Tema 4

Jerarquía de niveles estructurales

código máquina: es código binario puro que se ejecuta en la CPU: difícil de manejar poe le programador
Lenguaje ensamblador: es un lenjuage de programación de bajo nivel con correspondencia casi directa con el lenguaje máquina
add 7,3, $6 # registro 7 ⇐ registro 3 + registro 6

Un computador funciona ejecutnado continuamente instrucciones que operan sobre diferentes datos
Instrucciones y datos se almacenan en memoria y se codifican emplenado un número específico de bits
El computador está diseñado para transferir bloques de información de un determinado tamaño

- A cada uno de estos bloques se le denomina palabra
- El tamaño, en número de bits, de cada palabra se denomina ancho de palabra
  Todos los elementos del computador (registros, buses, memorias, periféricos, ALUs) están diseñados para transferir, almacenar o procesar palabras

Ejecución de instrucciones

El proceso de ejecución de una instrucción sigue una secuencia de 4 pasos:

1. Lectura o carga de la instrucción: memoria → registro de instrucción (IR, Instruction Register).
2. Decodificación de la instrucción: La unidad de control lee el contenido del IR y decodifica la instrucción a ejecutar.
3. Ejecución de la instrucción: Se activan las señales de control necesarias para la ejecución de la operación.
4. Determinación de la siguiente instrucción: actualizando el contador del programa (PC, Program Counter).

El funcionamiento de la CPU está determinado por las instrucciones que ejecuta
EL conjunto de instrucciones distintas que puede ejecutar se denomina repertorio de unstrucciones
El repertorio de instrucciones es la parte de la arquitectura de un computador relacionada con la

programación y condiciona la implementación
Distintos diseños de microprocesadores pueden tener el mismo repertorio de instrucciones (ejemplo: Intel Pentium y AMD Athlon)
Cada instrucción debe contener toda la información necesaria para su ejecución

Instrucciones máquina

- Son las instrucciones que ejecuta directamente la CPU
- El lenguaje máquina (no el lenguaje ensamblador) es el nivel más bajo de programación.
  Es código binario
  Depende completanmente del hardware
- Tipos de instrucciones:
  - De transferencias de datos: entre memoria y registros u operaciones con periféricos (E/S)
    - De operaciones aritméticas y lógicas
    - De control de flujo
  - Instrucciones de bifurcación y salto condicional
  - Instrucciones de llamada a subrutina
  - De control del sistema y otras
- Los elementos constitutivos de una instrucción máquina son:
  - Código de operación
  - Referencia a operandos fuente: uno o más
  - Referenica al operando resuelto: si produce alguno
  - Referencia a la siguiente instrucción: en la mayoría de los casos es implícito y se refiere a la siguiente instrucción de la lista. En caso contrario, se suministrará la dirección de memoria.
- Los operadores fuente y resultado estarán en alguna de las siguiente localizaciones:
  - Memoria: deberá indicarse la dirección
  - Registro de la CPU: identificados por un número único
  - Dispositivos de E/S: módulo y dispositivo E/S o dirección de memoria en caso de E/S asignada en memoria
- La instrucción se divide en campos correspondientes a los elementos constitutivos de la misma (código de operación, operandos, etc.)
- La división en cmapos y bits se denomina formato de instrucción
  - 4 bits Codop
  - 6 bits referencia a operando
  - 6 bits referencia a operando
  - 16 bits en total
- La mayoría de los repertorios de instrucciones emplean más de un formato

- Los aspectos más importantes en el diseño del repertorio de instrucciones son:
  - Repertorio de operaciones: Cuántas y qué operaciones considerar y cuán complejas deben ser.
  - Tipos de datos: Los distintos tipos de datos con los que se efectúan operaciones.
  - Formatos de instrucciones: Longitud de la instrucción (en bits), tamaño de los distintos campos, etc.
  - Registros: Número de registros de la CPU que pueden ser referenciados por unstrccuones y su uso.
  - Direccionamiento: El modo o modos de direccionamiento mediante los cuales puede especificarse la dirección de un operando.

Estrategias de diseño

- Dos filosofías a la hora de diseñar un repertorio de instrucciones:
  CISC (Complex Instruction Set Computer)
  - Juego de instrucciones muy rico
  - Modos de direccionamiento potentes y complejos
  - Programas con menos instrucciones, pero de ejecución compleja
    - RISC (Reduces Instruction Set Computer)
  - Número relativamente pequeño de instrucciones disponibles
  - Sencillas y con pocos modelos de direccionamiento
  - Programas con más instrucciones, pero de ejecución muy eficiente
- Estudiaremos el repertorio del MIPS (Microprocessor without Interlocked Pipeline Stages)
  Diseño pionero RISC presentado en 1981
  Usado durante años en supercomputadores y que en la actualidad se utiliza fundamentalmente en sistemas empotrados
  Empleado por NEC, Nintendo, SGI, Sony,...

Modelo de ejecución

- El modelo de ejecición especifica el dispositivo (memoria, registro, etc.) que almacena los operandos de las instrucciones.
- Una misma arquitectura suele dar soporte a varios modelos de ejecución
- Los modelos soportados dependen fuertemente de la arquitectura.
  - Por ejemplo una ALU exclusivamente conectada con registros quedará limitada al modo registro-registro.

    - Los modelos de ejecución posibles son:
        - Modelo de pila: POP y PUSH
        - Modelo registro-registro: el modo más rápido, Se necesita - trasferir los datos.
        - Modelo resgitro-memoria: un operador en registro, otro en memoria. EL resultado se almacena en un registro.
        - Modelo memoria-memoria: No se necesita transferir los datos. La ejecución requiere un alto número de ciclos.

Modos de Direccionamiento

- Son los algoritmos empleados por el procesador para calcular las direcciones de las instrucciones y datos.
- La dirección real de memoria especificada por el modo de direccionamiento se denomina dirección efectiva.
- Modos de direccionamiento:
    - Direccionamiento inmediato.
    - Direccionamiento directo.
    - Direccionamiento indirecto.
    - Direccionamiento de registro.
    - Direccionamiento indirecto con registro.
    - Direccionamiento con desplazamiento:
        - Desplazamiento relativo
        - Desplazamiento con registro-base.
        - Indexado

Modos de Direccionamiento

/// Meter fotos de la diapo 17

- El operando es una constante cuyo valor se almacena en el campo operando de la instrucción.
- Ventaja: no se requiere una referencia a memoria para obtener el operando.
  Desventaja: ek tamañi dek número está limitado por la longitud del campo de direcciones.
  /// Meter fotos de la diapo 18
- EL opereando se encuentra almacenado en memoria en la dirección indicada por el campo operando.
- Limitación: proporciona un espacio de direcciones reducido
  /// Meter fotos de la diapo 19

- El campo operando de la instrucción contiene la dirección a una posición de memoria que a su vez contiene la dirección del operando deseado.
- Desventaja: la ejecución de una instrucción requiere dos diferencias a memoria principal.
  /// Meter fotos de la diapo 20
- El operando referenciado se encuentra en un registro.
- El número de registro se especifica en el campo operando de la instrucción.
- Ventajas:
  - Sólo es necesario un campo pequeño de direcciones en la instrucción.
  - El tiempo de acceso a un registro es mucho menor que a memoria.
- Desventaja: espacio de direcciones muy limitado.

Direccionamiento indirecto con registro

/// Meter fotos de la diapo 21

- El operando de la instrucción contiene un número de registro que contiene la dirección de memoria del operando deseado.
- Ventaja: Emplea una referencia a memoria menos que el direccionamiento indirecto.

Direccionamiento con desplazamieto

/// Meter fotos de la diapo 22

- El campo operando contiene una dirección relativa o desplazamiento(D).
- La instrucción especifica otra dirección (usualmente un registro) donde se almacena información del direccionamiento.
- La dirección efectiva se calcula como (D+R).
  /// Meter fotos de la diapo 23
- Desplazamiento relativo:
  - El registro referenciado (implícitamente) es el contador de programa.
  - La dirección efectiva es por tanto un desplazamiento relativo a la dirección de la instrucción.
    /// Meter fotos de la diapo 24
- Direccionamiento con registro-base:
  - El registro referenciado (implícita o explícitamente) contiene una dirección de memoria.
  - El cmapo de dirección contiene un desplazamiento positivo o negativo desde esa dirección.
    /// Meter fotos de la diapo 25
- Indexado:
  - El operando referencia una dirección de memoria.

    - El registro referenciado contiene un desplazamiento positivo desde esa posición.
      /// Meter fotos de la diapo 26
      /// Meter fotos de la diapo 27

Arquitectura del MIPS R2000

Características

- Longitud de palabra fija de 32 bits
- Direcciona bytes individuales: para indicar la dirección de una palabra en memoria hay que dar la dirección del primer byte
    - 2 palabras consecutivas estarán separadas en 4 unidades
    - En las opreaciones de transferencia entre memoria y registro las direcciones de memoria han de ser múltiplos de 4: **restricción de alineamiento**
    - Las direcciones múltiplos de 4 se llaman direcciones alineadas
    - El intento de acceso a una unidad mayor de un byte en una dirección no alineada produce **un error de bus**

Orden de los bytes en una palabra

- Representación de los bytes en una palabra
    - **Big-endian**: la dirección del dato es el byte más significativo del mismo.
    - **Little-endian**: la dirección del dato es el byte menos significativo del mismo
    - **Bi-endian**: configurable (MIPS)
      /// Meter fotos de la diapo 29

Tipos de instruccines MIPS

- Tipos de instrucciones soportados:
    - Transferencia de datos
    - Aritméticas y lógicas
    - De control de flujo:
        - Salto condicional
        - Bifurcación
- Características:
    - Longitud de todas las instrucciones fija de 32 bits.
    - Los operandos de las operaciones aritméticas son siempre registros

- EL acceso a memoria se hace a través de operaciones de carga/almacenamineto (transferencia de datos).

Formatos de las instrucciones MIPS

/// Meter foto de la diapositivia 31

Instrucciones aritmético-lógicas:

- add, sub, or, and, ...
- Formato tipo R
- Direccionamiento de registro
  /// Meter foto de la diapositivia 32

Operaciones aritméticas con operandos inmediatos:

- addi, andi, ori, ...
- Formato tipo I
- Direccionamiento de registro + direccionamiento inmediato
  /// Meter foto de la diapositivia 33

Operacioens de transferencia:

- lw (load word) y sw (store word)
- Tipo I
- DIreccionamietno con desplazamiento
  /// Meter foto de la diapositivia 34

Operacioens de transferencia con operandos inmediatos:

- lui (load upper inmediate)
- Tipo I
- Direccionamiento inmediato
  /// Meter foto de la diapositivia 35

Instrucciones de salto condicional:

- beq (branch if equal) y bne (branch if not equal)
- Tipo I
- Direccionamiento con desplazamiento relativo

- beq rs,rt, despl salta a la instrucción que se encuentra en la dirección de memoria (PC + 4) + 4 * despl si $rs ==$rt.
- El ensamblador permite el uso de etiquetas para indicar la dirección de salto.
- Para jacer comparaciones del tipo *menor que* tenemos la instrucción lógca slt
- stl rd,rs, rt :rf = ($rs <$rt)
- stli rd,rs, inm: $rd =$ (rs < inm)
  /// Meter fotos de la diapositiva 38
- slt: Tipo R, direccionamiento de registro
- slti: Tipo I, direccionamiento de registro + inmediato

Instruccioens de bifurcación

- j (jump): Tipo J, direccionamiento pseudodirecito
  - PC ← $(PC + 4)_{31-28}IR_{25-0}00$
  - El emsamblador permite el uso de etiquetas
- jr (jump register): Tipo R, direccionamiento indirecto con registro.

Llamadas a subrutinas

- Subrutina: es una secuancia de instrucciones de un programa que realiza una tarea específica y que está empaquetada como una unidad
- Es una herramienta para la estructuración de programas
  - Aumenta la legibilidad del código
  - Permite su reutilización.
- Pasos para la llamada a una subrutina:
  - Situar los parámetros en un lugar donde la subrutina pueda acceder a ellos.
  - Transferir el control a la subrutina. Adquirir los recursos de almacenamiento necesarios para el procedimiento. Realizar la tarea deseada. Situal el valor del resultado en un lugar donde el programa que lo ha llamado pueda acceder a él
  - Retornar el control al punto de origen

Distribución de memoria en MIPS

/// Meter foto de la diapositiva 42

- Generalmente el invocador de la subrutina sitúa los parámetros en la pila.
- La subrutina colocará en la pila la infromación a guardar temporalmente.
- Al finalizar, liberarádicho espacio y dejará en la pila el resultado de la subrutina.
- Por útlimo, el invocador recupera la pila el resultado.

Paso de parámetros a través de la pila

- Pila: espacio de memoria con estructura LIFO.
- El registro $sp apunta a la dirección más reciente utilizada.
- Crece de las direcciones de memoria superiores hacia las inferiores.
- Ejemplo de utilización
  - Push salvando los registros en la pila:
    addi sp,sp, -8
    sw $v0, 0$(sp)
    sw $v1, 4$(sp)
  - Pop recuperando los datos de la pila a los registros:
    lw $v0, 0$(sp)
    lw $v1, 4$(sp)
    addi sp,sp, 8

Paso de parámetros a través de registros

- El acceso a los registros es más rápido que a memoria.
- El convenio de llamadas en MIPS se apoya en los registros para evitar accesos a pila:
  - $a0-$a3: paso de argumentos.
  - $v0-$v1: retorno de valores.
  - $ra: dirección de retorno.
- Si se necesita pasar o devolver más argumentos, será necesario utilizar la pila.

Instrucción de salto

- jal (jump and link): Salta a una dirección y simultáneamente salva la dirección de retorno ($ra = PC+4).
- Tipo J.
- Direccionamineto pseudodirecto. EL ensamblador permite el uso de etiquetas.
- EL salto de retorno se hará a través de jr $ra.
- Uso:
  - El invocador pone los parámetros en $a0-$a3 y llama a jal.
  - La subrutina realiza los cálculos , pone los resultados en $v0-$v1 y devuleve el control
  - Si se necesita pasar más de cuatro argumentos o devolver más de dos valores se utilizará la pila.

Salvaguarda de regustros

- Si la subrutina modifica los registros utilizados por la rutina invocadora, los valores debrán ser guardados y restaurados antes y después, repsectivamente, de la ejecución de la subrutina.
- Dos convenios estándar.
  - Guardar invocador (*caller save*).
  - Guardar invocado (*callee save*).
- Para evitar guardar y restaurar valores irrelevantes se usan las siguentes convenciones:
  - Registros temporales ($t0-$t9): no son preservados por el invocado. Es responsabilidad del invocador preservar su valor inicial si fuera necesario. Registros salvados ($s0-%s7): si el invocado los usa deberá salvar previamente su valor.
- La Unidad Central de Proceso (CPU) es la encargada de la ejecución de las instrucciones.
- La CPU se divide en ods bloques fundamentales:
  - Camino de datos: realiza las operaciones requeridas por las instrucciones máquina. Contiene:
    - Unidad aritmético-lógica.
    - Banco de registros.
    - Registros especiales: IR, PC, etc.
    - Buses internos: caminos de conexión entre los elementos de la CPU.
  - Unidad de control: gestiona el secuencionamineto de las operaciones ejecutadas en el camino de datos.
- La ejecución de una instrucción es un proceso dividid en 5 pasos
  i. **Captar la instrucción**: el procesador debe decodificarse para determinar qué acciçon es necesaria.
  ii. **Interpretar la instrucción**: la instrucción debe decodificarse para determinar qué acción es necesaria.
  iii. **Captar datos**: la ejecución puede exigir leer datos de la memoria o de un módulo de E/S.
  iv. **Procesar datos**: la ejecución de una instrucción puede exigir llevar a cabo alguna operación aritmética o lógica.
  v. **Escribr datos**: los resultados de la ejecución pueden tener que ser escritos en la memoria o en un módulo de E/S.

Introducción

- Estudiaremos un diseño sencillo de una CPU (camino de datos y unidad de control) capaz de ejecutar el siguiente subconjunto del repertorio de instrucciones MIPS:
  - Instrucciones de transferencia de datos: lw y sw.
  - Instrucciones aritmético-lógicas: add, sub, and, or y slt.

    - Instrucciones de control de flujo: beq y j.
- Comenzaremos por la construcción del camino de datos
    - Utilizaremos módulos digitales combinacionales y secuenciales
    - Para los módulos secuenciales aumiremos sincronización por flancos.

Construcción del camino de datos

Carga de la instrucción

// meter foto de la diapositiva 67

- La dirección de memoria en la que se localiza la instrucción a cargar la indica el PC.
- La siguiente instrucción se calcula incrementando el PC en 4 bytes.

Instrucciones aritmético-lógicas

// meter foto de la diapositiva 68

- Instrucciones tipo R que involucran tres resgistros: dis de lectura y uno de escritura.
- La operación se realiza a través de la ALU.
- Ejemplo típico: add $t1$,t2, $t3

// meter foto de la diapositiva 69

- Banco de registros:
    - Registros de 32 bits-
    - Dos puertos de lectura y uno de escritura.
    - Para escribir es necesario activar la señal EscribirReg.
    - Entradas (direcciones a leer/escribir): 5 bits.
    - Salidas: 32 bits (tamaño de palabra).

Unidad artmético-lógica

// meter foto de la diapositiva 70

La implementación de esta ALU se estudií en el tema de sistemas combinacionales

Instrucciones de transferencia de datos

// meter foto de la diapositiva 71

- Instrucciones tipo I
- Modo de direccionamiento: registro-base
- Ejemplos típicos:
  - lw $t1, desp1$(t2)
  - sw $t1, desp1$(t2)

// meter foto de la diapositiva 72

- Se utiliza la ALU para el cálculo de la dirección de memoria.
- El campo desplazamiento de la instrucción se extiende de 16 a 32 bits antes de realizar la suma.
- Memoria de datos:
  - Entradas:
    - Señales de control de lectura y escritura.
    - Dirección a acceder (32 bits).
    - Palabra a escribit (32 bits).
  - Salidas:
    - Palabra leída (32 bits).

Instrucción de salto condicional

// meter foto de la diapositiva 74

- Instrucción de tipo I: beq $t1$,t2, despl
- Evalúa si el contenido de $t1 y$t2 son iguales
- SI la condición se cumple se realiza el salto

// meter foto de la diapositiva 75

- Comparación:
  - Se realiza una resta y se utiliza la salida Cero de la ALU como marcador de igualdad
    // meter foto de la diapositiva 76
- Dirección de salto:
  - Salto relativo al PC+4 con signo
  - Dirección de salto: (PC+4)+4*\despl
  - Los 16 bits del desplazamiento se extiende a 32 y se desplazan 2 posiciones a la izquierda

Instrucción de salto incondicional

// meter foto de la diapositiva 77

- Tipo J con direccionamineto pseudodirecto: j L1.
- Reemplaza los 28 bits de menor peso del PC con los 26 bits de menor peso de la instrucción desplazados 2 posiciones a la izquierda
- No es necesario la introducción de hardware adicional

Camino de datos completo

- Formado por la combinación de elementos necesarios por cada instrucción
- Cada una de las 5 etapas de ejecución de instrucciones se realiza en un ciclo de reloj diferente:
  - Se puede reutilizar una misma unidad funcional dentro de la misma instrcción si se utiliza en diferentes ciclos
    - No se necesita una memoria para instrucciones y otra para datos ya que el acceso a instrcciones y datos es en ciclos diferentes
    - Oara el cálculo de la dirección de la siguiente instrucción (PC+4 o dirección de salto) podemos uar la ALU en lugar de los sumadores
- Los datos que vayan a ytilizarse en los siguientes ciclos deben almacenarse en un elemento de almacenamiento (memoria o registro)
  - Los datos a utilizar por las siguientes instrucciones en ciclos posteriores se almacenarán en elementos de almacenamiento visibles al programador (banco de registros, PC o memoria)
  - Los datos a utilizar por la misma intrucción en cuclos posteriores se almacenarán en registros temporales

Visión de alto nivel del camino de datos completo

// meter foto de la diapositiva 80

- Una sola ALU en vez de 1 ALU y dos sumadores
- Una memoria única para instrucciones y datos
- Registros temporales tras cada unidad funcional para almacenar su salida hasta que ese valor se utilice en el siguiente ciclo

// meter foto de la diapositiva 81

- Operaciones de la ALU:
  - Carga de la instrucción Suma PC+4.

    - Instrucción de tranferencia de datos: cálculo de la dirección a acceder.
    - Instrucción tipo R: Operación aritmética o lógica.
    - Salto condicional:
        - Evaluación de la condición de salto.
        - Cálculo de la direwcción de destino.

          // meter foto de la diapositiva 82
- Registros temporales:
    - Registro de instrucción (IR) y regsitro de datos de memoria (MDR):
        - Almacenan la instrucción o dato leído de memoria respectivamente.
    - Registros A y B: operandos leídos del banci de registros.
    - Salida ALU: salida de la ALU.
    - El regustro IR requiere de una seña de control de escritura (necesita almacenar su valor más de un ciclo).

      // meter foto de la diapositiva 83
- Se introducen multiplexores para selección de entradas:
    - Origen de direccion de acceso a memoria:
        - Salida de la ALU (acceso a datos).
        - PC (acceso a instrucciones).
    - Registro de destino:
        - Instrucciones tipo R: rd (bits 11-15).
        - Instrucciones tipo I (cargas): rt (bits 16-20).
    - Dato a escribir:
        - De memoria (carga de datos).
        - ALU (instrucción aritmético-lógica).
    - Primera entrada de ALU:
        - registro A (acceso a datos e instrucción aritmético.lógica).
        - PC (carga de instrucción y saltos).
    - Segunda entrada de ALU:
        - Registro B (instrucción aritmético-lógica y salto condicional).
        - Constante 4 (incremento del PC).
        - Campo desplazamiento extendido (transferecnia de datos).
        - Campo desplazamiento extendido y desplazado (saltos condicionales).
    - La dirección de la siguiente instrucción a ejecutar:
        - Resultado ALU: PC+4
        - Salida ALU: La dirección de salto calculada por la ALU en caso de salto condicional

          // meter foto de la diapositiva 87

Carga de una instrucción

// meter foto de la diapositiva 88

1. Se direcciona la memoria utilizando el PC.
2. Se efectua la lectura y se escribe en el registro de instrucción.
3. Se incrementa el PC en 4 bytes.
4. Sincronización por flancos: el incremento del PC no será visible hasta el siguiente ciclo.

Instrucciones aritmético-lógicas

// meter foto de la diapositiva 89

1. Lectura de los regstros indicados por rs y rt y almacenamiento en los registros temporales A y B.
   // meter foto de la diapositiva 90
2. Ejecución por parte de la ALU de la operación especificada en el código de operación y almacenamiento del resultado en el registro de SalidaALU.
   // meter foto de la diapositiva 91
3. Escrutura del resultado en el registro especificado por rd.

Instrucción lw

// meter foto de la diapositiva 92

1. Lecura del registro base y almacenamiento en el regitro temporal A.
   // meter foto de la diapositiva 93
2. Cálculo de la dirección de acceso como registro base más desplazamiento.
   // meter foto de la diapositiva 94
3. Acceso a memoria y carga del dato leído en el registro MDR.
   // meter foto de la diapositiva 95
4. Escritura del valor leído en blanco de registros,

Instrucción sw

// meter foto de la diapositiva 96

1. Carga en paralelo del registro base (registro temporal A) y dato a almacenar (registro temporal B).
   // meter foto de la diapositiva 97
2. Cálculo de la dirección de acceso como registro base más desplazamiento.
   // meter foto de la diapositiva 98

3. Acceso a memoria y almacenamiento del dato en el registro temporal B.

Instrucción beq

// meter foto de la diapositiva 99

1. Carga en paralelo de los operandos a comparar (registros temporales A y B). Al mismo tiempo, cálculo de la dirección de destino del salto.
   // meter foto de la diapositiva 100
2. Resta de los operandos. La señal de Cero de la ALU se utilizará para activar la escritura en el PC. Si se cumple la condición se escribe el PC con el nuevo valor.

Instrucción j

- El único paso a realizar es la escritura del PC con la dirección de salto.
- Serían precisos tan solo dos ciclos para su ejecución, pero como veremos se ejecutarán en 3 para simplificar la unidad de control

Señales de control

- Señales de control de escritura: PC, IR, memoria, banco de registros.
- Señal de lectura para la memoria.
- MUXes:
  - 2 entradas: 1 línea de control.
  - 4 entradas: 2 líneas de control.
- Unidad de control de la ALU:
  - Código de función + ALUop = 3 bits de control.

Unidad de control de la ALU

- 00: Usado en instrucciones lw, sw y beq (para calcular el destino del salto).
- 01: Usando en beq para determinar la igualdad de los operandos.
- 10: En instrucciones tipo R, la operación a realizar depende del campo de función de la instrucción.
- ALUop, SelALUB y Fuente PC son de 2 bits.
- El resto son de 1 bit.
- Los único registros que requiere señal de escritura son IR y PC.

Camino de datos y unidad de control

- Condiciones de escritura del PC:
  - Carga de la instrucción - Salto incondicial (EscrPC = 1).
  - Salto condicional con igualdad de operandos (EscrPCCond = 1 y Cero ALU = 1).

Diseño de la unidad de control

- La Unidad de Control debe especificar:
  - Las señales que se van a activar en caa paso
  - El paso siguientes de la secuencia
- Es un sistema secuencial síncrono:
  - Cada estado dura un ciclo de reloj y representa una etapa de la ejecución de una instrucción.
  - Entradas: los bits de la instrucción a ejecutar y los indicadores internos (p. ej. la salida Cero de la ALU).
    Salidas: las señales de control a activar.
- Los dos primeros estados son comunes a todas las instrucciones:
  - Carga de la instrucción.
  - Decodificación.
- El resto dependen del tipo de instrucción y sus operandos.
- Tras terminar la ejecución de una instrucción el autómata vulve al estado inicial para procesar la siguiente instrucción

Etapa 2 - Decodificación y búsqueda de los registros

- Se decodifica la instrucción
- Se adelantan acciones que puedan ser útiles
  - Se leen los registros rt y rs y se almacenan en A y B.
  - Se calcula la dirección de salto potencial (beq).

Etapa 3 (lw o sw). Cálculo de la dirección de memoria

- SelALUA = 1: Entrada A de la ALU es el registro A (registro base).
- SelALUB = 10: Entrada B es el campo de desplazamiento con el signo extendido
  ALUop = 00: suma.

Etapa 4 (lw). Acceso a memoria

- LeerMem: Activar la memoria memoria para lectura.
- IoD = 1: Memoria direccionada por SalidaALU.
- El registro MDR no necesita señal de activación.

Etapa 5 (lw): Escritura del dato MDR al banco de registros

- EscrReg: Activar la escritura en el banco de registros.
- Mem2Reg = 1: El dato a escribir proviene del MDR.
- RegDest = 0: El registro destino proviene de los bits 16-20 de la instrucción (campo rt).

Etapa 4 (sw): almacenamiento del dato en memoria

- EscrMem: Activar la escritura en memoria
- IoD = 1 Memoria direccionada por SalidaALU.

Instrucciones de salto condicional (beq)

Etapa 3: Comparación y determinación del nuevo PC

- Comparación:
  - SelALUA = 1: Entrada A de la ALU es el registro A (rs).
  - SelALU = 00: Entrada B de la ALU es el registro B (rt).
  - ALUop = 01: Resta.
- Nuevo PC:
  - EscrPCCond: Activa la escritura del PC siempre y cuando, además la salida Cero de la ALU sea 1.
  - FuentePC = 01: El nuevo PC se encuentra en SalidaALU.

Instrucciones de salto incondicional (jump)

Etapa 3: Se escribe el nuevo PC

- EscrPC: Activar la escritura incondicional del PC.
- FuentePC = 10: El nuevo PC se obtiene tras desplazar dos bits a la izquierda los bits 0-25 de la instrucción, y concatenarlos con los 4 bits más significativos del PC.

Autómata de control complejo

- Dos técnicas diferentes de implementación:
  - Unidad de control cableada: se basa en alguno de los métodos clásicos de diseño
  - Unidad de control microprogramada: el autómata de control se representa en forma de programa de control
    - Más fácil de depurar, modificar y ampliar
    - Mejor opción para sistemas grandes y/o complejos

Control microprogramado

- El autómata se transforma en un programa de control
- Las instrucciones del programa de control se denominan microinstrucciones
- Cada microinstrucción define el conjunto de señales de control que se deben activar en un estado determinado
- El conjunto de las microinstrucciones se denomina microprogramada
- El microprograma se almacenará en una memoria de solo lectura

2 formas:

- Existen 2 formas de realizar el secuenciamento:
  - Explícito: cada microinstrucción incluye la dirección de la microinstrucción siguiente
  - Implícito: existen dos tipos de microinstrucción, de control y de salto.

Secuenciamiento

- Vamos a implementar la UC microprograma para el conjunto básico de instrucciones MIPS considerando secuenciamiento explícito
- Cada estado del autómata de control se transformará en una microinstrucción
- Elección de la siguiente microinstrucción, tres opciones:
  - Incrementar la dirección de la microinstrucción actual
  - Saltar a la primera microinstrucción para iniciar la ejecíón de una nueva instrucción
  - Elegir la siguiente microinstrucción en función del estado en el que nos encontramos y la instrucción que estamos ejecutando
    - Tabla de envío: tabla con las direcciones destino
    - Una tabla para cada estado con múltiples estados destino

- Para la implementación de la UC del MIPS necesitamos dos tablas de envío
- Se necesitan 2 buts (CtrlDir) para especificar la siguiente microinstrucción
  - CtrlDir = 00: Salta a la primera microinstrucción para comenzar una nueva instrucción
  - CtrlDir = 01: Utiliza la tabla del envio 1
  - CtrlDir = 10: Utiliza la tabla del envio 2
  - CtrlDir = 11: Selecciona la siguiente microinstrucción

Implementación

- Cada punto de control del camino de datos se corresponde con un bit de las microinstrucciones
  - Cada microinstrucción consta de 18 bits
  - Los puntos de control podrían estar codificados para ahorrar bits

Temporización

Ciclo de reloj

//// Meter foto de la tabla de la 148

- El ciclo del reloj debería ser al menos tan largo como la más larga de las operaciones del camino de datos (Ejemplo Ciclo de reloj >= 35ns)
  - lw: 175 ns (cinco ciclos)
  - sw e instrucciones A-L: 140ns (cuatro ciclos)
  - beq u j: 105 ns (tres ciclos)

Procesamiento de excepciones

Excepciones

- Excepción: Suceso inesperado ocurrido en el procesador.
- Acciones a llevar a cabo ante una excepción:
  - Guardar la dirección de la instrucción causante en el registro contador de programa de excepción (EPC.)
  - Transferir el control al SO, que ejecutará alguna ruta específica.

    - Finalizar el programa o continuar su ejecución (utilizando EPC).
- Tipos de excepciones en nuestra implementación de MIPS:
    - Ejecución de una instrucción no definida.
    - Desbordamiento aritmético.
- La rutina del SO a ejecutar dependerá del tipo de excepción.
- En el MIPS se incluye un registro denominado Registro de Causa (Causa) para almacenar la razón de la excepción
    - Causa = 1: instrucción no definida
    - Causa = 0: desbordamineto aritmético
- EPC: registro de dirección de la instrucción causante (32 bits).
- Causa: Registro para almacenar la causa de la excepción (1 bit).
    - Causa = 1: Instrucción no definida.
    - Causa = 0: Desbordamineto aritmético.

Camino de datos modificado

/// Foto de la 153

- Señales de control:
    - EscrEPC.
    - EscrCausa
    - CausaInt: valor del registro Causa (1 bit).
- Nueva entrada al MUX del PC: 0xC0000000 (entrada al código del SO)

Procesamiento de una excepción

- Escribir en el registro Causa un 0 o un 1.
- Guardar la dirección de la instrucción causante (PC-4) en EPC.
- Poner en el PC la dirección de procesamiento de excepción del SO.

Unidad de control completa

//// foto de la 157

- Instrucción no definida:
    - Se detecta al no estar definido el estado siguiente al estado 1 para el código de operación encontrado.
    - El estado siguiente será el estado 11.

- Desbordamiento aritmético:
  - Se detecta cuando la ejecución de una instrucción arimético-lógica (estado 6) arroja un desbordamiento
  - El estado siguiente será el estado 10.

Microprogramación

- Modificaciones a la UC microprogramada:
  - 2 nuevas microinstrucciones (estados 10 y 11)
  - Modifican tabla envío 1
  - Nueva tabla de envío 3 para bifurcación estado 6
  - Microinstrucciones de tamaño 22 bits en vez de 18:
    - CrtlDir podrá tomar 5 valores diferentes: ampliación a 3 bits
    - Nuevas señales de control: CausaInt, EscrCausa, EscrEPC

Código de operación---Instrucción---Valor
000000----------Tipo R--------Rformat1
000010----------jump----------JUMP1
000100----------beq-----------BEQ1
100011----------lw------------Mem1
101011----------sw------------Mem 1
otro------------otra----------Nodef1

Señal de desbordamiento-----Valor
----------0----------Arti3
----------1----------Ov3

Mejoras del rendimiento en los procesadores actuales

- ## Hay dos posibilidades para incrementar el rendimiento:

  Mejor la tecnología de los circuitos integrados, lo cual reduciría el tiempo necesario para ejecutar cada una de las etapas y permitiría reducir el ciclo de reloj.
  - Mejorar el diseño del procesador:
    - Procesador Segmentado: Trabaja con varias instrucciones a la vez, cada una en una etapa diferente. Se pretende conseguir la ejecución de una instrucción por ciclo (límite máximo).

        - Procesador Superscalar: Implementa paralelismo a nivel de instrcción. Ejecuta más de 1 instrucción por ciclo de reloj gracias a la replicación de unidades funcionales.
- Todas las CPUs de propósito general actuales son segmentadas y superscalares.

Tema 5

Introducción

- Dispositivos de E/S o periféricos: permiten la comunicación del computador con su entorno
- La conexiones entre los periféricos, el procesador y la memoria se llavan a cabo mediante los buses
- Para realizar la conexión entre la CPU y los periféricos se necesita de módulos de E/S debido a que:
    - Velocidad de transferencia periféricos << velocidad CPU o memoria
    - Ancho de palabra no coincide con ancho de palabra CPU
    - Existen una amplia variedad de dispositivos -> muy distintas formas de funcionamiento
      //Foto de la 4

Buses

- Bus: vía de comunicación que conecta 2 o más dispositivos
- Es un medio de transmisión compartido
- En un mismo instante de tiempo se permite la recepción por varios dispositivos pero solamente uno puede transmitir -> se necesitan mecanismos de arbitraje
  // foto de la 5

Módulos E/S

- Actúan como interfaz entre la CPU y uno o más dispositivos periféricos
    - Controlador de E/S: módulo de E/S básico
    - Procesador de E/S: módulo de E/S con capacidad para ejecutar programas
- Funciones:
    - Control y temporización
    - Comunicación con el procesador
    - Comunicación con los dispositivos

    - Almacenamiento temporal de datos
    - Detección de errores

Direccionamineto de los módulos de E/S

- La comunicación con los dispositivos de E/S se realiza mediante los registros
- Dos aproximaciones para los registros:
    - E/S asignada en memoria:
        - Un único espacio de direcciones en el sistema
        - Parte del espacio de direcciones se reserva para E/S
        - Mismas instrucciones para acceso a memoria y a dispositivos de E/S -> mismos modos de direccionamiento disponibles
    - E/S aislada:
        - Dos espacios de direcciones diferentes: memoria y E/S
        - Se utiliza una señal de control (M/IO) para indicar si la dirección en el bus de direcciones es de E/S o de memoria, o bien se usa un bus de E/S propio
        - Instrucciones especuales para operaciones de E/S

Gestión de la E/S

La comunicación entre los módulos de E/S y el procesador se puede hacer utilizando:

- E/S programada: La CPU toma la iniciativa de contactar con el dispotivo de E/S y gestiona las tranferencias de datos
- E/S mediante interrupciones: Los dispositivos de E/S realizan una petición de interrupción al procesador
- Acceso directo a memoria: Realizada por un controlador especializado que se encarga de la transferencia de datos sin intervención del procesador

E/S programada

- CPU tiene control completo y directo sobre la opeación de E/S
    - Envío de comandos de lectura y escritura
    - Comprobación del estado del dispositivo
    - Transferencia de datos
- Encuesta (polling): cada vez que la CPU envía un comando al dispositivo de E/S espera a que la opeación sobre su estado
- Velocidad procesador >> Velocidad E/S -> gran desperdicio de ciclos de reloj en las encuestas

- Es la técnica de gestión más simple y más ineficiente desde el punto de vista del procesador

E/S con interruptores

- El dispositivo avisa al procesador cuando está lista para la transferencia
- El aviso se produce activando una línea de petición de interrupción
- Interrupción: suceso inesperado que afecta al procesador. Al contrario que las excepciones:
  - Son suceso externos
  - Se pueden producir en cualquier momento
- La E/S con interrupciones permite al procesadore trabajar en otro proceso mientras se espera por una operación de E/S
- Pasos en el procesamiento de una interrupción:
  - El dispositivo envía una señal de interrupción al procesador
  - El procesador termina la ejecución de la instrucción en curso
  - El procesador envía una señal de reconocimiento al dispositivo que originó la interrupción
  - El procesador salva el PC y los registros necesarios en la pila
  - El procesador carga en el PC la dirección de la rutina de tratamiento de la interrupción solicitada (ISR, Interrupt Service Routine) y ejecuta esa rutina
  - Una vez que la rutina termina, el procesador restaura el estado guardado en la pila y retorna al programa que se estaba ejecutando
- Si más de un dispositivo puede generar la petición de una interrupción habrá que:
  - Identificar el dispositivo que generó la interrupción
  - Establecer prioridades
  - Proteger los servicios de interrupción de otras interrupciones

Identificación del dispositivo

- La identificación del dispositivo se puede realizar mediante:
  - Líneas de interrupción dedicadas: limita el número de dispositivos que se pueden conectar
  - Líneas de interrupción compartidas: la identificación del dispositivo se puede hacer usando:
    - Encuesta software: se consulta el valor del registro de estado de los diferentes dispositivos
    - Encadenamiento: la línea de reconocimiento de interrupción se conecta en cadena a los distintos dispositivos
    - Arbitraje de bus: solo con un módulo puede activar la línea de interrupción en un instante dado.
- Encuesta software

    - Cuando se activa la línea de interrupción la CPU ejecuta una rutina genérica de manejo de interrupciones
    - La rutina se encarga de interrogar a los diferentes dispositivos comprobando sus registros de estado
    - Una vez identificado el dispositivo se ejecuta la rutina de tratamiento específica
- Encadenamiento:
    - Cuando se activa la línea de interrupción, la CPU activala línea de reconocimiento de interrupción que se conecta en cadena a todos los módulos de E/S
    - El primer módulo que recibe la señal y ha generado una interrupción deja de propagar la señal y envia la dirección de su rutina de tratamiento por el bus de datos
- Diferentes modos de direccionamiento para especificar la dirección:
    - Direccionamiento absoluto: se manda la dirección completa
    - Direccionamineto relativo: se envia parte de la dirección
    - Direccionamineto relativo indirecto (interrupciones vectorizadas): manda la posición relativa a una tabla de direcciones residente en memoria
- Arbitraje de bus:
    i. El módulo de E/S se hace con el bus antes de activar la línea de interrupción
    ii. La CPU activa la línea de reconocimiento de interrupción
    iii. El módulo de E/S sitúa la dirección de la rutina de tratamiento en el bus de datos

Prioridades

- Líneas de interrupción dedicadas: el procesador selecciona la línea con mayor prioridad
- Líneas de interrupción compartidas:
    - Consulta software: el orden en el que se consulta establece la prioridad
    - Encadenamiento: el orden en que se encadenan establece la prioridad
    - Arbitraje de bus: en el propio mecanismo de arbitraje se pueden establecer prioridades

Interrupciones anidadas

- Sistema de interrupciones único: La ejecución del programa de servicio de una interrupción contunúa hasta el final antes de aceptar una segunda petición de interrupción
- Sistema de interrupciones multinivel
    - A cada causa de interrupción se le asigna un nivel de prioridad
    - Una interrupción se atiende si su nivel es superior al de la interrupción cuyo programa de servicio se está ejecutando
- La forma de ignorar interpretaciones es mediante el enmascaramiento
- Las excepciones más críticas no son enmascarables

- Para el manejo de interrupciones y excepciones el MIPS utiliza el coprocesador 0
  - Registro EPC (*Exception Program Counter*): dirección de la instrucción que se estaba ejecutando cuando se produjo la excepción
  - Registro Status: máscara de interrupciones y bits de autorización
  - Registro Cause: motivo de la excepción o interrupción

Acceso directo a memoria

- E/S mediante interrupciones ineficientes para grandes transferencias de datos -> Acceso directo a memoria
- El acceso directo a memoria (*DMA, Direct Memory Access*) se realiza con un controlador especializado que transfiere los datos entre dispositivos y memoria sin intervención del procesador

Pasos en una transferencia DMA:

1. El procesador inica el DMA proporcionando:
   - Identidad del dispositivo
   - Operación a realizar
   - Dirección de memoria fuente o destino de los datos a transferir
     Número de bytes a transferir
2. El DMA inicia la operación:
   - Obtiene el acceso al bus e inicia la transferencia
   - Si se requiere más de una transferencia por el bus el controlador de DMA se encarga de realizarlas sin molestar al procesador

- 3º Una vez finalizada la transferencia el DMA interrumpe al procesador
- Opciones para el acceso al bus por parte del DMA:
  - Por **ráfagas**: cuando el DMA toma el control del bus no lo libera hasta haber transmitido todo el bloque de datos
  - Por **robo de ciclos**: el DMA retiene el control del bus solo un ciclo, transmite una palabra y lo libera
  - DMA **transparente**: solo roba ciclos cuando la CPU no usa el bus

Procesadores de E/S

- Procesador de E/S: controlador inteligente que permite la ejecución de órdenes de E/S sin intervención de la CPU