LIBRO 2

DISEÑO SERIE

SUMADOR DE 1 BIT

$$x + y + c = 2\times c1 + s$$

Retardo del circuito: Es el tiempo en el peor caso, que tarda en estabilizarse la última de las señales de la salida, a partir del instante en que se han estabilizado todas las señales de entrada

En este caso sería 30 => 15 XOR + 15 XOR

Figura 2.1 *Sumador de 1 bit. En cada puerta se especifica el retardo.*

SUMADOR DE VECTORES DE BITS

$$x = \sum_{i=0}^{n-1} x_i * 2^i$$

Esquema lógico del sumador

Figura 2.2 *Sumador de 2 bits con propagación serie del acarreo. S1b es un sumador de 1bit.*

En este caso el retardo sería de 50 => 25 para c1 y luego 10 AND + 15 OR (c2)
Porque s1 se ejecuta en paralelo con s0, y tarda 15 XOR más (por c1)

Metodología de reloj

Define cuando los elementos de memorización pueden leerse y cuando pueden escribirse

Figura 2.4 *Señal de reloj.*

REGISTRO

Un registro es un elemento básico de almacenamiento que actualiza su estado en el flanco ascendente o descendente de la señal de reloj. Un registro elemental almacena un 1 bit.

Figura 2.6 *Registro.*

Retardo de propagación (t_p):
Es el intervalo de tiempo desde el flanco ascendente hasta la estabilización del nuevo estado del registro

Figura 2.7 *Retardo de propagación de un registro.*

FUNCIONAMIENTO

En un dispositivo serie hay que esperar a que finalice una tarea antes de iniciar una nueva tarea

Figura 2.8 *Diagrama temporal del procesado de varias sumas en un sumador serie.*

El tiempo para procesar una secuencia de n tareas en un dispositivo series es $n*T_{seire_}$ donde T_{serie} es el periodo de la señal de reloj

RESTRICCIONES EN EL TIEMPO DE CICLO

Antes de iniciar una nueva tarea, la lógica combinacional, debe disponer de tiempo suficiente para obtener señales estables en sus salidas. El tiempo mínimo de ciclo, para un dispositivo serie, es:

$T \geq t_p + t_{logica}$

Figura 2.9 *Restricción en el periodo de la señal de reloj.*

Para el caso del sumador de 2 bits, suponiendo que el retardo de propagación es de 2, el tiempo de ciclo debe ser mayor o igual a 52. Entonces, el número de operaciones por segundo es 1/52 = 0.02

SEGMENTACIÓN

La técnica de segmentación permite diseñar un dispositivo que empieza a procesar una nueva tarea antes de haber finalizado el procesado de la tarea previa.

Primero se identifica los módulos combinacionales que se efectúa en cada fase. Después se añaden registros de desacoplo con tal de guardar información hasta el siguiente ciclo de reloj

Figura 2.11 *Sumador de vectores de 2 bits segmentado en dos etapas.*

FUNCIONAMIENTO Y CARACTERIZACIÓN

Un registro de desacoplo transfiere la información de su entrada a la salida en nuestro caso en el flanco ascendente.

Figura 2.12 *Elementos de una etapa en un diseño segmentado.*

Tabla de reserva: es un diagrama que muestra la ocupación de las etapas de un dispositivo segmentado en función del tiempo expresado en ciclos. Hay dos tipos

Figura 2.14 *Tabla de reserva de un sumador segmentado en dos etapas (E0, E1). a) bidimensional, b) unidimensional.*

SOLAPAMIENTO EN EL PROCESADO DE TAREAS

Figura 2.16 *Solapamiento del cálculo de varias sumas de vectores de 2 bits.*

Para una operación tarda 2 ciclos en procesarse, pero finaliza una por ciclo

RESTRICCIONES EN EL TIEMPO DE CICLO

$$T \geq max(t_{Ei}) + t_p$$

En el caso del sumador de 2 bits el tiempo de tapa de E0 y E1 es el mismo, por lo que el tiempo de ciclo sería 30 + 2 (t_p)

RESUMEN

La idea de la segmentación es identificar fases en una tarea que efectúa un dispositivo. Entonces se diseñan o identifican módulos combinacionales para cada una de las fases o grupos de ellas. Estos módulos se conectan ente si, insertando registros de desacoplo en las conexiones entre módulos, y se establece una secuencia de utilización de los módulos que de lugar a la tarea original. El sistema funciona de forma sincronizada con una señal de reloj y el estado de los registros de desacoplo se actualiza en el flanco ascendente de la señal de reloj. El desacoplo de las etapas, soportado por los registros de desacoplo, permite solapar el procesado de varias tareas en el dispositivo segmentado. El resultado es que, respecto a un funcionamiento serie, se consigue incrementar el número de tareas procesadas por unidad de tiempo.

GANANCIA

$$Ganancia = \frac{Productividad_{seg}}{Productividad_{serie}} = \frac{t_p + t_{logica}}{t_p + t_{logica}/NE} ¿$$

En el caso ideal la Ganancia es igual o menor que NE (Ganancia ≤ NE). Porque suponemos que $t_p << t_{logica}$. Pero normalmente, no se cumple este caso ideal. En el caso del sumador de 2 bits la ganancia sería 52/32 = 1.62. El segmentado es un 62% más rápido.

ELEMENTOS DE DESACOPLO

Distinguiremos dos elementos de memorización sincronizados con la señal de reloj. Uno utiliza el nivel de la señal de reloj para determinar cunado se almacena la señal de entrada y el otro el flanco.

CELDA D (LATCH D)

Nivel 1 => entrada se transmite a la salida
Nivel 0 => entrada se cierra, se mantiene el valor de la salida

Figura 2.20 *Celda D.*

Figura 2.21 *Evolución temporal, en una celda D, de la señal de salida en función de la señal de entrada y la señal de reloj.*

REGISTRO D O BIESTABLE D (FLIP-FLOP D)

Són dos celdas D que actúan con niveles de la señal de reloj opuestos.
Nivel 0 => maestro abierto, esclavo cerrado
Nivel 1 => maestro cerrado, esclavo abierto
Las partes oscurecidas es cuando maestro o esclavo estan cerrados

Figura 2.22 *Registro D.*

Figura 2.23 *Evolución temporal, en un registro D, de la señal de salida (Q) en función de la señal de entrada (D) y la señal de reloj.*

TIPOS DE UNIDADES FUNCIONALES SEGMENTADAS

Dispositivo segmentado unifunción:
Unicamente procesa un tipo de tarea
Dispositivo segmentado lineal:
Cada etapa se conecta exclusivamente a etapas sucesoras y ninguna etapa tarda más de un ciclo (a)
Dispositivo multiciclo o seudolineal:
Alguna etapa se utiliza de forma repetida en ciclos consecutivos (b)
Dispositivo segmentado no lineal:
Alguna etapa se utiliza más de una vez y al menos una de las utilizaciones es en ciclos no consecutivos (c)

Figura 2.24 *Ejemplos de segmentación lineal y no lineal.*

Dispositivo segmentado multifunción:
Dispositivo que puede procesar tareas de varios tipos
Dispositivo segmentado multifunción dinámico:
Dispositivo que puede procesar de forma solapada varias tareas de distinto tipo

REPLICACIÓN

Utilizar varias réplicas de un dispositivo serie y su objetivo es incrementar el número de tareas procesadas por unidad de tiempo. Las réplicas procesan tareas de forma concurrente y cada réplica procesa tareas de forma serie.

Figura 2.26 *Diagrama temporal que muestra el procesado de una secuencia de sumas cuando se utilizan dos sumadores serie.*

GANANCIA

La productividad en el caso ideal se multiplica por el número de réplicas cuando se utiliza replicación. Ganancia = N.réplicas

INTERPRETACIÓN DE INSTRUCCIONES

LENGUAJE MÁQUINA

Tipos de instrucciones:
MEM: leer o escribir memoria
ENT: cálculo aritmético-lógico
IS: secuenciamiento implícito

Clases	Tipo	Campos de la instrucción (31 . . . 26 25 ... 21 20 ... 16 15 ... 12 11 ... 5 4 ... 0)						Descripción
		6	5	5	4	7	5	
ENT	RR	CoOp	ra	rb	0 ... 0	func	rc	
	RI	CoOp	ra	literal	1	func	rc	Cálculo
MEM	Lo | St	CoOp	ra	rb	literal			Acceso a memoria
IS	BR	CoOp	ra	literal				Saltos relativos
	JMP	CoOp	ra	rb	func	0 ... 0 0 ... 0		Saltos indexados

Figura 2.27 *Formato de las instrucciones de lenguaje máquina. Todas las instrucciones son de 32 bit de longitud con un campo de código de operación de 6 bits, ubicado en los bits <31:26> de la instrucción.*

CAMINO DE DATOS DE UN PROCESADOR SERIE

MID: Elementos de memoria
BR: Banco de Registros
ALU: Unidad artimético-lógica
CP: Contador de Programa
Decodificador y control

Figura 2.30 *Camino de datos simplificado de un procesador secuencial.*

La memoria almacena el código y los datos. El control del procesador se encarga de encadenar las fases de interpretación. Un elemento puede utilizarse varias veces al interpretar una instrucción. Un procesador serie no empieza a interpretar una instrucción hasta que no ha finalizado la anterior.

Fases	Recursos utilizados
Determinar la dirección de la instrucción	ALU y CP
Búsqueda de la instrucción	memoria (MID)
Decodificación	decodificador
Lecturas de los registros fuente	banco de registros (BR)
Ejecución	ALU y memoria (MID)
Almacenamiento del resultado	banco de registros (BR) y memoria (MID)

Figura 2.31 *Fases de ejecución de una instrucción y utilización de recursos.*

Supondremos que cada fase tarda 1 ciclo excepto la de ejecución. **CP, BUS, D/L, ALU, M** indican las fases:
- Determinar dirección de una instrucción
- Búsqueda de la instrucción
- Decodificación y lectura de los registros fuente
- Utilización de la ALU
- Acceso a memoria para leer/escribir un dato

ES se utiliza para indicar escritura del resultado en el BR y/o almacenar en registros temporales la dirección que determina una instrucción de secuenciamiento, además del resultado de evaluar la condición si es el caso

Figura 2.32 *Interpretación de una secuencia de instrucciones en un procesador serie. La instrucción interpretada se muestra en la parte inferior de la figura.*

SEGMENTACIÓN DEL PROCESO DE INTERPRETACIÓN DE INSTRUCCIONES

Se utiliza para reducir el tiempo de ejecución de un programa se utiliza la técnica de segmentación en la implementación del camino de datos y se solapa la interpretación de instrucciones, con el objetivo de empezar a interpretar una instrucción en cada ciclo.

Problemas: recursos hardware disponibles y respetar la semántica del lenguaje con que se expresa el programa

Tipo	instrucción	ciclos 1	2	3	4	5	6
MEM	load	CP	BUS	D/L	ALU	M	ES
	store	CP	BUS	D/L	ALU	M	
ENT / IS		CP	BUS	D/L	A/E	ES	

Figura 2.33 *Segmentación del proceso de interpretación de instrucciones.*

Figura 2.34 *Tablas de reserva que muestran el ciclo en que se utilizan los recursos del camino de datos.*

RIESGO ESTRUCTURAL

Cuando un recurso se utiliza varias veces se crean situaciones, denominadas de riesgo estructural o conflicto de recurso, con otras tareas que se han empezado a procesar previamente.

Latencia de inició o repetición: n. de ciclos que hay que esperar entre el inicio de dos tareas consecutivas en una unidad funcional segmentada para que no se produzca un riesgo estructural. Suele ser distinta para una secuencia de tareas

Latencia media de inicio(): dada una secuencia de latencias de inicio se calcula el valor medio. Su inversa es la productividad en ciclos

Valor mínimo de la latencia media de iniciación: es mayor o igual que el máximo número de marcas en cualquiera de las filas de la tabla de reserva.

CONFLICTOS ENTRE INSTRUCCIONES CON EL MISMO PATRÓN

Figura 2.37 *Diagrama temporal para determinar una secuencia de latencias de inicio cuando se procesa una secuencia de instrucciones ENT.*

La secuencia de latencias de inicio para las instrucciones ENT es (1,4). Entonces la latencia medio de inicio LMI=(1+4)/2=2.5 ciclos y por lo tanto la productividad es 0.4 i/c.

Rendimiento

Ganancia = NE / LMI ciclos perdidos= LMI -1

Para el ej anterior seria ganancia = 5/2.5= 2 y los ciclos perdidos=2.5-1= 1.5 c

Eliminación de conflictos

- Replicar recursos
- El retardo del recurso sea inferior al tiempo de ciclo y que el recurso se pueda utilizar varias veces en un ciclo

Recursos que dan problemas:
ALU

En cada ciclo se utiliza para calcular la dirección de la instrucción que se va a buscar en el siguiente ciclo. Por otro lado, todas excepto las de tipo JMP, la utilizan para efectuar cálculos aritméticos o lógicos.

Sol: Añadir un sumador en el camino de datos segmentado, para calcular la siguiente @ de la instrucción que debe interpretarse, suponiendo secuenciamiento implícito (i+4)

Figura 2.38 *Ocupación de recursos cuando se dispone de un sumador para el secuenciamiento implícito.*

BR

Se utiliza para leer los operandos fuentes (3º ciclo) y escribir (5º ciclo)

Sol: Replicamos el camino. Tenemos dos caminos de lectura, dos registros fuente (BRL) y un camino de escritura, un registro destino (BRE)

Figura 2.39 *Replicación de caminos de acceso al banco de registros para reducir riesgos estructurales.*

MID

En instrucciones tipo load/store se utiliza dos veces el recurso MID.

Sol: Como se tiene en cuenta que el código de un programa no se suele modificar. Entonces, en el nivel de la jerarquía de memoria más cercano al procesador se distingue una cache de instrucciones (MI) y una cache de datos (MD) y cada una de las caches tiene un camino de acceso (Arquitectura Harvard).

Figura 2.40 *Recursos y ocupación para que no existan conflictos entre instrucciones con el mismo patrón de ocupación de los recursos.*

CONFLICTOS ENTRE INSTRUCCIONES CON DISTINTO PATRÓN

Las instrucciones ENT y load representan el 50-60% de las instrucciones interpretadas y normalmente se interpretan de forma entrelazada. Y se produce conflicto con el recurso BRE. La solución menos costosa es modificar el instante de escritura en las ENT (se retarda 1 ciclo), con la idea de conseguir el mismo patrón que las load.

Figura 2.42 *Mismo patrón de ocupación de las etapas para todas las instrucciones.*

NOTA: Cuando el diseño tiene una segmentación unificada para todas las instrucciones; todas utilizan los recursos en el mismo ciclo, un recurso se utiliza una sola vez, no es necesario detectar riesgos estructurales (simplifica el control). Se dice que el procesador está segmentado de forma lineal

CONTROL DE UN DISPOSITIVO SEGMENTADO

En el control distinguimos dos partes:
- Determinar cuando una nueva tarea se inicia o prosigue a la siguiente etapa
- Controlar la lógica de cada etapa y el encaminamiento entre las etapas

Figura 2.43 *Control de un dispositivo segmentado. Las lógicas A y B siempre efectúan la misma función.*

CONTROL DE INICIACIONES

Utiliza las tablas de reserva (permite caracterizar las latencias de inicio permitidas y prohibidas) y la señal de solicitud de tarea pendiente.

Latencia prohibida: latencia de inicio que da lugar a un riesgo estructural

Actuación cuando se detecta un futuro riesgo estructural: el control no permite que se inicie o prosiga la tarea . Las tareas previas siguen utilizando las etapas en función del tipo de tarea. El bloqueo de la tarea, que produciría un riesgo estructural, perdura hasta que se determina que no se producirá riesgo estructura.

PROCESADOR SEGMENTADO

Se intenta minimizar la posibilidad de conflicto de recursos, pero a veces no es posible por el coste respecto al rendimiento que proporciona.

Los conflictos entre inst. Load y ENT debido a su distinta LC (2c y 1c). Pero la técnica de retardo no siempre sirve, pues la LC de una op de coma flotante es mayor (4c - 10c) por lo que el conflicto se produce entre más tipos de inst.

Latencia de cálculo (LC): Número de ciclos desde que se dispone de los datos para efectuar un cálculo hasta que finaliza

La solución es controlar el recurso de escritura al BR. Este requiere conocer el tipo de inst (etapa D/L), donde se decodifica y el conflicto se produciría después de esta. Por lo que se controla si una inst prosigue la interpretación o se queda en la etapa D/L hasta quwe desaparecen los riesgos.

Etapa de retención Etapa en la que se retiene una inst. (con un nop)

Las inst nop no modifican el estado del procesador. También debemos retener CP y BUS por culpa de la retención D/L, ya que al siguiente ciclo, estas estarán ocupadas y se produciría un conflicto

Figura 2.46 *Bloqueo de la interpretación segmentada de instrucción para controlar un riesgo estructural.*

Los ciclos de bloqueo los contabilizaremos en la salida del dispositivo segmentado. Entonces, en un procesador se contabiliza un ciclo de bloqueo en el ciclo que finaliza una instrucción nop

MODOS EN UN ELEMENTO DE DESACOPLO

Los reg de desacoplo junto a sus multiplexores, que indican el modo de funcionamiento, controlan las retenciones

Modo normal: La información de la entrada se transfiere a la salida cuando hay flanco ascendente

Modo bloqueo: La salida no se modifica cuando hay flanco ascendente.

Modo burbuja: En la salida hay una nop después del flanco ascendente.

Figura 2.47 *Lógica adicional en los registros de desacoplo para implementar los modos: normal, bloqueo y burbuja.*

CONTROL DE RIESGOS ESTRUCTURALES

Figura 2.48 *Esquema simplificado de las etapas CP, BUS y D/L.*

Figura 2.49 *Lógica de retención e inyección de instrucciones nop.*

Figura 2.50 *Circuito para detectar una latencia prohibida igual a tres.*

CONTROL DEL CAMINO DE DATOS

Control estacionario en el tiempo: Un elemento centralizado en cada ciclo envía las señales a todos los elementos del dispositivo segmentado que lo requieren

Figura 2.52 *Control estacionario en el tiempo.*

Control estacionario en los datos: La información de control fluye con los datos y el control está integrado en la segmentación. Cada vez que se inicia una operación se generan las señales necesarias, para que en cada etapa sean interpretadas y controlen la lógica y el encaminamiento. Una etapa a su vez puede generar o procesar información de control para la siguiente etapa

Figura 2.53 *Control estacionario en los datos.*

MEMORIA CACHE

ORGANIZACIÓN DE LA CACHE

$dirección_{bloque}$ = dirección / B B => tamaño de bloque en bytes

NC = C / B (Número de contenedores) C => capacidad de datos de la cache bytes, potencia de dos

Mapeo directo (el que utilizaremos) => entrada = $dirección_{bloque}$ **mod** NC
etiqueta = $dirección_{bloque}$ / NC

elemento = (dirección **mod** B) / TE TE => tamaño del elemento en bytes, potencia de dos

EL => Elemento accedido EN=> Indican el contenedor ETIQ=> Etiqueta

Figura 2.55 *Campos de la dirección de memoria utilizados para acceder a cache.*

Figura 2.56 *Esquema lógico del acceso a una cache con mapeo directo.*

SEGMENTACIÓN DEL ACCESO

ET => ETIQ DAT => Acceso a los bancos de datos

		ciclos	
		1	2
load/store	etapa	ET	DAT
	recursos	bET	bD

Figura 2.57 *Segmentación lineal en dos etapas del acceso a cache para las instrucciones load y store. bET indica banco de etiquetas y bD banco de datos (bD0, bD1).*

CAMINO DE DATOS SEGMENTADO

ML => Load MS => Store EsMem => Se activa cuando se escribe en el banco

Figura 2.58 *Camino de datos segmentado en dos etapas del acceso a cache.*

Carlos Rodríguez
08-04-2020