectorial. La eficiencia de las funciones solo es posible si las operaciones aritméticas a ejecutar son vectorizables, es decir arregladas como un flujo continuo de datos. Una dificultad que se presenta con el pipeline son las ramificaciones, es decir que es difícil lograr eficientes saltos y ramificaciones debido a que el contador de programa debe de continuar en otra dirección del programa perdiéndose la secuencia de búsqueda y por lo tanto la operación de pipeline.

SEGUNDA GENERACION DE LA FAMILIA

TMS320

Por: **Larry Escobar Salguero.**

Facultad de Ingeniería

UNAM

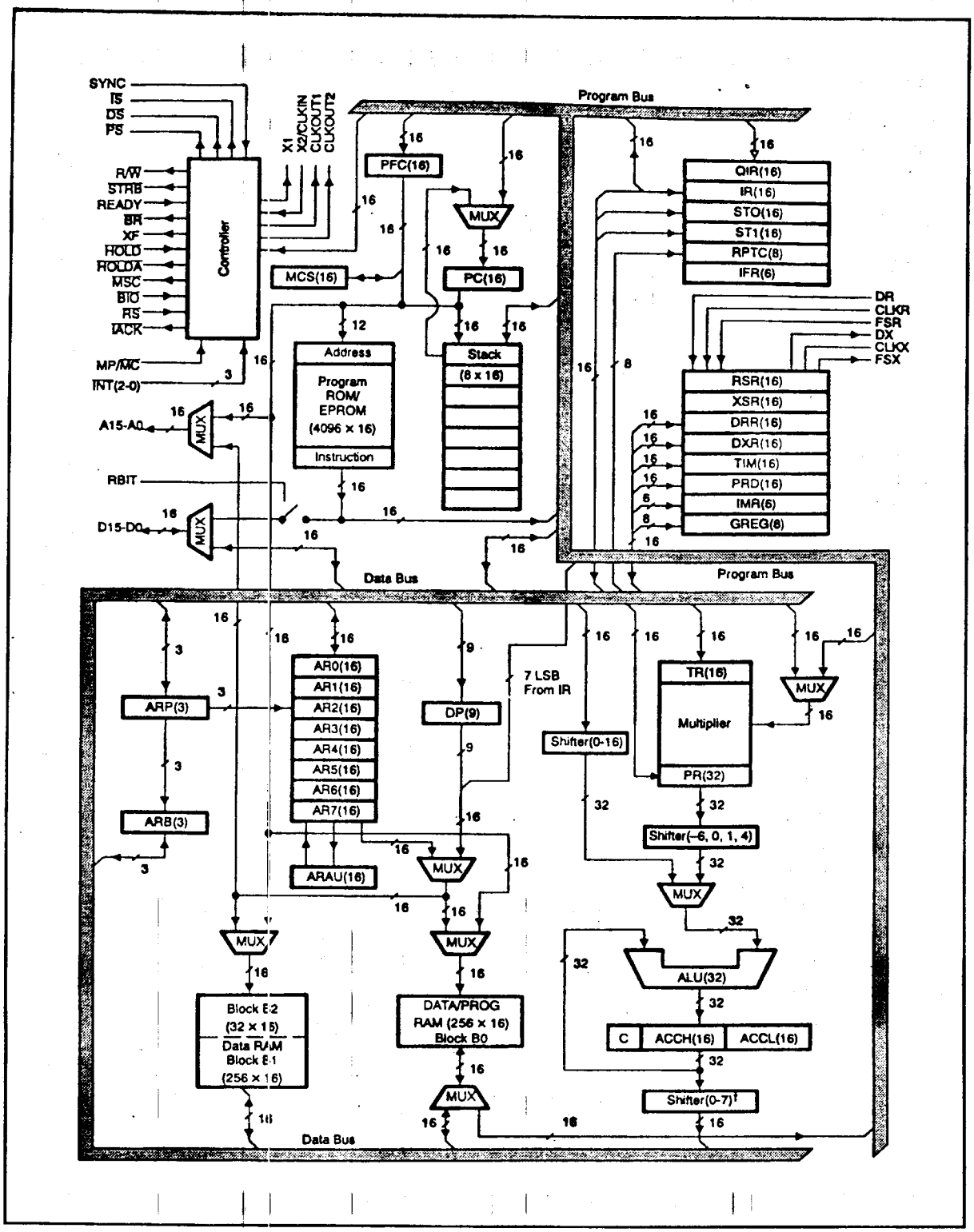
Junio del 2000

SEGUNDA GENERACION DE LA FAMILIA TMS320

La segunda generación de la familia TMS320 está compuesta de dos procesadores el TMS32020 y el TMS320C25, la arquitectura de ambos procesadores se basa en el TMS23010. El TMS32020 trabaja a 20 MHz y ejecuta el doble de instrucciones del TMS32010, mientras el TMS320C25 incrementa la funcionalidad de la arquitectura en relación al TMS32020.

CARACTERISTICAS DEL TMS320C25 (TMS25)

- Ciclo de Instrucción 100 ns (10 MIPS).
- 544-Palabras de datos programables en memoria RAM interna.
- 4K-Palabras de programa en memoria ROM interna.
- 128K-Palabras de espacio total de memoria DATOS y PROGRAMA.
- ALU/Acumulador de 32 bits.
- Multiplicador paralelo de 16x16 con producto de 32 bits.
- Instrucciones multiplicación/acumulación ejecutadas en un ciclo de instrucción.
- Instrucciones de repetición para uso eficiente del espacio de programa.
- Movimiento de bloques para el manejo de DATOS/PROGRAMA.
- Timer integrado para operaciones de control.
- Ocho registros auxiliares con una unidad aritmética propia .
- Un stack por hardware de ocho niveles.
- Dieciseis puertos de entrada o salida.
- Un registro de corrimiento paralelo de 16 bits.
- Posibilidad de generar tiempos de espera para comunicación a periféricos o memorias de respuesta lenta.
- Un puerto serie para interfaz directa a codecs.
- Sincronía en configuraciones de múltiples procesadores.
- Interfaz de memoria de datos global.
- Compatibilidad de códigos fuente del TMS32010.
- Acceso directo a memoria (DMA).
- Instrucciones para implementar filtros, la Transformada Rápida de Fourier, y aritmética de precisión extendida.
- Generador de reloj interno.
- Suministro de potencia 5 Volts.
- Tres niveles de pipeline.



Arquitectura del TMS320C25

DESCRIPCION DE LA ARQUITECTURA

La arquitectura tipo Harvard del TMS25 maximiza el procesamiento mediante dos estructuras de buses de memoria separadas (memoria programa y memoria datos), para incrementar la velocidad de ejecución, contando con instrucciones de transferencia de datos entre ambos espacios. Externamente, las memorias de programa y datos son multiplexadas sobre el mismo bus externo para maximizar el rango de direccionamiento para dichos espacios mientras se minimizan los pines del dispositivo.

La flexibilidad en el diseño del sistema es incrementada al contar con tres bloques de datos internos en RAM (un total de 544-Palabras), donde uno de ellos puede ser configurado como memoria programa o memoria dato. El TMS25 es capaz de direccionar externamente 64K-Palabras en un espacio de memoria dato, para facilitar la implementación de algoritmos para DSP.

Los programas de 4K-Palabras pueden ser mascarables en la memoria ROM interna y de esta forma pueden ser ejecutados a alta velocidad desde este espacio de memoria. Externamente el espacio de memoria de programa direccionable es de 64K-Palabras. El TMS25 funciona con una aritmética en modo dos complemento, empleando una ALU y un Acumulador de 32 bits.

El multiplicador realiza operaciones de 16x16 bits en modo dos complemento con un resultado de 32 bits en un solo ciclo de instrucción. El multiplicador está compuesto de tres elementos: el registro T, el registro P y un arreglo multiplicador.

El registro de corrimiento del TMS25 tiene una entrada de 16 bits y está conectado al bus de datos, mientras que su salida está conectada a la ALU 32 bits. Este registro proporciona corrimientos a la izquierda de 0 a 16 bits sobre el datos de entrada y son programados mediante instrucciones.

La interfaz local de la memoria del TMS25 consiste de un bus de datos paralelo de 16 bits (D15-D0), un bus de direcciones de 16 bits (A15-A0), tres pines para selección de memoria dato/programa o espacio de entrada/salida (/DS,/PS e /IS), y varias señales de control del sistema. La señal R/w controla el sentido de transferencia de datos, y la señal /STRB provee un tiempo válido para la transferencia de información. La señal READY sirve para generar tiempos de espera para comunicarse con memorias externas lentas.

El stack por hardware de ocho niveles es empleado para almacenar el contenido de contador de programa durante interrupciones y llamadas de subrutinas. Las operaciones de control son proporcionadas en el TMS25 por un timer interno de 16 bits mapeado en memoria, un contador de repetición, tres interrupciones externas mascarables, y una interrupción interna generada por el puerto serie o el timer.

Un puerto serie interno full-duplex provee comunicación directa con dispositivos seriales como codecs, convertidores seriales A/D, y otros dispositivos seriales.

El TMS25 soporta Acceso Directo a Memoria (DMA) para su memoria externa dato/programa empleando las señales /HOLD y /HOLDA. Otro procesador puede tomar por completo el control de la memoria externa. Sin embargo, de manera concurrente al modo DMA, el TMS50 continuará ejecutando su programa si éste opera con la RAM y ROM internas.

El TMS320C25 tiene un alto grado de paralelismo, es decir, que mientras está operando sobre la CALU, puede implementar operaciones aritméticas en la Unidad Aritmética de Registros Auxiliares (ARAU).

ORGANIZACIÓN DE LA MEMORIA

Las 544-Palabras de RAM interna se dividen en tres bloques separados: B0, B1 y B2. El bloque B0 de 256-Palabras es configurado como memoria programa o dato por medio de instrucciones CNFP o CNPD respectivamente. Como memoria dato, B0 reside en las páginas 4 y 5 del mapa de memoria de datos, y como memoria de programa de FF00h a FFFFh, pudiéndose utilizar la instrucción BLKP (movimiento de bloque de memoria programa a memoria dato) para cargar la información del programa al bloque B0 cuando éste es configurado como RAM de datos. Las 288-Palabras (bloques B1 y B2) siempre son configuradas como memoria dato. B1 está localizado en las páginas 6 y 7 abarcando 256 localidades mientras que B2 está en las treinta y dos localidades superiores de la página 0.

La memoria dato interna y las localidades reservadas para registros son mapeadas dentro de las primeras 1024-Palabras del espacio total de memoria dato.

CONJUNTO DE INSTRUCCIONES

El TMS25 cuenta con un conjunto de 133 instrucciones, 97 de ellas son ejecutadas en un ciclo, 36 requieren ciclos adicionales, 21 son para el control de flujo del programa, otras siete son de dos palabras (palabras largas inmediatas), las ocho restantes soportan transferencia I/O y de datos entre espacio de memoria. Además posee tres formas de direccionamiento: el indirecto, directo e inmediato.

QUINTA GENERACION DE LA FAMILIA

TMS320

EL DSP TMS320C50

Por: **Larry Escobar Salguero.**

Facultad de Ingeniería

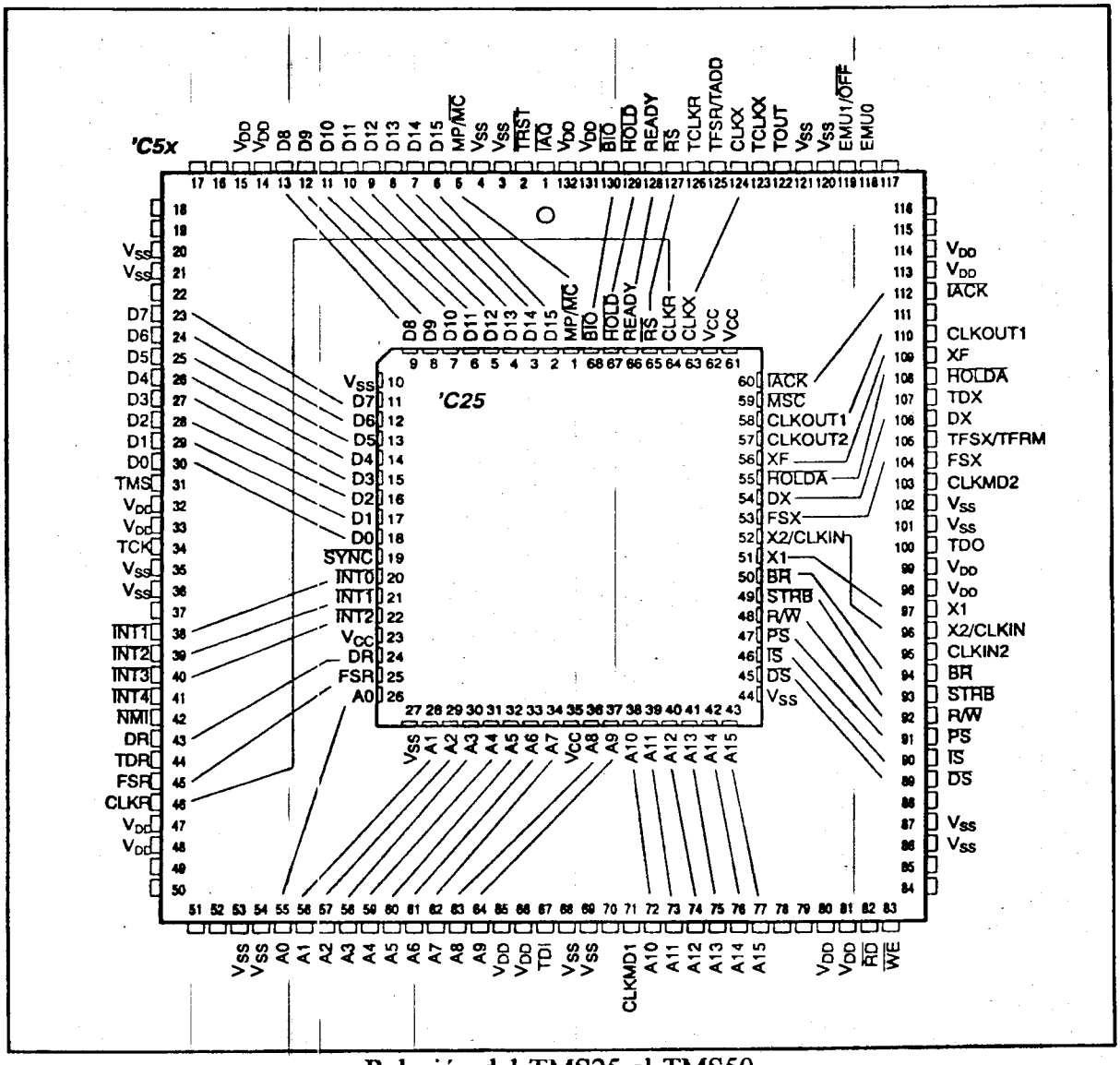
UNAM

Junio del 2000

QUINTA GENERACION DE LA FAMILIA TMS320
EL DSP TMS320C50

El TMS320C50 (TMS50) es un procesador de punto fijo de la familia TMS320 que está basado en el TMS320C25 con una arquitectura adicional que mejora el desempeño, entre estas caraterísticas están:

- Ciclo de instrucción más rápido
- Manipulación de bits vía la unidad paralela lógica.
- Gran cantidad de memoria interna.
- Estados de espera por software.
- Respuesta rápida a interrupciones.
- Bloques de repetición.
- Buffer circular.
- Bloques de repetición.
- Corrimientos de 0 a 16 bits en el acumulador.
- Registros shadow.



Relación del TMS25 al TMS50

Los dispositivos de la generación 'C5x son capaces de efectuar operaciones en dos veces la velocidad de la generación 'C2x y su código es compatible hacia arriba con las familias 'C1x y 'C2x. La familia 'C5x está diseñada para ejecutar 28 Millones de instrucciones por segundo (MIPS).

RASGOS PRINCIPALES DEL DSP TMS320C50

- 35/50 ns de ciclo de instrucción para instrucciones en punto entero (28.6/20 MIPS).
- 9K x 16 bits de memoria RAM interna, con acceso en un ciclo (SARAM).
- 2K x 16 bits en memoria ROM acceso en un ciclo.
- 1056 x 16 bits en memoria RAM interna, puede accesarse dos veces por ciclo de máquina (DARAM).
- 224k x 16 bits máximo espacio direccionable en memoria externa (64 K palabras[2] de programa, 64 k palabras de datos 64 k puertos I/O y 32 k palabras globales)
- Acumulador (ACC) de 32 bits y un acumulador buffer (ACCB) de 32 bits.
- Acumulador (ACCB) de 32 bits.
- Unidad Lógica Paralela (PLU) de 16 bits.
- Multiplicador paralelo de 16 x 16 con capacidad de productos de 32 bits.
- Instrucciones de multiplicación/acumulación en un ciclo simple.
- Ocho registros auxiliares con una unidad aritmética para direccionamiento indirecto.
- Ocho niveles de pila por hardware.
- 0 a 16 corrimientos a la izquierda.
- Dos buffer de direccionamiento indirecto para direccionamiento circular.
- Instrucciones de repetición de bloques.
- Instrucciones para el manejo de movimiento de bloques entre datos y programa.
- Puerto serial síncrono full-duplex para comunicación directa con otros 'C5x y otros dispositivos seriales.
- Puerto serial de múltiple acceso por división de tiempo (TDM).
- Temporizador
- 64k puertos I/O paralelos, con 16 puertos mapeados en memoria.
- 16 estados de espera programables por software, para programa, datos o espacio I/O.
- Operación de DMA.
- Cuatro niveles de Pipeline.
- Direccionamiento indexado.
- Direccionamiento de bit reverso para efectuar FFT's radix 2.
- Generador interno de reloj.
- Polarización de 5 volts, con modo power down.
- Empacado en 132 pines.
- 28 registros mapeados en memoria.
- Cuatro interrupciones externas y cinco internas una del timer y cuatro para puerto serie.
- 32 interrupciones por software.

[2] una palabra equivale a 16 bits

La familia TMS320C5x está disponible en diferentes versiones que se caracterizan por la distribución de la memoria interna:

	SARAM	ROM
TMS320C50	10K	2K
TMS320C51	2K	8K
TMS320C52	1K	4K
TMS320C53	4K	16K
TMS320C56	7K	32K

Algunas de estas versiones están disponibles para polarización de 5 y 3.3 volts.

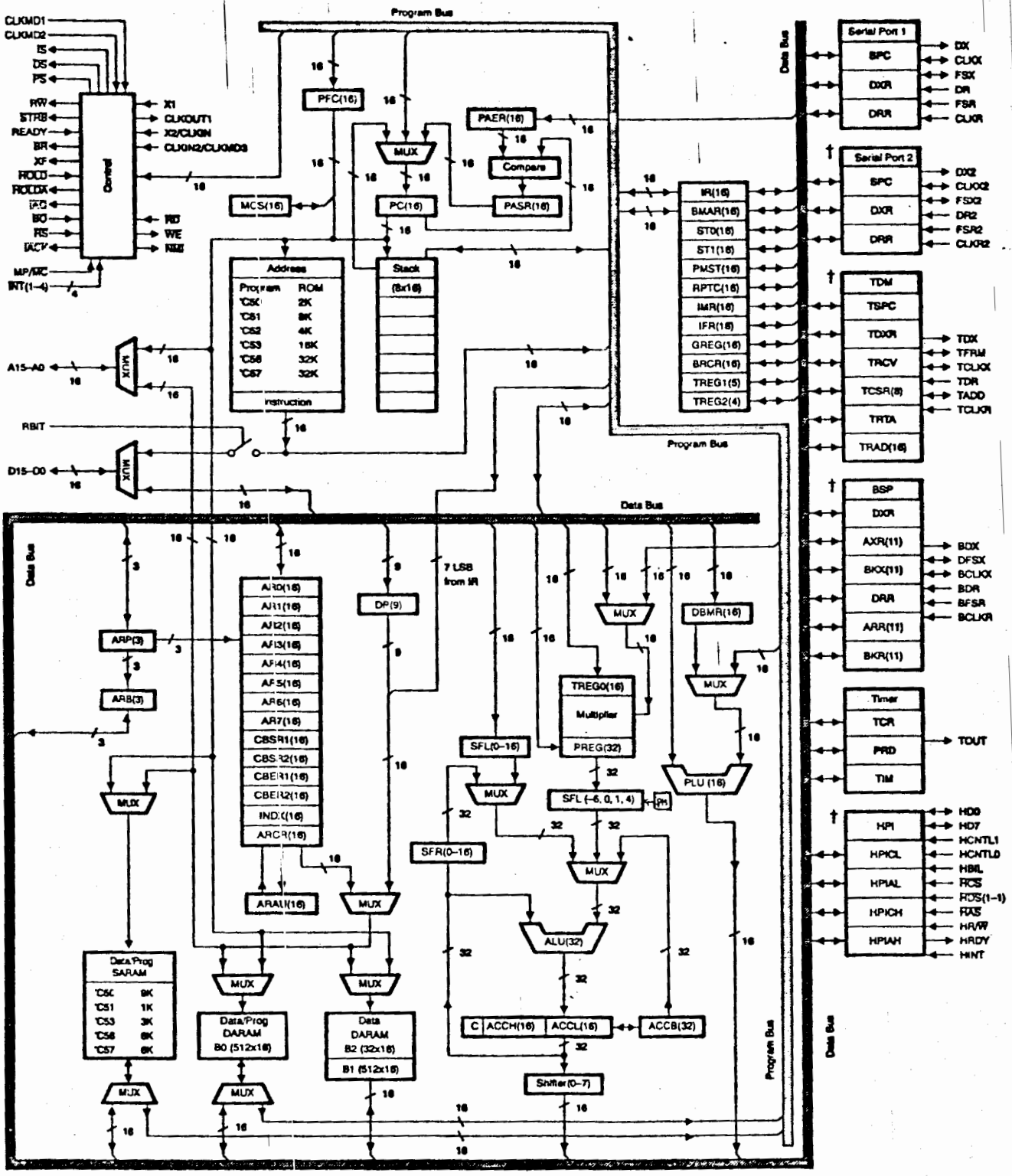
ARQUITECTURA

El TMS50 consiste de tres bloques básicos:

- Unidad Central de Proceso (CPU)
- Memoria
- Interfaz a circuitos periféricos.

Su arquitectura utiliza dos buses separados, datos y programa, con instrucciones que aceptan transferencia de datos entres ambos espacios. El TMS50 efectúa aritmética de complemento a dos, utilizando la ALU y el acumulador de 32 bits. Contiene una Unidad Lógica Paralela (PLU), que ejecuta operaciones lógicas sobre la memoria de datos sin afectar el contenido del acumulador.

Su **Multiplicador** efectúa multiplicaciones de 16 x 16 bits en complemento a 2 con resultado de 32 bits en un simple ciclo de instrucción. El multiplicador consiste de tres elementos: un arreglo multiplicador, un registro producto (PREG almacena el producto de 32 bits) y un registro temporal (TREG0 que almacena un multiplicando).



Arquitectura del TMS320C50

Existen dos bloque de **corrimientos**, uno a la entrada de los datos de 16 bits y otro a la salida de datos de la ALU. A la entrada de los datos puede producir corrimientos de 0 a 16 bits a la izquierda, según se programe en la instrucción o definido en el registro contador de corrimiento TREG1.

Contienen una **Pila** de 8 niveles de profundidad, que salva el contenido del apuntador de programa (PC) durante las interrupciones o llamados de subrutinas. En las interrupciones los registros estratégicos (ACC, ACCB, ARCR, INDX, PMST, PREG, STO, ST1 y TREGs) son guardados en los registros shadow y recuperados en el retorno de interrupción.

El TMS50 posee alto grado de paralelismo, ya que mientras se están operando datos en la Unidad Central Aritmético Lógica (CALU), se pueden estar ejecutando operaciones aritméticas en la Unidad Aritmética de Registros Auxiliares (ARAU), estas operaciones simultáneas se efectúan en un ciclo de instrucción.

REGISTROS

El TMS50 consta de 28 registros mapeados en la parte baja de la memoria dato, en las localidades 04h a 5Fh, estos se listan a continuación:

Registro	Dirección (hex.)	Descripción
-	0-3	Reservado
IMR	4	Registro de mascara de interrupción.
GREG	5	Reg. para localización global de memoria.
IFR	6	Reg. de banderas de Interrupción.
PMST	7	Reg. de modo estado del procesador.
RPTC	8	Reg. contador de repetición.
BRCR	9	Contador de repetición de bloque.
PASR	A	Dirección inicial de programa para bloque de repetición.
PAER	B	Dirección final de programa para bloque de repetición
TREG0	C	Reg. temporal para multiplicando.
TREG1	D	Reg. temporal para contador de corrimiento dinámico.
TREG2	E	Reg. temporal usado como apuntador de bit en prueba dinámica de bit.
DBMR	F	Manipulación dinámica de bit.
AR0-AR7	10-17	Registros auxiliares.
INDX	18	Indice.
ARCR	19	Reg. Auxiliar de comparación.
CBSR1	1A	Dirección inicial de buffer circular 1.
CBER1	1B	Dirección final de buffer circular 1.
CBSR2	1C	Dirección inicial de buffer circular 2.

CBER2	1D	Dirección final de buffer circular 2.
CBCR	1E	Registro de control de buffer circular.
BMAR	1F	Registro de direccionamiento dinámico.
	20-35h	Periféricos mapeados en memoria.
	36-4Fh	Reservado.
	50-5F	Puertos mapeados en memoria PA0-PA15.

Estos registros pueden cargarse al acumulador en direccionamiento indirecto o directo a través de la instrucción LAMM (carga el acumulador con registro mapeado en memoria) sin necesidad de hacer cambio de página. Para cambiar el contenido de estos registros se utiliza el acumulador salvándolo con la instrucción SAMM (salva el acumulador en registro mapeado).

Ejemplo:

```
 LACL #1Ah      ; ACC = 1Ah, carga el ACC con la constante 1Ah.
SAMM BRCR       ; Carga el ACC en el registro contador de bloque.
LAMM RPTC       ; ACC = RPTC, carga el ACC con el contenido del
                ; registro RPTC.
```

DESCRIPCION GENERAL DEL TMS320C50

La destacada funcionalidad del TMS320C50 (TMS50 o C50) para el PDS, se debe a su tipo de arquitectura Harvard que maximiza el procesamiento mediante el establecimiento de dos estructuras de buses de memoria separadas (memoria programa y memoria datos) para incrementar la velocidad de ejecución, contando con instrucciones para realizar transferencia de datos entre ambos espacios. Externamente, las memorias de programa y datos son multiplexadas sobre el mismo bus externo para maximizar el rango de direccionamiento para dichos espacios mientras se minimizan los pines del dispositivo.

La flexibilidad en el diseño del sistema es incrementada al contar con tres bloques de datos internos en memoria RAM de doble acceso DRAM (un total de 1056-palabras), donde uno de ellos puede ser configurado como memoria programa o memoria dato y 9 k-palabra de simple acceso (SRAM), además el TMS50 es capaz de direccionar externamente 64 k-palabras en un espacio de memoria dato, para facilitar la implementación de algoritmos para DSP. La memoria interna ROM de 2 k-palabras puede ser usada para reducir el costo de sistemas. Los programas en memoria ROM interna pueden ejecutarse a alta velocidad desde este espacio de memoria. Externamente el espacio de memoria de programa direccionable es de 64 k-palabras.

El TMS50 funciona con una aritmética en modo dos complemento, empleando una ALU y un Acumulador de 32 bits. La ALU es una unidad aritmética de propósito general que opera con palabras de 16 bits provenientes de la RAM de datos o derivadas de instrucciones inmediatas o por el empleo del registro producto (PR) del multiplicador de 32 bits. La ALU puede efectuar operaciones booleanas. El acumulador almacena los resultados de la ALU y a su vez es una segunda entrada a la ALU. La longitud total del Acumulador es de 32 bits, la cual es dividida en parte alta (bits 31 al 16 ACCH) y parte baja (bits 15 al 0 ACCL), y para el manejo de datos se tiene instrucciones de almacenamiento para cada una de las partes (alta y baja) del acumulador.

El multiplicador realiza operaciones de 16x16 bits en modo dos complemento con un resultado de 32 bits en un solo ciclo de instrucción. El multiplicador está compuesto de tres elementos: el registro TREG0 (de 16 bits) para almacenar temporalmente el multiplicado, el registro P (de 32 bits) para almacenar el producto y un arreglo multiplicador. Los valores del multiplicador provienen de la memoria dato, memoria programa (cuando se emplean las instrucciones MAC/MACD), o son derivados de la instrucción MPY #constante (multiplicación inmediata). La rapidez del multiplicador integrado permite la ejecución de operaciones fundamentales en el DSP como la convolución, correlación y filtrado.

El registro de corrimiento de entrada está conectado al bus de datos y su salida de 32 bits está conectada a la ALU. Este registro proporciona corrimientos a la izquierda de 0 a 16 bits sobre el datos de entrada y son programados mediante las instrucciones. Adicionalmente, se tiene la capacidad de que el procesador funcione con escalamiento numérico, extracción de bits, aritmética extendida y prevención de sobreflujo.

La interfaz local de la memoria del TMS consiste de un bus de datos paralelo de 16 bits (D15-D0), un bus de direcciones de 16 bits (A15-A0), tres pines para selección de memoria dato/programa o espacio de entrada/salida (/DS,/PS e /IS), y varias señales de control del sistema. La señal R/w controla el sentido de transferencia del dato, y la señal /STRB provee un tiempo válido para la transferencia de información, además contiene las señales de salida /RD y /WE que se pueden conectar directamente a memorias RAM con pines de entrada /OE y /WE. Cuando emplea las memorias ROM, RAM internas o memoria de programa externa de alta velocidad, el TMS50 ejecuta instrucciones a la máxima velocidad sin tiempos de espera. Para direccionar memorias externas lentas emplea de la señal READY para generar tiempos de espera suficientes para la transferencia de información.

El stack (o pila por hardware de ocho niveles) es empleado para almacenar el contenido de contador de programa durante interrupciones y llamadas de subrutinas. Las instrucciones PUSH y POP permiten salvar y recuperar el acumulador parte baja (ACCL) contenida en el stack. Las interrupciones empleadas en el dispositivo son mascarables.

Las operaciones de control son proporcionadas en el TMS50 por: un timer interno de 16 bits mapeado en memoria, un contador de repetición, cuatro interrupciones externas mascarables, y una interrupción interna generada por el puerto serie o el timer.

Un puerto serie interno full-duplex provee comunicación directa con dispositivos seriales como codecs, convertidores seriales A/D, y otros dispositivos seriales. Los registros del puerto serial mapeados en la memoria (registros de transmisión/recepción de dato) pueden operar en modo byte (ocho bits) o modo word (16 bits). Cada registro tiene una entrada de reloj externa, una entrada de sincronía y registros de corrimiento. Contiene un puerto serial (TDM, time división multiplexed) para aplicaciones de múltiples procesadores.

El TMS50 para aplicaciones de múltiples procesadores tiene la capacidad de distribuir el espacio global de memoria de dato y de comunicación con este espacio mediante las señales /BR (requerimiento de bus) y READY. El registro de distribución de la memoria global de dato (GREG) de ocho bits especifica hasta 32K-Palabras de memoria dato como memoria global externa. El contenido del registro determina el tamaño del espacio global de memoria. Si la instrucción actual direcciona un operando dentro de este espacio, /BR es insertado para requerir el control del bus. La extensión del ciclo de lectura/escritura de memoria es controlado con la línea READY.

El procesador TMS50 soporta Acceso Directo a Memoria (DMA) para su memoria externa dato/programa empleando las señales /HOLD y /HOLDA. Otro procesador puede tomar por completo el control de la memoria externa del TMS por medio de la señal /HOLD (activa baja), esta señal provoca que el TMS mantenga en estado de alta impedancia sus líneas de direccionamiento, datos y control. Sin embargo, de manera concurrente al modo DMA el TMS50 continuará ejecutando su programa si éste opera con la RAM y ROM internas.

La arquitectura del TMS50 está construida alrededor de dos buses: el de programa y el de datos.

El bus de programa proporciona el código de instrucción y operandos inmediatos de la memoria de programa. El bus de datos interconecta varios elementos, como lo son la Unidad Aritmética Lógica Central (CALU) y los registros auxiliares a los datos RAM.
Tanto el bus de programa como el de datos, pueden transferir datos de memoria dato RAM interna y de la memoria programa interna o externa al multiplicador en un ciclo de instrucción para operaciones de multiplicación/acumulación.

El TMS50 tiene un alto grado de paralelismo, es decir, que mientras está operando sobre la CALU, puede implementar operaciones aritméticas en la Unidad Aritmética de Registros Auxiliares (ARAU). Tal paralelismo resulta en un poderoso conjunto de operaciones aritméticas, lógicas y manipulación de bits que se ejecutan en un solo ciclo de máquina.

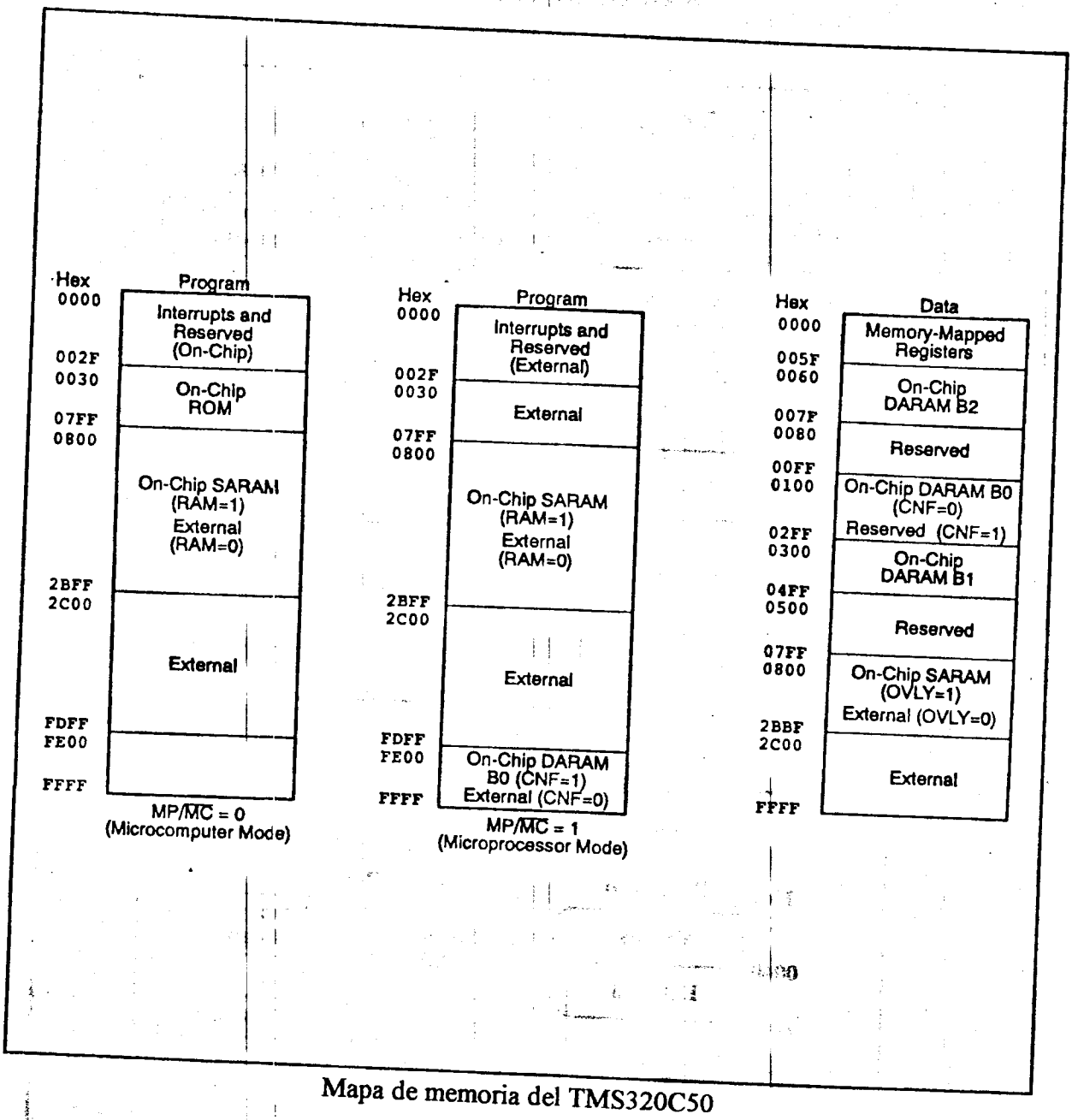
ORGANIZACIÓN DE LA MEMORIA DEL TMS50

El TMS50 posee un total de 1056-Palabras de 16 bit de DARAM interna, de las cuales 544-Palabras son siempre memoria dato y las 512 restantes pueden ser configuradas como memoria programa o dato. También provee 2k-Palabras ROM.

Las 1056 Palabras en DARAM interna se dividen en tres bloques separados: B0, B1 y B2. El bloque B0 de 512-Palabras es configurado como memoria programa (por instrucción SETC CNF) o dato (por medio de instrucciones CLRC CNF). Como memoria dato, B0 reside en las páginas 2 a 5 del mapa de memoria de datos (0100h-02FFh), y como memoria de programa de localidades FDFFh a FFFFh. Las 544 Palabras (bloques B1 y B2) siempre son configuradas como memoria dato. B1 está localizado en las páginas 6 a 9 (300h-4FFh) abarcando 512 localidades mientras que B2 está en las treinta y dos localidades superiores de la página 0 (60h-7Fh), cada página consta de 128 palabras, es decir que el mapa completo tiene 512 páginas.

Las 4k-palabras de ROM interna pueden ser un programa mascarable. Esta área de memoria permite la ejecución de programas a máxima velocidad sin necesidad de memorias externas veloces para almacenar el programa. El mapeo de estas 4k palabras es hecho por medio del pin de selección MP/mc (Microprocesador/microcomputadora). Manteniendo MP/mc en alto mapea este bloque de memoria como externo, y MP/mc en bajo lo mapea en ROM interna.

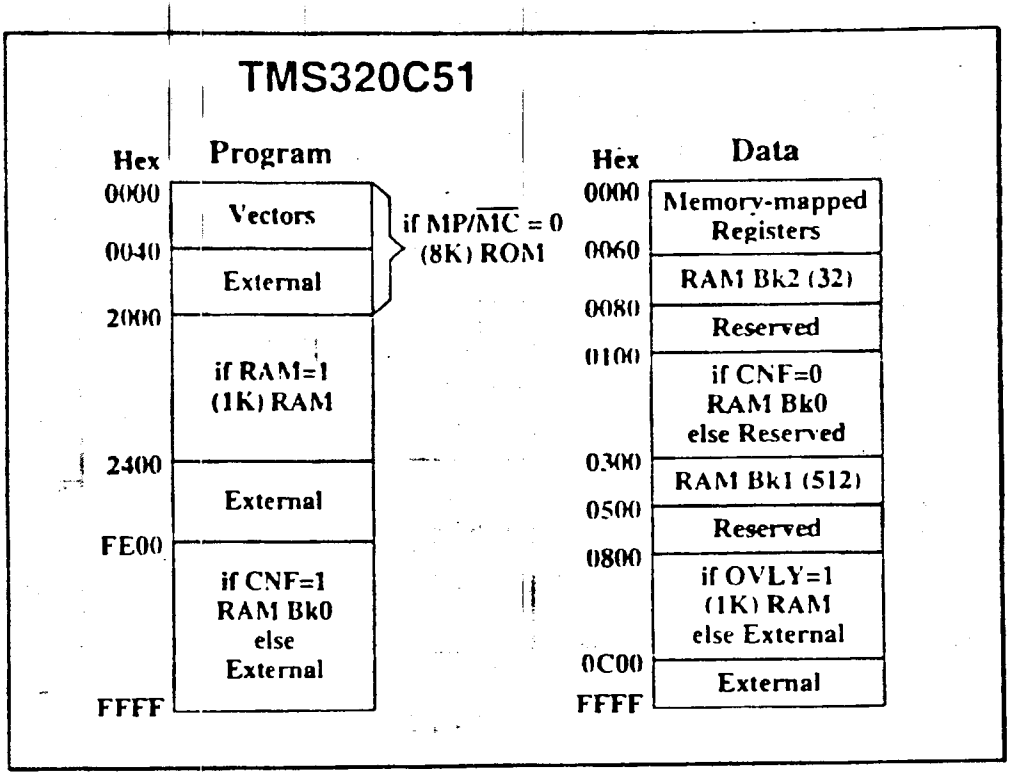
El TMS50 provee tres espacios de direccionamiento, para memoria programa, memoria dato e I/O. Estos espacios son distinguidos externamente por medio de las señales /PS (programa), /DS (dato) e /IS (puertos I/O). Las señales /PS, /DS, /IS Y /STRB son activadas sólo cuando un espacio externo de memoria está siendo direccionado.

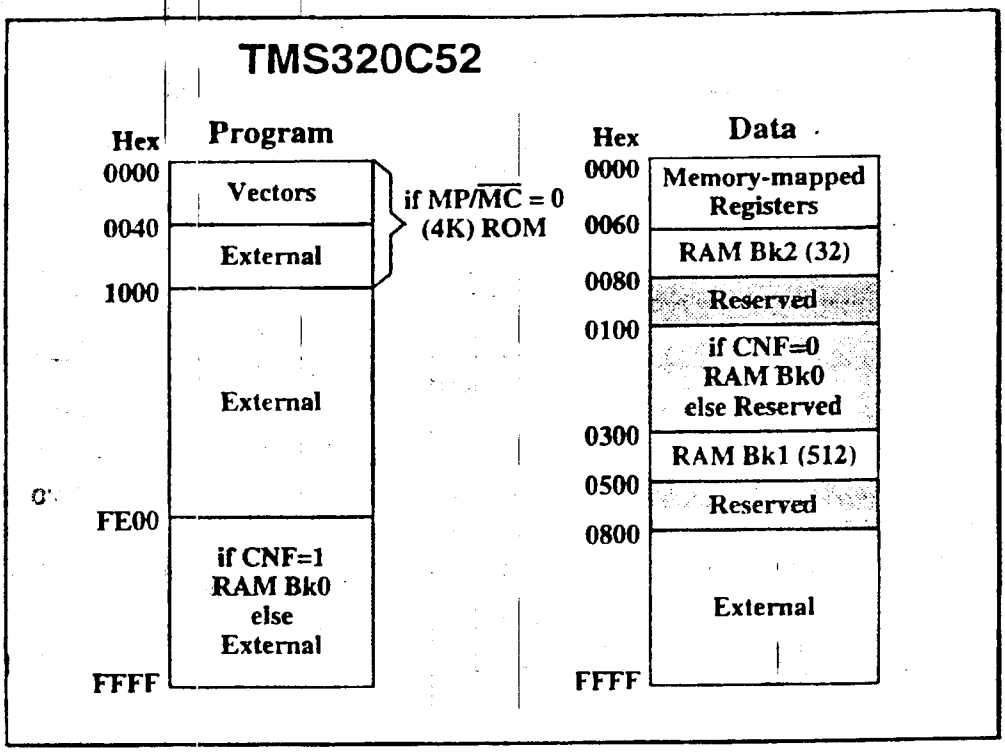


Mapa de memoria del TMS320C50

Durante un direccionamiento interno estas señales permanecen en estado inactivo alto para prevenir conflictos de direccionamiento cuando el B0 es configurado como memoria programa. La señal R/w determina el sentido del flujo de los datos, es decir, lectura (alto) y escritura (bajo).

MAPA DE MEMORIA DEL TMS320C51 y TMS320C52





MODOS DE DIRECCIONAMIENTO DE MEMORIA

Por: Larry Escobar Salguero.

Facultad de Ingeniería

UNAM

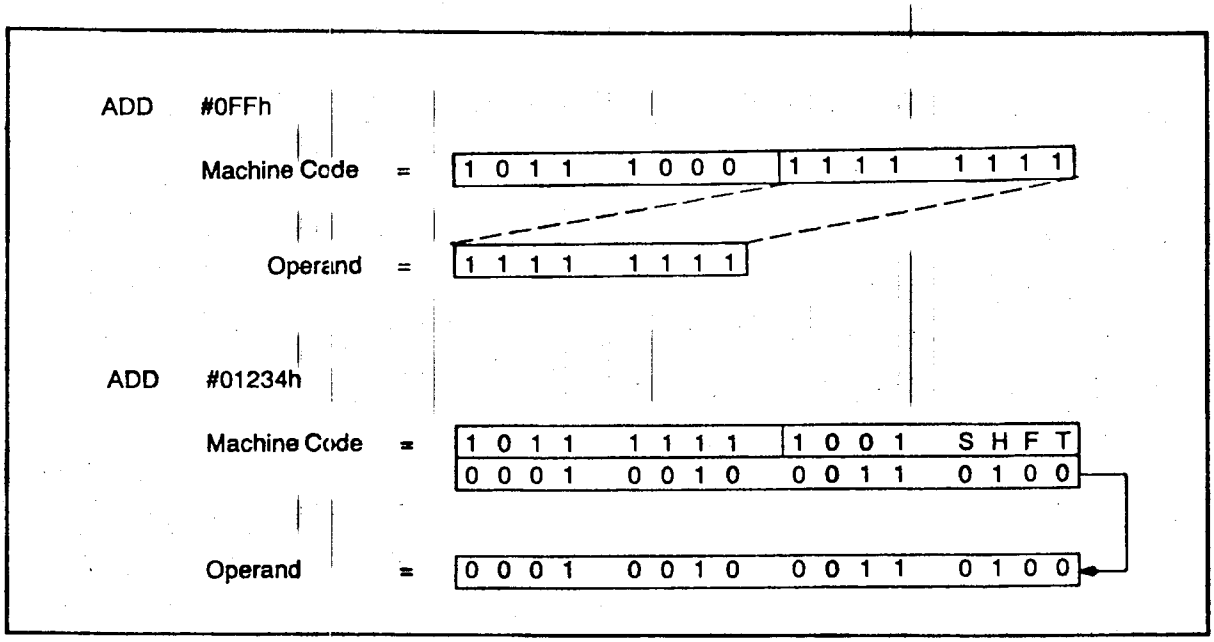
Junio del 2000

MODOS DE DIRECCIONAMIENTO DE MEMORIA

El TMS50 puede direccionar un total de 64k-palabras de memoria de programa y 64k-palabras de memoria de datos. Los 16 bits del bus de direcciones (DAB) direccionan la memoria de datos de dos formas: modo de direccionamiento directo e indirecto.

MODO INMEDIATO CORTO Y LARGO

Cuando un operando inmediato es empleado, el operando está contenido en la instrucción, el operando corto es una constante de 1 a 13 bits, en este caso el código de la instrucción es de 16 bits. Para caso de operandos inmediatos de 16 bits permite hace corrimientos a la izquierda sobre el operando de hasta 15 bits, el código de instrucción es de 32 bits, el corrimiento está en la parte baja de la primera palabra y el operando queda contenido en la segunda palabra de la instrucción, es decir que su ejecución es más lenta.



Direccionamiento inmediato corto y largo

La diferencia entre un operando corto o largo es el tamaño del operando y si existe corrimiento el operando es de tipo largo.

Ejemplos:

Operando corto

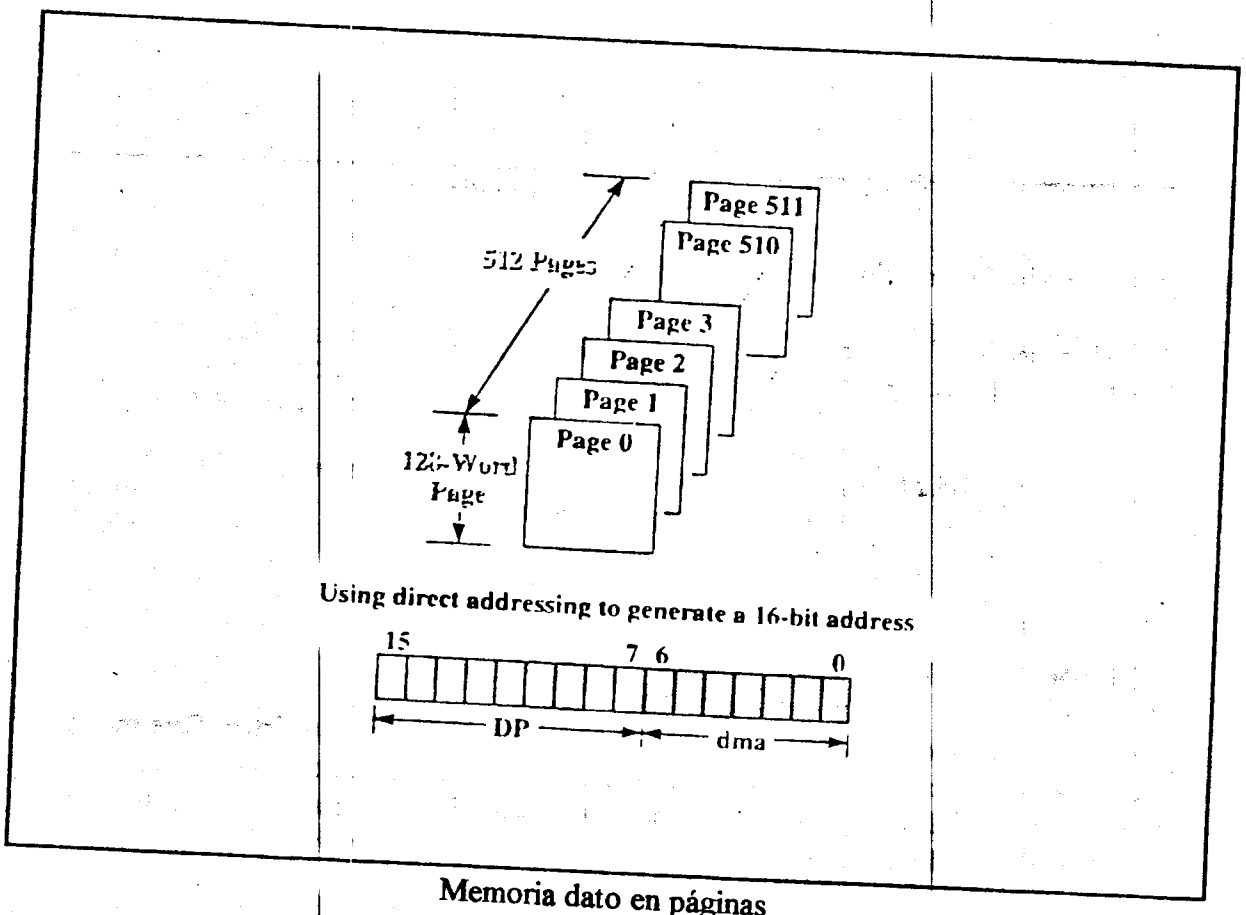
```
ADD  #0C1h ; suma al ACC una constante corta 0Ch
MPY  #03Ah ; multiplica el registro TREG0 por la constante 3Ah
```

Operando largo

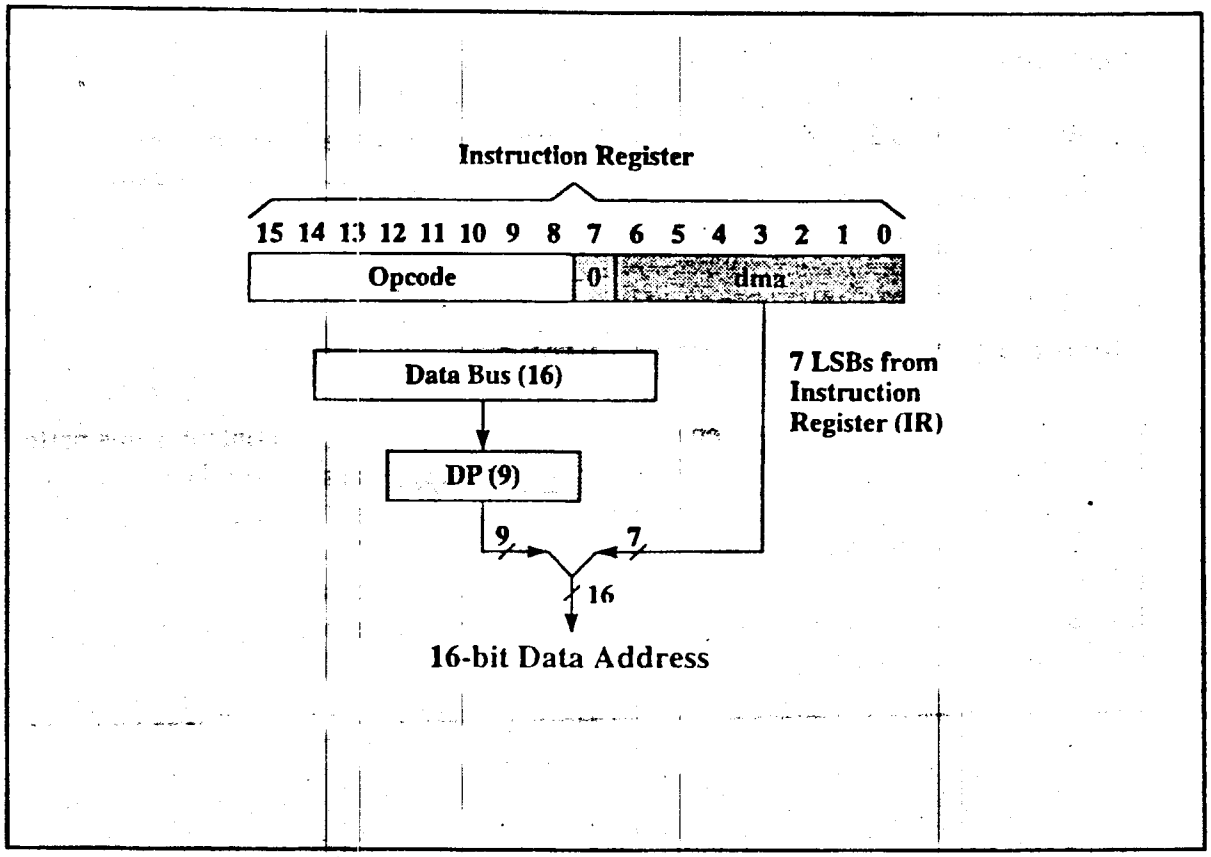
```
ADD   #111Fh,1   ;suma al ACC una constante larga con
                 ;corrimiento de un bit a la izquierda
MPY   #0124Ah    ;multiplica el registro TREG0 por la constante
                 ;124Ah
```

MODO DE DIRECCIONAMIENTO DIRECTO

A este modo se le llama directo ya que en el código de la instrucción está contenida una parte de la dirección del dato a operar. En este modo la memoria total de datos está dividida en páginas. Los 9 bits del apuntador de página DP pueden apuntar 512 páginas de 128 palabras cada una (64 k palabras). El dato de memoria direccionado es especificado por los 7 bits menos significativos de la instrucción para apuntar la palabra deseada dentro de la página (opera como una especie de offset sobre la página). Antes de usar este modo es necesario cargar el número de página en el registro DP, esto se hace con la instrucción LDP #pag.



Memoria dato en páginas



Modo de direccionamiento directo

Ejemplos de modo de direccionamiento directo:

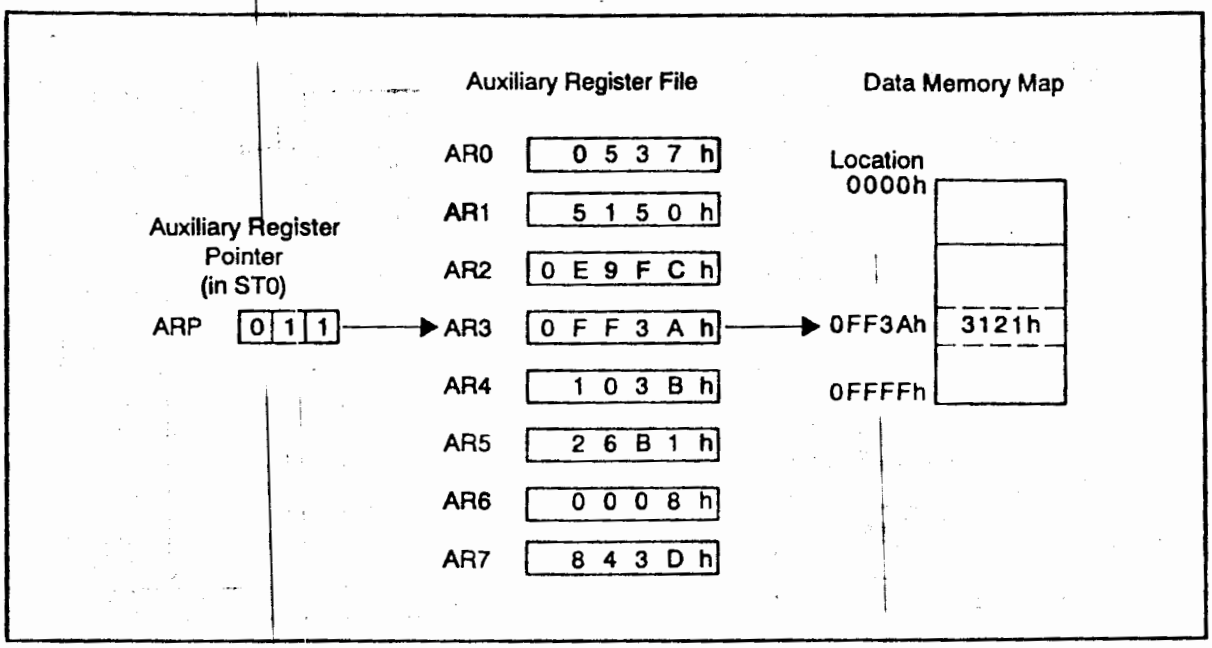
```
LACL  dir_mem        ; carga el ACCL con el dato en dir_mem, ACCH=0
LACC dir_mem,shift   ; carga un dato al ACC con corrimiento (shift)
ADD                  suma al ACC un dato en memoria con
                     ; corrimiento
SUB dir_mem,shift    ; resta del ACC un dato en memoria
                     ; con corrimiento
```

MODO DE DIRECCIONAMIENTO INDIRECTO

En este modo este modo, la dirección del dato a operar está en otro registro. Este modo es muy eficiente para el direccionamiento de tablas y arreglos, en la instrucción no se expresa ninguna dirección de memoria. Los 16 bits de la dirección son seleccionados por el registro auxiliar en uso, direccionando el dato de memoria a través del bus de registro auxiliar buffer (AFB).

Registros Auxiliares

El TMS50 posee 8 Registros Auxiliares: AR0-AR7, los cuales pueden ser utilizados para direccionamiento indirecto de memoria dato (como se observa en la figura), o almacenamiento temporal de datos. Estos registros son seleccionados por un apuntador de Registros Auxiliares de tres bits (ARP), cargándose con los valores de 0 a 7 para designar desde AR0 a AR7 respectivamente.

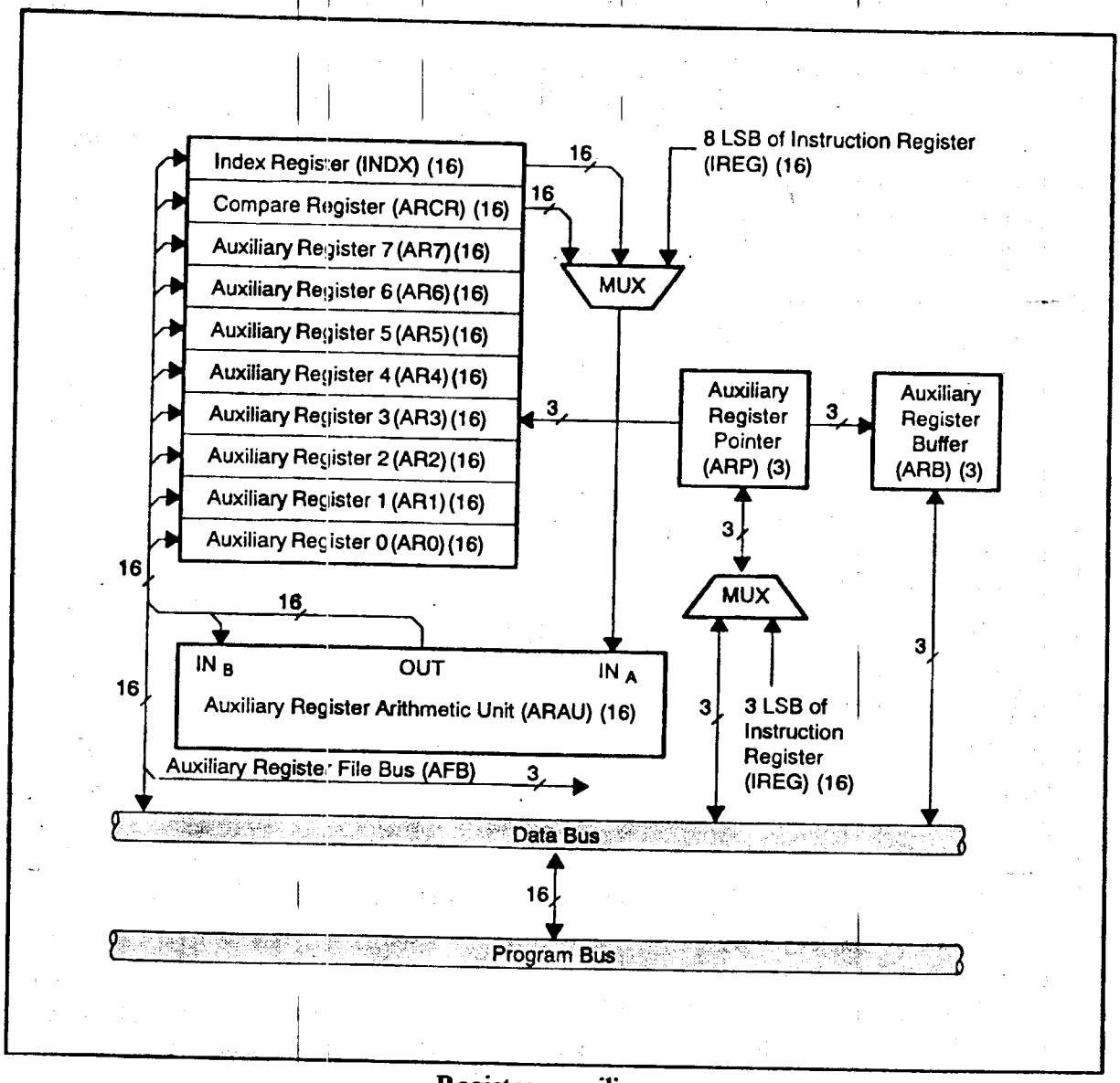


Direccionamiento indirecto

Los registros auxiliares están conectados a una Unidad Aritmética de Registros Auxiliares (ARAU). Esta unidad puede autoindexar el registro actual mientras que la localización del dato está siendo direccionada. Como resultado la CALU no accesa a tablas de información para manipuleo de direcciones, de este modo efectúa libremente otras operaciones. Los registros auxiliares deben de inicializarse y seleccionarse antes de ser utilizados, para cargarlos se utiliza la instrucción LAR y se seleccionan con la instrucción MAR.

Ejemplo:

```
LAR AR0,#100   ; Carga al registro AR0 con la dirección 100 d.
LAR AR1,#060h  ; Carga al registro AR1 con la dirección 60 h.
MAR *,AR1      ; Selecciona el registro AR1,
               ; no modifica al anterior ARi.
```



Registros auxiliares

Operaciones que efectúa la unidad aritmética de registros auxiliares ARAU operaciones

Operacion de ARAU	Comentario
AR(ARP) + INDX => AR(ARP)	al registro ARi actual en uso se le post-suma el contenido del registro INDX, ejem. ADD *0+
AR(ARP) - INDX ==> AR(ARP)	al registro ARi actual en uso se le post-resta el contenido del registro INDX, ejem. ADD *0-
AR(ARP) + 1 ==> AR(ARP)	el registro ARi actual en uso se postincrementa en uno, ejem. ADD *+
AR(ARP) - 1 ==> AR(ARP)	el registro ARi actual en uso se postdecrementa en uno, ejem. ADD *-
AR(ARP) ==> AR(ARP)	ARi sin cambios, ejem. SUB *
AR(ARP)+IR(7-0) => AR(ARP)	suma una constante de 8 bits inmediatos al ARi, ejem. ADRK #03ah
AR(ARP)-IR(7-0) => AR(ARP)	resta una constante de 8 bits inmediatos al AR(ARP), ejem. SBRK #06Bh
AR(ARP)+rc(INDX)=>AR(ARP)	le suma el registro INDX al ARi en uso con la propagación de carry inverso.
AR(ARP)-rc(INDX)=>AR(ARP)	le resta el registro INDX al ARi en uso con la propagación de carry inverso.

En la comparación del registro actual ARi con el registro ARCR, si la condición es cierta el bit TC del registro de estado ST1 se fija a uno, si es falsa se limpia. Con la instrucción CMPR #CM (donde CM puede valer 0,1,2,3) se pueden comprobar las condiciones respectivas al valor de CM:

CM	Comparación
00	(AR(ARP)) = (ARCR)
01	(AR(ARP)) < (ARCR)
10	(AR(ARP)) > (ARCR)
11	(AR(ARP)) ≠ (ARCR)

EJEMPLOS DE DIRECCIONAMIENTO INDIRECTO

1) Desarrollar un programa que realice la siguiente suma, suponer que los datos de x están almacenados a partir de la localidad "x".

$$y=\sum_{n=1}^{100} X(n)$$

```
LAR  AR1,#x   ; carga el registro AR1 con la dirección de x
MAR  *,AR1 ; selecciona al registro AR1 como registro
           ;auxiliar en uso, no modifica el anterior ARi
ZAP        ; ACC=0,    PR=0
RPT  #99   ; repite la siguiente instrucción 100 veces (n+1vez)
ADD  *+    ; suma al ACC lo que apunta AR1,  y AR1=AR1+1
SACL *     ; salva el ACCL en localidad apuntada por AR1
```

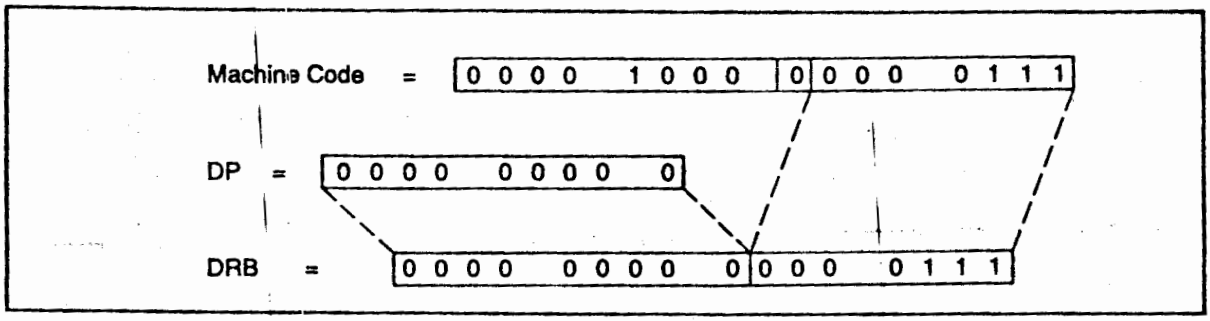
2) Desarrollar un programa que realice la siguiente suma de los valores el localidades pares de x , suponer que los datos de x están almacenados a partir de la localidad "x".

$$y=\sum_{n=0}^{49} X(2n)$$

```
LDP   #y   ; selecciona página donde está la variable y
LACL #2    ; ACC=2
SAMM  INDX ; Registro INDX = 2
LAR  AR1,#x ; carga el registro AR1 con la dirección de x
MAR  *,AR1 ; selecciona al registro AR1 como registro
           ;auxiliar en uso, no modifica el anterior ARi
ZAP        ; ACC=0,    PR=0
RPT  #49   ; repite la siguiente instrucción 100 veces (n+1vez)
ADD  *0+   ; suma al ACC lo que apunta AR1, y AR1=AR1+INDX
SACL y     ; salva el ACC en localidad de y en modo directo
```

DIRECCIONAMIENTO DE REGISTROS MAPEADOS

Este direccionamiento permite un acceso rápido a los registro mapeados en memoria. Opera similar al direccionamiento directo y se utiliza para acceder a los registros mapeados, en este caso se forza a los 9 bits más significativos de la dirección sean cero independiente del valor que tenga el registro de página, esto permite hacer un acceso directo a estos registros sin necesidad de hacer un cambio de página. La instrucciones utilizadas para este direccionamiento son LAMM y SAMM.



Direccionamiento de registros mapeados

Algunas instrucciones:

```
LAMM  dir_mem  ; el ACCL es cargado con el contenido de memoria direccionado modo directo.
LAMM  *,ARi    ; el ACCL es cargado con el contenido de memoria direccionado modo indirecto
                 ; solo toma los siete bits menos significativos del ARi
SAMM  dir_mem  ; el ACCL es copiado en el registro mapeado (modo directo)
SAMM  *,ARi    ; el ACCL es cargado en memoria direccionada en modo indirecto
LMMR reg_map,#dir_mem ; el registro mapeado es cargado con el contenido de dir_mem
                 ; (modo directo)
LMMR *,#dir_mem,ARi  ; el registo mapeado y direcionado por los 7 bits LSB del ARi en uso, es
                 ; cargado con el contenido de dir_mem
SMMR reg_map,#dir_mem  ; el registro mapeado es guardado en  dir_mem (modo directo)
LMMR *,#dir_mem,ARi ; el registo mapeado y direcionado por los 7 bits LSB del ARi en uso, es
                 ; almacenado en de dir_mem
```

DIRECCIONAMIENTO CIRCULAR

El direccionamiento circular es utilizado para direccionar eficientemente una ventana de datos que se se están procesando repetidamente, los siguientes registros son utilizados para el direccionamiento circular:

CBSR1	1A	Dirección inicial de buffer circular 1.
CBER1	1B	Dirección final de buffer circular 1.
CBSR2	1C	Dirección inicial de buffer circular 2.
CBER2	1D	Dirección final de buffer circular 2.
CBCR	1E	Registro de control de buffer circular.

El registro CBCR es de 8 bits que se definen:

Bit	Nombre	Función
0-2	CAR1	Identifica que registro auxiliar es utilizado para el buffer circular 1.
3	CENB1	Habilita buffer circular 1 con un uno, con cero lo deshabilita.
4-6	CAR2	Identifica que registro auxiliar es utilizado para el buffer circular 2.
7	CENB1	Habilita buffer circular 2 con un uno, con cero lo deshabilita.

Antes de utilizar el direccionamiento circular hay que cargar los registros de inicio y final de buffer y escribir al registro de control CBCR. Esto registro se pueden cargar con las instrucciones:

```
LACC  #0100h
SAMM CBSR1
LACC  #010Ah
SAMM CBER1
```

Estos registros también se pueden cargar con la instrucción SPLK que significa carga en paralelo una constante larga inmediata (ver unidad PLU), ejemplo:

```
SPLK  #100h,CBSR1
SPLK  #10Ah,CBER1
```

El control del buffer circular lo efectúa la unidad ARAU haciendo la comparación:

```
Si (AR(ARP))=(CBER)     entonces AR(ARP)=CBSR
                        de lo contrario AR(ARP) = AR(ARP) + PASO
```

significa, que si el buffer circular ha terminado el registro índice en uso se recarga con la dirección inicial del direccionamiento circular.

Aunque la unidad ARAU es útil para la manipulación de direccionamiento en paralelo con otras operaciones, ésta puede servir como una unidad aritmética adicional de propósito general ya que los registros auxiliares pueden comunicarse directamente con la memoria de datos. La unidad ARAU implementa aritmética no signada de 16 bits en comparación a la unidad central aritmético lógica (CALU) que implementa aritmética en modo dos complemento de 32 bits.

MODO DE DIRECCIONAMIENTO A REGISTROS

Este es un tipo de direccionamiento especial que opera con la unidad lógica paralela (PLU), la cual efectúa operaciones lógicas directamente sobre cualquier localidad de memoria dato sin afectar el contenido del ACC, permite direccionamiento directo e indirecto y operaciones con el registro de manipulación dinámica de bits (DBMR). Esto se verá cuando se estudie la unidad PLU.

MOVIMIENTO DE MEMORIA A MEMORIA

Son instrucciones que para el movimiento de datos y programa permiten utilizar eficazmente la configuración de la RAM interna. La instrucción BLDP mueve un bloque de datos de memoria dato a memoria programa y BLPD mueve un bloque de programa a memoria de datos. Para el empleo eficiente de estas instrucciones, se utilizan las instrucciones de repetición RPT y RPTZ.

La instrucción DMOV permite mover una palabra de la dirección actual en la memoria dato en RAM interna a la próxima localización alta mientras que el dato de la dirección de localización está siendo operado en el mismo ciclo por CALU. Una operación de la ARAU también puede ejecutarse en el mismo ciclo cuando se usa el modo de direccionamiento indirecto. La función de DMOV es útil en la implementación de algoritmos donde se utilizan operadores de retardo Z^{-1}, tales como convolución y filtrado digital donde el dato es pasado a través de una ventana de tiempo. La función de movimiento de dato puede ser utilizada dentro de los bloques B0, B1 y B2. Las instrucciones TBLR y TBLW (tabla de lectura y escritura) permiten transferir palabras entre el espacio de programa y de dato.

UNIDAD CENTRAL ARITMETICA LOGICA

CENTRAL (CALU)

Por: Larry Escobar Salguero.

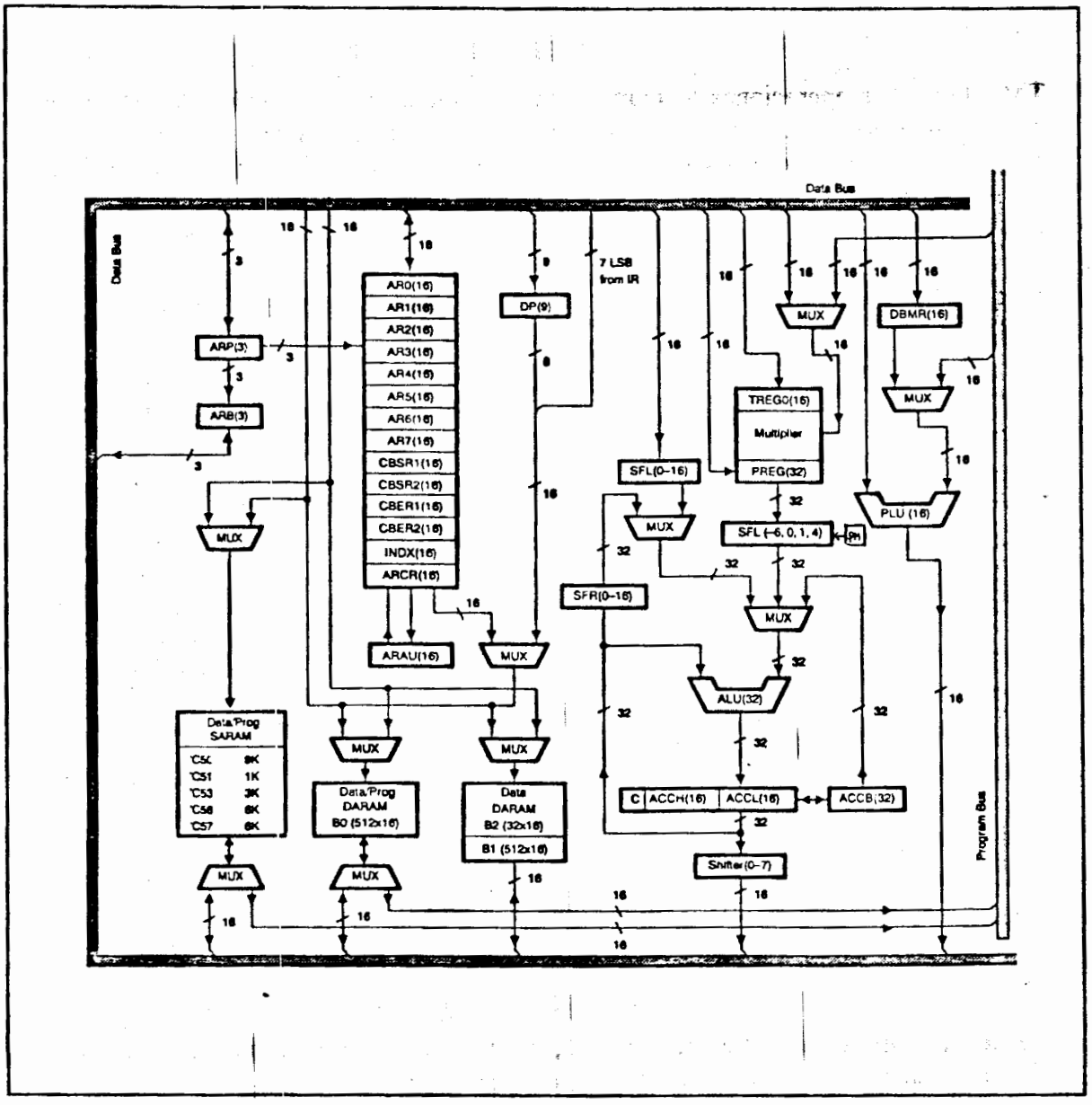
Facultad de Ingeniería

UNAM

Junio del 2000

UNIDAD CENTRAL ARITMETICA LOGICA CENTRAL

La Unidad Central Aritmética Lógica (CALU) del C50 contiene un registro de corrimiento de 16 bits, un multiplicador paralelo de 16x16 bits, una Unidad Aritmética Lógica de 32 bits, un acumulador de 32 bit (ACC) el cual está dividido en parte alta ACCH y parte baja ACCL, un acumulador buffer (ACCB) de 32 bits, un registro de corrimientos de salida para el acumulador y el multiplicador. Las instrucciones SFL y SFR realizan los corrimientos a la izquierda y derecha respectivamente del ACC.



Unidad central aritmética lógica

Pasos en la ejecución de una instrucción típica en la ALU:

1) El dato es buscado de RAM a través del bus de datos.
2) El dato es pasado a través del registro de corrimiento y en la ALU se ejecuta la operación.
3) El resultado es movido dentro del acumulador.

Una entrada a la ALU es siempre dada por el acumulador y la otra puede ser transferida del Registro Producto (PR) del multiplicador, o del registro de corrimiento que es cargado de la memoria dato o del ACCB. Después de ejecutar la operación aritmética o lógica, la ALU almacena el resultado en el acumulador.

El TMS50 soporta operaciones de punto flotante requeridas para grandes rangos dinámicos. La instrucción NORM (normalizar) es utilizada para normalizar un número signado (en puntos fijo) contenido en el acumulador por ejecución de corrimientos a la izquierda. La instrucción LACT desnormaliza un número de punto flotante por corrimiento aritmético a la izquierda de la mantisa donde el exponente está en los cuatro bits menos significativos del registro TREG1.

El modo de saturación de sobreflujo en el acumulador puede ser programado a través de la instrucciones SETC OVM (fija el modo) y CLRC OVM (quita el modo). Cuando el TMS50 está en modo de saturación y si un sobreflujo ocurre entonces la bandera de sobreflujo es fijada y el acumulador es cargado con cada uno de los números más positivos o negativos, dependiendo de la dirección de sobreflujo. (7FFFFFFFh para positivo, y 80000000h para negativo).

Se pueden implementar una variedad de instrucciones de salto a una dirección específica dependiendo del estado de la ALU y del acumulador. El acumulador tiene asociado un acarreo (carry) el cual puede ser puesto es un uno (set) o cero (reset) dependiendo de las operaciones a implementar. También existen instrucciones que permiten hacer corrimientos y rotaciones del acumulador a la izquierda o a la derecha.

Registro de Corrimiento

Puede producir corrimientos de 0 a 16 bits a un dato de entrada según sea programado en la instrucción o en el registro TREG1. Los bits menos significativos se llenan de ceros y los más significativos pueden ser llenados con ceros o signo extendido, dependiendo del estado programado en el modo de signo extendido SXM.

Unidad Aritmético Lógica

Esta unidad es la encargada de efectuar sumas, substracciones y operaciones boolenas, el resultado de la ALU siempre queda en el registro ACC. Una entrada a la ALU siempre viene del ACC y la otra entrada puede venir de el ACCB, o del registro producto o del bus de datos o programa. Para efectuar sus operaciones utiliza los modos de direccionamiento inmediato, directo e indirecto.

Los cuatro del registro TREG1 se utilizan como para indicar un corrimiento sobre un dato de entra a operarse en la ALU con las instrucciones ADDT, LACT y SUBT

Las instrucciones SFL y SFR efectúan un corrimiento del ACC un bit a la izquierda o derecha respectivamente. Si el modo signado está puesto (SXM = 1) la instrucción SFR efectúa un corrimiento aritmético a la derecha, es decir mantiene el signo del acumulador, cuando SXM = 0 la instrucción SFR efectúa un corrimiento lógico, el modo SXM no afecta a la instrucción SFL. Las instrucciones ROL (rotación a la izquierda) y ROR (rotación a la derecha) efectúa rotaciones del acumulador a través del bit de acarreo.

Las instrucciones SFLB, SFRB, RORB y ROLB pueden efectuar corrimiento y rotaciones de 65 bits al utilizar el bit de carry, el ACC y el ACCB.

El ACC puede ser corrido a la derecha en 0 a 31 bit en dos ciclos de instrucción o de 1 a 16 bits en un ciclo con la instrucción BSAR.

Cuando se obtiene un resultado de alguna operación aritmétoca o lógica éste se almacena en el ACC, si es necesario guardarlo en algún lugar de memoria se utilizan las instrucciones SACH (salva la parte alta del acumulador) y SACL (salva la parte baja del acumulador) con corrimientos de 0 a 7 bits a la izquierda, estos corrimientos se efectúan durante la transferencia de los datos sin afectar al contenido del ACC.

```
SACH VAR1,1        ; salva la parte alta del ACCH en VAR1 con corrimiento de uno.
```

Acumulador Buffer

Este registro de 32 bits acompaña al acumulador para efectuar operaciones de 32 bits, intercambio de datos y comparación de datos de una tabla, entre las instrucciones que utilizan:

```
LACB    ; carga al ACC con el valor del ACCB
SACB    ; salva el ACC en el registro ACCB
EXAR    ; intercambia el contenido del ACC con el ACCB
ADDB    ; ACC  = ACC + ACCB
ANDB    ; ACC  = ACC and ACCB
SBB     ; ACC = ACC - ACCB
ORB     ; ACC = ACC or ACCB
CRLT    ; ACC = ACCB con el valor menor que contenían ambos.
CRGT    ; ACC = ACCB con el valor mayor que contenían ambos.
XORB    ; ACC = ACC (XOR) ACCB
```

Estas instrucciones no utilizan operandos y se ejecutan en un ciclo de instrucción.

Multiplicador y Registros T y P

El TMS50 utiliza un hardware capaz de efectuar multiplicaciones de 16x16 bits en un ciclo de máquina. Todas las instrucciones de multiplicación exceptuando MPYU (multiplicación no signada) efectúan operaciones de multiplicación signada en el multiplicador. Es decir, que dos números pueden ser multiplicados siendo tratados como números en modo dos complemento y el resultado es un número de 32 bits complementado a dos. Los registros asociados a la multiplicación son:

- TREG0 registro temporal de 16 bits, que mantiene uno de los operandos a ser multiplicado.
- PR registro producto de 32 bits, que mantiene el resultado del producto.

La salida del registro producto puede ser corrida a la izquierda 1 ó 4 bits, esto es útil para la implementación de aritmética fraccionaria o justificación de productos fraccionales. La salida PR también puede ser corrida a la derecha en 6 bits habilitando la posibilidad de ejecución de 128 multiplicaciones/acumulación sucesivas sin sobreflujo. El registro TREG0 normalmente es es cargado con la instrucción LT, la cual le provee uno de los operandos (del bus de datos), y la instrucción MPY (multiplicación) provee el segundo operando también del bus de datos. Una multiplicación también puede ser ejecutada con un operando inmediato (de 13 bits) utilizando la instrucción MPY. En cada uno de los dos casos el producto es obtenido cada dos ciclos.

Las instrucciones de multiplicación/acumulación (MAC y MACD) utilizan totalmente el ancho de banda de cálculo de la multiplicación permitiendo que dos operandos sean procesados simultáneamente.

Las instrucciones SQRA (eleva al cuadrado y acumula un producto previo) y SQRS (eleva al cuadrado y resta del ACC el producto previo) pasan el mismo valor a ambas entradas de la 1multiplicación para elevar al cuadrado un valor de la memoria dato.

La instrucción MPYU permite multiplicaciones no signadas, las cuales facilitan grandemente la precisión de operaciones aritméticas.
Algunas instrucciones para efectuar operaciones de multiplicación:

```
LT  x       ; Carga en modo directo el registro TREG0 con el contenido del dato x
LT  *,AR2   ; Carga el contenido de memoria apuntada por el reg. AR en uso al registro TREG0
              y cambia al registro auxiliar AR2
LTA x       ; Carga el registro TREG0 con el dato x, y acumula el producto previo
              ACC = ACC + PR
LTD x       ; Carga el registro TREG0 con el dato x, acumula el producto previo y mueve el dato
              x a la siguiente localidad de memoria. El movimiento de dato sólo es válido en
              memoria RAM interna.
```

LTP x ; Carga el registro TREG0 con el dato x, y almacena el producto previo al ACC.
LTS x ; Carga el registro TREG0 con el dato x, y resta del ACC el producto anterior.
MAC 0FF00h,x ; Multiplica el contenido de FF00h en memoria programa con x, y acumula el producto previo.
MACD 0FF00h,x ; Multiplica el contenido de FF00h en memoria programa con x, y acumula el producto previo, y mueve el dato x a la localidad siguiente en memoria..
MPY x ; Multiplica el contenido de TREG0 con el contenido de x, el resultado lo deja en el registro PR. Esta instrucción se puede utilizar en direccionamiento directo, indirecto, inmediato corto y largo (sin corrimiento)
MPYA x ; Multiplica el contenido de TREG0 con el contenido de x, el resultado lo deja en el registro PR, y el producto previo es sumado al acumulador. Esta instrucción se puede utilizar en direccionamiento directo e indirecto.
MPYS x ; Multiplica el contenido de TREG0 con el contenido de x, el resultado lo deja en el registro PR, y el producto previo es restado al acumulador. Esta instrucción se puede utilizar en direccionamiento directo e indirecto.
MPYU x ; Multiplica el contenido de TREG0 con el contenido no signado de x , el resultado lo deja en el registro PR, puede utilizarse en direccionamiento directo e indirecto.

En la operaciones anteriores cuando el registro PR opera con el ACC, el contenido del registro PR es corrido tal como se establece en registro de modo de corrimiento de PR. Se disponen de cuatro modos de corrimiento a través del Registro Producto (PR), los cuales son útiles cuando se ejecutan operaciones de multiplicación/acumulación, aritmética fraccional o justificación fraccional de productos. PM ocupa dos bits en el registro estado ST1 especificando el modo del producto (PM) como se indica:

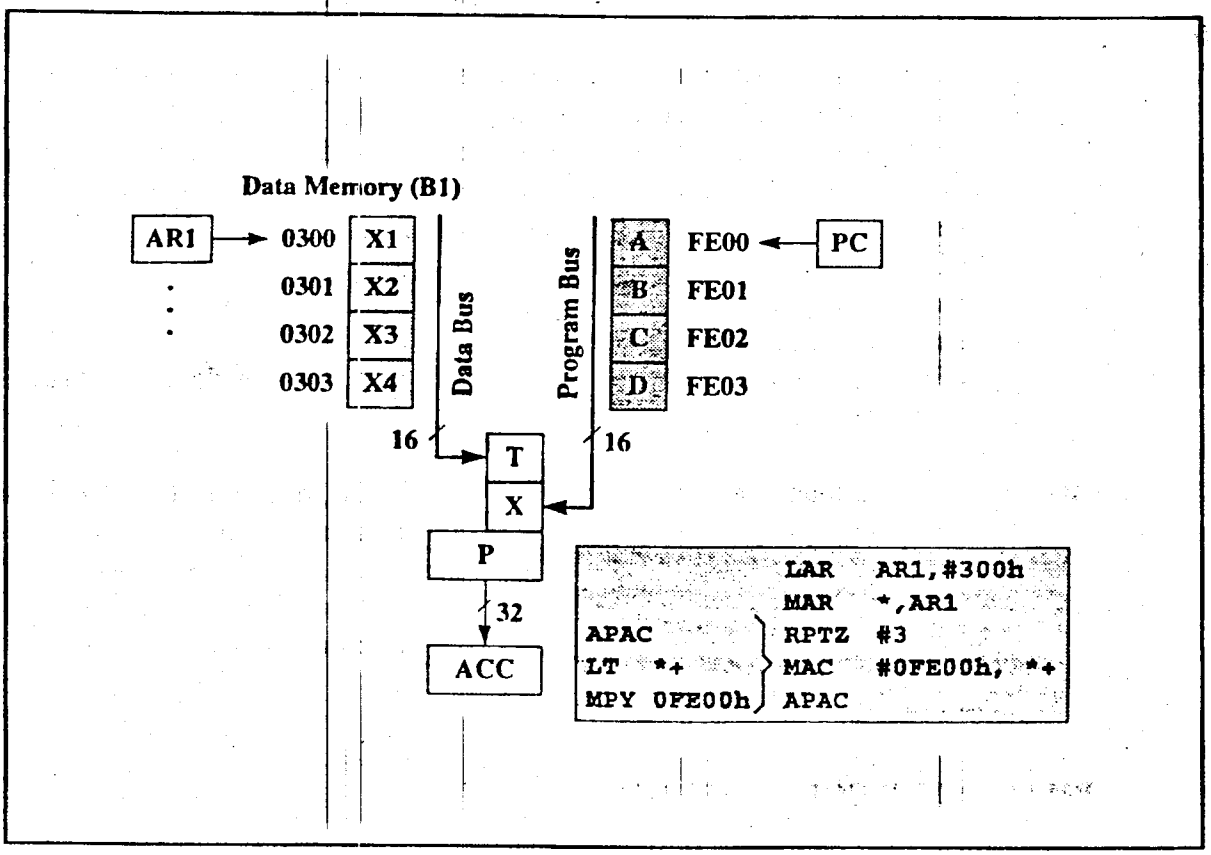
PM MODO DE CORRIMIENTO

PM	RESULTADO
00	No hay corrimiento
01	Corrimiento a la izquierda de 1 bit
10	Corrimiento a la izquierda de 4 bit
11	Corrimiento a la derecha de 6 bit

El resultado de una multiplicación (en el registro PR) puede transferirse al acumulador de varias formas por diferentes instrucciones (sin operandos):

PAC ; El contenido del registro PR corrido como se especifica en PM es cargado al ACC.
APAC ; El contenido del registro PR corrido como se especifica en PM es sumado al ACC.
SPAC ; El contenido del registro PR corrido como se especifica en PM es restado al ACC.

La unidad CALU del TMS50 alcanza su máximo desempeño cuando las instrucciones de multiplicación acumulación (MAC) o multiplicación acumulación y movimiento de datos (MACD) son utilizadas en conjunto con las instrucciones de repetición (RPT Y RPTZ) para efectuar operaciones de convolución.



Suma de productos con instrucción MAC

SISTEMA DE CONTROL

Por: Larry Escobar Salguero.

Facultad de Ingeniería

UNAM

Junio del 2000

SISTEMA DE CONTROL

El sistema de control del TMS50 está dado por el contador de programa (PC), el stack, la señal de reset externo, las interrupciones, los registros de estado y el contador de repetición (RPTC).

El Contador de Programa y el Stack

El contador de programa consta de 16 bits, lo que permite direccionar hasta 64K direcciones de memoria programa externas o internas en la búsqueda de las instrucciones. El stack consta de 8 localidades de 16 bits para almacenar el PC durante las interrupciones o las transferencias a subrutinas.

El PC direcciona la memoria programa vía el Bus de Dirección de Programa (PAB). A través del PAB una instrucción es buscada de la memoria programa y cargada en el Registro de Instrucción (IR). Cuando el IR es cargado, el PC está listo para iniciar un nuevo ciclo de búsqueda. El PC puede direccionar la RAM interna del bloque B0 cuando éste ha sido configurado como memoria de programa. El PC también direcciona la memoria de programa cuando está en memoria externa a través del bus externo de direcciones A15-A0 y el bus externo de datos D15-D0. Al inicio del nuevo ciclo de búsqueda, el PC es cargado con PC+1 o con una dirección de salto. En el caso de no tomar el salto el PC es incrementado una vez más de la localización de la instrucción de salto.

El TMS50 permite la ejecución de GOTO computados al efectuar saltos en la dirección del acumulador con la instrucción BACC, o saltos al subrutina con la instrucción CALA a una dirección dada por el ACC.

EL TMS50 permite el movimiento de datos en memoria dato a través de la instrucción BLDD, entre memoria dato y programa por medio de la instrucción BLDP y de programa a datos por medio de la instrucción BLPD. Además también se pueden utilizar las instrucciones de movimiento de tablas TBLR y TBLW.

El TMS50 permite efectuar saltos condicionales con la instrucción BCND donde puede probar hasta 4 condiciones o llamadas condicional a subrutina CC, una por cad grupo de las siguientes posibilidades :

Grupo 1	Grupo 2	Grupo 3	Grupo 4
EQ ⟹ ACC = 0 NEQ ⟹ ACC ≠ 0 LT ⟹ ACC < 0 LEQ ⟹ ACC ≤ 0 GT ⟹ ACC > 0 GEQ ⟹ ACC ≥ 0	OV ⟹ OV = 1 NOV ⟹ OV = 0	C ⟹ C = 1 NC ⟹ C = 0	BIO ⟹ /BIO = 0 TC ⟹ TC = 1 NTC ⟹ TC = 0

UNC ==> Sin condición

Para que el salto sea válido se deben de cumplir todas las condiciones de la instrucción.
Ejemplo:

```
BCND  BANDERA,LT,OV  ; Salta a BANDERA si ACC < 0  y si existe overflow.
CC    SUB1,EQ        ; Salta a la subrutina SUB1 si el ACC = 0
```

Hay que notar que no todas la combinaciones de condiciones son válidas, por lo que debe de probarse sólo una de cada grupo si es necesario, ya que sólo es necesario llenar un campo de las condiciones.

Con la instrucción RPT #N el TMS50 puede ejecutar en el próximo ciclo N+1 veces la siguiente instrucción, donde la constante N es cargada en el contador de repeticiones RPTC (la constante N puede ser de 8 o 16 bits, la instrucción RPT también puede operar en modo directo e indirecto).

El stack es accesado a través de las instrucciones PUSH y POP. Las instrucciones adicionales de PSHD y POPD permiten utilizar el stack para el almacenamiento temporal de la memoria de datos.

OPERACION DE PIPELINE A CUATRO NIVELES

El TMS50 contiene cuatro niveles de Pipeline (búsqueda, decodificación, operando y ejecución), donde un contador de búsqueda contiene la dirección de la próxima instrucción a ser prebuscada. Una vez que una instrucción es buscada, entonces es cargada en un Registro de Instrucción (IR), al menos que IR contenga la instrucción en ejecución.

El PC contiene la dirección de la próxima instrucción a ser ejecutada, sin ser usada directamente en las operaciones de búsqueda, pero sirve como un apuntador de referencia para la posición actual dentro del programa. El PC es incrementado cada vez que una instrucción se ejecuta. Cuando ocurre una interrupción o una llamada de subrutina, el contenido del PC es guardado en el stack para preservar su contenido al regreso de una interrupción o subrutina.

La prebúsqueda, decodificación y ejecución de la operación de pipeline son independientes, permitiendo así la ejecución de operaciones solapadas. Durante algún ciclo, tres instrucciones diferentes pueden ser activadas cada una a diferente estado de completez.

Cuando existen estados de espera, sólo retardan directamente la operación de pipeline. Cada estado de espera insertado durante la operación de búsqueda contribuye a agregar ciclos adicionales de máquina en la ejecución de pipeline de la instrucción.

INSTRUCCIONES DE SALTO Y SUBRUTINAS

Salto incondicional:

B ETIQUETA ; salta incondiconalmente a la dirección de programa donde esta ETIQUETA

Salto condicional:

BCND ETIQUETA, Condición1,Condición2,Condición3,Condición4 ;
; salta a ETIQUETA si se cumplen las cuatro dondiciones.

Salto computado o GOTO computado:

BACC ; el PC salta incondicionalmente a la dirección que contiene el ACCL,
; obviamente, antes se ha cargado una dirección en el ACCL.

Salto incondicional a subrutina:

CALL SUBRUTINA ; va a ejecutar la SUBRUTINA y retorna a la siguiente instrucción.

Llamada computado a subrutina o subrutina computada:

CALA ; salta incondicionalmente a una dirección contenida en el ACCL, el programa
; retorna a la siguiente instrucción después de haber ejecutado la subrutina.

Llamada condicional a subrutina:

CC ETIQUETA ; salta a ETIQUETA si se cumplen las cuatro dondiciones, después de
; ejecutar la subrutina el programa regresa a la siguiente instrucción.

Retorno de subrutina:

RET ; esta instrucción debe estar al final de toda subrutina para el programa regrese
; a donde fue llamada la subrutina.

Ejecución condicional:

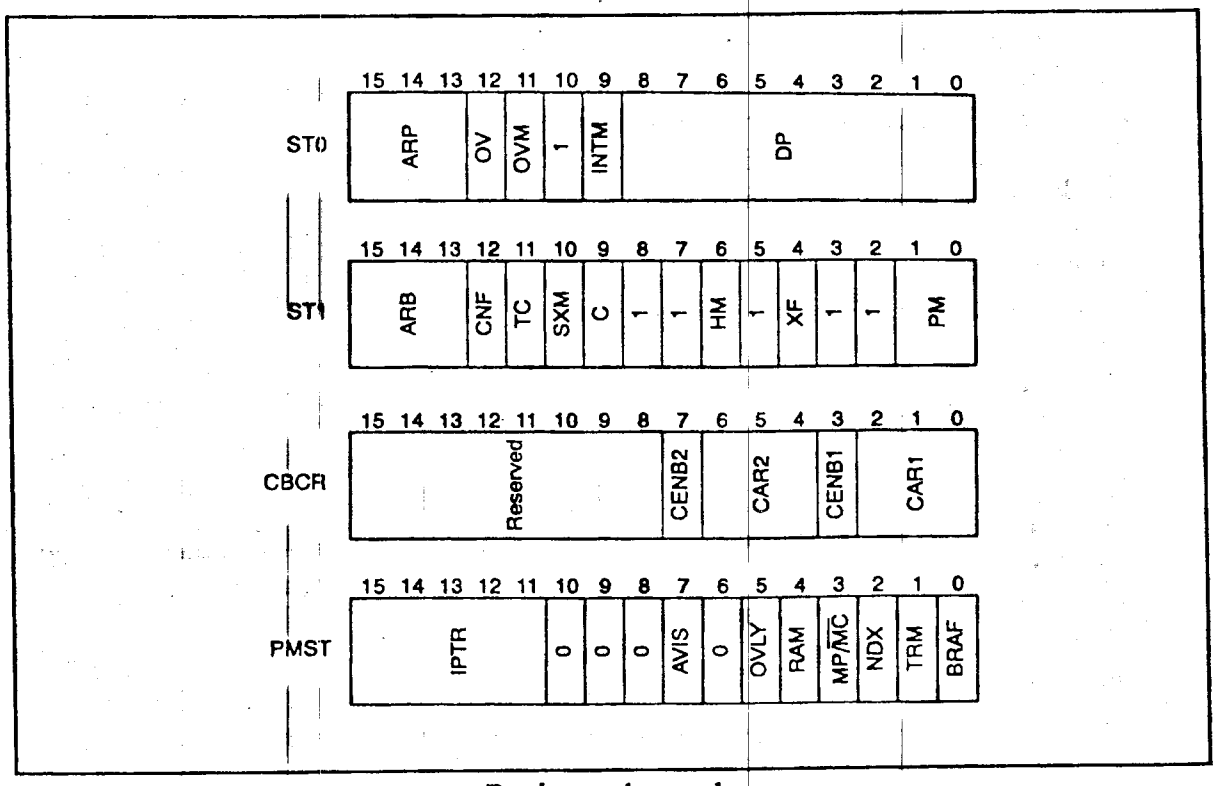
XC K,Condición1,Condición2,Condición3,Condición4 ;

Donde K=1,2 , el programa ejecuta las siguientes K instrucciones si se cumplen las condiciones, de lo contrario se salta esas instrucciones. Hay que hacer notar que si K=1, la siguiente instrucción debe ser de una palabra y si K=2 la instrucción a ejecutar puede ser de 2 palabras o 2 instrucciones de una palabra.

De las instrucciones anteriores la mayoría se pueden ejecutar con retardo para la recuperación de pipeline, sin embargo, estas posibilidades son para programación avanzada.

REGISTROS DE ESTADO

El TMS50 contiene cuatro registros de estado, los registros de estado ST0 y ST1, contiene el estado de varias condiciones y modos de operación del TMS50 y los registros PMST y CBCR contienen estados extras y control de la información.



Registros de estado

Los registros de estado pueden ser almacenados dentro de la memoria dato y cargados de memoria dato, permitiendo así que el estado de la máquina sea salvado y restaurado para interrupciones y subrutinas. Todos los bits de estado pueden ser cargados y salvados utilizando las instrucciones LST #(0-1) y SST #(0-1) respectivamente en direccionamiento directo e indirecto.

Los registros ST0, ST1 y PMST tiene asociado un registro sombra para salvarse automáticamente cuando una interrupción ocurre, en la rutina de atención de interrupción cualquiera de la instrucciones RETI o RETE le devuelve a los registros de estado sus valores originales antes de la interrupción.

Significado de los bits de registro de estado:

ARB Buffer Apuntador de Registros Auxiliares.
ARP Apuntador de Registros Auxiliares.
AVIS En uno permite la visualización de las líneas de dirección externas cuando el programa efectúa direccionamientos internos.
BRAF Bandera de repetición de bloque, en uno indica que se está repitiendo un bloque
C Bit de acarreo para una suma o préstamo para una resta.
CAR1 Tres bits que identifican que registro auxiliar AR se utiliza en el buffer circulara uno.
CAR2 Tres bits que identifican que registro auxiliar AR se utiliza en el buffer circulara dos.
CENB1 En uno habilita el buffer circulara uno.
CENB2 En uno habilita el buffer circulara dos.
CNF Bit de control de configuración de RAM interna.
DP Apuntador de página de Memoria Dato.
HM Bit de Modo Hold.
INTM Habilita interrupciones mascarables, 0 = habilitado, 1 = deshabilitado.
IPTR Apuntador de vector de interrupción. Cinco bits que apuntan a 2000 localidades donde reside el vector de interrupción. Permite el remapeo de los vectores de interrupción.
MP/mc Bit de modo microprocesador (1) /microcomputadora (0).
NDX Habilita el registro índice.
OV Bit de bandera de sobreflujo.
OVLY Habilita el acceso de programa en memoria RAM, si es uno, el bloque de memoria es mapeado en el espacio de dato, si es cero, el bloque de memoria no es direccionable en memoria dato. Fijado a cero en el reset.
OVM Bit de modo de sobreflujo.
PM Modo de corrimiento del registro producto al cargarse en el ACC.
RAM Habilita el programa en memoria RAM. Habilita la SARAM mapearse en el espacio de memoria programa. Fijado a cero en el reset.
SXM Bit de modo de signo extendido y extensión de signo.
TC Bandera de control de prueba de bit.
TRM Habilita múltiples TREGs, En uno habilita el uso de los registro TREG1 y TREG2.
XF Bit de estado del pin externo XF.

Configuración de la memoria SARAM

OVLY	RAM	Configuración de SARAM
0	0	Deshabilitada
0	1	Mapeada en el espacio de programa.
1	0	Mapeada en el espacio de datos.
1	1	Mapeada en ambos espacios, datos y programa.

Instrucciones asociadas don los registros de estado y los bits individuales:

```
SETC  BIT        ; el BIT de control especificado es puesto a 1.
CLRC  BIT        ; el BIT de control especificado es limpiado.

LST   n,dir_mem  ; el registro de estado STn es cargado en modo directo con dato en
                 ; dir_mem.
LST   n,*,ARi    ; el registro de estado STn es cargado en modo indirecto con el dato
                 ; apuntado por el ARi en uso.
SST   n,dir_mem  ; el registro de estado STn es almacenado en modo directo en dir_mem.
LST   n,*,ARi    ; el registro de estado STn es almacenado en modo indirecto en
                 ; dirección apuntada por el ARi en uso.
```

JERARQUIA DE CICLOS

En múltiples aplicaciones es necesario anidar varios ciclos, por lo que es necesario conservar cierta jerarquía para obtener un buen desempeño:

```
        LAR   ARi,#N1
ETIQ1   INSTRUCCION
        INSTRUCCION
        ...
        ...
        LACC #NB
        SAMM BRCR
             RPTB  FINB
             INSTRUCCION
             INSTRUCCION
               . . .
               RPT #N
               INSTRUCCION ; ciclo de una instrucción del  RPT
             INSTRUCCION
               .....
FINB         INSTRUCCION      ; fin del ciclo del bloque de instrucciones.
        INSTRUCCION
        INSTRUCCION
        ....
        MAR *,ARi            ; asegura al ARi para que sirva de contador
        BANZ  ETIQ1,*-;ARi   ; salta a ETIQ1 si ARi en uso es diferente de cero.
```

CONTADOR DE REPETICION (RPTC)

RPTC es un registro de 16 bits, que cuando es cargado con un número N en la instrucción RPT, causa que la siguiente instrucción sea ejecutada N+1 veces. El RPTC puede ser cargado con un número e 0 a 65,536 usando las instrucciones RPT #N o RPTZ #N. Es utilizado en las instrucciones de multiplicación/acumulación, movimiento de bloques, transferencias de I/O y tablas de lectura/escritura. La instruccción RPTZ limpia el acumulador y el registro producto (PR) antes de empezar la repetición de la siguiente instrucción. Cuando la instrucción de repetición (RPT o RPTZ) es decodificada todas las interrupciones mascarables son deshabilitadas, sin embargo el TMS5O responde a señales de HOLD mientras efectúa un ciclo de repetición. Las instrucciones de repetición se utilizan en conjunto con las instrucciónes MAC, MACD, BLDD, BLPD, TBLR y TBLW.

Ejemplo:

Desarrollar un programa que realice la siguiente suma de productos entre 50 puntos de la señal X(n) y los coeficientes h(n), suponer que los datos de X(n) están almacenados a partir de la localidad "x" y los coeficientes h(n) están almacenados a partir de la localidad "h", el resltado lo salva en y.

$$y=\sum_{n=0}^{49} X(n)\,h(n)$$

```
LDP  #y    ; carga página donde está y
LAR  AR1,#x ; carga el registro AR1 con la dirección de x
MAR  *,AR1 ; selecciona al registro AR1 como registro
           ;auxiliar en uso, no modifica el anterior ARi
ZAP        ; ACC=0,   PR=0
  RPT  #49    ; repite la siguiente instrucción 50 veces
  MAC  #h,*+ ; suma al ACC lo que apunta AR1,  y AR1=AR1+1
APAC         ; rescata el último producto
SACL y     ; salva el ACCL en localidad "y"
```

Hay que hacer notar que notar que no todas las instrucciones se pueden utilizar con las instrucciones de repetición de ciclo y algunas no tienen significado (Ver el manual de usuario pag. 3-42).

REPETICION DE BLOQUES

El TMS50 permite la implementación de ciclos DO y FOR. Para su ejecución utiliza los registros PASR, PAER, BRCR y el bit BRAF en el rgistro PMST.

El registro contador de repetición de bloque BRCR es cargado con la cuenta del ciclo desde 0 a 65,536. La instrucción de repetición de bloque RPTB es la que inicia el bloque y carga la dirección de la siguiente instrucción en el registro de inicio de bloque PASR y la dirección del última instrucción del bloque la carga en el registro PAER. Cuando se inicia la ejecución de bloque el bit BRAF se pone en uno. La dirección de PAER es comparada con el PC, si son iguales entonces el contenido de BRCR es comparado a cero, si el registro BRCR es igual a cero el ciclo termina, de lo contrario éste se decrementa y el PC es cargado con la dirección de PASR.

Ejemplo:

Desarrollar un programa que realice la misma suma que se hizo con RPT, pero ahora con RPTB

$$y=\sum_{n=0}^{49} X(n)\,h(n)$$

```
      LDP   #y    ; selecciona página donde está la variable y
      LAR  AR1,#x ; carga el registro AR1 con la dirección de x
      LAR  AR2,#h ; carga el registro AR2 con la dirección de h
      MAR  *,AR1  ; selecciona al registro AR1 como registro
                  ;auxiliar en uso, no modifica el anterior ARi
      LACC  #49   ; ACC=49
      SAMM  BRCR  ; el registro contador de bloque BRCR = 49
      ZAP         ; ACC=0,    PR=0

      RPT  FINB ; repite las instrucciones de blque 50 veces
       LT  *+,AR2 ; TREG0 = X(n)
       MPY *+,AR1 ; PR = X(n)*h(n)
FINB    APAC       ; ACC=ACC+PR
      SACL y     ; salva el ACC en localidad de y en modo directo
```

Algunas consideraciones en la repetición de bloques:

- RPTB si es interrumpible.
- El ciclo RPT puede anidarse dentro del ciclo RPTB
- El anidamiento de ciclos RPTB no es recomendado, ya que hay que estar salvando y restaurando registros ocasionando más consumo de tiempo.
- Las interrupciones y llamadas a subrutina son permitidos dentro de un bloque de repetición.
- Un bloque debe tener al menos tres instrucciones de longitud.

MODO POWER-DOWN

Este modo le permite al TMS50 entrar al modo de consumo de baja potencia. Este modo se invoca con las instrucciones IDLE e IDLE2 o por la entrada /HOLD en bajo cuando el bit de estado HM está en uno.

En el modo Power-down detiene la ejecución del TMS50 manteniendo el contenido de todos sus registros internos. Si este modo se invocó con la instrucciones IDLE o IDLE2 entonces el TMS50 sale de este modo al recibir alguna interrupción, si el bit INTM = 0 (interrupciones mascarables habilitadas) el TMS50 va a atender la rutina de interrupción y después continúa con el programa, si INTM = 1, entonces el TMS50 continúa con la intrucción seguida a IDLE o IDLE2.

UNIDAD LOGICA PARALELA

Por: Larry Escobar Salguero.

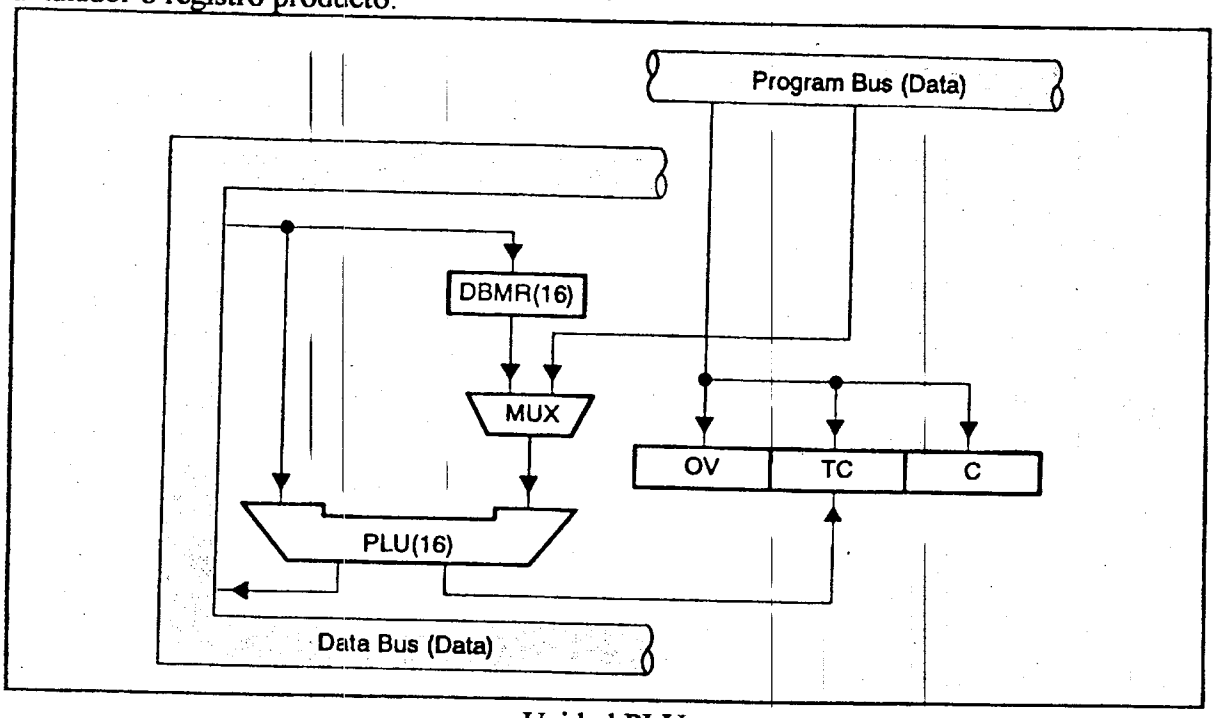
Facultad de Ingeniería

UNAM

Junio del 2000

UNIDAD LOGICA PARALELA

La unidad lógica paralela (PLU) efectúa operaciones lógicas independientemente de la unidad ALU y ARAU, es decir, que no utiliza el acumulador para efectuar las operaciones. Ésta puede fijar, limpiar o cambiar múltiples bits de los registros de control o de estado o de cualquier localidad en memoria dato. La unidad PLU efectúa operaciones lógicas directas sin afectar el contenido del acumulador o registro producto.



Unidad PLU

En las operaciones de la unidad PLU un operando es buscado en memoria dato y el otro proviene del operando inmediato en la instrucción, al efectuar la operación lógica entre los operando el resultado es escrito de nuevo en la localización de la memoria dato direccionado, normalmente todas estas operaciones se efectúan en un ciclo de instrucción

Esta unidad permite la prueba de bits individuales con la intrucción BIT, o comparación de una constante con una localidad de memoria o comparación dinámica utilizando el registro de manipulación dinámica (DBMR), si las cantidades a comparar son iguales la comparación entonces el bit de estado TC se pone a uno.

Ejemplos:

```
APL  #000Fh,x      ; efectúa operación AND entre la constante 000Fh y el dato x
OPL  #0FF0h,y      ; efectúa operación OR entre la constante 000Fh y el dato y
XPL  #01,*         ; cambia el bit 0 del dato apuntado por el reg. ARi en uso
SPLK #0ABh,dato    ; almacena la constante 0ABh en variable dato.
```

INTERRUPCIONES

Por: Larry Escobar Salguero.

Facultad de Ingeniería

UNAM

Junio del 2000

INTERRUPCIONES

Las interrupciones son un recurso para suspender la actividad del TMS50 (o un microprocesador) para atender algún requerimiento por hardware o software, esto evita la necesidad de estar encuestando por software un evento externo. Las interrupciones típicas son generadas por dispositivos externos que quieren transferir información al TMS50, tal el es el caso de convertidores A/D y D/A. El TMS50 tiene cuatro interrupciones externas mascarables (/INT4, /INT3, /INT2 e /INT1), disponibles para dispositivos externos que interrumpan al microprocesador. Dos interrupciones internas que son generadas por el puerto serie (RINT y XINT), una por el timer (TINT) y una por software a través de la instrucción TRAP. Aunque esta última no es priorizable incluye su vector de interrupción. Cada dirección de interrupcción consta de dos localidades para ubicar una instrucción de salto y la dirección de la subrutina de atención de interrupciones.

Tabla de vectores de interrupción

Las fuentes de interrupción del TMS50 son seis externas, cinco internas, una de TRAP y por software:

Interrupción	Dirección	Descripcción
RESET	0h	Reset externo
/INT1	2h	Interrupcción externa de usuario N. 1
/INT2	4h	Interrupcción externa de usuario N. 2
/INT3	6h	Interrupcción externa de usuario N. 3
TINT	8h	Interrupcción interna de timer
RINT	Ah	Interrupcción de recepción del puerto serie.
XINT	Ch	Interrupcción de transmisión del puerto serie.
TRNT	Eh	Interrupcción de recepción del puerto TDM.
TXNT	10h	Interrupcción de transmisión del puerto TDM.
/INT4	12h	Interrupcción externa de usuario N. 4
	14-21 h	Reservado.
TRAP	22h	Vector de instrucción TRAP
NMI	24h	Interrupcción externa no mascarable.

Las interrupcciones externas son activadas por una señal en nivel bajo al menos tres ciclos en el respectivo pin externo.

Las interrupciones internas son generadas por eventos internos del TMS50.

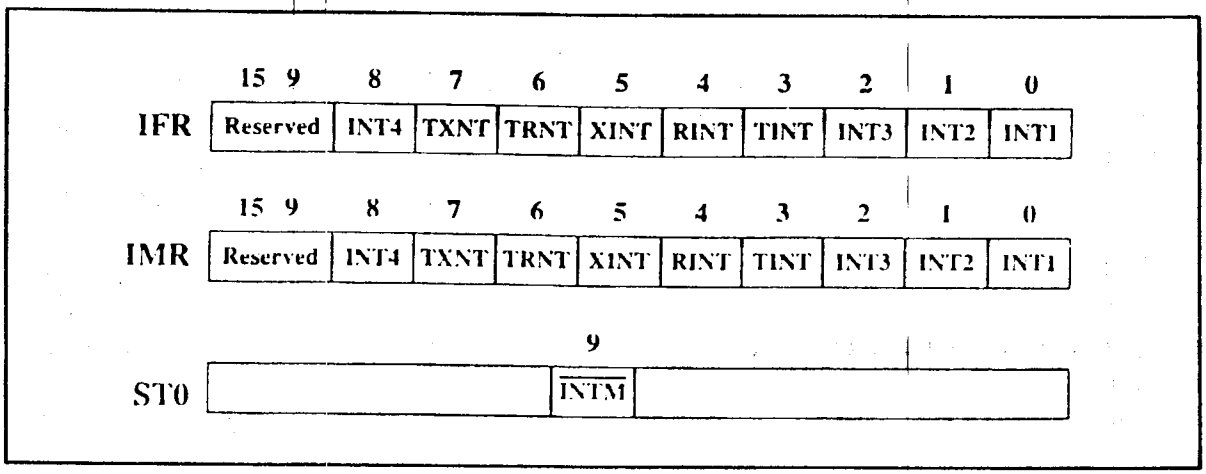
Reset (/RS)

Es una interrupción externa no mascarable, con la más alta prioridad sobre las demás interrupciones. Puede ejecutarse en cualquier instante y es aplicada típicamente en el encendido cuando los estados de máquina son aleatorios.

La señal /RS causa que el TMS50 termine la ejecución y obligue al contador de programa ir a cero afectando varios registros y los bits de estado. Para la correcta operación del sistema, la señal de reset debe ser activada baja por lo menos tres ciclos de reloj. El bus de datos es puesto en alta impedancia, y las señales de control se van a alto mientras dura el reset.

Operación de las interrupciones

Cuando una interrupción ocurre, ésta es almacenada en un Registro de Bandera de Interrupción de 16 bits (IFR). Este registro es fijado por el uso de interrupciones externas /INT(4-1) y las interrupciones RINT, XINT y TINT. Cada interrupción es almacenada en IFR hasta que ésta es reconocida y automáticamente borrada por el Reconocimiento de Interrupciones (/IACK) o el reset (/RS) o se escribe un uno en el bit correspondiente en el registro IFR. El reset no es almacenado en IFR.



Registro de banderas de interrupcion IFR e IMR

El TMS50 tiene una mapa de memoria para Registro de Interrupciones Mascarables (IMR) para enmascarar las interrupciones internas y externas. Un uno en cualquiera de los bits 0-15 del registro IMR habilita la correspondiente interrupción siempre que el bit de estado INTM esté en cero (habilita interrupciones mascarables) . Obviamente las interrupciones NMI y /RS no están incluidas en el registro IMR.

El bit /INTM en el registro de status ST0 habilita o deshabilita todas las interrupciones mascarables (INTM=0 habilitado y INTM=1 deshabilitado). INTM es fijado a uno por la señal /IAK. Si una interrupción ocurre durante un ciclo múltiple de instrucciones, la interrupción no puede ser procesada hasta que la instrucción es completada, este mecanismo de protección también es aplicado a instrucciones de ciclo múltiple debido a la señal de READY. Tampoco se permite que se procesen interrupciones cuando una instrucción está siendo repetida por medio de RPT, la interrupción es almacenada en IFR hasta que el ciclo de repetición sea terminado, después la interrupción es procesada. Las interrupciones no pueden ser procesadas entre la instrucción CLRC INTM y la próxima instrucción en la secuencia del programa. También una interrupción no es reconocida cuando el /HOLD está activado.

La instrucción RETE permite terminar una rutina de atención de interrupción, reestablecer los registros principales salvados y habilitar interrupciones (INTM = 0) .

Salvado de registros

Cuando una interrupción es ejecutada ciertos registros estratégicos son salvados automáticamente, y en el retorno de la interrupción (con instrucciones RETE y RETI) estos registros son reestablecidos. En el TMS50 existe una región llamada de registros shadow donde se almacenan los registros: ACC, ACCB, PREG, STO, ST1, PMST, TREG0, TREG1, TREG2, INDX y ARCR. Esto permite salvar el constexto de registros que está trabajando el programa al llegar una interrupción.

Implementación de Stack por software

Cuando se anidan interrupciones o subrutinas más allá de los ocho niveles, entonces se puede implementar un stack por software por medio de la instrucciones POPD y PSHD, permintiendo a la parte alta del stack (TOS) sea salvado o recuperado de memoria dato. La intrucción POPD salva el contenido del PC en TOS en memoria dato y los siete valores bajos del stack son recorridos una localidad hacia arriba, la intrucción PSHD pone un valor de la memoria dato en el TOS del stack y los demás valores en el stack son recorridos hacia bajo una localidad de memoria (el valor más bajo del stack es perdido). Esto implica que el stack puede ser expandido en memoria dato.

Interrupciones por software

La instrucción de interrupciones por software (INTR #k) permite efectuar hasta 32 (k de 0 a 31) interrupciones por software con las mismas características de una interrupción por hardware. El programador sólo invoca la rutina de atención de interrupción de acuerdo a la tabla siguiente, lo importante de esta forma de interrupción es que el usuario puede definir sus propias rutinas de antección de interrupción.

Interrupciones por software

K	Interrupción	Localización	K	Interrupción	Localización
0	RESET	0h	16	Reservado	20h
1	/INT1	2h	17	TRAP	22h
2	/INT2	4h	18	NMI	24h
3	/INT3	6h	19	Reservado	26h
4	TINT	8h	20	De usuario	28h
5	RINT	Ah	21	De usuario	2Ah
6	XINT	Ch	22	De usuario	2Ch
7	TRNT	Eh	23	De usuario	2Eh
8	TXNT	10h	24	De usuario	30h
9	/INT4	12h	25	De usuario	32h
10	Reservado	14h	26	De usuario	34h
11	Reservado	16h	27	De usuario	36h
12	Reservado	18h	28	De usuario	38h
13	Reservado	1Ah	29	De usuario	3Ah
14	Reservado	1Ch	30	De usuario	3Ch
15	Reservado	1Eh		De usuario	3Eh

PERIFERICOS Y PUERTO SERIE

Por: Larry Escobar Salguero.

Facultad de Ingeniería

UNAM

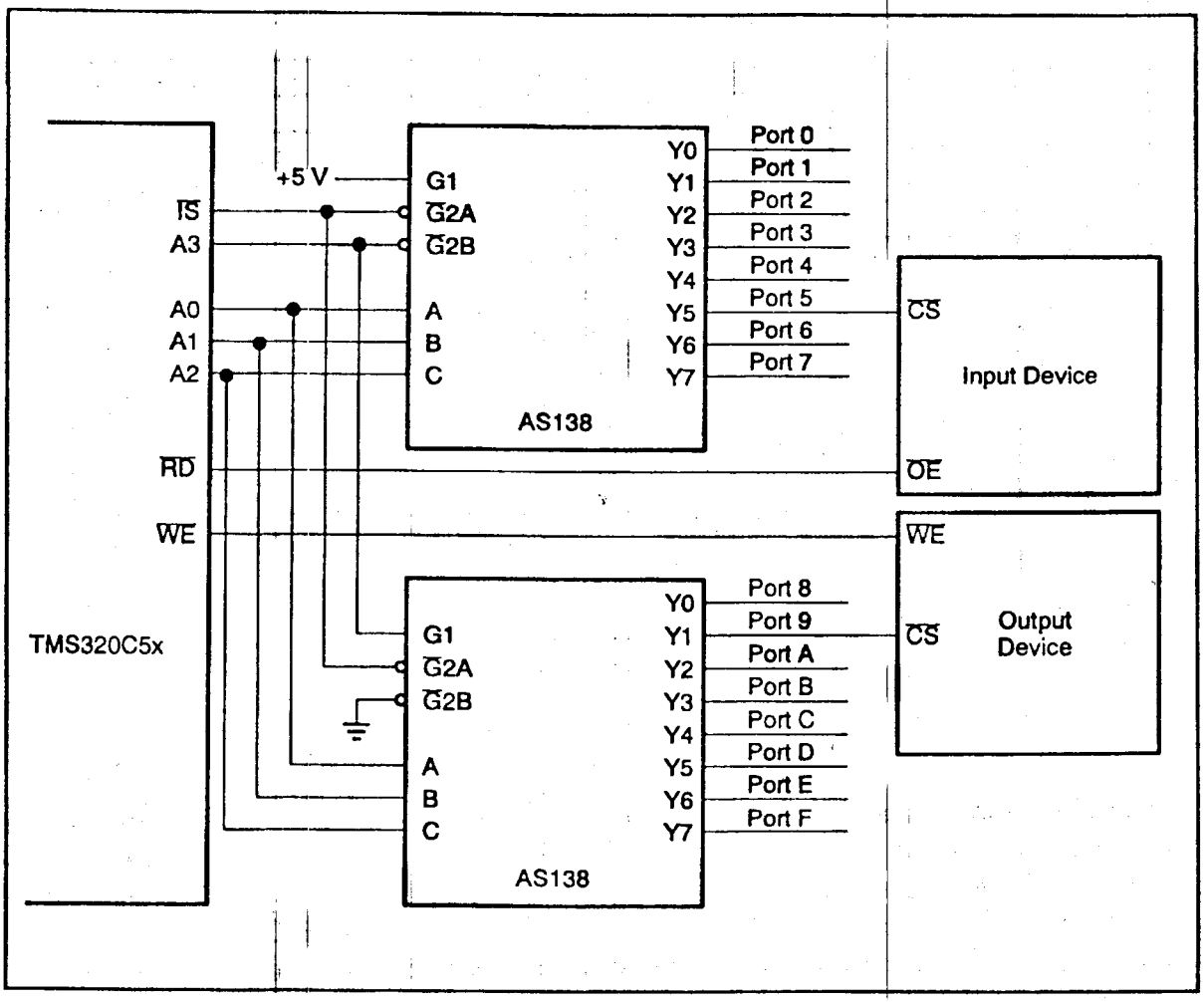
Junio del 2000

PERIFERICOS Y PUERTO SERIE

Los periféricos que maneja el TMS50 son controlados por algunos registros mapeados, para la inicialización de estos periféricos, en sus registros deben de escribirse las palabras necesarias para que operen correctamente. Dentro de estos periféricos están el puerto serie (transmisión / recepción), el puerto serie multiplezado por división (TDM), 64 k-puertos paralelos y el timer.

PUERTOS PARALELOS DE ENTRADA SALIDA

El TMS50 puede direccionar hasta 64k puertos de entrada salida (I/O), éstos son seleccionados cuando la señal externa /IS es activada en bajo cuando el TMS50 ejecuta alguna instrucción IN u OUT. La señal /RD y /WE se pueden utilizar en conjunto con /IS para accesar a dispositivos externos tal como se ve en la figura siguiente:



Interfaz a puertos paralelos I/O

GENERACION DE ESTADOS DE ESPERA POR SOFTWARE

El TMS50 es un procesador de señales muy rápido, sin embargo, brinda la posibilidad de acceder a dispositivos lentos al generar hasta 7 estados de espera por medio de la línea externa READY, pero aún más estos estados de espera se pueden generar por software evitando hardware externo.

La programación por software de los estados de espera puede ser controlado por dos registros generadores de estado de espera (PDWSR para dato y programa y IOWSR para I/O) y un registro de control de 5 bits (CWSR).

Register	Bits	Space	Address Range	
PDWSR	0–1	Program	0000h–3FFFh	
	2–3		4000h–7FFFh	
	4–5		8000h–0BFFFh	
	6–7		0C000h–0FFFFh	
	8–9	Data	0000h–3FFFh	
	10–11		4000h–7FFFh	
	12–13		8000h–0BFFFh	
	14–15		0C000h–0FFFFh	
IOWSR			BIG = 0	BIG = 1
	0–1	I/O	Port 0/1, Port 10/11, etc.	0000h–1FFFh
	2–3		Port 2/3, Port 12/13, etc.	2000h–3FFFh
	4–5		Port 4/5, Port 14/15, etc.	4000h–5FFFh
	6–7		Port 6/7, Port 16/17, etc.	6000h–7FFFh
	8–9		Port 8/9, Port 18/19, etc.	8000h–9FFFh
	10–11		Port 0A/0B, Port 1A/1B, etc.	0A000h–0BFFFh
	12–13		Port 0C/0D, Port 1C/1D, etc.	0C000h–0DFFFh
	14–15		Port 0E/0F, Port 1E/1F, etc.	0E000h–0FFFFh

Generación de estados de espera

El mapa de memoria para dato y programa se divide en 8 bloques de 16k palabras y se le asocia en el registro PDWSR dos bits a cada bloque, es decir, que a cada bloque se le puede programar su propio estado de espera independientemente de los otros bloques. El espacio para I/O puede configurarse en dos formas dependiendo como se especifique el bit BIG en el registro CWSR. Si BIG=0, se mapean ocho pares de periféricos I/O por cada dos bits en IOWSR, si BIG=1, ocho bloques de 8 K palabras y a cada bloque se le puede programar su estado de espera independiente.

Cuatro bits en el registro CWSR permiten al usuario seleccionar uno de dos mapas de estados de espera y el número de estados de espera correspondientes (de 0 a 7 estados). Los valores de los bits se escriben tanto en los registros PDWSR, IOWSR y CWSR como se especifica a continuación:

Wait-State Field† of PDWSR or IOWSR (Binary Value)	No. of Wait States (CWSR Bit n = 0)	No. of Wait States (CWSR Bit n = 1)
00	0	0
01	1	1
10	2	3
11	3	7

n (Bit Position In CWSR)	Space
0	Program
1	Data
2	I/O (lower-half: Port 0–Port 7 if BIG=0, 0000h–7FFFh if BIG=1)
3	I/O (upper-half: Port 8–Port F if BIG=0, 8000h–0FFFFh if BIG=1)
4	BIG mode bit

Registros PDWSR, IOWSR y CWSR

El registro CWSR debe escribirse antes que los registros PDWSR e IOWSR.

PUERTO SERIE

El TMS50 consta de un puerto serie bidireccional full duplex, este puerto provee una comunicación directa a dispositivos tales como codecs, convertidores seriales A/D, otros sistemas seriales y para aplicaciones multiprocesos a través del puerto serial TDM.

Las señales del puerto serie son compatibles con codecs y otros dispositivos seriales. La máxima frecuencia de operación del puerto serie cuando utiliza el reloj interno es de 5 Mbits/s a 50 ns y 7.14 Mbits/s a 35 ns.

Para su operación el TMS50 contiene los pines externos:

CLKX	Señal de reloj de transmisión.
CLKR	Señal de reloj de recepción.
DX	Señal de transmisión de dato serial, envía el dato actual
DR	Señal de recepción de dato serial, recibe el dato actual.
FSX	Señal de sincronización de frame de transmisión, inicia la transferencia de un frame.
FSR	Señal de sincronización de frame de recepción.

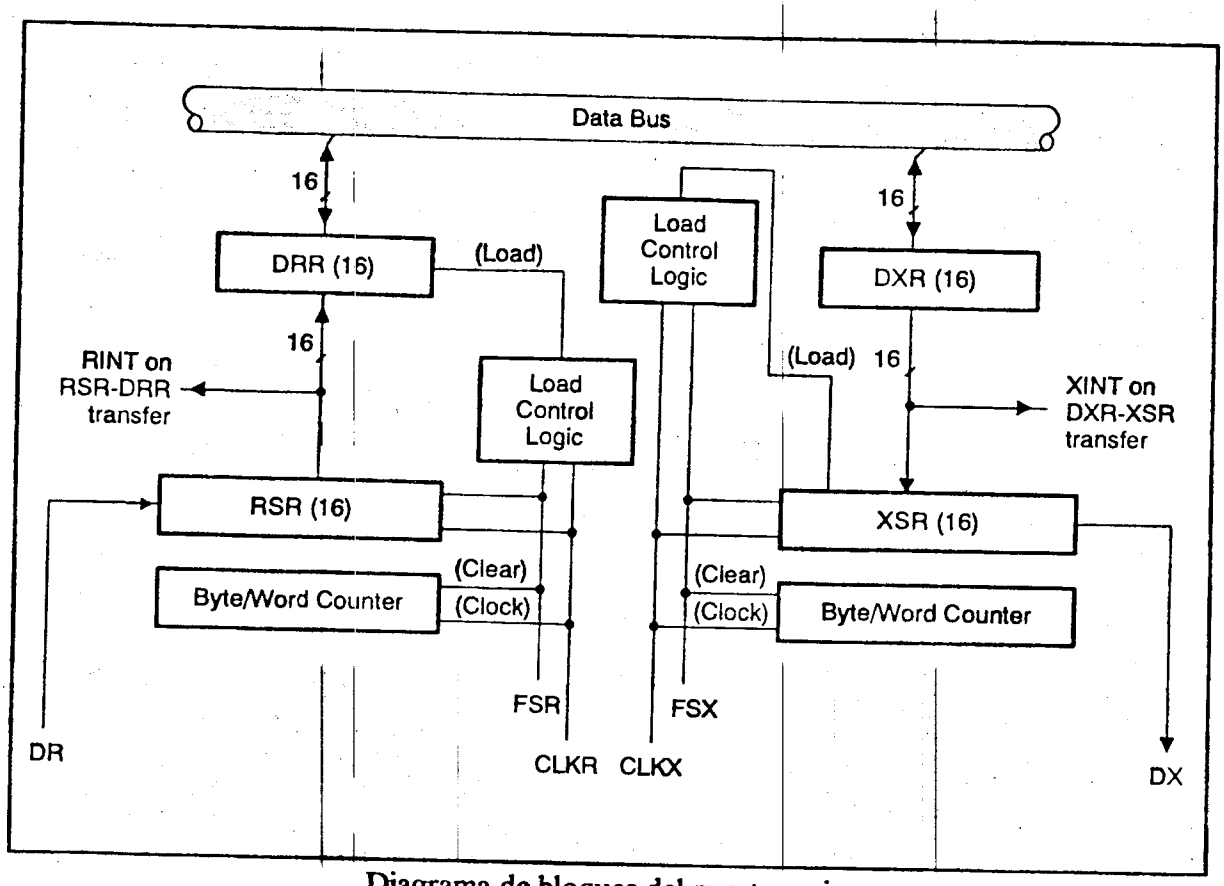


Diagrama de bloques del puerto serie

Para la operación del puerto serie se utilizan los registros:

SPC	Registro de control del puerto serie.
DXR	Registro de transmisión de datos.
DRR	Registro de recepción de datos.
XSR	Registro de corrimiento en la transmisión.
RSR	Registro de corrimiento en la recepción.

Un dato a transmitir es escrito en el registro DXR el cual copia el dato en el registro XSR que efectúa el corrimiento de bits para hacer la transmisión serial sobre el pin DX, tan pronto como la copia DXR a XSR se permite escribir otro dato en el registro DXR, a la vez ocurre una transición 0-1 en el bit de transmisión lista (XRDY bit 11) del registro de control SPC y se genera una interrupción de transmisión de puerto serie (XINT).

Para la recepción un dato recibido en el pin DR es corrido dentro del registro RSR, el cual es copiado en el registro de recepción de dato (DRR) y el dato puede ser leído. Una vez completada la copia de RSR a DRR existe una transición 0-1 en el bit de recepción lista (RRDY bit 10) del registro de control SPC y a la vez se genera una interrupción de recepción de puerto serie (RINT). El puerto serie es doblemente buffereado porque los datos puede ser transferidos de DXR o DRR mientras otra transmisión o recepción se efectúa.

El registro de control SPC es un registro mapeado y se describe a continuación:

Bit	Name	Description
0		Reserved
1	DLB	Digital Loopback
2	FO	Word length 0 = 16 bits 1 = 8 bits
3	FSM	Frame sync mode: 0 = continuous mode 1 = one sync pulse/word
4	MCM	CLKX source: 0 = CLKX 1 = CLKOUT1/4
5	TXM	0 = FSX is an input 1 = FSX is an output
6	XRST	0 = reset transmiter 1 = operate transmitter

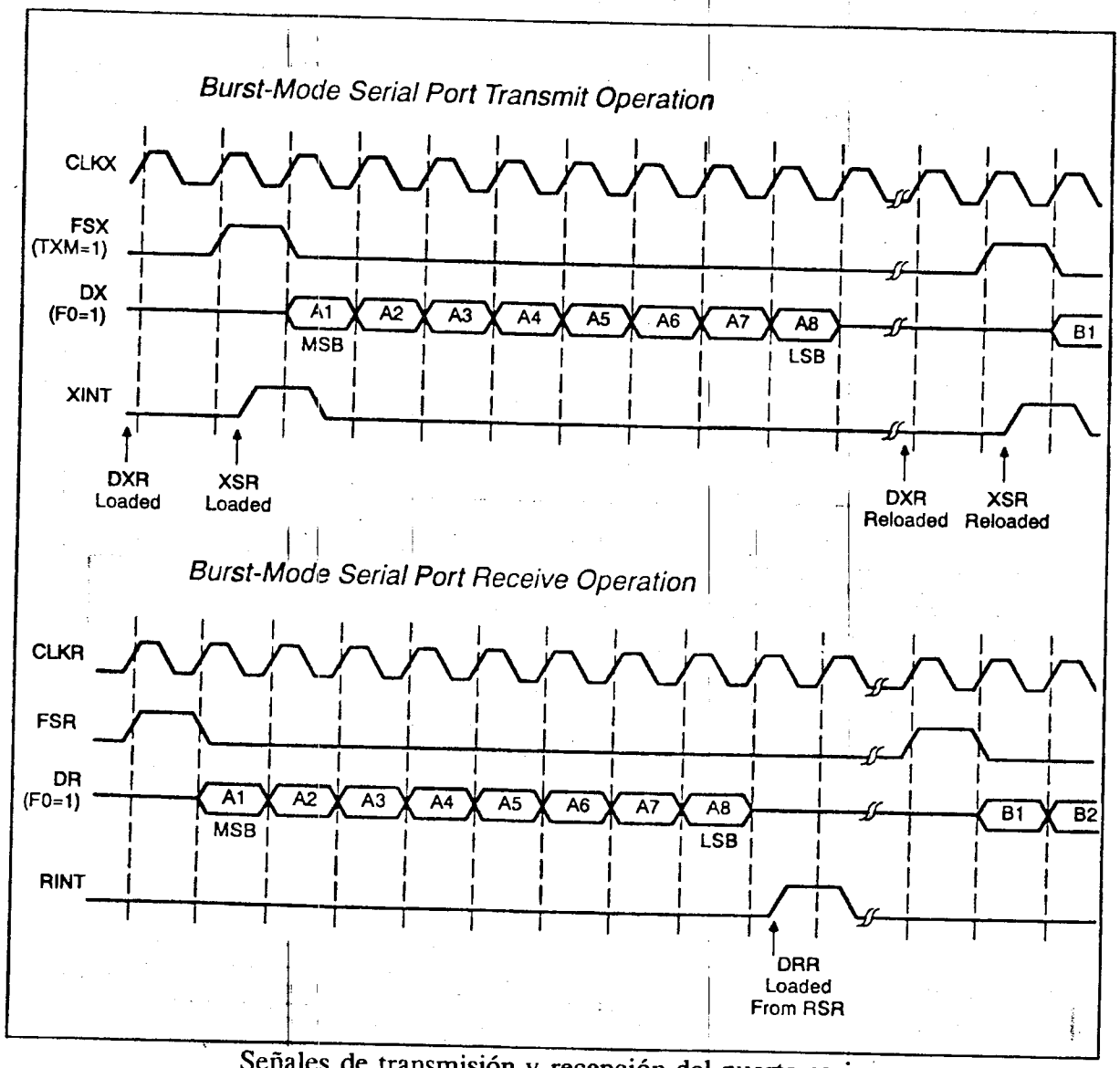
Bit	Name	Description
7	RRST	0 = reset receiver 1 = operate receiver
8	IN0	Input level at CLKR
9	IN1	Input level at CLKX
10	RRDY	1 = receive data ready
11	XRDY	1 = transmitter ready
12	XSREMPTY	0 = transmiter underrun
13	RSRFULL	1 = receiver overflow
14 15	SOFT FREE	Determine state of serial port when a breakpoint is encountered in the HLL Debugger

FREE	SOFT	Action
1	x	Free run
0	0	Immediate stop
0	1	Stop after completing word

Registro de control de puerto serie (SPC)

Para la configuración del puerto serie es necesario escribir a los bits 1-5 del registro SPC, sin embargo, antes de escribir a estos bits hay que efectuar un reset sobre los bits 6 y 7 (XRST y RRST), es decir, escribir ceros a estos bits y luego escribir a los bits 1-5, para que la configuración del puerto quede habilitada se ponen los bits 6 y 7 a unos. Los bits 10-13 del registros SPC son bits de estado del puerto serie.

En la siguiente figura se muestran los diagramas de tiempo de las señales del puerto serie, tanto para la operación de transmisión como de recepción.



Señales de transmisión y recepción del puerto serie

EL TEMPORIZADOR (TIMER)

El timer es un contador interno del TMS50 que puede utilizarse para generar interrupciones periódicas con tiempos exactos.

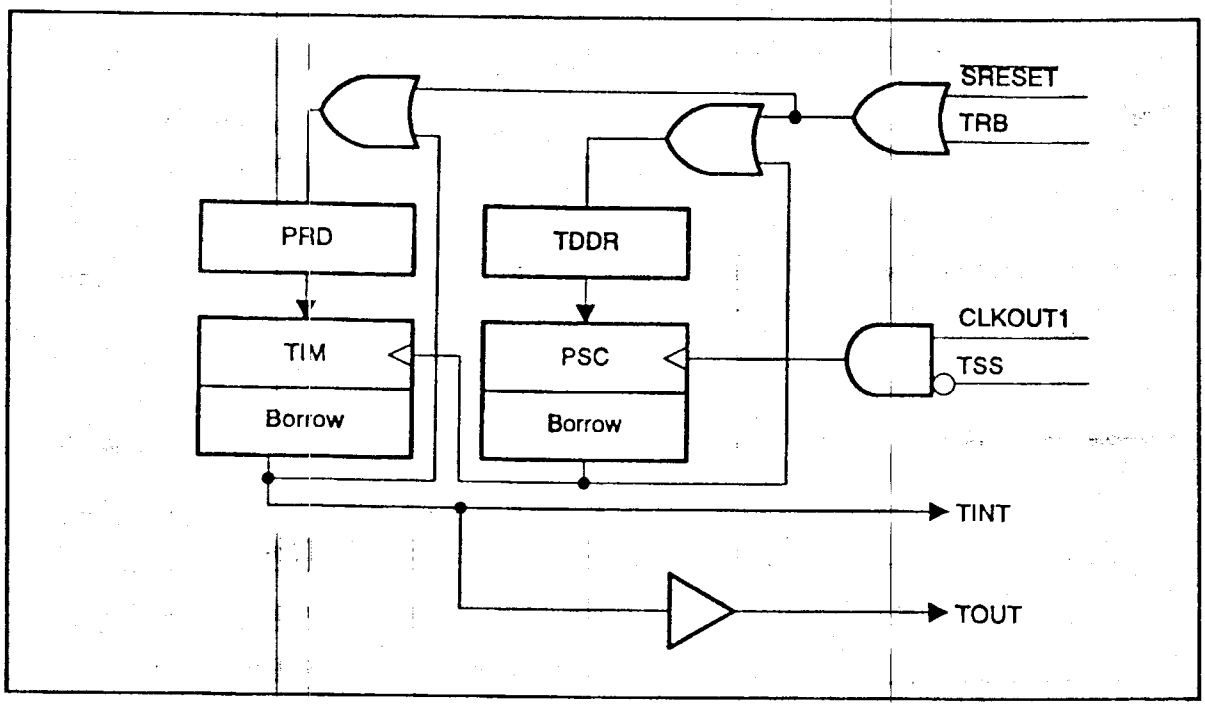


Diagrama de bloques de Timer

El TMS50 posee un Registro Timer (TIM) y un Registro Período (PRD) de 16 bits. El registro timer cuenta descendentemente en la caída de CLKOUT1. Los registros TIM y PRD son fijados a su máximo valor FFFFh en el reset. Una interrupción de Timer (TINT) es generada cada vez que el TIM llega a cero, el TIM es recargado con el valor de PRD en el próximo ciclo. Esta característica es útil para operaciones de control y sincronización de muestreo o escritura de periféricos.

Si el timer no es utilizado, TINT debe ser mascarable o todas las interrupciones deben ser deshabilitadas por la instrucción SETC INTM.

La razón de tiempo de interrupción del timer está dada por:

$$f_{TINT} = \frac{1}{t_{c(C)}\,(TDDR + 1)\,(PDR + 1)}$$

donde:

f_{TINT} : razón de interrupción de timer
$t_{c(C)}$: período del reloj CLKOUT1
TDDR + 1 : escalamiento de 4 bits
PRD + 1 : escalamiento de 16 bits

El primer contador (descendente) TDDR se encuentra en el registro de control de timer (TCR bits 0-3), los bits 6-9 del registro TCR corresponden al contador preescalador. Cuando el registro TCR llega a cero es cargado con el valor del registro TDDR.

La operación del timer es controlada por el registro TCR, el bit TSS permite detener la operación del timer al escribirle un uno, el bit TRB permite cargar de nuevo el período.

Registro de control del Timer (TCR)

Bit	Nombre	Descripción
0-3	TDDR	Razón de división del Timer
4	TSS	Detiene al Timer =1, Reinicializa al Timer =0
5	TRB	Reinicia el Timer con el PRD y TDDR
6-9	PSC	Contador preescalador

15-12	11	10	9-6	5	4	3-0
Reservado	SOFT	FREE	PSC	TRB	TSS	TDDR

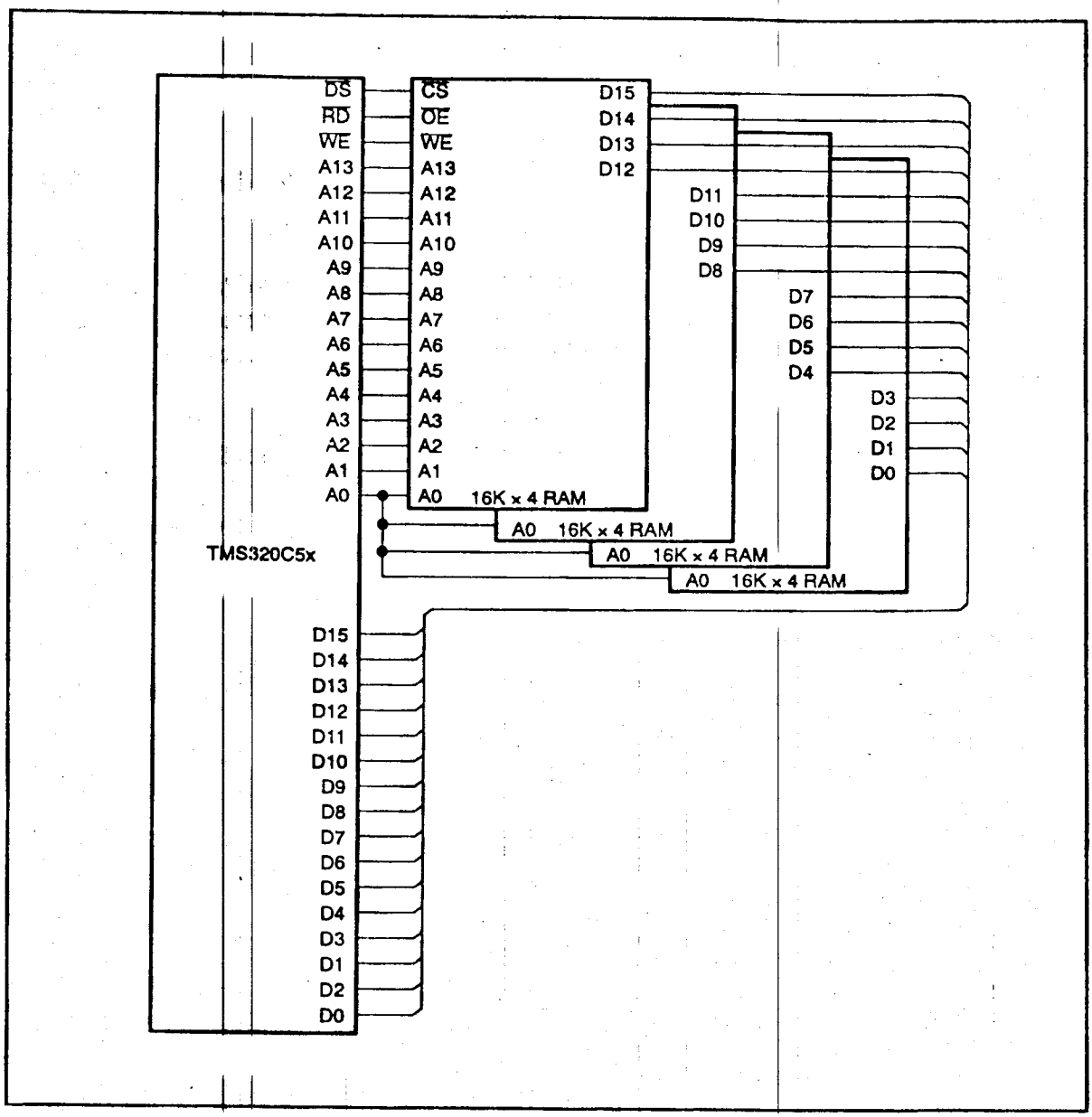
Los bits SOFT y FREE determinan el estado del timer cuando éste es detenido.

La secuencia para utilización del timer sería:

- escribir los valores necesarios al registro PRD y el TDDR al registro TCR .
- habilitar la interrupción del timer (desenmascararla) en el registro IMR.
- limpiar cualquier interrupción pendiente en el registro IFR.
- Habilitar interrupciones limpiando el bit de estado INTM.

INTERFACE A MEMORIA RAM EXTERNA

En la siguiente figura se observa la conexión directa del TMS50 a una memoria RAM, el TMS50 provee las señales necesarias para leer y escribir en la memoria RAM.



ESPACIO DE ENTRADA SALIDA (I/O)

El TMS50 puede direccionar hasta 64 k puertos I/O de 16 bits permitiendo el acceso a periféricos utilizados en aplicaciones para el PDS.

El acceso a puertos es multiplexado sobre el mismo bus de direcciones y datos utilizados para accesar a memoria programa y dato, el espacio I/O es distinguido por la activación de la señal externa /IS en bajo cuando el TMS50 ejecuta una instrucción de OUT o IN:

```
IN    DATO,01FFh ;     lee un dato del puerto 1FFh y lo escribe en la localidad de
                       memoria DATO
OUT   SALIDA,01F1h;    escribe un dato de la localidad e memoria SALIDA y lo
                       escribe en el puerto 1F1h.
```

De los 64 k puertos 16 son mapeados en el espacio de memoria dato (PA0 a PA15 de 50h a 5Fh), estos puertos son tratados como memoria, sin embargo externamente son distinguidos por la activación de la señal /IS. Estos puertos pueden accesarse con instrucciones que se utilizan para direccionar memoria:

```
SACL  52h        ; salva el acumulador al puerto externo 52h (DP = 0)
```

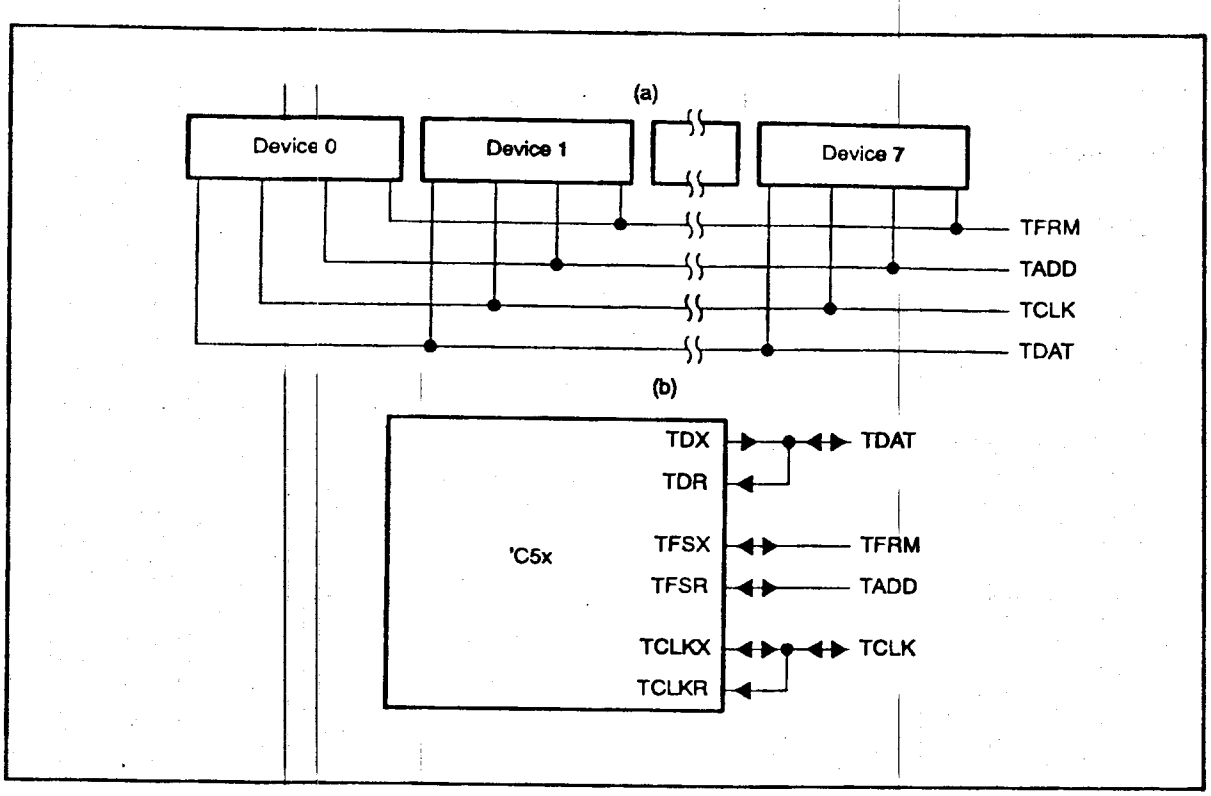
Las señales /RD y /WE pueden utilizarse para seleccionar un dispositivo como puerto de entrada o salida. También hay que recordar que a los puertos se les puede programar estados de espera por software.

PUERTO SERIE TDM

El TMS50 tiene un puerto serie multiplexado por división de tiempo (TDM) que le permite la comunicación serial con otros siete TMS50 proveyendo una poderosa interface para aplicaciones multiprocesos.

Para la operación del puerto serie TDM se pone un uno en el bit TDM del registro de configuración de puerto serie TDM (TSPC). Si el bit TDM = 0, el puerto TDM puede operar como otro puerto serie convencional del TMS50. El registro TSPC es similar al registro SPC del puerto serie a excepción de que el bit TDM indica la operación de este puerto.

El concepto de división de tiempo es que cada dispositivo toma una parte del tiempo disponible a compartir. El puerto TDM requiere juntar las líneas de transmisión de datos (TDX) y de recepción (TRD) en una línea simple llamada línea de datos (TDAT), para que los datos fluyan sobre esta línea. Similarmente los relojes se unen en una línea de reloj TDM. Una señal simple de frame es utilizada para indicar el inicio de un evento de 8 palabras TDM. Hasta ocho TMS50 pueden conectarse a cuatro líneas para la comunicación por el puerto TDM.



Conexión del TMS50 por puerto TDM

La línea TADD es manejada por un TMS50 particular para un tiempo particular y determina qué dispositivo de la conexión TDM puede efectuar una recepción válida en el tiempo de slot.

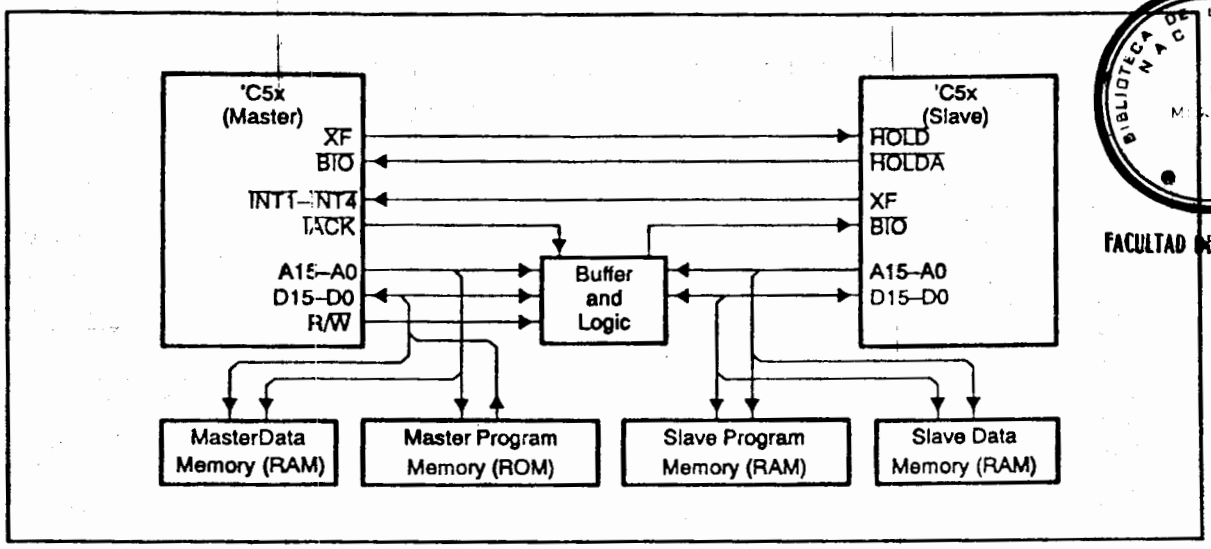
El puerto TDM tiene asociados seis registros mapeados en memoria. En una lectura válida TDM, el dato es transferido del registro TRSR al registro TRCV y una interrupción de recepción es generada indicando que el registro TRCV tiene un dato válido y que puede ser leído. Todos los puertos TDM operan sincronizados por las líneas TCLK y TFRM las cuales son generadas por cada dispositivo.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRCV	Receive Data															
TDXR	Transmit Data															
TSPC	FREE	SOFT	X	X	XRDY	RRDY	IN1	IN0	RRST	XRST	TXM	MCM	FSM	FO	DLB	TDM
TCSR	X	X	X	X	X	X	X	X	CH7	CH6	CH5	CH4	CH3	CH2	CH1	CH0
TRTA	TA7	TA6	TA5	TA4	TA3	TA2	TA1	RA0	RA7	RA6	RA5	RA4	RA3	RA2	RA1	RA0
TRAD	X	X	X2	X1	X0	S2	S1	S0	A7	A6	A5	A4	A3	A2	A1	A0

Registros del puerto TDM

ACCESO DIRECTO A MEMORIA

El TMS50 soporta multiprocesamiento utilizando Acceso Directo de Memoria (DMA). La característica del DMA es que puede utilizarse para multiprocesamiento deteniendo temporalmente la ejecución de uno o más procesos que permita a otro proceso leer o escribir a la memoria externa o interna de otro TMS50.



Configuración Maestro-esclavo utilizando DMA

Las dos señales /HOLD y /HOLDA permiten a otros dispositivos tomar el control de los buses del TMS50 esclavo. Cuando el TMS50 recibe una señal de /HOLD de un dispositivo externo, el TMS50 envía una señal de reconocimiento /HOLDA (activa baja), entonces pone sus buses de direcciones, datos y las señales de control (/PS, /DS, /IS, R/w y /STRB) en alta impedancia. Los pines de puerto serie (DX y FSX) no son afectados. /HOLD es muestreado en el tercer cuarto de ciclo de reloj, si es encontrado activo, éste toma tres ciclos de máquina antes que los buses y señales se vayan a alta impedancia, permitiendo terminar la instrucción que está siendo ejecutada.

Esto significa que el TMS50 maestro toma el control completo de la memoria externa del TMS50 esclavo, la señal de /HOLDA es encuestada en el pin /BIO del TMS50 maestro. La señal externa de salida XF del esclavo puede utilizarse para interrumpir al maestro indicándole la finalización de alguna tarea o el requerimiento de información adicional para seguir trabajando.

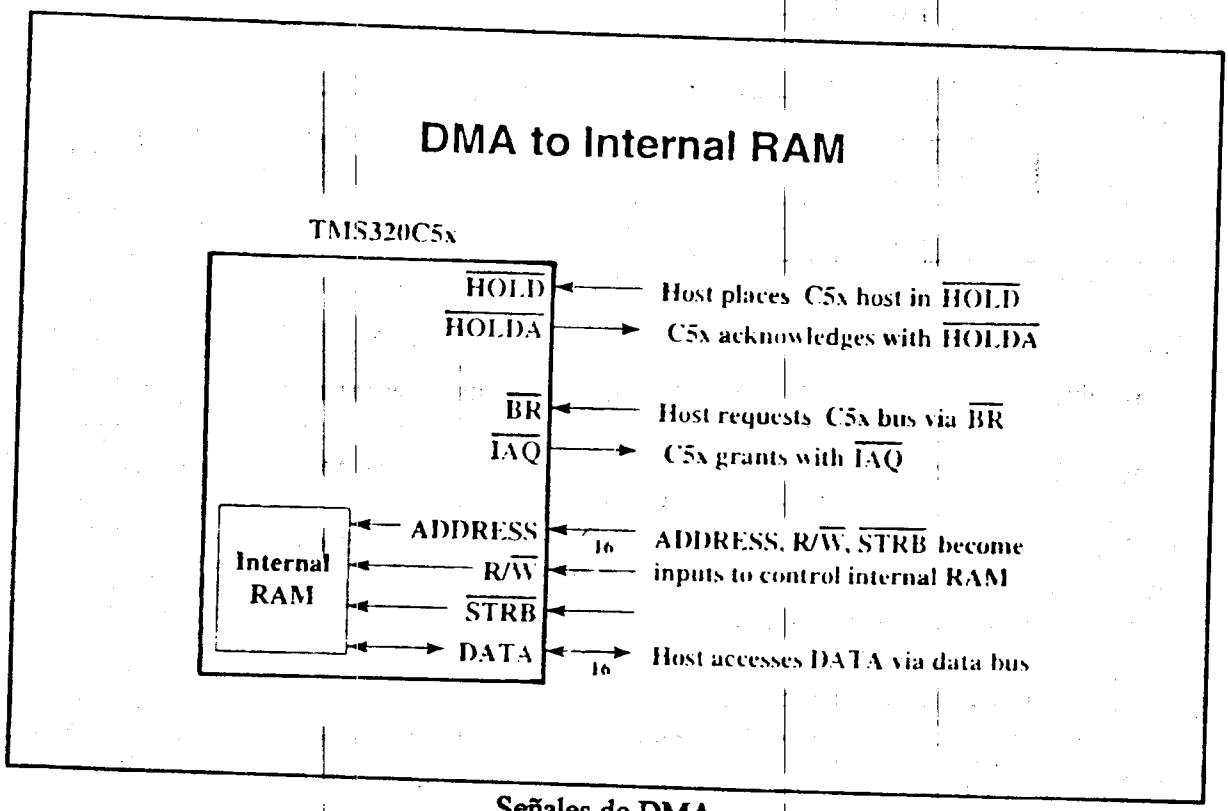
El modo de operación es seleccionada por HM (modo hold) en el registro de estado. Cuando HM=1, el TMS50 detiene la ejecución del programa e ingresa al estado de hold directamente. Cuando HM = 0 el procesador puede continuar la ejecución del programa en memoria interna pero pone la interface externa en alta impedancia. Si el programa en ejecución es de memoria interna y no se accesa dato externo de memoria, el procesador entra a estado externo de hold, pero el programa

sigue ejecutándose internamente (si HM = 0). Esto permite una operación más eficiente del sistema, dado que puede continuarse la ejecución mientras que una operación de DMA está siendo ejecutada.

Todas la interrupciones son deshabilitadas mientras que /HOLD es activada con HM = 1. Si HM=0 las interrupciones funcionan normalmente.

El dispositivo maestro puede accesar las localidades internas del TMS50 (esclavo), cuando el esclavo envía el reconocimiento de /HOLDA, el maestro puede enviar una señal baja a la entrada /BR del TMS50 esclavo, éste último responderá con la señal /IAQ reconociendo el acceso a su memoria interna. La dirección de transferencia es manejada por señal R/w por medio del maestro. Con la señal /STRB se selecciona el acceso a memoria, es decir que esta señal determina la duración del acceso a memoria.

La finalización de la transferencia DMA se da cuando se desactiva la señal /HOLD (se pone en alto) y en seguida también se desactiva la señal de /HOLDA.



Señales de DMA

MEMORIA GLOBAL

Para aplicaciones de multiprocesamiento, el TMS50 tiene la capacidad de localizar un espacio de memoria global de datos y comunicarse con este espacio a través de la señal /BR (requerimiento de bus) y la señal de control READY.

La memoria global es compartida por más de un procesador, por lo tanto para su acceso debe ser arbitrado. Cuando se utiliza memoria global, el espacio de direccionamiento del procesador es dividido en secciones local y global. La sección local es usada por el procesador para ejecutar funciones individuales y la sección global para comunicarse con otros procesadores. A diferencia del DMA, la lectura o escritura de memoria global no requiere que uno de los procesadores esté detenido.

El Registro de Mapeo de Memoria Global (GREG) especifica la parte de memoria de datos del TMS50 como memoria global externa. El registro GREG es mapeado como memoria de datos en la localización 5h, es un registro de 8 bit conectados con los 8 bits menos significativos del bus de datos.

El contenido de GREG determina el tamaño del espacio de memoria global. El valor de GREG y el correspondiente espacio es indicado en la tabla de configuración de memoria global:

Valor de GREG	LOCALIZACION DE MEMORIA		MEMORIA GLOBAL	
	Intervalo	No. de Palbras	Invervalo	No. de Palabras
000000xx	0-FFFFh	65,536	--	O
10000000	0-7FFFh	32,768	8000-FFFFh	32,768
11000000	0-BFFFh	49,152	C000-FFFFh	16,384
11100000	0-DFFFh	57,344	E000-FFFFh	8,192
11110000	0-EFFFh	61,440	F000-FFFFh	4,096
11111000	0-F7FFh	63,488	F800-FFFFh	2,048
11111100	0-FBFFh	64,512	FC00-FFFFh	1,024
11111110	0-FDFFh	65,024	FE000-FFFFh	512
11111111	0-FEFFh	65,280	FF000-FFFFh	256

Se observa que se puede compartir memoria global desde 256 hasta 32,768 palabras.
Cuando un dato de memoria es direccionado en modo directa o indirectamente correspondiente a direccionamiento de memoria global (definido por GREG), /BR es acertado bajo con /DS para indicar al procesador que se desea hacer una acceso de memoria global. La lógica externa arbitra el control de memoria global acertando una señal READY controlando el tiempo de acceso a la memoria. También se pueden generar estados de espera por software como ya se especificó. La memoria global es compartida por varios procesadores pero es accesada individualmente sólo por un procesador. Las señales /BR y /DS son utilizadas para habilitar el espacio de memoria global.

BIBLIOGRAFIA

- Edward A. Lee. Arquitectura programable DSP. IEEE ASSP MAGAZINE, octubre de 1988 y enero 1989.
- Texas Instruments. Digital Signal Processing Applications with the TMS320 Family, Theory, Algorithms, and implementations, vol.1,2 y 3. USA 1990.
- Texas Instruments. TMS30C1x, User's Guide. USA 1990.
- Texas Instruments. TMS30C2x, User's Guide. USA 1991.
- Texas Instruments. TMS30C3x, User's Guide. USA 1992.
- Texas Instruments. TMS320C5x, User's Guide. USA 1997.
- Texas Instruments. TMS320C5x, General-purpose applications user's guide. USA 1997.
- Texas Instruments. TMS320C5x DSP starter Kit. User's Guide. USA 1994.
- Alcántara Silva Rogelio. Introducción al procesamiento digital de señales. Septiembre 1989.
- Psenicka B. y Escobar S. L. Procesamiento Digital de Señales, Segunda parte, Microcontroladores y realización de los filtros digitales con TM320Cxx.
- Escobar S. Larry. Manual de laboratorio de procesamiento digital de señales. Facultad de Ingeniería, UNAM, abril del 2000.
- Escobar S. L. & Alcántara S. R. Fixed Point Arithmetic using Digital Signal Processors. International Conference on Telecommunications, ICT 2000. 22-25 may 2000, Acapulco, México.
- Escobar S. L. Algoritmos de filtrado adpatable: implementación, evaluación, comparación y aplicaciones en telecomunicaciones. Tesis de maestría en Ingeniería Eléctrica. Facultad de Ingeniería, UNAM, nov. 1997.

Esta obra se terminó de imprimir
en agosto del 2000
en el taller de imprenta del
Departamento de Publicaciones
de la Facultad de Ingeniería,
Ciudad Universitaria, México, D.F.
C.P. 04510

Secretaría de Servicios Académicos

El tiraje consta de 300 ejemplares
más sobrantes de reposición.