

**UNIVERSITAT POLITÈCNICA DE CATALUNYA**

*Departament d'Arquitectura de Computadors*

**RECURSOS ANCHOS:  
UNA TÉCNICA DE BAJO COSTE  
PARA EXPLOTAR PARALELISMO  
AGRESIVO EN CÓDIGOS  
NUMÉRICOS**

Autor: David López Alvarez  
Directores: Mateo Valero Cortés  
Josep Llosa i Espuny

Dedicado

A Silvia,  
que ha sufrido con elegancia  
el horrible tándem marido-doctorando.

A mi Tete,  
por hacer que me picara  
el gusanillo de la informática.

A mi madre,  
la persona más cuerda que conozco,  
y quizá la más sabia.  
Por enseñarme un montón de cosas  
que no se aprenden en ninguna universidad.

A mi padre, *in memoriam*.  
Disfruta esta tesis papá, allá donde estés.  
Aquí te seguimos añorando.

## **Agradecimientos**

*Estos son mis principios. Si a usted no le gustan, tengo otros.*

*Julius "Groucho" Marx*

### Sonetillo de rima rara.

Quienes tesis hayan hecho  
ya saben que es cosa sin par  
llegar al agradecimiento  
y no saber dónde empezar.

Hay tantos con derecho  
que quisieras nombrar  
que, en verdad no miento,  
¡espacio me ha de faltar!

Otras dos tesis necesitaría,  
para agradecer a quien se debe,  
pero ¡las va a hacer su tía!

Que otros nuevas tesis aborden,  
que yo procuraré ser breve,  
(pero dentro de un orden).

(sigue...)

## A los amigos

*I get by with a little help from my friends, mm ...  
I get high with a little help from my friends, mm ...  
Going to try with a little help from my friends.  
J. Lennon & P. McCartney.*

### Canción sin pirata

A mi familia, para empezar,  
pues no puede ser de otro modo  
os agradezco de verdad  
todo, absolutamente todo.

A mis compinches del DAC,  
colegas en ambos sentidos  
más que colegas, amigos  
y compañeros de adversidad.

A Jordi Tubella, casi un confesor,  
que ha aguantado (y lo que pesa)  
a un pesado como yo  
al otro lado de la mesa.

Por los buenos momentos, a Agustín  
por largas conversaciones ante una cerveza.  
Y a un escritor que ahora empieza,  
al querido y elegante Fermín.

A Montse, por tenerme como amigo  
a pesar de perder la apuesta.  
A Anna, por tener siempre dispuesta  
una palabra amable para conmigo.

A Roger y Marta, a Toni y Dolors.  
A Joan Carles, por los crucis infinitos.  
A los Enrics (a los dos).  
A Pepe y Pedro, mis PBCs favoritos.

A J. Smith, a Paqui, a Cristina,  
y a un monton de gente,  
que no me es indiferente  
pero que ofrecen mala rima.

A Edu, el tercer director.  
Pues aunque no aparezca  
en la portada como autor  
no es porque no lo merezca.

A Josep, que me enseñó  
a compaginar amistad y trabajo.  
Y a Laura que le obligó  
a leer mi tesis a destajo.

A Mateo, que en mí creyó  
todo lo que se puede creer.  
Por todo lo que me empujó  
con frases como: "Esto, para ayer".

Y como el lector debe estar hartito  
y a punto de enviarme a hacer puñeta  
se despide este poeta maldito,  
también llamado "maldito poeta".

David, Octubre de 1998.

# Índice

<b>Prefacio</b> .....	<b>1</b>
<b>Capítulo 1. Conceptos básicos de paralelismo a nivel de instrucciones.</b> .....	<b>5</b>
1.1. Introducción .....	5
1.2. Perspectiva histórica .....	6
1.3. Evolución de la arquitectura desde la perspectiva del ILP .....	8
1.4. Dependencias .....	14
1.5. Hardware de soporte para ILP .....	15
1.6. Compilación para ILP: planificación y asignación de registros .....	17
<b>Capítulo 2. Entorno de trabajo.</b> .....	<b>21</b>
2.1. Introducción .....	21
2.2. Definiciones básicas .....	22
2.2.1 Representación de un bucle .....	22
2.2.2 Extensión del grafo de dependencias: el concepto de Stride .....	25
2.2.3 Representación de una arquitectura .....	27
2.3. Planificación de instrucciones mediante segmentación software .....	30
2.4. Ictíneo .....	45
2.5. Juego de pruebas: el Perfect Club .....	49
2.6. Sumario .....	50

**Capítulo 3. Límites en ILP y técnicas para alcanzarlos . . . . . 53**

3.1. Introducción .....53

3.2. Limitaciones: recursos y recurrencias .....55

3.3. Incrementando el rendimiento de bucles limitados por los recursos .....59

    3.3.1 Replicación.....59

    3.3.2 Widening .....62

    3.3.3 Clasificación de los bucles .....66

    3.3.4 Caracterización del juego de pruebas.....72

3.4. Unidades funcionales complejas para bucles limitados por las recurrencias .....74

3.5. Sumario y contribuciones.....77

**Capítulo 4. Métodos de compactación . . . . . 79**

4.1. Introducción .....79

4.2. Compactación estática.....80

    4.2.1 Unrolling .....81

    4.2.2 Algoritmo de deducción de strides.....83

    4.2.3 Algoritmo de compactación .....85

4.3. Compactación dinámica .....86

4.4. Sumario y contribuciones.....91

**Capítulo 5. Rendimiento de las técnicas . . . . . 93**

5.1. Introducción .....93

5.2. Límites teóricos .....95

    5.2.1 Comportamiento de los buses anchos .....95

    5.2.2 Comportamiento de las unidades funcionales anchas .....99

    5.2.3 Límites de la técnica widening .....101

    5.2.4 Técnicas mixtas .....105

    5.2.5 Añadiendo unidades funcionales FMA .....106

5.3. Añadiendo spill code .....108

5.4. Sumario y contribuciones.....117

<b>Capítulo 6. Coste de una arquitectura .....</b>	<b>119</b>
6.1. Introducción .....	119
6.2. Coste en área .....	120
6.3. Tiempo de ciclo.....	126
6.4. El lenguaje máquina.....	133
6.5. La memoria cache .....	134
6.6. Sumario y contribuciones .....	136
<b>Capítulo 7. Escogiendo un modelo: relación rendimiento/coste .....</b>	<b>139</b>
7.1. Introducción .....	139
7.2. Configuraciones implementables.....	140
7.3. Adaptando la latencia de las unidades funcionales.....	143
7.4. Efectos observados.....	146
7.5. Configuraciones con mejor rendimiento por generación tecnológica. ....	150
7.6. Sumario y contribuciones .....	155
<b>Capítulo 8. Conclusiones y trabajo futuro .....</b>	<b>157</b>
8.1. Sumario y conclusiones .....	157
8.2. Contribuciones .....	160
8.3. Trabajo futuro. ....	163
<b>Referencias .....</b>	<b>167</b>
<b>Crucigrama .....</b>	<b>177</b>



# Prefacio

**(O breve resumen de lo que encontrare aquél que se atreviere a leer este mamotreto)**

En este trabajo se presenta un estudio de diversos métodos para la explotación del paralelismo a nivel de instrucciones (*Instruction Level Parallelism*, ILP) para bucles de código numérico en arquitecturas *Very Long Instruction Word* (VLIW).

En nuestro trabajo se proponen modificaciones en la arquitectura del procesador. Para poder evaluar el rendimiento de las modificaciones propuestas, es necesario garantizar un buen aprovechamiento de los recursos, por lo que se han empleado técnicas de segmentación software (*software pipelining*). Dichas técnicas se basan en efectuar una planificación (*scheduling*) de bucles buscando un patrón de ejecución tal que se aprovechen al máximo los recursos disponibles.

Estudiando el rendimiento de la planificación de un bucle sobre una arquitectura determinada, se puede observar cómo dicho rendimiento está limitado tanto por las recurrencias del bucle como por los recursos de la arquitectura.

Para aumentar el rendimiento de bucles limitados por las recurrencias, se puede reducir la latencia de las unidades funcionales (lo que está más allá de los objetivos de esta tesis) o bien fusionar diversas operaciones en una sola operación compleja (técnica de *fusión*). Existen, por ejemplo, procesadores con unidades aritméticas que fusionan una multiplicación y una suma asociada en una sola operación (operación *Fused Multiply-and-Add*, FMA).

Una solución para incrementar el rendimiento de bucles limitados por los recursos ha sido incrementar el número de recursos (técnica de *replicación*). Otra solución es utilizar recursos anchos (técnica de *widening*), en donde no se incrementa el número de recursos, sino la cantidad de operaciones que puede resolver cada recurso por ciclo de procesador. El grado de anchura será el número de operaciones que se pueden ejecutar por recurso simultáneamente.

La técnica de *widening* ha sido utilizada en ciertos procesadores superescalares, como el IBM POWER2 (donde hay buses de 128 bits) o el coprocesador multimedia AltiVec (las unidades aritméticas y los recursos son de 128 bits). La idea también ha sido usada en procesadores vectoriales. En estos procesadores hay unidades funcionales que resuelven operaciones escalares (de ancho 1) y unidades que resuelven operaciones vectoriales (de ancho  $n$ , típicamente 64 y 128 elementos).

En este trabajo se propone el uso de buses, unidades aritméticas y banco de registros anchos para arquitecturas VLIW, donde las mismas unidades pueden resolver operaciones de ancho 1 o ancho  $n$ . Se ha realizado una comparación del rendimiento usando las técnicas de replicación, *widening* y fusión, y combinaciones de las mismas.

Muchos trabajos existentes en la literatura proponen una técnica y estudian su rendimiento teórico, sin tener en cuenta sus costes (a veces tan elevados, que pueden resultar inimplementables). En esta tesis se realiza una evaluación global de las técnicas propuestas, estudiando su relación rendimiento/coste, y efectuando una proyección de futuro según las previsiones de la *Semiconductor Industry Association (SIA)* para la tecnología y el área de chip de los próximos 12 años.

Los costes estudiados han sido:

- la necesidad de *spill code* (código añadido por la falta de registros físicos necesarios para la resolución del problema).
- el coste en área de chip.
- el coste en tiempo de ciclo del banco de registros.

Los resultados se obtienen a base de una exploración sistemática de un amplio espectro de configuraciones donde, para unidades aritméticas con FMA y sin FMA, se han variado los grados de replicación y *widening*, el tamaño del banco de registros y el particionado del mismo.

El documento se organiza de la siguiente manera:

- los capítulos 1 y 2 son introductorios. En el primero se presentan los conceptos básicos del paralelismo a nivel de instrucciones (ILP) y su evolución histórica, mientras que el segundo ya está más centrado en este trabajo, dándose una serie de definiciones básicas que se utilizarán a lo largo del resto de la tesis.
- el capítulo 3 presenta los factores que limitan el ILP y las técnicas que nos permiten atacarlos. En este capítulo se explican las técnicas de replicación, *widening* y fusión.
- el capítulo 4 está orientado a los métodos de compactación necesarios para aplicar la técnica de *widening*. En él se explican los algoritmos de compactación que hemos desarrollado para la compactación estática, así como un método de compactación dinámica para arquitecturas superescalares.
- el capítulo 5 está dedicado al estudio del rendimiento de las técnicas (en solitario o en combinación). Primero se estudia el rendimiento teórico de replicación, de *widening* y de configuraciones mixtas en que se aplican ambas técnicas. Asimismo, se analiza el rendimiento teórico de las técnicas mixtas con unidades aritméticas capaces de

implementar la operación FMA. Finalmente, se estudia la influencia del tamaño del banco de registros (y la pérdida de rendimiento respecto al teórico que produce la introducción de *spill code*).

- en el capítulo 6 se presentan diversos métodos para calcular el coste (en área y tiempo de ciclo) de las técnicas comparadas.
- ya en el capítulo 7, se aborda la relación rendimiento/coste. En este capítulo se propone una metodología que nos permite averiguar qué configuraciones obtienen el mejor rendimiento cuando estamos limitados por la tecnología, presentándose dichas configuraciones para cada una de las generaciones tecnológicas propuestas por la SIA.
- finalmente, el capítulo 8 resume los resultados, presentando las conclusiones y contribuciones de esta tesis, así como las líneas abiertas que deja este trabajo y en las que pensamos seguir investigando.