PAC- Performance-centered Adaptive Curriculum for Employment Needs
Programa ERASMUS: Acción Multilateral - 517742-LLP-1-2011-1-BG-ERASMUS-ECUE

MASTER DEGREE:

Industrial Systems Engineering

ASIGNATURA ISE3:

Electrónica para Sistemas Industriales (EIS)

MÓDULO 4:

Circuitos VLSI básicos en la arquitectura de ordenadores

TAREA 4-2:

ARQUITECTURA DE MICROPROCESADOR

Departamento de Ingeniería Eléctrica, Electrónica y de Control
http://www.ieec.uned.es/

Contenido

Índice de figuras

TAREA 4-2: ARQUITECTURAS DE MICROPROCESADOR

1. INTRODUCCIÓN Y OBJETIVOS.

El conjunto de registros que constituyen un sistema en particular y las transferencias de datos que sean posibles entre ellos forman la arquitectura del sistema. Los tipos de registro en el microprocesador y las posibles transferencias de datos entre ellos determinan la arquitectura del microprocesador.

Casi todos los microprocesadores contienen, según vimos antes, como mínimo lo siguiente:

- Unidad aritmético y lógica (ALU).
- Varios registros.
- Contador de programa
- Circuitería de decodificación de instrucciones.
- Sección de control y temporización.
- Cerrojos y buffers de datos.
- Líneas de control y buses internos.
- Varias entradas y salidas de control.

Además de estos elementos, una pastilla microprocesadora puede contener también algunas de las unidades funcionales siguientes:

- Memoria ROM.
- Memoria RAM.
- Puertos de entrada/salida serie.
- Circuitería de reloj interna.
- Temporizadores programables.
- Circuitería de arbitración de prioridad de interrupciones.
- Lógica de interfaz de comunicación de E/S serie a paralelo.
- Lógica de control de acceso directo a memoria.

La arquitectura de un PC también puede contar con coprocesadores. Estos son microprocesadores especializados en la ejecución de determinados cálculos, que descargan de estas funciones a la unidad central de proceso.

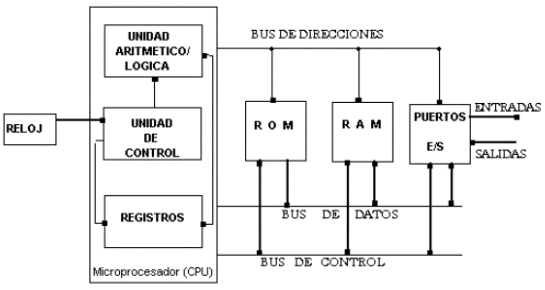

Figura 1: Esquema de un Microprocesador y de como se encuentran conectados los elementos de una pastilla microproceadora

La arquitectura de un microprocesador es esencial para determinar el camino para programarlo. La arquitectura unicamente especifica el camino de ejecución de las instrucciones, dadas por el procesador, el número y la organización de los registros de datos(las areas de almacenaje para datos durante la ejecución de una operación), y la configuración de los terminales input-output(canales físicos a través de los cuales los datos son transferidos de una parte de la computadora a otra).

Cuando la arquitectura esta ya definida, todas las opciones del microprocesador se encuentran fijadas (lo cual permite la elección del lenguaje de programación, correspondiente a la arquitectura, donde cada bit de instrucción correspode a material aviable o elemento lógicos...).

Este lenguaje de escritura se llama Emsamblador. El ensamblador es un lenguaje de programación que , por la traducción directa de los mnemónicos a instrucciones máquina, permite realizar aplicaciones rápidas, solucionando situaciones en las que los tiempos de ejecución constituye el factor principal para que el proceso discurra con la suficiente fluidez. Para consecución de esta fluidez es por la que han aparecido diferentes tipos de arquitecturas, y al elegir una para el diseño de nuestro microprocesador definiremos el uso y lenguaje a usar sobre este.

2. ARQUITECTURA DEL MICROPROCESADOR

Antes de comenzar con el estudio de las cuatro arquitecturas que va a ocupar el objeto de estudio vamos a introducir un par de arquitecturas primitivas sobre las que se basan hoy en día todas.

2.1 ARQUITECRURA VON NEUMANN

El origén de esta arquitectura nace de la colaboración del matemático húngaro John Von Neumann, en el proyecto ENIAC (primera computadora de la historia). En concreto Von Neumann se intereso por la problemática de tener que recablear la máquina para cada nueva tarea.

Así en 1949 encontró y desarrollo la solución a este problema, consistió en poner la información sobre las operaciones a realizar en la misma memoria utilizada para los datos, escribiendola en la misma forma, es decir en código binario. A partir de este momento todo los ordenadores construidos con esta estructura se denominarían que utilizaban la arquitectura de Von Neumann. El primero que se construyo fue el UNIVAC I para la oficina de censo de los estados unidos.

Esta arquitectura que estara formada por las partes básica y elementales vistas anteriormente:

- Unidad aritmético lógica
- Unidad de control
- Unidad de entrada y salida
- La memoria
- El bus de datos

Las principales limitaciones que nos encontramos con la arquitectura Von Neumann son:

- La limitación de la longitud de las instrucciones por el bus de datos, que hace que el microprocesador tenga que realizar varios accesos a memoria para buscar instrucciones complejas.

- La limitación de la velocidad de operación a causa del bus único para datos e instrucciones que no deja acceder simultáneamente a unos y otras, lo cual impide superponer ambos tiempos de acceso operación

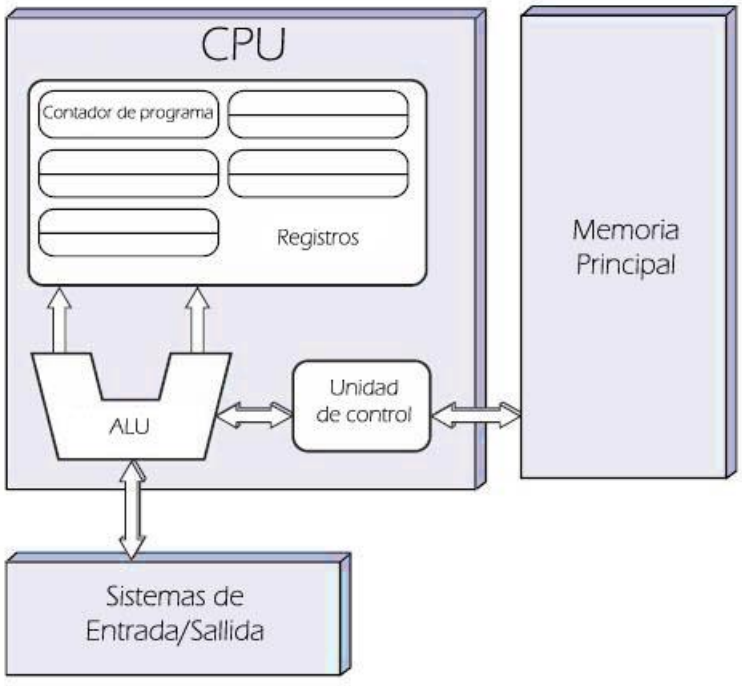

Figura 2: ARQUITECTURA DE VON NEUMANN

Un ordenador con arquitectura Von Neumann realizará o emulará los siguientes pasos secuencialmente:

- Obtiene la siguiente instrucción desde la memoria en la dirección indicada por el contador deprograma y la guarda en el registro de instrucción.

- Aumenta el contador de programa en la longitud de la instrucción para apuntar a la siguiente.

- Descodifica la instrucción mediante la unidad de control. Ésta se encarga de coordinar el resto de componentes del ordenador para realizar una función determinada.

- Se ejecuta la instrucción. Ésta puede cambiar el valor del contador del programa, permitiendo así operaciones repetitivas. El contador puede cambiar también cuando se cumpla una cierta condición aritmética, haciendo que el ordenador pueda 'tomar decisiones', que pueden alcanzar cualquier grado de complejidad, mediante la aritmética y lógica anteriores.

- Vuelve al paso N° 1.

2.2 ARQUITECTURA HARVARD

En este modelo, que utilizan los Microcontroladores PIC, tiene la unidad central de proceso (CPU) conectada a dos memorias (una con las instrucciones y otra con los datos) por medio de dos buses diferentes.

Una de las memorias contiene solamente las instrucciones del programa (Memoria de Programa), y la otra sólo almacena datos (Memoria de Datos). Ambos buses son totalmente independientes lo que permite que la CPU pueda acceder de forma independiente y simultánea a la memoria de datos y a la de instrucciones. Como los buses son independientes éstos pueden tener distintos contenidos en la misma dirección y también distinta lóngitud.

También la longitud de los datos y las instrucciones puede ser distinta, lo que optimiza el uso de la memoria en general. Para un procesador de Set de Instrucciones Reducido, o RISC (Reduced Instrucción Set Computer), el set de instrucciones y el bus de memoria de programa pueden diseñarse de tal manera que todas las instrucciones tengan una sola posición de memoria de programa de longitud. Además, al ser los buses independientes, la CPU puede acceder a los datos para completar la ejecución de una instrucción, y al mismo tiempo leer la siguiente instrucción a ejecutar.

Figura 3: ARQUITECTURA DE HARVARD

Ventajas de esta arquitectura:

- El tamaño de las instrucciones no esta relacionado con el de los datos, y por lo tanto puede ser optimizado para que cualquier instrucción ocupe una sola posición de memoria de programa,logrando así mayor velocidad y menor longitud de programa.

- El tiempo de acceso a las instrucciones puede superponerse con el de los datos, logrando una mayor velocidad en cada ciclo

Estas arquitecturas son las que han abierto el camino al resto, y sobre estos dos sitemas de organización podemos encontrar basados todas las arquitectuturas actualmente. Vamos a a pasar a estudiar arquitecturas más modernas y que han revolucionado el mundo de los procesadores.

2.3 ARQUITECTURA CISC

CISC es un tipo de arquitectura de computadoras que promueve el uso de gran número de instrucciones, permitiendo operaciones complejas entre operandos situados en memoria o en registros internos.

La tecnología CISC nació de la mano de Intel, creado en 1971, permitiría el nacimiento de la informática personal. Más concretamente, sería en 1972 cuando aparecería el “8080” (primer chip capaz de procesar 8 bits, suficiente para representar números y letras).

Los microprocesadores CISC tienen un conjunto de instrucciones que se caracteriza por ser muy amplio y permitir operaciones complejas entre operando situados en la memoria o en los registros internos. Esta arquitectura se basa en que cada instrucción puede corresponder a varias operaciones de bajo nivel, tales como leer de memoria, operación aritmética, escribir en la memoria, sumar datos… todo en una sola instrucción.

Este tipo de arquitectura dificulta el paralelismo entre instrucciones, por lo que, en la actualidad, la mayoría de los sistemas CISC de alto rendimiento implementan un sistema que convierte dichas instrucciones complejas en varias instrucciones simples del tipo RISC, llamadas generalmente microinstrucciones.

La microprogramación es una característica importante y esencial de casi todas las arquítecturas CISC. La microprogramación significa que cada instrucción de máquina es interpretada por una microprograma localizada en una memoria en el circuito integrado del procesador. Las instrucciones compuestas son decodificadas internamente y ejecutadas con una serie de microinstrucciones almacenadas en una ROM interna. Para esto se requieren de varios ciclos de reloj, al menos uno por microinstrucción. Es así entonces como los chips CISC utilizan comandos que incorporan una gran diversidad de pequeñas instrucciones para realizar una única operación.

Cuando el sistema operativo o una aplicación requiere de una de estas acciones, envía al procesador el nombre del comando para realizarla junto con el resto de información complementaria que se necesite. Pero cada uno de estos comandos de la ROM del CISC varían de tamaño y, por lo tanto, el chip debe en primer lugar verificar cuanto espacio requiere el comando para ejecutarse y poder así reservárselo en la memoria interna. Además, el procesador debe determinar la forma correcta de cargar y almacenar el comando, procesos ambos que ralentizan el rendimiento del sistema.

El procesador envía entonces el comando solicitado a una unidad que lo descodifica en instrucciones más pequeñas que podrán ser ejecutadas por un nanoprocesador, una especie de procesador dentro del procesador. Y al no ser las instrucciones independientes, pues son instrucciones menores procedentes de la descodificación de una instrucción mayor, sólo puede realizarse una instrucción cada vez.

A través de la compleja circuitería del chip, el nanoprocesador ejecuta cada una de las instrucciones del comando. El desplazamiento por esta circuitería también ralentiza el proceso. Para realizar una sola instrucción un chip CISC requiere de cuatro a diez ciclos de reloj.

El acercamiento de CISC procura reducir al mínimo el número de instrucciones por programa, sacrificando el número de ciclos por la instrucción. El RISC hace lo opuesto, reduciendo los ciclos por la instrucción en el coste del número de instrucciones por programa.

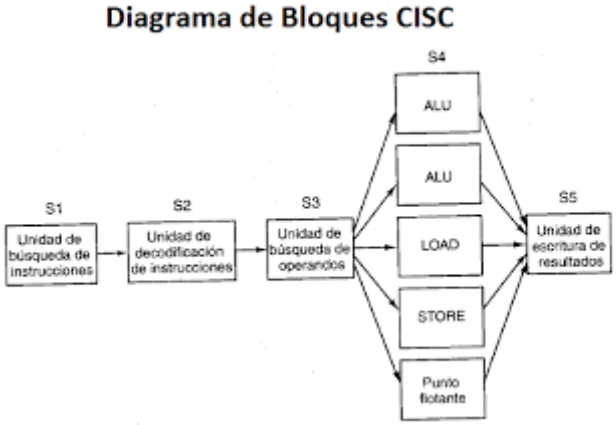

Figura 4: Arquitectura CISC

-CARACTERISTICAS DE LA ARQUITECTURA CISC-

El propósito esencial de una arquitectura CISC es intentar proporcionar única instrucción de máquina para cada enunciado que esté escrita en un lenguaje de alto nivel.

Otra característica de la arquitectura CISC es la incorporación de formatos de instrucciones de tamaño variable.

Las instrucciones que necesitan operandos de registros pueden tener sólo dos bytes de longitud, pero las instrucciones que necesitan direcciones de memoria pueden necesitar cinco bytes para incluir todo el código de la instrucción.

Las instrucciones en un procesador CISC típico proporcionan la manipulación directa de los operandos que residen en la memoria. Por ejemplo, una instrucción ADD puede especificar un operando en la memoria mediante un direccionamiento de índice y un segundo operando en la memoria por medio de un direccionamiento directo.

Aunque los procesadores CISC tienen instrucciones que sólo utilizan registros de procesador, la disponibilidad de otros modos de operaciones tiende a simplificar la compilación de lenguajes de alto nivel. Sin embargo, conforme se incorporan más instrucciones y modos de direccionamiento en una computadora, se necesita más circuitería lógica para implementarlos y soportarlos, y esto puede producir que los cálculos se hagan lentos.

La microprogramación es una característica importante y esencial de casi todas las arquítecturas CISC. Como por ejemplo:

Intel 8086, 8088, 80286, 80386, 80486.

Motorola 68000, 68010, 68020, 68030, 6840.

La microprogramación significa que cada instrucción de máquina es interpretada por un microprograma localizado en una memoria en el circuito integrado del procesador.

En la década de los sesenta la micropramación, por sus características, era la técnica más apropiada para las tecnologías de memorias existentes en esa época y permitía desarrollar también procesadores con compatibilidad ascendente. En consecuencia, los procesadores se dotarón de poderosos conjuntos de instrucciones.
Las instrucciones compuestas son decodificadas internamente y ejecutadas con una serie de microinstrucciones almacenadas en una ROM interna. Para esto se requieren de varios ciclos de reloj (al menos uno por microinstrucción).

Los microprocesadores CISC tienen un conjunto de instrucciones que se caracteriza por ser muy amplio y permitir operaciones complejas entre operandos situados en la memoria o en los registros internos.
Este tipo de arquitectura dificulta el paralelismo entre instrucciones, por lo que en la actualidad la mayoría de los sistemas CISC de alto rendimiento implementan un sistema que convierte dichas instrucciones complejas en varias instrucciones simples, llamadas generalmente microinstrucciones.

Las instrucciones compuestas son decodificadas internamente y ejecutadas con una serie de microinstrucciones almacenadas en una ROM interna. Para esto se requieren de varios ciclos de reloj, al menos uno por microinstrucción. Es así entonces como los chips CISC utilizan comandos que incorporan una gran diversidad de pequeñas instrucciones para realizar una única operación.

El objetivo principal de la arquitectura CISC es completar una tarea en el menor número de líneas de código ensamblador posibles. Este objetivo es conseguido mediante la construcción de un microprocesador capaz de comprender y ejecutar una serie de operaciones complejas.

Una de las ventajas principales de esta filosofía es que el compilador tiene que hacer muy poco trabajo para traducir un lenguaje de alto nivel a ensamblador. Además, debido a que la longitud del código es relativamente corta, hace falta poca RAM para almacenar las instrucciones. Pero la dificultad está en construir instrucciones complejas directamente en hardware.

-REGISTROS CISC-

Los registros son mayoritariamente de uso dedicado

Dentro de un procesador existen dos tipos de registros. Los registros De propósito general son aquellos que el procesador puede utilizar para almacenar datos temporales, variables locales... Por su parte, los registros de uso dedicado son aquellos que están reservados para tareas muy específicas (almacenar el contador de programa, el stack pointer...).

El número de registros de propósito general es reducido en las arquitecturas CISC. Por un lado, se debe a que el elevado número de modos de direccionamiento provoca que casi todo el tránsito de datos se produzca de memoria a memoria. Por otro lado, la mayor parte del espacio del chip se utiliza para la decodificación y la ejecución, así como para el almacenamiento del microcódigo, dejando poco espacio para estos registros. El compilador que se use ha de ser capaz de maximizar el rendimiento de los pocos registros de propósito general que hay, con el fin de lograr una ejecución mucho más eficiente del programa.

En cambio, sí son abundantes los registros de uso dedicado que controlan el tránsito de datos, y el estado del procesador. Algunos de estos registros son utilizados para almacenar el stack pointer, para realizar la gestión de las interrupciones y para almacenar los códigos de condición.

Por último, cabe destacar que los computadores basados en arquitecturas CISC suelen incorporar una memoria intermedia rápida (caché) para agilizar cálculos y para almacenar datos temporales muy usados.

-VENTAJAS Y DESVENTAJAS-

CISC tiene un coste "razonable", que es alcanzado a nivel de usuario. Esto mismo, no ocurre con los RISC, que por el contrario tienen un coste elevado, por esto mismo esta tecnología ha sido enfocada a ventas a nivel de empresa y equipos de gama alta.

La utilidad que se le dé a la maquina es muy importante, ya que el usuario debe de encontrar un nivel óptimo en cuanto a calidad - precio. Y por qué pagar más si realmente no se le va a sacar partido al cien por cien.

El software utilizado es otro de los factores importantes, dado que un RISC no utiliza el mismo software que un CISC. Estos últimos, por lo general tienen un software más asequible.

Dada la compatibilidad hacia atrás dela familia CISC x86, los usuarios han podido renovar sus equipos sin por ello tener que abandonar software que ya

conocían, y reutilizar sus datos. Así mismo, los fabricantes han tenido en cuenta este factor, puesto que seguir con otra línea de procesadores suponía no solo un cambio muy radical, sino que además podía llevar un riesgo en cuanto a ventas. Estos son algunos de los motivos. Sin embargo, también hay que tener en cuenta el conflicto de intereses de algunos fabricantes, así como la opinión de distintas revistas, algunas de ellas asociadas a diferentes marcas. Se están estudiando las tendencias futuras, como pueden ser los híbridos, mejoras en los microprocesadores CISC, mejoras en los RISC

2.4 ARQUITECTURA RISC

Buscando aumentar la velocidad del procesamiento se descubrió en base a experimentos que, con una determinada arquitectura de base, la ejecución de programas compilados directamente con microinstrucciones y residentes en memoria externa al circuito integrado resultaban ser mas eficientes, gracias a que el tiempo de acceso de las memorias se fue decrementando conforme se mejoraba su tecnología de encapsulado.

Debido a que se tiene un conjunto de instrucciones simplificado, éstas se pueden implantar por hardware directamente en la CPU, lo cual elimina el micro-código y la necesidad de decodificar instrucciones complejas.
En investigaciones hechas a mediados de la década de los setentas, con respecto a la frecuencia de utilización de una instrucción en un CISC y al tiempo para su ejecución, se observó lo siguiente:
Alrededor del 20% de las instrucciones ocupa el 80% del tiempo total deejecución de un programa.
Existen secuencias de instrucciones simples que obtienen el mismo resultado que secuencias complejas predeterminadas, pero requieren tiempos de ejecución más cortos.

-CARACTERÍSTICAS DE LA ARQUITECTURA RISC-

Estos microprocesadores siguen tomando como base el esquema moderno de Von Neumann.

Las instrucciones, aunque con otras características, siguen divididas en tres grupos:

- Transferencia.
- Operaciones.
- Control de flujo.

Reducción del conjunto de instrucciones a instrucciones básicas simples, con la que pueden implantarse todas las operaciones complejas.

Arquitectura del tipo load-store (carga y almacena). Las únicas instrucciones que tienen acceso a la memoria son 'load' y 'store'; registro a registro, con un menor número de acceso a memoria.

Casi todas las instrucciones pueden ejecutarse dentro de un ciclo de reloj. Con un control implantado por hardware (con un diseño del tipo load-store), casi todas las instrucciones se pueden ejecutar cada ciclo de reloj, base importante para la reorganización de la ejecución de instrucciones por medio de un compilador.

Pipeline (ejecución simultánea de varias instrucciones). Posibilidad de reducir el número de ciclos de máquina necesarios para la ejecución de la instrucción, ya que esta técnica permite que una instrucción puede empezar a ejecutarse antes de que haya terminado la anterior.

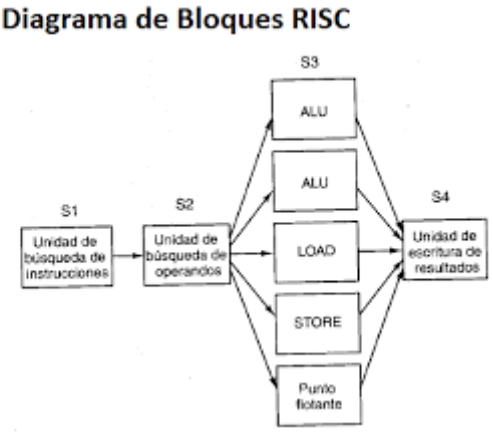

Figura 5: Arquitectura RISC

PROPIEDADES DEL SITEMA TIPO RISC

Como se discutió en la sección anterior algunas de las condiciones necesarias para archivar una operación reducida en un sistema RISC son:

- Estándar, tamaño arreglado de instrucción, igual para la computadora la longitud de la palabra y para el ancho de el bus de datos (con la estipulación algunos nuevos sistemas la longitud de la palabra y/o del bus de datos en un entero múltiple del tamaño de la instrucción como en estos nuevos sistemas de 64 bits y en algunos sistemas de 32 bits con un bus de datos de 64 bits).

- El tiempo de la ejecución es estandart para todas las instrucciones, preferiblemente dentro un ciclo singular del CPU (con la estipulación esta minoría de instrucciones igual como divide tendrá que ser ejecutado en mas de un solo ciclo.

- RISC es una arquitectura de tipo load/store

Como hemos visto, el número de referencias por instrucción en un lenguaje de alto nivel es elevado y la mayoría de las instrucciones sólo requieren un simple flujo de datos. Como resultado, la gran mayoría de direccionamientos en las instrucciones RISC son de tipo registro-registro. Es decir se cargan los operandos en los registros mediante una operación de tipo load, se realizan las operaciones pertinentes entre los registros y los resultados se almacenan en memoria mediante una instrucción de tipo store. A este modelo de arquitectura, exclusiva de los RISC, se le conoce con el nombre de load/store.

Evidentemente, los registros del procesador son los que brindan los tiempos de acceso más cortos debidos tanto a la tecnología de acceso como al número de bits necesarios para direccionarlos. En una arquitectura de tipo load/store Lo ideal sería poder tener todos los operandos que se necesiten en la ejecución de un programa ubicados en registros. Pero entonces el coste del hardware se vería incrementado notablemente.

Se necesita una estrategia para ubicar aquellos operandos a los que se accede con más frecuencia y reducir el tráfico registro-memoria generado en las operaciones de tipo load y store.
Una estrategia consistiría en confiar al compilador la maximización del uso de los registros. La otra estrategia, más acorde con la filosofía RISC, es disponer de una cantidad elevada de registros para poder mantener ubicadas las variables durante un período de tiempo mayor.

PROCESOS DE INSTRUCCIONES

Los microprocesadores basados en esta arquitectura poseen instrucciones de tamaños fijos y presentados en un número reducido de formatos y en donde sólo las instrucciones de carga y almacenamiento acceden a la memoria por datos. También suelen disponer de muchos registros de propósito general.

Debido a que se tiene un conjunto de instrucciones simplificado, éstas se pueden implantar por hardware directamente en la CPU, lo cual elimina el microcódigo y la necesidad de decodificar instrucciones complejas.

Podemos ver las siguientes características pra el proceso de instrucciones:

- El objetivo es que se ejecuten rápido, de ser posible en un único ciclo de maquina (luego de ser captadas y decodificadas, por supuesto).

- Pipeline de tres etapas para las instrucciones sin referencia a memoria: FI – DI – EI.

- Se puede utilizar control cableado porque son simples.

- La unidad de control es simple, por lo tanto puede funcionar con mayor frecuencia de clock.

- El pipeline es eficiente si las instrucciones son de tiempo de ejecución similar en casa etapa.

- Usualmente instrucciones de ancho fijo son de 32 bits. Esto hace que la carga y decodificación de instrucciones sea simples y rápida. No se necesita esperar a que se conozca el largo de la instrucción actual para empezar a decodificar la próxima.

- El formato uniforme simplifica la decodificación porque el código de operación y el campo de dirección están ubicados en la misma posición para todas las instrucciones.

REGISTRO RISC

Registros de parámetros: Estos registros contienen el conjunto de parámetros que se pasan del procedimiento padre al que está en curso. A través de ellos un procedimiento hijo se puede comunicar con su padre sin que exista un flujo real de datos entre ambos.

Registros locales: Los registros locales sirven para almacenar aquellos operandos que el compilador haya seleccionado con fines optimizadores, debido sobre todo al gran número de asignaciones en las que se ven involucrados.

Registros temporales: Los registros temporales sirven para realizar el paso de argumentos desde un procedimiento a sus hijos. Del mismo modo que ocurría con los registros de parámetros, no es necesario un flujo real de datos entre los dos procedimientos.
Tal como podemos observar, existe solapamiento entre diversas ventanas siempre y cuando se realicen llamadas y exista paso de argumentos.

Sin embargo, el número de procedimientos activos en un sistema y el nivel de anidamiento son imprevisibles y es imposible disponer de un número infinito de ventanas.

Por ello, el aspecto real del fichero de registros es el de un buffer circular. Sólo se mantienen marcos o ventanas para aquellos procedimientos que sean más recientes.

Los más antiguos se han de guardar en memoria para, posteriormente, cargarlos de nuevo en el fichero de registros.

VENTAJAS DE LA ARQUITECTURA RISC

La estrategia del RISC también trae algunas ventajas muy importantes. Porque cada instrucción requiere solamente un ciclo de reloj ejecutarse, el programa entero se ejecutará en aproximadamente la misma cantidad de tiempo que el comando del multiciclo "MULT".

Este el RISC "instrucciones reducidas" requiere menos transistores del espacio del hardware que las instrucciones complejas, saliendo de más sitio para los registros de fines generales. Porque todas las instrucciones se ejecutan en una cantidad de tiempo uniforme (es decir un reloj), el canalizar es posible.

La separación de las instrucciones de la "CARGA" y del "ALMACÉN" reduce realmente la cantidad de trabajo que la computadora debe realizar. Después del CISC-estilo "MULT" se ejecuta un comando, el procesador borra automáticamente los registros. Si uno de los operandos necesita ser utilizado para otro cómputo, el procesador debe recargar los datos del banco de memoria en un registro. En el RISC, seguirá habiendo el operando en el registro hasta que otro valor se carga en su lugar.

RISC Vs CISC

Es que los procesadores CISCx86 corren a DOS, Windows 3.1 y Windows 95 en el modo nativo; es decir, sin la traducción de software que disminuya el desempeño. Pero CISC y RISC también reflejan dos filosofías de computación rivales. El procesamiento de RISC requiere breves instrucciones de software de la misma longitud, que son fáciles de procesar rápidamente y en tandém por un CPU.

En contraste, un procesador de CISC tiene que procesar instrucciones más largas de longitud desigual. Es más difícil procesar múltiples instrucciones de CISC a la vez.

Figura 6: logotipos de Arquitecturas

Los que proponen RISC mantienen que su método de procesamiento es más eficiente y más escalable, por lo que los arquitectos pueden añadir unidades de ejecución más fácilmente a un diseño existente y aumentar el rendimiento (las unidades de ejecución son los circuitos dentro del chip que hacen gran parte del trabajo). Similarmente, RISC facilita el multiprocesamiento verdadero, donde varios CPUs trabajan simétricamente mientras dividen, ejecutan y ensamblan una cadena de instrucción; los chips CISC pueden hacer lo mismo, pero no son tan efectivos. La simplicidad de las instrucciones de RISC también significa que requieren menos lógica para ejecutar, reduciendo el costo del chip. Pocos en el campo del CISC discuten estos hechos, prefiriendo apuntar a la realidad.

Todo el debate de CISC/RISC puede ser irrelevante pronto debido a que nuevas técnicas están convergiendo. El Pentium Pro, el Nx586 y el K5 son básicamente procesadores RISC en su núcleo. Toman las instrucciones de CISC y las traducen a instrucciones estilo RISC. Para la generación que sigue al Pentium Pro, Intel y Hewlett-Packard están colaborando en un CPU híbrido que pueda aceptar instrucciones RISC y CISC. Si ese chip crea un estándar, puede acelerar el cambio hacia el software optimizado para RISC. Un mundo de RISC significaría CPUs más poderosos, y más baratos. Cuando quiera mejorar, simplemente puede añadir otro CPU en lugar de desprenderse de su viejo CPU.

¿RISC O CICS?

El conflicto surge al evaluar las ventajas netas ¿qué es más apropiado, usar muchas instrucciones de un solo ciclo aprovechadas al máximo, o pocas de múltiples pasos de reloj en las que existe infrautilización?

La cuestión, es que hasta el momento, el estudio de prestaciones de ambas tecnologías, nos ha llevado a concluir que hoy en día los RISC obtienen mas prestaciones, es decir, son más potentes y rápidos que los CISC. Sin embargo, el mercado se ha decantado por la tecnología CISC en cuanto a volumen de ventas.

Habría que preguntarse porque ha optado el mercado por la tecnología CISC, aquí vemos algunas razones.

- Por experiencia, se puede comprobar que un CISC tiene un coste "razonable", que es alcanzado a nivel de usuario. Esto mismo, no ocurre con los RISC, que por el contrario tienen un coste elevado, por esto mismo esta tecnología ha sido enfocada a ventas a nivel de empresa y equipos de gama alta.

- La utilidad que se le dé a la maquina es muy importante, ya que el usuario debe de encontrar un nivel optimo en cuanto a calidad - precio. Y por qué pagar más si realmente no se le va a sacar partido al cien por cien.

- El software utilizado es otro de los factores importantes, dado que un RISC no utiliza el mismo software que un CISC. Estos últimos, por lo general tienen un software más asequible.

- Dada la compatibilidad hacia atrás de la familia CISC x86, los usuarios han podido renovar sus equipos sin por ello tener que abandonar software que ya conocían, y reutilizar sus datos. Así mismo, los fabricantes han tenido en cuenta este factor, puesto que seguir con otra línea de procesadores suponía no solo un cambio muy radical, sino que además podía llevar un riesgo en cuanto a ventas.

Estos son algunos de los motivos. Sin embargo, también hay que tener en cuenta el conflicto de intereses de algunos fabricantes, así como la opinión de distintas revistas, algunas de ellas asociadas a diferentes marcas.

Existen varios mitos que contraponen las ventajas de la tecnología RISC frente a la CISC, que es importante descalificar:

- Los procesadores RISC ofrecen peor soporte para los lenguajes de alto nivel o HLL (High Level Language) que lo CISC. Esta creencia se argumenta en que un conjunto de instrucciones de "alto nivel" (CISC) es mejor soporte para lenguajes de alto nivel. Sin embargo, la característica fundamental de los lenguajes de alto nivel, implica que el programador sólo interacciona con el ordenador a través del propio lenguaje de alto nivel (programación, depuración, mensajes del sistema, etc.), por lo que todos los problemas a "bajo nivel", deben de ser transparentes y desconocidos para el. Por ello, son de nulas consecuencias para el programador y los lenguajes de alto nivel, como se implementan las funciones, en función del tipo de CPU.

- Es más complicado escribir compiladores RISC que CISC. Dado que los procesadores CISC tienen un mayor número de instrucciones y modos de direccionamiento, existen por tanto más formas de hacer la misma tarea, lo que puede confundir tanto al compilador como al que lo escribe. Por ello, subjetivamente es posible escoger una forma de hacerlo poco adecuada, por el tipo de instrucciones o por el tiempo de ejecución que requieren. En cambio, en un procesador RISC, hay menos opciones, por lo que el compilador es más simple, aunque se genere, habitualmente, un 20-30% más código; a cambio, se consigue un

incremento de la velocidad de hasta un 500%.

- Un programa es más rápido cuanto más pequeño. La velocidad a la que un programa puede ser ejecutado no depende en absoluto de su tamaño, sino del tiempo de ejecución de cada una de sus instrucciones. Dado que las instrucciones RISC son más rápidas, y admiten mejor los pipelines, puede haber mayor paralelismo y simultaneidad en la ejecución de pequeñas secciones de código. Dichas secciones de código pueden ser ejecutadas en una fracción del tiempo que requiere una sola instrucción CISC.

2.5 Arquitectura VLIW

Ante la problemática de poder aumentar el rendimiento de los procesadores. Está se vio limitada al hecho de que estos solo podian realizar una instrucción por ciclo de reloj, para poder superar este punto se presentó la arquitectura VLIW , la cuál presenta la posibilidad de realizar instrucciones en paralelo esto quiere decir que podrián realizarse más operaciones por ciclo de reloj.

VLIW es una arquitectura para diseño de CPUs que implementa una forma de paralelismo a nivel de instrucción. Utiliza varias unidades funcionales (varias ALU, varios multiplicadores, etc.) para lograr paralelismo.

Los procesadores VLIW se caracterizan por tener instrucciones de gran tamaño. Esto se debe a que en cada instrucción se especifica el estado de todas las unidades funcionales del sistema. Esto se hace para simplificar el diseño del hardware, dejando el trabajo de planificación del código al programador/compilador. Esto es contrario a la arquitectura superescalar, en donde es el hardware el encargado de planificar las instrucciones en tiempo de ejecución. Una de las ventajas de esta arquitectura, es la simplificación de la arquitectura de hardware al no tener que planificarse el código.

Otra ventaja es la reducción del número de instrucciones de los programas. En tanto una de sus desventajas es que requiere compiladores más complejos, además, cualquier mejora en la arquitectura de hardware implica un cambio en las instrucciones, eso significa que no existe compatibilidad hacia atrás.

Cuando los diseñadores han de dedicar los cada vez más transitores que es posible integrar en la fabricación de un procesador, tienen pocas posibilidades. Por un lado, incrementar las cachés internas llega a un punto a partir del cual no mejora la tasa de aciertos de forma significativa. Por otro lado se puede incrementar el grado de paralelismo superescalar, añadiendo más unidades de ejecución, sin embargo la complejidad y la circuitería necesaria aumentan exponencialmente. Complejisdad que sirve para mejorar las

prestaciones de la máquina, ya que, como vimos en el tema anterior en la implementación superescalar se pueden emitir 5 o más instrucciones en no más del 10 % de los casos(En las condiciones más favorables).

Figura 7: Procesador Itanium (ARQUITECTURA VLIW)

Para solucionar el problema, Intel y HP han apostado por la implementación de un gran número de unidades funcionales en paralelo, pero utilizando el enfoque de paralelísmo explicito. En esta estrategia, el compilador planifica estáticamente las instrucciones en tiempo de compilación, en lugar de que lo haga dinámicamente el procesador en tiempo de ejecución. Así es el compilador el que determina qué instrucciones pueden ejecutarse en paralelo, e incluye esa

Así Intel y HP lanzarón en 2002 un microprocesador tipo VLIW, el IA-64. El Itanium 2 es un procesador que usa direcciones de memoria de 64 bits y está basado en el modelo EPIC. Los procesadores Intel Itanium 2 representan el diseño de producto más complejo del mundo con más de 1.700 millones de transistores. Esto permite obtener sólidas capacidades de virtualización, mejorar la confiabilidad y niveles de rendimiento líderes del mercado.

Características de la arquitectura VlIW:

- Gran número de registros. El formato de instrucción de la arquitectura IA-64 permite la utilización de hasta 256 registros de 64 bits, 128 de uso general para datos enteros y lógicos y 128 para datos en coma flotante y gráficos. También dispone de 64 registros predicado de un bits.

- Múltiples unidades de ejecución. Puede utilizar muchas más unidades paralelas que las máquinas superescalares típicas, de paralelismo implícito a nivel de instrucciones, que suelen soportar cuatro cauces. Típicamente se usan ocho o más unidades funcionales.

P. ej., Itanium 2 tiene 11 unidades funcionales diferentes.

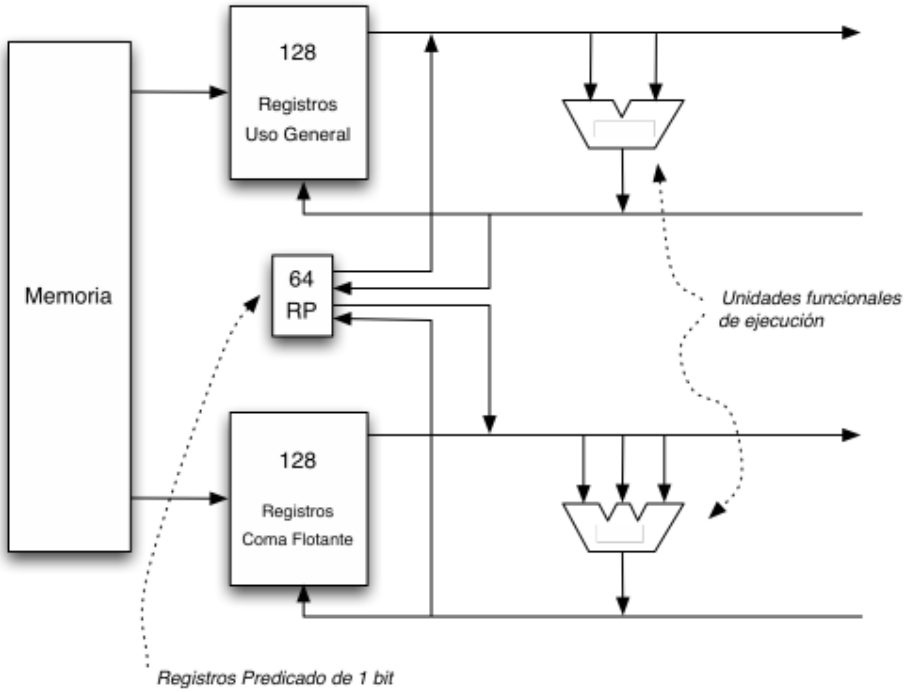

Figura 8: Diagrama de bloques para una arquitectura VLIW

-FORMATO DE INSTRUCCIÓN-

Las instrucciones de la arquitectura IA-64 responden al tipo VLIW, very long instruction word, ya que cada instrucción es en realidad un paquete de 128 bits que contiene 3 instrucciones de 40 bits más un campo plantilla de 8 bits.Figura de abajo

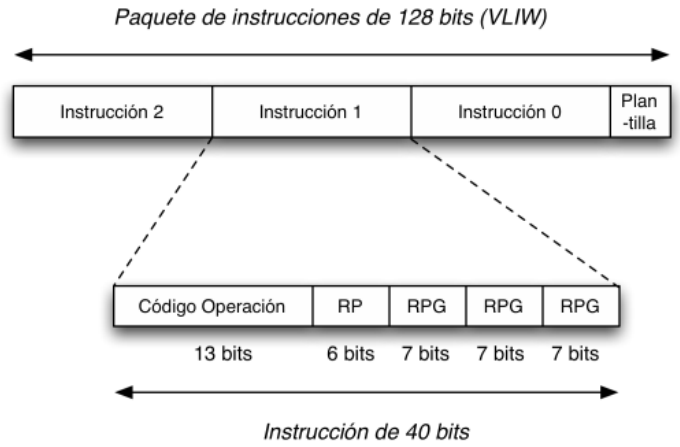

Figura 9

RP: Registro Predicado de 1 bit

RG: Registro de uso general

El campo plantilla indica qué instrucciones pueden ejecutar en paralelo, de forma que su interpretación no se limita a un solo paquete. Así, si por ejemplo el flujo de instrucciones permite la ejecución de ocho instrucciones en paralelo, el compilador las reordenará para que abarquen paquetes contiguos, y ajustará los bits de plantilla para informar al procesador de cuales son las instrucciones independientes. Las instrucciones agrupadas no tienen por qué estas en el orden original.

El campo plantilla proporciona una gran flexibilidad. El compilador puede mezclar instrucciones dependientes e independientes en el mismo paquete, con tán solo indicarlo en el campo plantilla. Esto no ocurre en otros diseños VLIW, que necesitan insertar instrucciones NOP hasta completar el paquete para evitar que instrucciones dependientes compartan el mismo paquete.

La figura 9 muestra el formato de cada una de las instrucciones del paquete. Las instrucciones son de 40 bits, lejos de los 32 bits típicos de las instrucciones RISC. Esto es debido a que un mayor número de registros implica más bits para codificarlos y además, se añade la referencia a un campo predicado. Este campo es de utilidad ya que se necesitara realizar la operación de ejecución con predicados, la cuál consiste en que la ejecución de instrucciones puedan referirse a registros predicados de un bit. Esto indica que la ejecución de dicha instrucción será especulativa, y una vez se sepan los valores de los registros predicado, la instrucción se retira si el valor final del registro predicado correspondiente es verdadero(valor 1, booleano TRUE), y se descarta si es falso. Por otro lado, todas aquellas instrucciones sin asignación de predicado se ejecutan incondicionalmente.

2.6 ARQUITECTURA EPIC

Procesamiento de instrucciones explícitamente en paralelo (del inglés EPIC: Explicitly Parallel Instruction Computing) es un paradigma de programación que comenzó a investigarse a principios de los años 80 y se convirtió en una patente estadounidense 4,847,755 (Gordon Morrison, et. al).

Este paradigma también se conoce como arquitecturas de Independencia. El objetivo de EPIC era aumentar la capacidad de los microprocesadores para ejecutar instrucciones de software en paralelo mediante el uso del compilador, en lugar de la compleja circuitería en cápsula (die), para identificar y aprovechar las oportunidades para la ejecución en paralelo. Esto permitiría escalar el rendimiento más rápidamente en los futuros diseños de procesadores, sin tener que recurrir a frecuencias de reloj cada vez más altas, las cuales se han convertido desde ese momento en una problemática importante debido a problemas de alimentación y refrigeración.

Las arquitecturas EPIC añaden numerosas características para paliar las deficiencias de VLIW:

- Cada grupo de instrucciones de software múltiples recibe el nombre de paquete (bundle en inglés). Cada uno de los paquetes contiene información que indica si algún paquete subsiguiente depende de este conjunto de operaciones. Con esta habilidad, se pueden construir implementaciones posteriores que envíen varios paquetes en paralelo. El compilador se encarga de calcular la información de dependencia de modo que libere al hardware de la tarea de ejecutar comprobaciones sobre la dependencia de operandos.
- Se utiliza una instrucción de carga especulativa como un tipo de búsqueda anticipada de datos. Esta búsqueda anticipada aumenta las probabilidades de éxito de encontrar los datos requeridos en la caché primaria para las cargas normales.
- Así mismo, una instrucción de carga de comprobación ayuda a las cargas especulativas mediante la comprobación de que una carga no depende de un almacenamiento anterior.

La arquitectura EPIC también incluye un cajón desastre de conceptos sobre arquitecturas para aumentar el ILP (paralelismo en el nivel de instrucciones):

- Se utiliza la ejecución de saltos predicados para reducir el número de ocurrencias de los saltos y para incrementar la ejecución especulativa de instrucciones. Gracias esta característica, las condiciones de los saltos se convierten en registros de predicados que se utilizan para terminar los resultados de instrucciones ejecutadas desde la parte del salto que se desprecia.
- Las excepciones retardadas (mediante el uso de un bit Not-A-Thing, Negación de algo, dentro de los registros de propósito general) también permiten una mayor ejecución especulativa incluso después de posibles excepciones.
- El uso de archivos de registro de arquitectura muy amplios elimina la necesidad de renombrar los registros.
- Instrucciones de salto de múltiples direcciones.

Figura 10: Microprocesador bajo arquitectura VLSI

3. CONCLUSIONES

Podemos observar como la tecnología de un computador esta directamente relacionanada con la arquitectura que use. Además el tipo de función que requiera un computador también se vera influenciado por el tipo de arquitectura con la que se implemente.

Por último decir que hoy en día la evolución de los microprocesadores, debido a que la miniaturización y aprovechamiento máximo del espacio esta llegando a su fin debido a los límites de la física. Por lo tanto el campo de la arquitectura de ordenadores se esta conviertiendo en un campo de estudio cada vez en má aumento para aprovechar el espacio de una forma más eficiente y así conseguir que los transistores instalados en un microprocesador puedan organizarse de una forma más óptima.

4. BIBLIOGRAFÍA Y/O REFERENCIAS

[1] M.Morris Mano. Arquitectura de ordenadores

[2] Ignacio Angulo Martinez y otros. Arquitectura de microprocesadores, los pentium a fondo.

[3]Jean-Loup Baer. Microprocessor Architecture From Simple Pipelines to Chip Multiprocessors.

5. ENLACES DE INTERÉS

- **http://library.thinkquest.org/26207/class/class1.html**
- **http://www.tu-plovdiv.bg/en/**