



**Escuela Técnica Superior
de Ingeniería Informática**

Universidad de La Laguna



SIJEM

Simulador Didáctico de Jerarquías de Memoria

**Memoria de Proyecto de Fin de Carrera
de Ingeniería Superior en Informática**

Fernando Alesanco Pérez

Diciembre 2004

Directores:

Dr. Lorenzo Moreno Ruiz

Dra. Carina Soledad González

Agradecimientos

En primer lugar quiero agradecer el entusiasmo, apoyo, trabajo y consejos de mis directores, Dr. Lorenzo Moreno Ruiz y Dra. Carina Soledad González. Gracias a ellos y al esfuerzo de sus alumnos, este proyecto se ha convertido en mucho más de lo que se pensó inicialmente.

También quiero agradecer a mi familia todo el apoyo prestado durante estos años de carrera, en los buenos y en los malos momentos, dándome fuerzas para seguir adelante.

A todos los compañeros de Facultad, con los que he compartido innumerables horas delante de un ordenador y que se han convertido en amigos para toda la vida.

A mis amigos del instituto, que a pesar de las distancias han sabido mantener el contacto, demostrándome que cuento con ellos.

Y a los muchos amigas y amigos que he hecho durante estos años, de los que espero que sigan ahí por mucho tiempo.

A mi familia y a mis verdaderos amigos,

Lorenzo Moreno Ruiz y Carina Soledad González, catedrático y profesor asociado del Departamento de Física Fundamental, Electrónica y Sistemas de La Universidad de La Laguna.

CERTIFICAN:

Que el proyecto de Fin de Carrera de título: ***“SIJEM: Simulador didáctico de Jerarquías de Memoria”*** presentado por D. Fernando Alesanco Pérez ha sido realizado bajo nuestra dirección.

Y para que así conste, firmamos la presente en La Laguna, a 15 de Diciembre de 2004.

Dr. Lorenzo Moreno

Dra. Carina Soledad González

Índice.

Capítulo 1. Introducción	1
1.1. Conceptos de SIJEM.	1
1.2. Los simuladores en la enseñanza de materias relacionadas con los ordenadores	1
1.3. Justificación del proyecto	3
1.4. Resumen del documento	3
1.5. Disponibilidad del simulador	3
Capítulo 2. Fundamentos teóricos	5
2.1. Jerarquías de memoria	5
2.2. Localidad de referencias	6
2.3. Memorias caché	6
2.3.1. Aciertos y fallos de caché	6
2.3.2. Medida de prestaciones de las memorias caché	6
2.3.3. Fuentes de fallos de caché	7
2.3.4. Cachés divididas frente a cachés conjuntas	8
2.3.5. Cachés multinivel	8
2.3.6. Políticas de mapeado	9
2.3.6.1. Estrategias de colocación	9
2.3.6.2. Estrategias de búsqueda	10
2.3.6.3. Estrategias de reemplazamiento	11
2.3.6.4. Estrategias de coherencia	12
2.3.7. Bits de control	13
2.4. Memoria principal	13
2.4.1. Medida de prestaciones de la memoria principal	13
2.4.2. Anchura de la memoria principal	14
2.4.3. Memoria principal entrelazada	14
2.5. Memoria secundaria	15
2.6. Memoria virtual	15
2.6.1. Direcciones virtuales y direcciones reales	15
2.6.2. Transformación en bloques	16
2.7. Paginación	17
2.7.1. Selección del tamaño de página	18
2.7.2. Mecanismos de traducción de direcciones	18

2.7.2.1. Traducción de direcciones por transformación directa	18
2.7.2.2. Traducción de direcciones por transformación asociativa	19
2.7.2.3. Traducción de direcciones por transformación asociativa-directa con TLB conjunta	20
2.7.2.4. Traducción de direcciones por transformación asociativa-directa con TLB dividida	21
2.7.3. Estrategias de administración del sistema de memoria virtual paginado	22
2.7.3.1. Estrategias de búsqueda	22
2.7.3.2. Estrategias de colocación	23
2.7.3.3. Estrategias de reemplazamiento	23
2.7.4. Prestaciones del sistema de memoria virtual	24
2.8. Un poco de historia	24
 Capítulo 3. Especificación de requisitos	 27
3.1. Requisitos funcionales	27
3.2. Requisitos no funcionales	27
 Capítulo 4. Decisiones de diseño	 29
4.1. Elección del lenguaje de programación	29
4.2. Características del simulador	30
4.3. Control de la simulación	31
4.4. Asistente	31
4.4.1. Asistente - Configurar	32
4.4.1.1. Ficheros de configuración	33
4.4.2. Asistente – Cargar fichero	36
4.4.2.1 Ficheros de traza	36
4.4.3. Asistente – Simular	38
4.5. Simulación	38
4.5.1. Traducción de direcciones	38
4.5.2. Búsqueda de páginas	40
4.6. Estrategias	42
4.6.1. Estrategias de administración del sistema de memoria virtual	42
4.6.2. Estrategias de administración de la memoria caché	42
4.7. Estadísticas	45
4.7.1. Estadísticas de traducción de direcciones	45
4.7.2. Estadísticas de búsqueda de páginas	45
4.8. Ciclos	45
4.8.1. Ciclos en traducción de direcciones	45
4.8.2. Ciclos búsqueda de páginas	46

Capítulo 5. Ayuda del programa	47
5.1. Instalación y ejecución del programa	47
5.2. Archivo de ayuda	47
5.2.1. Introducción a SIJEM	48
5.2.2. Interfaz de usuario de SIJEM	49
5.2.2.1. Asistente > Configurar	50
5.2.2.2. Asistente > Cargar programa	67
5.2.2.3. Asistente > Simular	69
5.2.3. Ficheros de SIJEM	90
5.2.3.1. Tipos de ficheros de SIJEM	90
5.2.3.2. Ficheros de ejemplo de SIJEM	93
Capítulo 6. Tecnología educativa	97
6.1. ¿Qué es tecnología educativa?	97
6.2. Tecnologías de la información y la comunicación (TIC)	97
6.2.1. Las TIC en la enseñanza	97
6.2.1.1. Uso de las TIC en la enseñanza	97
6.2.1.2. Objetivo de las TIC en la enseñanza	98
6.3. Los medios didácticos	98
6.3.1. Tipos de medios	99
6.4. Teorías de enseñanza y aprendizaje aplicadas al diseño de software educativo	99
6.4.1. Aproximación conductista al diseño de software educativo	99
6.4.1.1. La enseñanza programada	100
6.4.1.2. La máquina de enseñar	100
6.4.2. Aproximación cognitivista al diseño de software educativo	100
6.4.2.1. Aprendizaje significativo de Ausubel	101
6.4.2.2. Teorías del procesamiento de la información	101
6.4.3. Aproximación constructivista al diseño de software educativo	102
6.4.4. Consideraciones en la aplicación de las teorías de enseñanza y aprendizaje	103
6.5. Características esenciales del software educativo	103
6.6. Estructura básica del software educativo	104
6.7. Funciones de los programas educativos	104
6.8. Ventajas potenciales del software educativo	105
6.9. Inconvenientes potenciales del software educativo	105
6.10. Criterios de evaluación del software educativo	106
6.11. La tecnología educativa en SIJEM	106

Capítulo 7. Evaluación de la herramienta	107
7.1. Pruebas	107
7.2. Encuesta personal	108
7.2.1. Resultados de la encuesta	110
7.2.1.1. Muestra	110
7.2.1.2. Dedicación al uso del programa	110
7.2.1.3. Interés por las jerarquías de memoria	111
7.2.1.4. Aspecto educativo del software	111
7.2.1.5. Funcionalidad del software	112
7.2.1.6. Funcionamiento del software	113
7.2.1.7. Mensajes de error	114
7.2.1.8. Usabilidad del programa	114
7.2.1.9. Otras cuestiones	115
7.2.1.10. Críticas y posibles mejoras	115
7.2.2. Conclusiones	116
7.3. Test adaptativo	116
7.3.1. Tests adaptativos	116
7.3.2. Características del test adaptativo de PORTAD	118
7.3.3. Test adaptativo para la evaluación de SIJEM	120
7.3.4. Realización del test adaptativo	121
7.3.5. Resultados del test adaptativo	121
7.3.5.1. Muestra	121
7.3.5.2. Nota final	121
7.3.5.3. Número de preguntas	122
7.3.5.4. Aciertos y fallos	122
7.3.6. Resultados del test adaptativo	123
7.4. Encuesta de grupo	123
7.4.1. Resultados de la encuesta de grupo	124
7.4.1.1. Muestra	124
7.4.1.2. Interés mostrado por el test	125
7.4.1.3. Aspecto educativo del test	125
7.4.1.4. Información sobre el test	126
7.4.1.5. Comprensión del simulador y las jerarquías de memoria	127
7.4.1.6. Preguntas	127
7.4.1.7. Funcionalidad del test	129
7.4.1.8. Otras cuestiones	129
7.5. Modificaciones	130
7.5.1. Ficheros de traza	130
7.5.2. Búsqueda de direcciones	131

Capítulo 8. Conclusiones	133
8.1. Tecnología educativa	133
8.2. Proceso de evaluación	133
8.2.1. Participación de los alumnos	133
8.2.2. Integración con otros proyectos	133
8.3. Líneas abiertas	134
Anexo A. Guión de prácticas con el simulador	136
Anexo B. Preguntas del test adaptativo	137
A.1. Concepto 1 – Nivel simple – Entorno de simulación	137
A.1.1. Preguntas de nivel bajo	137
A.1.2. Preguntas de nivel medio	138
A.1.3. Preguntas de nivel alto	140
A.2. Concepto 2 – Nivel intermedio – Traducción de direcciones	141
A.2.1. Preguntas de nivel bajo	141
A.2.2. Preguntas de nivel medio	142
A.2.3. Preguntas de nivel alto	143
A.3. Concepto 3 – Nivel avanzado – Traducción de direcciones con TLB dividida	143
A.3.1. Preguntas de nivel bajo	143
A.3.2. Preguntas de nivel medio	144
A.3.3. Preguntas de nivel alto	145
A.4. Concepto 4 – Nivel intermedio – Estrategias de colocación y reemplazamiento	145
A.4.1. Preguntas de nivel bajo	145
A.4.2. Preguntas de nivel medio	148
A.4.3. Preguntas de nivel alto	151
A.5. Concepto 5 – Nivel complejo – Estrategias de coherencia	151
A.5.1. Preguntas de nivel bajo	151
A.5.2. Preguntas de nivel medio	152
A.5.3. Preguntas de nivel alto	152
Bibliografía	155

Índice de figuras.

<i>Figura 2.1. Esquema típico de una jerarquía de memoria</i>	5
<i>Figura 2.2. Jerarquía de memoria con caché multinivel</i>	8
<i>Figura 2.3. Estrategias de colocación</i>	10
<i>Figura 2.4. Bloque de memoria caché</i>	11
<i>Figura 2.5. Bloque de memoria caché con bits de control</i>	13
<i>Figura 2.6. Transformación dinámica de direcciones</i>	16
<i>Figura 2.7. Entrada de la tabla de páginas</i>	18
<i>Figura 2.8. Traducción de direcciones por transformación directa</i>	19
<i>Figura 2.9. Traducción de direcciones por transformación asociativa</i>	19
<i>Figura 2.10. Traducción de direcciones por transformación asociativa-directa con TLB</i>	21
<i>Figura 2.11. Traducción de direcciones por transformación asociativa-directa con TLB dividida</i>	22
<i>Figura 4.1. Estructura de la jerarquía de memoria implementada</i>	40
<i>Figura 5.1. Estructura de la jerarquía de memoria implementada</i>	84
<i>Figura 7.1. Red bayesiana</i>	118
<i>Figura 7.2. Red bayesiana del test adaptativo</i>	120

Índice de tablas

<i>Tabla 1.1. Algunos simuladores relacionados con la arquitectura de ordenadores</i>	2
<i>Tabla 2.1. Paginación frente a segmentación</i>	17
<i>Tabla 2.2. Jerarquías de memoria en los sistemas actuales</i>	26
<i>Tabla 4.1. Ficheros de traza de ejemplo</i>	36
<i>Tabla 4.2. Formatos de direcciones</i>	36
<i>Tabla 5.1. Ficheros de configuración de ejemplo</i>	93
<i>Tabla 5.2. Ficheros de configuración de ejemplo</i>	94
<i>Tabla 5.3. Ficheros de configuración de ejemplo</i>	95
<i>Tabla 5.4. Ficheros de direcciones de ejemplo</i>	96
<i>Tabla 7.1. Niveles de usuario y concepto</i>	119
<i>Tabla 7.2. Probabilidades a priori</i>	119
<i>Tabla 7.3. Baremo de puntuación</i>	120

Índice de capturas

<i>Captura 4.1. Control de la simulación</i>	<i>31</i>
<i>Captura 4.2. Asistente</i>	<i>32</i>
<i>Captura 4.3. Resumen de configuración</i>	<i>32</i>
<i>Captura 4.4. Resumen de configuración</i>	<i>33</i>
<i>Captura 4.5. Traducción de direcciones</i>	<i>39</i>
<i>Captura 4.6. Búsqueda de páginas</i>	<i>41</i>
<i>Captura 4.7. Cálculo de la posición o conjunto</i>	<i>42</i>
<i>Captura 4.8. Etiqueta (Tag) de la caché</i>	<i>43</i>
<i>Captura 4.9. Puntero de la política de reemplazamiento Clock</i>	<i>43</i>
<i>Captura 4.10. Estado, estadísticas y ciclos de traducción de direcciones</i>	<i>45</i>
<i>Captura 4.11. Estado, estadísticas y ciclos de búsqueda de páginas</i>	<i>45</i>
<i>Captura 5.1. Inicio del programa SIJEM y su Ayuda</i>	<i>47</i>
<i>Captura 5.2. Interfaz de usuario</i>	<i>49</i>
<i>Captura 5.3. Asistente de configuración – Bienvenido</i>	<i>50</i>
<i>Captura 5.4. Asistente de configuración – Página 1</i>	<i>51</i>
<i>Captura 5.5. Carga de un fichero de configuración</i>	<i>52</i>
<i>Captura 5.6. Asistente de configuración – Página 2</i>	<i>53</i>
<i>Captura 5.7. Asistente de configuración – Página 3</i>	<i>55</i>
<i>Captura 5.8. Asistente de configuración – Página 4</i>	<i>59</i>
<i>Captura 5.9. Asistente de configuración – Página 5</i>	<i>63</i>
<i>Captura 5.10. Asistente de configuración – Resumen</i>	<i>67</i>
<i>Captura 5.11. Asistente – Cargar programa</i>	<i>68</i>
<i>Captura 5.12. Abrir fichero de traza</i>	<i>68</i>
<i>Captura 5.13. Asistente – Simular</i>	<i>69</i>
<i>Captura 5.14. Simular traducción de direcciones por transformación directa</i>	<i>70</i>
<i>Captura 5.15. Tabla de páginas</i>	<i>71</i>
<i>Captura 5.16. Dirección virtual</i>	<i>71</i>
<i>Captura 5.17. Dirección real</i>	<i>72</i>
<i>Captura 5.18. Información de la instrucción actual</i>	<i>72</i>
<i>Captura 5.19. Estado actual de la máquina</i>	<i>73</i>
<i>Captura 5.20. Control de la simulación</i>	<i>73</i>
<i>Captura 5.21. Traducción de direcciones por transf. Asociativa-directa con TLB</i>	<i>74</i>
<i>Captura 5.22. Tabla de páginas</i>	<i>75</i>
<i>Captura 5.23. TLB</i>	<i>75</i>
<i>Captura 5.24. Dirección virtual</i>	<i>76</i>
<i>Captura 5.25. Dirección real</i>	<i>76</i>

<i>Captura 5.26. Información de la instrucción actual</i>	77
<i>Captura 5.27. Estado actual de la máquina</i>	77
<i>Captura 5.28. Control de la simulación</i>	78
<i>Captura 5.29. Traducción de direcciones por transf. asociativa directa con TLB dividida</i>	78
<i>Captura 5.30. Tabla de páginas</i>	80
<i>Captura 5.31. TLB Dividida</i>	80
<i>Captura 5.32. Dirección virtual</i>	81
<i>Captura 5.33. Dirección real</i>	81
<i>Captura 5.34. Información de la instrucción actual</i>	82
<i>Captura 5.35. Estado actual de la máquina</i>	83
<i>Captura 5.36. Control de la simulación</i>	83
<i>Captura 5.37. Búsqueda de direcciones/páginas</i>	85
<i>Captura 5.38. Memoria principal</i>	85
<i>Captura 5.39. Memoria caché</i>	86
<i>Captura 5.40. CPU</i>	87
<i>Captura 5.41. Estado actual de la máquina</i>	88
<i>Captura 5.42. Información de la instrucción actual</i>	89
<i>Captura 5.43. Control de la simulación</i>	90
<i>Captura 7.1. Error inicial del programa</i>	129
<i>Captura 7.2. Error final del programa</i>	130
<i>Captura 7.3. Búsqueda de páginas</i>	131

Índice de ejemplos

<i>Ejemplo 4.1. Fichero de configuración</i>	35
<i>Ejemplo 4.2. Extracto de un fichero de traza</i>	37
<i>Ejemplo 4.3. Extracto de un fichero de páginas</i>	37
<i>Ejemplo 5.1. Ficheros de configuración</i>	93

Índice de gráficos

<i>Gráfico 7.1. Muestra de la encuesta inicial</i>	110
<i>Gráfico 7.2. Resultados de ¿Cuánto tiempo ha dedicado a usar este software?</i>	111
<i>Gráfico 7.3. Interés de las jerarquías de memoria frente al resto de temas de la</i>	

<i>asignatura</i>	111
<i>Gráfico 7.4. Aspecto educativo del software</i>	112
<i>Gráfico 7.5. Funcionalidad del software</i>	113
<i>Gráfico 7.6. Traducción de direcciones</i>	113
<i>Gráfico 7.7. Búsqueda de páginas</i>	113
<i>Gráfico 7.8. Funcionamiento del software</i>	114
<i>Gráfico 7.9. Funcionamiento del software</i>	114
<i>Gráfico 7.10. Usabilidad del programa</i>	115
<i>Gráfico 7.11. Distribución de las notas del test</i>	121
<i>Gráfico 7.12. Número de preguntas del test</i>	122
<i>Gráfico 7.13. Aciertos y fallos de preguntas de nivel simple</i>	122
<i>Gráfico 7.14. Aciertos y fallos de preguntas de nivel intermedio</i>	123
<i>Gráfico 7.15. Muestra de la encuesta de grupo</i>	125
<i>Gráfico 7.16. Resultados de ¿El test ha despertado mi interés?</i>	125
<i>Gráfico 7.17. Resultados de ¿He aprendido usando este test?</i>	126
<i>Gráfico 7.18. Resultados de ¿La información proporcionada al inicio del test me ha parecido suficiente?</i>	126
<i>Gráfico 7.19. Resultados de ¿El test ayuda a comprender mejor el simulador? y ¿El test ayuda a comprender mejor los contenidos de la asignatura?</i>	127
<i>Gráfico 7.20. Resultados de ¿Las preguntas formuladas son suficientemente claras?</i>	128
<i>Gráfico 7.21. Resultados de ¿Los ejemplos aportados junto a las preguntas me han ayudado a contestarlas?</i>	128
<i>Gráfico 7.22.. Resultados de Funcionalidad del test</i>	128

Capítulo 1.

Introducción

SIJEM es un simulador de jerarquías de memoria cuyo objetivo es servir de apoyo al aprendizaje de todos los conceptos relacionados con las jerarquías de memoria, fundamentales en materias como Sistemas Operativos, Estructura y Arquitectura de Ordenadores.

1.1. Conceptos de SIJEM

A través de un interfaz altamente visual trata de ilustrar los conceptos de jerarquías de memoria:

Memoria virtual

- Traducción de direcciones virtuales a direcciones reales.
- Paginación.
- Uso de tabla de páginas y TLBs.
- Estrategias de búsqueda, colocación y reemplazamiento en memoria principal.

Niveles de memoria

- Memoria secundaria y principal.
- Memorias caché multinivel (Nivel 3, Nivel 2 y Nivel 1 Conjunta o Separada en datos e instrucciones).
- Estrategias de colocación, reemplazamiento y coherencia entre los diferentes niveles.

1.2. Los simuladores en la enseñanza de materias relacionadas con los ordenadores

A la hora de formar alumnos en materias relacionadas con los ordenadores surgen una serie de complicaciones que difícilmente pueden ser solventadas con las herramientas clásicas de docencia.

Un aspecto clave es que los alumnos consigan trasladar sus conocimientos teóricos sobre el funcionamiento interno de las máquinas a implementaciones reales

cuando en éstas los mecanismos que las hacen funcionar quedan ocultos bajo una complicada combinación de software y hardware integrados. Otro es que puedan afianzar los conocimientos observando pequeños ejemplos funcionales que les proporcionen una visión más clara de todos los conceptos involucrados.

En estos casos es cuando los simuladores proporcionan una valiosa herramienta para la docencia. En la siguiente tabla se muestran algunos de los simuladores existentes relacionados con la Estructura y Arquitectura de Ordenadores, así como de los Sistemas Operativos¹.

<i>Simulador</i>	<i>Descripción</i>
Dinero IV: Trace Driven Uniprocessor Cache Simulator	Simulador de cachés para trazas con referencias a memoria
Simula Cache 1.0	Simulador de cachés separadas y multinivel
Visual Cache	Simulador de cachés realizado en Java, para ser ejecutado directamente desde Internet.
SMP Cache	Simulador de cachés para máquinas multiprocesador
ACME (Adaptative caching using multiple experts)	Plataforma de simulación que permite incluir nuevos algoritmos de reemplazamiento para el estudio de su comportamiento
SoSim (Simulator for Operating Systems Education)	Pretende ser una herramienta didáctica para la enseñanza de los conceptos de los modernos sistemas operativos.
SimOS	Simulador de máquinas Digital y HP DEC
SPIM	Simulador de máquinas MIPS
RSIM	Simulador de ILP (Paralelismo a nivel de instrucción)
WinDLX	Simulador de la máquina ficticia DLX, para explicar el funcionamiento de las máquinas que utilizan Pipeline.
WinMIPS64	Simulador del juego de instrucciones de las máquinas MIPS, enfocado también a la comprensión del pipeline de las máquinas segmentadas.
MIDAS	Simulador de arquitectura superescalar
SIMDE	Simulador de Arquitecturas ILP con planificación estática y dinámica (Máquinas Superescalares y VLIW)

Tabla 1.1. Algunos simuladores relacionados con la arquitectura de ordenadores

¹ En la dirección web <http://www.sosresearch.org/caale/caalesimulators.html> [11] puede encontrarse una larga lista de simuladores relacionados con la arquitectura de ordenadores y enlaces para su descarga.

1.3. Justificación del proyecto

Cómo se puede observar en la tabla 1.1, el número de simuladores es elevado, lo que demuestra que son un recurso de mucha utilidad en el estudio de la arquitectura de ordenadores.

Sin embargo, la mayoría de ellos no han sido orientados a la docencia. Esa es la razón por la que no cuentan con una interfaz sencilla, que facilite el aprendizaje de un posible alumno, ni una ayuda que lo guíe eficazmente en el manejo del programa.

Así, se concibió la idea de desarrollar un simulador didáctico que fuera más allá de lo implementado hasta la fecha, incluyendo no sólo conceptos de memorias caché, como la mayor parte de los simuladores, sino todos los conceptos asociados con las jerarquías de memoria. Además, se quiso que en ningún momento se perdiera el enfoque didáctico de la herramienta y se consiguiera un software educativo de gran calidad, para lo que se siguieron las pautas del campo de estudio conocido como tecnología educativa.

Como un paso más en pos de asegurar la calidad, una vez finalizada la implementación del simulador, se desarrolló un proceso de evaluación tanto de los aspectos funcionales de la herramienta como de los aspectos educativos. Su objetivo era aprovechar que el desarrollo del proyecto coincidía con el periodo lectivo de la asignatura de Arquitectura de Computadores de la ETSII, para que los alumnos, usuarios finales de la herramienta, la validaran.

Para finalizar, dentro del proceso de evaluación se incluyó un estudio sobre el posible uso de la herramienta como un software tanto de enseñanza como de evaluación del alumnado. Para ello se combinó el simulador con otra herramienta didáctica desarrollada como proyecto en la ETSII, PORTAD², utilizando los test adaptativos conjuntamente con el simulador SIJEM en una prueba en el aula.

1.4. Resumen del documento

En el presente documento se hace un recorrido por todos los conceptos asociados por las jerarquías de memoria, pasando posteriormente a explicar, cómo estos han sido implementados en el simulador y mostrando la ayuda incluida en el programa.

Finalmente se analizan los resultados obtenidos en el proceso de evaluación de la herramienta como recurso didáctico, utilizando los conceptos agrupados bajo el campo de estudio de la Tecnología Educativa, y evaluando las posibilidades de combinación de ésta herramienta de simulación con otros recursos didácticos como los test adaptativos.

1.5. Disponibilidad del simulador

El simulador se encuentra disponible para su descarga a través del servidor FTP de la ETSII de la Universidad de La Laguna, en la siguiente dirección:

<ftp://ftp.etsii.ull.es/asignas/ARQUIT/SIJEM/>

² Portal para la realización de test adaptativos. Información adicional sobre PORTAD y los test adaptativos puede ser encontrada en la memoria del proyecto de fin de Carrera [9]

Capítulo 2.

Fundamentos teóricos

En este capítulo se hace un pequeño resumen de todos los conceptos teóricos incluidos en el simulador, ya que resultan imprescindibles para la comprensión total de la herramienta.

2.1. Jerarquías de memoria

Desde el inicio de la informática los usuarios y programadores han deseado disponer de una cantidad de memoria ilimitada que fuera lo más rápida posible. Sin embargo, una solución basada en el uso masivo de la memoria más rápida disponible, tendría un coste tan alto que la haría inviable.

Una solución más económica pasa por la implementación de una jerarquía de memoria, que se aproveche de la localidad de los programas y de las relaciones coste/rendimiento de diferentes tecnologías para *“proporcionar un sistema de memoria con un coste casi tan bajo como el del nivel más barato de memoria con una velocidad tan alta como la del nivel más rápido”*.¹

Una jerarquía de memoria se organiza en diferentes niveles, cada uno de ellos más pequeño pero más rápido que el anterior. La información contenida en cada nivel suele ser un subconjunto de la del nivel anterior, de manera que todos los datos de un nivel están a su vez en el nivel inferior.

La importancia de las jerarquías de memoria ha aumentado con el avance en prestaciones de los procesadores, cada vez más rápidos y más dependientes de la velocidad del sistema de memoria.

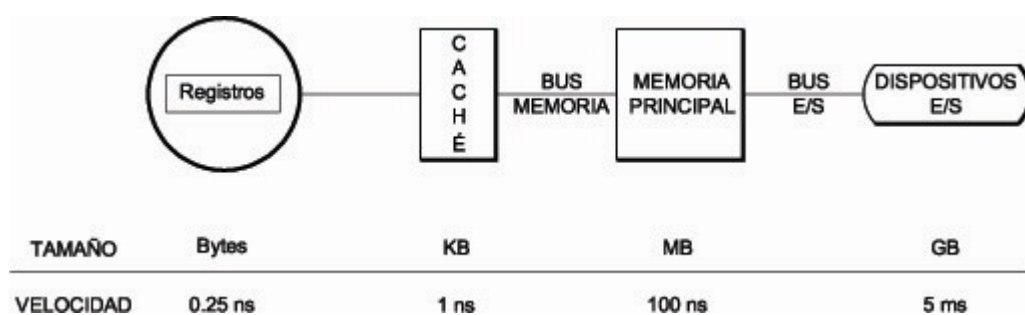


Figura 2.1. Esquema típico de una jerarquía de memoria

¹ Traducción de la cita de la página 390 de Computer Architecture: A Quantitative Approach [1]

2.2. Localidad de referencias

El análisis de los programas muestra que la mayor parte de su tiempo de ejecución se invierte en rutinas en las que muchas de sus instrucciones se ejecutan repetidas veces, de manera que instrucciones de zonas concretas del programa se ejecutan repetidamente durante largos periodos de tiempo mientras que al resto del programa se accede de manera poco frecuente. Este fenómeno se conoce como localidad de referencias y se manifiesta de dos formas:

- *Localidad temporal*: Es probable que si se referencia un elemento, éste vuelva a ser referenciado pronto.
- *Localidad espacial*: Es probable que si se referencia un elemento, los elementos próximos a él también sean referenciados pronto.

2.3. Memorias caché

La memoria principal es un recurso de acceso lento, lo que significa que el procesador debe esperar mucho tiempo cada vez que desea obtener un dato o instrucción desde la memoria. Para paliar el efecto de esta espera y así aumentar las prestaciones, se incluye en la jerarquía una memoria de acceso muy rápido que se intercala entre el procesador y la memoria principal, a la que denominamos caché. Su tamaño suele ser bastante menor que el de la memoria principal y normalmente se implementa utilizando memorias SRAM en lugar de DRAM².

Su función principal será almacenar la información usada en el pasado, pues según la propiedad de localidad de referencias, es muy posible que se necesite usarla en un futuro próximo.

2.3.1. Aciertos y fallos de caché

Cuando el procesador solicita un dato a la caché y éste se encuentra en la misma, se produce lo que denominamos “Acierto en caché” mientras que cuando no se encuentra, se produce un “Fallo de caché”. Esto provoca que una colección de datos denominada bloque, sea transferida desde la memoria principal hasta la memoria caché. En consecuencia se deduce que, para aumentar el rendimiento, debemos reducir el número de “Fallos de caché”.

2.3.2. Medida de prestaciones de las memorias caché

Para medir la influencia de la caché dentro de la jerarquía de memoria vamos a utilizar la fórmula de tiempo de CPU por instrucción:

$$\text{Tiempo CPU} = (\text{Ciclos de reloj para ejecutar una instrucción} + \text{Ciclos de reloj espera por memoria}) \times \text{Duración del ciclo de reloj}$$

² Más información sobre memorias DRAM y SRAM puede encontrarse en Carl Hamacher, Zvonko Vranesic, Safwat Zacky, *Organización de Computadores* [4]

Donde los *Ciclos de reloj para ejecutar una instrucción* representan el tiempo necesario para ejecutar una instrucción en caso de “Acuerdo de caché” y los *Ciclos de reloj espera por memoria* representan los ciclos que la CPU debe esperar a que el bloque sea transferido desde memoria principal a caché en caso de “Fallo de caché”.

Debemos obtener ahora por tanto los *Ciclos de reloj espera por memoria*. Para hacerlo nos valemos de la siguiente fórmula:

$$\begin{aligned} \text{Ciclos de reloj espera por memoria} = & \\ & (\text{Instrucciones lectura} / \text{Instrucciones de programa}) \\ & \times \text{Frecuencia fallos lectura} \times \text{Penalización por fallo de lectura} \\ & + (\text{Instrucciones escritura} / \text{Instrucciones de programa}) \\ & \times \text{Frecuencia fallos escritura} \times \text{Penalización por fallo de escritura} \end{aligned}$$

Simplificando los términos obtenemos esta otra fórmula:

$$\begin{aligned} \text{Ciclos de reloj espera por memoria} = & \\ & (\text{Instrucciones acceso a memoria} / \text{Instrucciones de programa}) \\ & \times \text{Frecuencia fallos} \times \text{Penalización por fallo} \end{aligned}$$

Añadiéndolo a la fórmula inicial queda de la siguiente forma:

$$\begin{aligned} \text{Tiempo CPU} = & (\text{Ciclos de reloj para ejecutar una instrucción} \\ & + (\text{Instrucciones acceso a memoria} / \text{Instrucciones de programa}) \\ & \times \text{Frecuencia fallos} \times \text{Penalización por fallo}) \\ & \times \text{Duración del ciclo de reloj} \end{aligned}$$

Vemos que dado que los *Ciclos de reloj para ejecutar una instrucción*, la relación $(\text{Instrucciones acceso a memoria} / \text{Instrucciones de programa})$ y la *Duración del ciclo de reloj* son independientes del sistema de memoria, son la *Frecuencia fallos* y la *Penalización de fallos* los parámetros que debemos reducir para incrementar las prestaciones.

2.3.3. Fuentes de fallos de caché

Un estudio del comportamiento de la caché revela que todos los fallos se producen debido a una de estas tres causas:

- *Primera referencia:* En el primer acceso a un bloque es seguro que éste no se encuentra en la memoria caché, ya que debe ser traído desde memoria principal a caché.
- *Capacidad:* Si la caché no puede contener todos los bloques necesarios durante la ejecución de un programa, se presentarán fallos que obligarán a descartar bloques que posteriormente deberán ser traídos de nuevo.
- *Conflicto:* Según el tipo de estrategia de colocación utilizado es posible que se puedan descartar bloques que posteriormente deban ser traídos de nuevo

Una caché eficiente deberá implementar por tanto mecanismos orientados a evitar estas situaciones.

2.3.4. Cachés divididas frente a cachés conjuntas

Un estudio más concienzudo de la localidad de referencias de los programas, muestra que éstos manifiestan diferentes comportamientos según se trate de instrucciones o datos. Por ello, las cachés a veces se dividen en caches de instrucciones y caches de datos, con lo que se tiene la posibilidad de optimizarlas de forma separada (diferentes capacidades, tamaños de bloque y asociatividades).

Aunque el hecho de dividir provoca un incremento en el coste, ya que es necesario duplicar memorias y buses, separando instrucciones y datos, conseguimos eliminar un gran número de conflictos y fijar un espacio en el que direcciones y datos aprovechen mejor la capacidad disponible.

2.3.5. Cachés multinivel

Actualmente los procesadores se van volviendo cada vez más rápidos, con lo que la tecnología necesaria para que la caché se mantenga a la altura y supere el creciente desnivel entre el procesador y la memoria principal es demasiado costosa.

Siguiendo entonces el principio de las jerarquías de memoria se recurre a añadir nuevos niveles entre la caché original y la memoria principal. Surgen entonces los “niveles de caché”, basados en introducir memorias más pequeñas pero más rápidas del lado del procesador y memorias mayores aunque sensiblemente más lentas del lado de la memoria principal.

De esta manera conseguimos llevar a cabo de forma eficiente el principio de las jerarquías de memoria, “*proporcionar un sistema de memoria con un coste casi tan bajo como el del nivel más barato de memoria con una velocidad tan alta como la del nivel más rápido*”.

En los sistemas actuales de escritorio es corriente encontrar dos niveles de memoria, subiendo hasta tres en el caso de servidores:

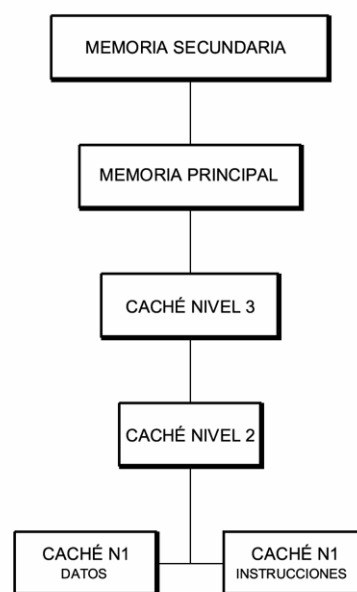


Figura 2.2. Jerarquía de memoria con caché multinivel

Si pasamos de nuevo a evaluar las prestaciones de las memorias caché multinivel, en este caso de dos niveles, nos encontramos con la siguiente fórmula:

$$\begin{aligned} \text{Tiempo medio de acceso a memoria} = \\ \text{Tiempo empleado si acierto en nivel 1} + \text{Frecuencia fallos nivel 1} \\ \times (\text{Tiempo empleado si acierto en nivel 2} + \\ \text{Frecuencia fallos nivel 2} \times \text{Penalización por fallo en nivel 2}) \end{aligned}$$

Cómo vemos, el éxito de la frecuencia de fallos del segundo nivel se mide con las sobras del primer nivel, lo cual da lugar a dos tipos de frecuencia de fallos que debemos reducir para aumentar las prestaciones:

- *Frecuencia local de fallos:* El número total de fallos dividido por el número total de accesos a esta caché.
- *Frecuencia global de fallos:* El número de fallos de la caché dividido por el número total de accesos a memoria generados por el procesador.

2.3.6. Políticas de mapeado

A la hora de decidir que bloques entrarán en memoria caché y cuáles permanecerán en memoria principal, se plantean el siguiente conjunto de estrategias:

- *Estrategias de colocación:* Deciden en que posición debe ser colocado el nuevo bloque dentro de la memoria caché.
- *Estrategias de búsqueda:* Determinan en qué posición se encuentra un determinado bloque de memoria caché.
- *Estrategias de reemplazamiento:* Deciden que bloque debe ser reemplazado cuando la posición es solicitada por un nuevo bloque.
- *Estrategias de coherencia:* Tratan la coherencia entre los datos que han sido modificados cuando el bloque se encontraba en caché y los datos contenidos en la memoria principal.

2.3.6.1. Estrategias de colocación

La organización de la memoria caché determina la posición en que un nuevo bloque puede ser colocado dentro de memoria caché. Así se distinguen tres tipos de estrategias de colocación.

- *Mapeado directo:* Cada bloque de memoria principal tiene un solo lugar donde ser colocado en memoria caché según la fórmula.

$$(\text{Bloque de memoria principal}) \text{ MOD } (\text{Número de bloques en caché})$$

- *Memoria completamente asociativa:* Un bloque de memoria principal puede ser colocado en cualquier lugar de la memoria caché.
- *Memoria asociativa por conjuntos:* A un bloque de memoria principal le corresponde un conjunto de bloques en memoria caché, de manera que puede ser colocado en cualquier bloque dentro del conjunto determinado por la fórmula.

(Bloque de memoria principal) MOD (Número de conjuntos en caché)

Cuando un conjunto tiene n bloques decimos que tenemos una memoria asociativa por conjuntos de n -vías, es decir, que tiene asociatividad n .

Como se puede deducir, las diferentes estrategias no son más que un incremento de los niveles de asociatividad por conjuntos: de manera que la caché de mapeado directo es simplemente una caché asociativa por conjuntos de una vía, mientras una memoria completamente asociativa con m bloques es una memoria caché asociativa por conjuntos de m vías.

En la siguiente figura se muestra qué posición ocuparía un hipotético bloque 12 según las diferentes organizaciones en una jerarquía compuesta por una caché de 8 bloques y una memoria principal de 32 bloques.

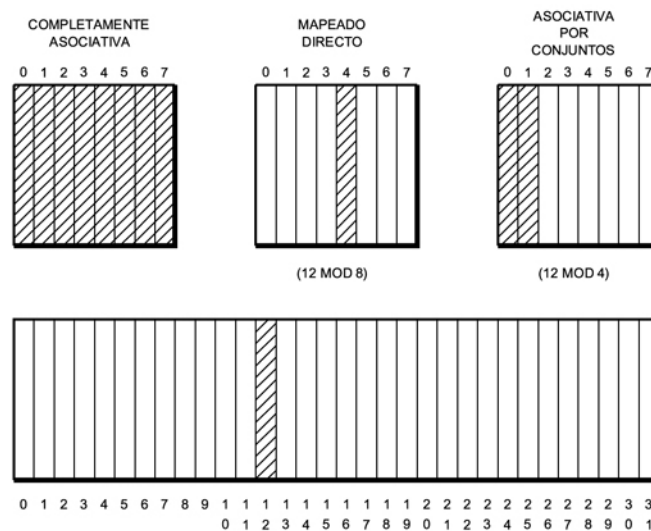


Figura 2.3. Estrategias de colocación

2.3.6.2. Estrategias de búsqueda

Para encontrar un bloque dentro de la caché, debemos encontrar primero la correspondencia entre la dirección de la CPU y la dirección de caché.

Mientras que una dirección de CPU está compuesta por bloque de memoria principal y desplazamiento del bloque, en las cachés, normalmente bastante más pequeñas que la memoria principal y teniendo en cuenta la asociatividad se componen de etiqueta, índice y desplazamiento.

El índice se utiliza para seleccionar el conjunto en mapeado directo y memoria asociativa por conjuntos, mientras que en memoria completamente asociativa no existe.

La etiqueta se incluye en un campo y se utiliza para comparar si el bloque encontrado es el bloque que buscamos.

Según esto podemos deducir que si mantenemos igual el tamaño de la caché, incrementamos la asociatividad, reducimos el tamaño del índice e incrementamos el tamaño de la etiqueta; aceleramos las búsquedas pero ralentizamos las comparaciones. Debemos por tanto encontrar un equilibrio para obtener las máximas prestaciones del sistema de caché.

ETIQUETA	ÍNDICE	DESPLAZAMIENTO
----------	--------	----------------

Figura 2.4. Bloque de memoria caché

2.3.6.3. Estrategias de reemplazamiento

En ocasiones el nuevo bloque que entra en caché se encuentra con que la posición ya está ocupada por un bloque anterior. Debe entonces reemplazarse un bloque, de decidir cual se van a encargar las estrategias de reemplazamiento.

Las estrategias de colocación influyen también en la decisión, así en el mapeado directo la posición de un bloque ya se encuentra determinada, de manera que el bloque antiguo en esa posición debe ser reemplazado con el nuevo.

Con las restantes estrategias de colocación deben tenerse en cuenta otros factores, pues es posible variar la posición del nuevo bloque dentro del conjunto, dando lugar a variadas estrategias de reemplazamiento tales como:

- *Óptima*: Esta estrategia nos dice que el rendimiento óptimo se obtendría cuando el bloque reemplazado fuese aquel que no se va a utilizar durante más tiempo en el futuro. Dado que los ordenadores no pueden conocer el futuro, no es una estrategia implementable en un sistema real.
- *Azar*: Esta estrategia decide el bloque a reemplazar al azar, no es la más eficiente pero su simplicidad de implementación y ausencia de contador o sello de tiempo la hacen válida para ciertos sistemas.
- *FIFO (First in – First out o Primero en entrar – Primero en salir)*: Esta estrategia reemplaza aquel bloque que ha estado durante más tiempo en memoria. Se implementa asociando a cada entrada un sello de tiempo que marca el instante en que el bloque ha entrado en memoria.
- *LRU (Least Recently Used o Menos recientemente usado)*: Esta estrategia reemplaza aquel bloque que lleva más tiempo en memoria sin ser usado. Se implementa asociando a cada entrada un sello de tiempo que marca el último instante en que el bloque fue usado.

- *Clock*: Según su forma de implementación esta estrategia puede ser una aproximación a FIFO o a LRU.
 - *Aproximación a FIFO*: Reemplaza aquel bloque que ha estado durante más tiempo en memoria. Se implementa utilizando un puntero que determina que bloque debe ser reemplazado avanzando siempre en el sentido de las agujas del reloj, de ahí que se denomine “Clock”, reloj en inglés.
 - *Aproximación a LRU*: Reemplaza aquel bloque que ha estado durante más tiempo en memoria sin ser usado. Se implementa, al igual que la aproximación a FIFO, utilizando un puntero que determina qué bloque debe ser reemplazado avanzando siempre en el sentido de las agujas del reloj, pero además añade un bit que marca si el bloque ha sido usado desde la última vez que el puntero pasó por allí, para evitar que un bloque recientemente usado sea reemplazado.
- *LFU (Least Frequently Used o Menos frecuentemente usado)*: Esta estrategia reemplaza aquel bloque que ha sido usado el menor número de veces desde que ha entrado en memoria. Se implementa asociando a cada entrada un contador del número de veces que el bloque ha sido usado.
- *NUR (Not used recently o No recientemente usado)*: Esta estrategia reemplaza aquel bloque que no ha sido usado recientemente. Para determinar esta situación utiliza dos factores: si el bloque ha sido referenciado y si el bloque ha sido modificado. Se implementa asociando a cada entrada una etiqueta de dos bits, lo que da lugar a cuatro posibles grupos de bloques para ser reemplazados.
 - Bit 1: Bit de referencia – (0 no referenciado y 1 referenciado)
 - Bit 2: Bit de modificación – (0 no modificado y 1 modificado)
 - 00 - Bloque no referenciado y no modificado
 - 01 - Bloque no referenciado pero modificado
 - 10 - Bloque referenciado pero no modificado
 - 11 - Bloque referenciado y modificado

Los bloques son reemplazados en este mismo orden.

2.3.6.4. Estrategias de coherencia

Dado que los datos contenidos en memoria caché son copias de los datos contenidos en memoria principal, debe existir algún método que nos permita mantener la coherencia entre el contenido de memoria principal y el de la memoria caché. De esto se encargan las estrategias de coherencia, que son de dos tipos:

- *Write Through o Escritura directa:* Una escritura en memoria actualiza tanto la copia en memoria principal como la copia en memoria caché.
 - *Victim Buffer:* Para minimizar el efecto de la lenta escritura en memoria principal o en niveles de caché de diferentes velocidades se suele utilizar una memoria (buffer) la cual permitirá actualizar la memoria principal y los niveles de caché más lentos sin que el procesador tenga que detener la ejecución.
- *Write Back o Escritura retardada:* Una escritura en memoria actualiza sólo la copia en caché, mientras que la copia de la memoria principal se actualizará cuando el bloque de caché sea reemplazado. La copia modificada de caché se marcará con un bit (*Bit dirty o Bit Sucio*) indicando que la copia en memoria principal no es válida.

Además de estas estrategias los sistemas de caché suelen implementar mecanismos para invalidar un determinado bloque de caché en los diferentes niveles de la jerarquía.

2.3.7. Bits de control

Cómo hemos visto, según las estrategias implementadas, cada entrada de memoria caché contará con una serie de bits, denominados bits de control.

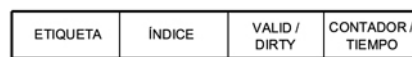


Figura 2.5. Bloque de memoria caché con bits de control

2.4. Memoria principal

Una vez estudiada la memoria caché, pasamos al siguiente nivel en la jerarquía, la memoria principal.

Esta memoria desempeña una doble función dentro de la jerarquía, por un lado satisface las demandas de las cachés, y por el otro sirve como interfaz de entrada/salida, ya que es tanto el destino de la entrada como la fuente para la salida.

2.4.1. Medida de prestaciones de la memoria principal

Desde el punto de vista de la caché debemos de tener en cuenta dos parámetros:

- *Tiempo de acceso:* Tiempo desde que se ordena una lectura hasta que se recibe el dato.
- *Duración del ciclo:* Tiempo mínimo transcurrido entre peticiones a memoria.

Este último parámetro se encuentra influenciado por otro, el tiempo de refresco, ya que actualmente las implementaciones de memoria principal se realizan en DRAM³. Esto implica que la memoria puede no estar disponible ocasionalmente porque se está enviando una señal que regenere de nuevo los datos.

El conjunto de éstos parámetros suele agruparse en lo que denominamos *latencia*.

Por otro lado, desde el punto de vista de los dispositivos de E/S el factor primordial es el *ancho de banda*, que determina las velocidades de transferencia con los dispositivos de entrada/salida.

Debemos por lo tanto, reducir la *latencia* y aumentar el *ancho de banda* para incrementar las prestaciones.

2.4.2. Anchura de la memoria principal

Las cachés a menudo se organizan con una anchura de una palabra, porque la mayoría de los accesos a la CPU son de ese tamaño, y a su vez la memoria principal tiene el ancho de una palabra para que coincida con la anchura de la caché.

Sin embargo, duplicar o cuadruplicar el ancho de la memoria disminuirá la penalización por fallos, por lo que el coste de implementar un bus más ancho puede verse compensado con el aumento de las prestaciones producido al reducir la *latencia* y aumentar el *ancho de banda*.

2.4.3. Memoria principal entrelazada

Otra posible medida para aumentar el *ancho de banda*, y a su vez reducir la *latencia*, puede ser utilizar una memoria entrelazada, en la que los chips de memoria se organizan en bancos para leer o escribir múltiples palabras a la vez en lugar de una. De esta manera se disminuye de forma significativa la penalización por fallos.

Aunque la implementación hardware de esta solución plantea ciertas limitaciones, y su comportamiento ante escrituras muy seguidas no demasiado bueno, se ve compensado por la mejora cuando se efectúan múltiples accesos secuenciales.

2.5. Memoria secundaria

La memoria principal no puede usarse para cubrir toda la capacidad de almacenamiento necesaria en un ordenador, debido al elevado coste por bit de información almacenada. Se hace por tanto necesario utilizar otras tecnologías que nos permitan disponer de grandes cantidades de memoria aunque éstas sean de acceso más lento. Aparece entonces este nivel en la jerarquía, al que denominamos memoria secundaria.

En este nivel se suelen utilizar discos y cintas magnéticos, discos ópticos y cada vez más, memorias flash en forma de dispositivos fijos o extraíbles. Éstos se fijan a un bus de entrada/salida (IDE, SCSI, FibreChannel, PCI, USB, IEEE1394-FireWire) gestionado por un controlador que organiza las transferencias, comprueba

³ Más información sobre memorias DRAM y SRAM puede encontrarse en Carl Hamacher, Zvonko Vranesic, Safwat Zacky, *Organización de Computadores* [4]

los errores y evita los conflictos; y que en muchas ocasiones incluye una memoria semiconductor (buffer) capaz de almacenar unos cuantos megabytes de datos, actuando como una memoria caché, que de nuevo aprovecha el principio de localidad de los programas.

2.6. Memoria virtual

Aunque actualmente es poco frecuente, pues ya es corriente contar con grandes cantidades de memoria física, puede suceder que un programa sea demasiado grande para ser alojado en memoria principal. Además suele ocurrir que los ordenadores se encuentren ejecutando múltiples procesos al mismo tiempo, ya que la mayoría de los sistemas operativos actuales son multiusuario y multitarea. En éstas ocasiones la suma del espacio ocupado por todos los procesos puede ser mayor que el tamaño de la memoria principal.

Para solucionar estos problemas y evitar al programador la tarea de identificar que partes de un programa deben convivir en memoria y cuáles no, se ideó un sistema de memoria denominado memoria virtual.

La memoria virtual permite disponer de la capacidad de direccionar un espacio de almacenamiento mayor que el disponible en la memoria principal de un determinado sistema. De esta manera, tenemos la impresión de disponer de más memoria física de la que realmente tenemos.

La memoria virtual se implementa aprovechando principalmente los dos niveles superiores de la jerarquía, la memoria principal y la memoria secundaria. Así, cuando se va a ejecutar un proceso, su código y datos pasan desde la memoria secundaria hasta la memoria principal siguiendo alguna estrategia que permita mantener sólo una pequeña porción del programa en memoria principal.

Además, también es posible dividir la memoria virtual de manera que asignemos porciones a cada usuario, para que pueda ejecutar sus procesos sin interferir con los de los demás usuarios.

Para ello, debe implementar una serie de mecanismos que le permitan gestionar estas tareas eficientemente.

2.6.1. Direcciones virtuales y direcciones reales

La clave de un sistema de memoria virtual está en discernir entre las direcciones virtuales y las direcciones reales:

- *Direcciones virtuales:* Son las direcciones generadas en la CPU por el proceso en ejecución.
- *Direcciones reales:* Son las direcciones físicas disponibles en la memoria principal del ordenador.

El sistema debe, por tanto, implementar un mecanismo que permita traducir entre los dos tipos de direcciones mientras el proceso se encuentra en ejecución.

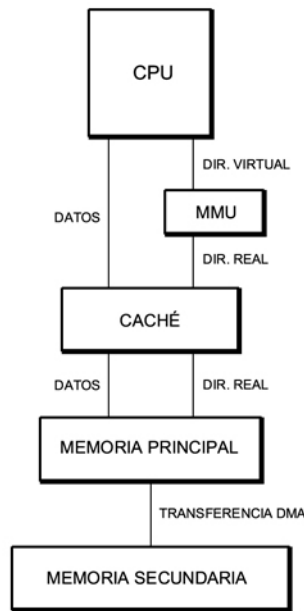


Figura 2.6. Transformación dinámica de direcciones

La tarea de realizar la traducción se suele implementar en una unidad hardware especial denominada MMU (Memory Management Unit o Unidad de Gestión de Memoria) para realizarla lo más rápidamente posible, y evitar la ralentización del sistema. A esta tarea se la suele conocer como “*Transformación dinámica de direcciones*”.

2.6.2. Transformación en bloques

Uno de los mecanismos de traducción dinámica de direcciones más utilizado es el de “*Transformación en bloques*”, que se basa en la idea de que el sistema de memoria virtual debe ser dividido en bloques, para evitar que la información del mapa de traducción sea demasiado voluminosa y difícil de manejar.

De esta manera, la información se agrupa en bloques que contienen elementos de información (instrucciones y datos), de los cuales el sistema conoce exactamente donde han sido colocados en la memoria principal o donde buscarlos en memoria secundaria si no se han cargado todavía.

El tamaño de estos bloques determina la técnica utilizada, distinguiéndose así dos tipos:

- *Paginación*: Compuesta por bloques de tamaño fijo (páginas).
- *Segmentación*: Compuesta por bloques de tamaño variable (segmentos).

La decisión de usar una técnica u otra lleva consigo una serie de ventajas y desventajas, que pasamos a enumerar en la siguiente tabla:

	Paginación	Segmentación
Palabras por dirección	Una	Dos (Segmento y desplazamiento)
¿Visible al programador?	No	Es posible
Reemplazamiento	Trivial, todos los bloques tienen el mismo tamaño	Difícil, debe localizarse una parte no utilizada y contigua en memoria principal
Uso de memoria	Fragmentación interna (Porciones de la página inutilizadas)	Fragmentación externa (Porciones de memoria principal sin usar)
Tráfico de disco	Eficiente (El tamaño de página se ajusta)	No siempre (Algunas transferencias llevan pocos datos)

Tabla 2.1. Paginación frente a segmentación

Dado que cada técnica aporta ventajas y desventajas, en los sistemas actuales se suele implementar un mecanismo mixto de paginación y segmentación (*Segmentación paginada*), en el que los bloques de tamaño variable (segmentos), alojan un grupo de bloques de tamaño variable (páginas). De esta manera, los segmentos tienen tamaños menos dispares, múltiplos del tamaño de página, que disminuyen la fragmentación externa y facilitan el reemplazamiento. Además, como contienen los datos asociados al proceso (segmento de código, segmento de datos) dentro del mismo segmento, aprovechan mejor las transferencias de disco.

En la siguiente sección vamos a dejar a un lado la segmentación, centrándonos en la paginación, ya que proporciona una manera más sencilla de comprender todos los mecanismos involucrados en un sistema de memoria virtual.

2.7. Paginación

Cómo ya se ha descrito, la paginación es un sistema de memoria virtual compuesto por bloques de tamaño fijo denominados páginas.

Dado que un proceso sólo puede ser ejecutado si su página actual se encuentra en memoria principal, las páginas deben ser transferidas desde memoria secundaria hasta memoria principal en bloques, llamados marcos de páginas, que tienen el mismo tamaño que las páginas. De esta manera, a cada página que se traslade a memoria principal, se le asignará un marco de página de entre todos los disponibles. De determinar cuál, se encargarán las estrategias de colocación que veremos más adelante.

Una dirección virtual en un sistema de paginación es un par ordenado (p, d) , donde p es el número de página en el sistema de memoria virtual en el que reside el elemento al que se está haciendo referencia y d es el desplazamiento dentro de la página p donde está el elemento referenciado.

2.7.1. Selección del tamaño de página

En un sistema de memoria virtual paginado el primer parámetro que incide en las prestaciones es el tamaño de página. Elegir su tamaño es cuestión de encontrar el equilibrio entre:

- Páginas grandes: favorecen que el tamaño de la tabla de páginas sea menor y la transferencia entre memoria secundaria y memoria principal sea más rápida.
- Páginas pequeñas: aunque incrementan el tamaño de la tabla de páginas, el porcentaje de memoria y ancho de banda malgastados en cada bloque disminuye, pues un bloque demasiado grande provoca que éste no se llene completamente provocando lo que se denomina “*Fragmentación interna*”.

De aquí concluimos que, según la naturaleza de los procesos y la arquitectura concretas de una máquina, puede resultar más ventajoso un tamaño de página grande que otro pequeño, o viceversa.

2.7.2. Mecanismos de traducción de direcciones

Existen varios mecanismos para llevar a cabo la traducción de direcciones virtuales a direcciones reales en los sistemas de memoria virtual paginados, los cuales se detallan a continuación.

2.7.2.1. Traducción de direcciones por transformación directa

Este mecanismo basa su funcionamiento en una tabla de páginas, normalmente alojada en la propia memoria principal.

La tabla de páginas almacena la siguiente información para cada una de las páginas de la memoria virtual de los procesos:

Bit de residencia de página	Dirección de almacenamiento secundario (si la página no está en el almacenamiento real)	Nº del marco de página (si la página está en el almacenamiento real)
r	s	p'

r = 0 si la página no está en el almacenamiento real
r = 1 si la página está en el almacenamiento real

Figura 2.7. Entrada de la tabla de páginas

Como vemos en la siguiente figura, la parte p de la dirección virtual se utiliza para indexar la tabla de páginas, alojada a partir de la dirección b en memoria principal, en busca de la entrada correspondiente para determinar si la página se encuentra o no en la memoria principal.

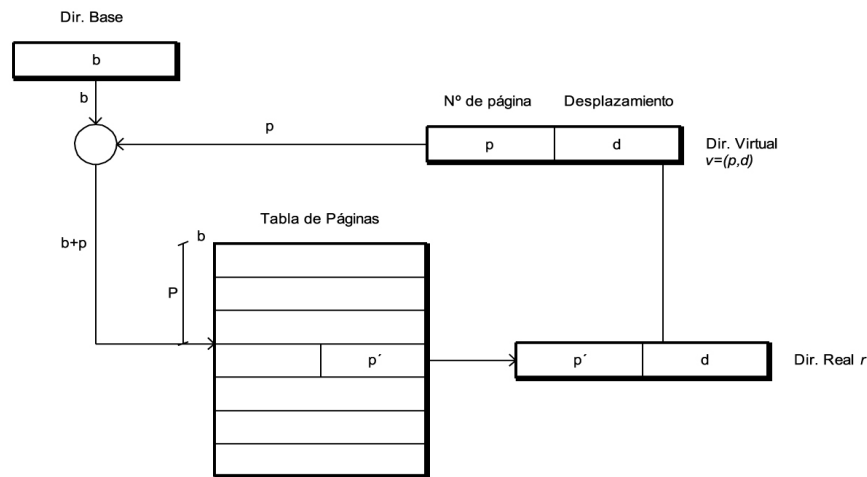


Figura 2.8. Traducción de direcciones por transformación directa

Si el bit de residencia r está a:

- 0 : La página no está en memoria principal, de manera que debemos transferirla desde el almacenamiento secundario en la dirección s hasta un marco libre en memoria principal y modificar la entrada de la tabla de páginas. A este caso lo denominamos "Fallo de página".
- 1 : La página está ya cargada en la memoria principal en el marco p' . Debemos entonces acceder a él y sumarle el desplazamiento d para obtener el dato o instrucción. A este caso lo denominamos "Acierto en tabla de páginas".

2.7.2.2. Traducción de direcciones por transformación asociativa

La transformación directa presenta el problema de que el acceso a la tabla de páginas es muy lento, ya que se encuentra en una memoria de acceso aleatorio.

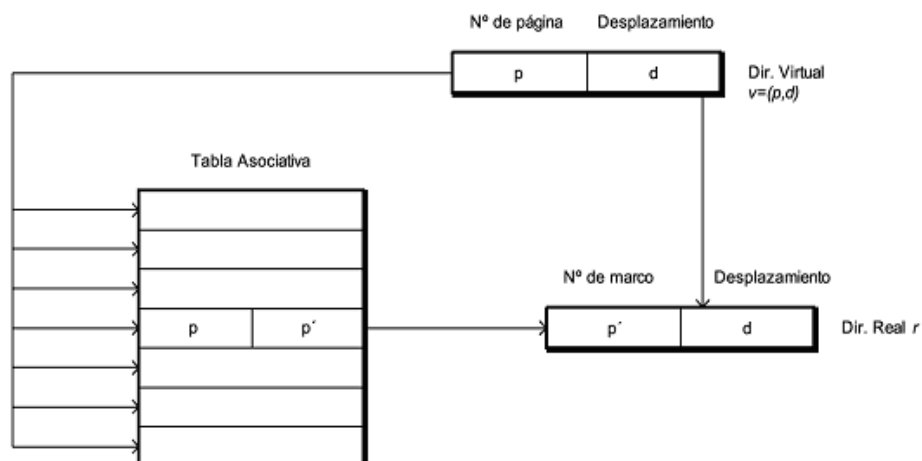


Figura 2.9. Traducción de direcciones por transformación asociativa

Una primera solución puede ser acelerar el acceso, colocando la tabla de páginas en una memoria asociativa en la que se acceda a todas las posiciones al mismo tiempo.

Sin embargo, dado el tamaño normal de las tablas de página, esta solución suele ser demasiado costosa, por lo que no se implementa en la realidad.

2.7.2.3. Traducción de direcciones por transformación asociativa-directa con TLB

Otra posible manera de acelerar las traducciones puede ser recordar las últimas traducciones incluyéndolas en una pequeña memoria asociativa de acceso muy rápido a la que se denomina TLB (Translation Lookaside Buffer o Buffer de traducciones anticipadas).

Esta técnica aprovecha el principio de localidad de los programas para obtener un rendimiento parecido al de la técnica anterior, pero con una memoria de tamaño muchísimo menor.

Para implementar la búsqueda usando una TLB hacemos uso del siguiente algoritmo:

- Extraemos el número de página p de la dirección virtual.
- Buscamos el número de página virtual dentro de la TLB:
 - Si se encuentra, accedemos al marco p' indicado de memoria principal y le sumamos el desplazamiento para obtener el dato o instrucción. A este caso lo denominamos “Acierto en TLB”.
 - Si no se encuentra, debemos acceder a la tabla de páginas para obtener el marco p' correspondiente, usando la parte p de la dirección virtual para indexar la entrada correspondiente de la tabla de páginas. Si el bit de residencia r está a:
 - 0 : La página no está en memoria principal, de manera que debemos transferirla desde el almacenamiento secundario en la dirección s hasta un marco libre en memoria principal y modificar las entradas de la tabla de páginas y de la TLB. A este caso lo denominamos “Fallo de página”.
 - 1 : La página está ya cargada en la memoria principal en el marco p' . Debemos entonces acceder a él y sumarle el desplazamiento d para obtener el dato o instrucción, y a su vez introducir la entrada en la TLB. A este caso lo denominamos “Acierto en tabla de páginas”.

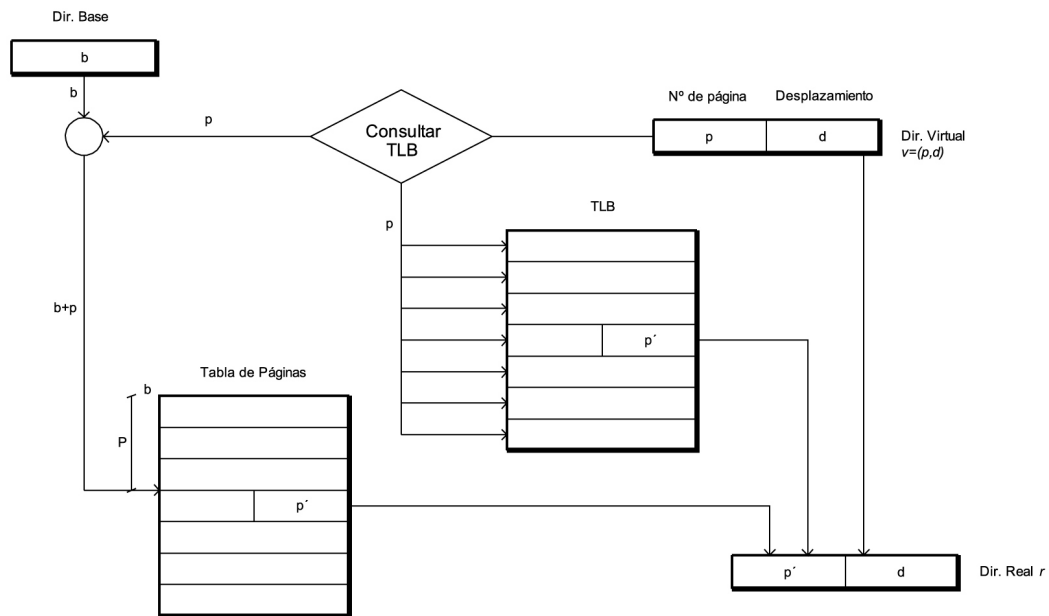


Figura 2.10. Traducción de direcciones por transformación asociativa-directa con TLB

Según el algoritmo anterior vemos que en los casos en que se produce “Fallo de página” o “Acierto en tabla de páginas” el tiempo efectuado en acceder a la TLB ha sido tiempo malgastado, por lo que debemos tratar de encontrar un tamaño de TLB adecuado a la localidad de los programas y la arquitectura concreta de la máquina.

Otro factor a tener en cuenta será la política de reemplazamiento que seguiremos cuando todas las entradas se encuentren llenas. En una TLB se pueden implementar políticas tales como:

- o *Azar*: Se reemplaza una entrada al azar de la TLB.
- o *FIFO (First in – First out o Primero en entrar – Primero en salir)*: La entrada a reemplazar es la que lleva más tiempo en la TLB.
- o *LRU (Least Recently Used o Menos recientemente usada)*: La entrada a reemplazar es la que lleva más tiempo sin usarse.
- o *LFU (Least Frequently Used o Menos frecuentemente usada)*: La entrada a reemplazar es la que ha sido usada un menor número de veces.

2.7.2.4. Traducción de direcciones por transformación asociativa-directa con TLB dividida

Una posible mejora sobre el esquema anterior puede ser el uso de dos TLB, una para almacenar las traducciones correspondientes a datos y otra para almacenar

las correspondientes a instrucciones. De esta manera podrían aprovecharse mejor las diferentes localidades de referencia de las instrucciones y los datos.

El funcionamiento es similar al caso anterior, pero efectuando la búsqueda en una TLB u otra según se trate de instrucciones o datos.

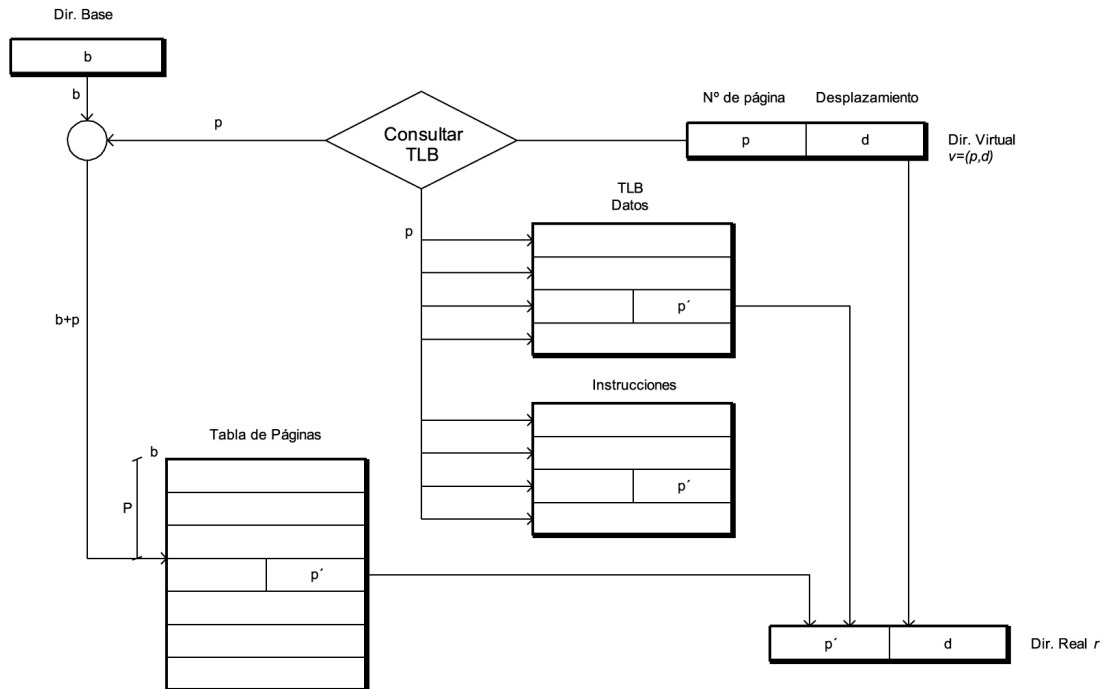


Figura 2.11. Traducción de direcciones por transformación asociativa-directa con TLB dividida

2.7.3. Estrategias de administración del sistema de memoria virtual paginado

A la hora de decidir que páginas entrarán en memoria principal y cuáles permanecerán en memoria secundaria se plantean el siguiente conjunto de estrategias:

- *Estrategias de búsqueda:* Deciden en que casos una nueva página debe ser traída desde memoria secundaria hasta memoria principal.
- *Estrategias de colocación:* Determinan en qué posición se colocará una nueva página en memoria principal.
- *Estrategias de reemplazamiento:* Deciden que página se debe trasladar desde memoria principal a memoria secundaria cuando es necesario el marco en memoria principal.

2.7.3.1. Estrategias de búsqueda

Para que un proceso pueda ejecutarse, es necesario que sus páginas se encuentren cargadas en memoria principal, y en los múltiples niveles de caché, si

estos existen. Por lo tanto, debe existir un mecanismo que determine qué páginas deben ser traídas desde memoria secundaria a memoria principal. Son las estrategias de búsqueda, que se dividen en:

- *Paginación por demanda:* Las páginas son trasladadas desde memoria secundaria a memoria principal cuando son referidas por el proceso en ejecución, presentando el problema de que al principio del programa el procesador debe esperar la transferencia página por página.
- *Paginación anticipada:* El sistema predice qué páginas deben ser trasladadas desde memoria secundaria a memoria principal sin que éstas tengan que ser referenciadas por el proceso en ejecución. Normalmente, para aprovechar la localidad de los programas, el sistema suele traer, además de la página referenciada:
 - La *Página anterior*: Es muy probable que la siguiente página referenciada sea la página anterior (bucles).
 - La *Página siguiente*: Es muy probable que la siguiente página referenciada sea la página contigua (datos contiguos).
 - Las *Páginas anterior y siguiente*: Es la combinación de las dos anteriores.

2.7.3.2. Estrategias de colocación

Cuando una página es trasladada desde memoria secundaria a memoria principal, debe determinarse en qué marco va a ser colocarla, de eso se encargan las estrategias de colocación.

Normalmente, puesto que al inicio de la máquina todas las posiciones de memoria están vacías, y cuando se llena son las estrategias de reemplazamiento las que determinan donde colocar la nueva página; se suele usar la estrategia *FIRST FIT* basada en ir colocando las páginas consecutivamente desde la posición de memoria más baja.

2.7.3.3. Estrategias de reemplazamiento

Normalmente no todas las páginas de un proceso caben en memoria principal, o el proceso en ejecución cambia y es necesario cargar todas sus páginas. Por ello, debe implementarse un mecanismo eficiente para gestionar los casos en los que todos los marcos de página se encuentran ocupados y es necesario trasladar una nueva página desde memoria secundaria hasta memoria principal. De decidir qué página reemplazar, se encargan las estrategias de reemplazamiento, que pueden ser de diferentes tipos:

- *Óptima:* Esta estrategia nos dice que el rendimiento óptimo se obtendría cuando la página reemplazada fuese aquella que no se va a utilizar

durante más tiempo en el futuro. Dado que los ordenadores no pueden conocer el futuro, no es una estrategia implementable en un sistema real.

- *Azar*: Esta estrategia decide la página a reemplazar al azar. No es la más eficiente, pero su simplicidad de implementación y ausencia de contador o sello de tiempo la hacen válida para ciertos sistemas.
- *FIFO (First in – First out o Primera en entrar – Primera en salir)*: Esta estrategia reemplaza aquella página que ha estado durante más tiempo en memoria. Se implementa asociando a cada entrada un sello de tiempo que marca el instante en que la página ha entrado en memoria.
- *LRU (Least Recently Used o Menos recientemente usada)*: Esta estrategia reemplaza aquella página que lleva más tiempo en memoria sin ser usada. Se implementa asociando a cada entrada un sello de tiempo que marca el último instante en que la página fue usada.
- *Clock*: Según su forma de implementación esta estrategia puede ser una aproximación a FIFO o a LRU.
 - *Aproximación a FIFO*: Reemplaza aquella página que ha estado durante más tiempo en memoria. Se implementa utilizando un puntero que determina que página debe ser reemplazada avanzando siempre en el sentido de las agujas del reloj, de ahí que se denomine “Clock”, reloj en inglés.
 - *Aproximación a LRU*: Reemplaza aquella página que ha estado durante más tiempo en memoria sin ser usada. Se implementa, al igual que la aproximación a FIFO, utilizando un puntero que determina que página debe ser reemplazada avanzando siempre en el sentido de las agujas del reloj; pero además añade un bit que marca si la página ha sido usada desde la última vez que el puntero pasó por allí, para evitar que una página recientemente usada sea reemplazada.
- *LFU (Least Frequently Used o Menos frecuentemente usada)*: Esta estrategia reemplaza aquella página que ha sido usada el menor número de veces desde que ha entrado en memoria. Se implementa asociando a cada entrada un contador del número de veces que la página ha sido usada.
- *NUR (Not used recently o No recientemente usada)*: Esta estrategia reemplaza aquella página que no ha sido usada recientemente. Para determinar esta situación utiliza dos factores: si la página ha sido referenciada y si la página ha sido modificada. Se implementa asociando a cada entrada una etiqueta de dos bits, lo que da lugar a cuatro posibles grupos de páginas para ser reemplazados.
 - Bit 1: Bit de referencia – (0 no referenciado y 1 referenciado)

- o Bit 2: Bit de modificación – (0 no modificado y 1 modificado)
- o 00 - Página no referenciada y no modificada
- o 01 - Página no referenciada pero modificada
- o 10 - Página referenciada pero no modificada
- o 11 - Página referenciada y modificada

Las páginas son reemplazadas en este mismo orden.

2.7.4. Prestaciones del sistema de memoria virtual

Cómo se ha visto en esta sección, son muchos los mecanismos involucrados en un sistema de memoria virtual, de manera que las características de cada uno influyen en las prestaciones globales del sistema.

Algunas de las técnicas vistas mejoran sensiblemente las prestaciones, aprovechando sobre todo la localidad de los programas, pero añaden complejidad al hardware. De manera que los mecanismos elegidos para implementar un sistema real deben representar el equilibrio entre coste del hardware y prestaciones globales.

2.8. Un poco de historia

Aunque desde un principio se supo que disponer de enormes cantidades de memoria sería una necesidad, y que posiblemente las jerarquías de memoria fueran el camino para cubrirla, la implementación generalizada no se produjo hasta que las máquinas Atlas e IBM 360 demostraron su gran eficacia.

Desde entonces, sólo algunos superordenadores, máquinas embebidas o viejos ordenadores personales, prescinden de las jerarquías de memoria.

Hoy día, gracias a los avances en tecnología de procesadores, los niveles superiores de la jerarquía, la memoria caché de nivel 1 y de nivel 2, suelen ir integrados en la misma pastilla. Gracias a ello, pueden funcionar a la mitad o la misma velocidad que el núcleo del procesador, mientras que los siguientes niveles suelen añadirse en forma de módulos de memoria a la placa madre del sistema.

Son también muchos los avances que se han introducido para mejorar el ancho de banda y la latencia de la memoria principal, apareciendo cada año memorias más rápidas:

- Memorias SDR: desde 66 hasta 133Mhz
- Memorias RAMBUS: 800 y 1066Mhz
- Memorias DDR: desde 266 hasta 400Mhz
- Memorias DDR2: 533Mhz

También los dispositivos de memoria secundaria y buses de E/S están en plena evolución, con la generalización del almacenamiento en memorias flash de gran tamaño y el almacenamiento en unidades compartidas en redes de alta velocidad.

Para demostrar el uso masivo de las jerarquías de memoria en los sistemas actuales basta con observar la siguiente tabla:

	Intel Pentium 3	Intel Pentium 4	AMD Athlon	IBM PowerPC 405	Sun UltraSparc III
Juego de instrucciones	80x86			PowerPC	Sparc v9
Posibles aplicaciones	Sistemas de escritorio y servidores			Sistemas embebidos	Servidores
Velocidad reloj	Hasta 1.4Ghz	Por encima de 3Ghz	Por encima de 2Ghz	266Mhz	900Mhz
Caché L1 instrucciones	16Kb asociativa por cjtos de 2 vías	96KB Trace Caché	64Kb asociativa por cjtos de 2 vías	16Kb asociativa por cjtos de 2 vías	32Kb asociativa por cjtos de 4 vías
Caché L1 datos	16Kb asociativa por cjtos de 2 vías	8Kb asociativa por cjtos de 4 vías	64Kb asociativa por cjtos de 2 vías	8Kb asociativa por cjtos de 2 vías	64Kb asociativa por cjtos de 4 vías
TLB Datos / Instrucciones	32 / 64	128	280 / 288	4 / 8	128 / 512
Tamaño mínimo página	8 KB	8 KB	8 KB	1 KB	8 KB
Caché L2	256Kb asociativa por cjtos de 8 vías	256 / 2048 Kb asociativa por cjtos de 8 vías	256 / 512 Kb asociativa por cjtos de 16 vías	No tiene	8192 Kb mapeado directo
Caché L3	No tiene	En XEON hasta 2048 kb asociativa por cjtos de 8 vías	No tiene	No tiene	No tiene
Tamaño del bloque caché	32	64 / 128	64	32	32
Ancho banda memoria en bits	64	64	64	64	128
Velocidad memoria	133Mhz	400 a 1066 Mhz	100 a 400 Mhz	133 Mhz	150 Mhz

Tabla 2.2. Jerarquías de memoria en los sistemas actuales

Capítulo 3.

Especificación de requisitos

Cómo paso previo a la realización del simulador, se establecieron una serie de requisitos que deberían ser cumplidos por el simulador para llevar a cabo su función didáctica.

3.1. Requisitos funcionales

Como requisitos funcionales se estableció que:

- El programa contaría con una estructura coherente y robusta que permitiera garantizar una buena base sobre la que realizar las simulaciones.
- Deberían incluirse el mayor número de conceptos relacionados con las jerarquías de memoria.
- Los parámetros de configuración podrían ser introducidos a mano o mediante ficheros de configuración predefinidos.
- Las trazas de los programas necesarios para llevar a cabo la simulación se incluirían en ficheros estructurados que pudieran ser creados manualmente por el usuario.
- La simulación incluiría la posibilidad de avanzar, volver atrás, parar, reiniciar y variar la velocidad en cualquier momento.

3.2. Requisitos no funcionales

Dado que se trata de un simulador orientado a usuarios de distinto tipo y es de vital importancia su capacidad didáctica, se consideraron primordiales las condiciones de usabilidad y apariencia.

Así se orientó el diseño del simulador a conseguir un programa:

- *Visual:* El programa mostraría cada uno de los pasos de la simulación apoyándose en todas las ayudas gráficas posibles para conseguir un entorno de fácil comprensión para el usuario.
- *Informativo:* El programa mostraría en todo momento toda la información implicada en cada paso de la simulación.
- *Configurable:* El programa contaría con la posibilidad de configurar el mayor número de parámetros que influyan en la simulación.
- *Fácil de usar:* El programa contaría con todas las ayudas posibles para proporcionar al usuario un entorno de fácil manejo en el que realizar la simulación.
- *Modulable:* El programa sería diseñado como una serie de módulos independientes que desempeñan funciones específicas que faciliten su revisión y ampliación futura.
- *Ampliable:* El programa sería un proyecto abierto que permitiera incluir nuevas características a posteriori.
- *Documentado:* El programa contaría con una documentación que ayudara al usuario en su uso y le proporcionase información de todos los conceptos implicados.

Capítulo 4.

Decisiones de diseño

Una vez establecidos los requisitos, comenzó el proceso de implementación del simulador. En esta etapa debieron tomarse una larga serie de decisiones importantes, encaminadas a conseguir una herramienta muy didáctica, a la vez que altamente funcional.

4.1. Elección del lenguaje de programación

Una de los requisitos básicos del simulador era que fuera una herramienta muy visual, de manera que dentro de las opciones de programación posibles se contemplaron:

- Utilizar *programación web*: pese a las ventajas de portabilidad, distribución y accesibilidad del programa que proporciona la programación web, la velocidad de ejecución que supondría el uso de lenguajes interpretados como Java mermarían la funcionalidad del programa.
- Utilizar un *lenguaje visual* como Visual Basic, Borland Delphi Pascal, Borland C++ Builder, etc... donde se perdía en portabilidad (aplicaciones Windows, aunque también para Linux gracias a Borland Kylix) pero se ganaba en funcionalidad, velocidad de ejecución y eficiencia en el uso de memoria.

Al final la elección se decantó hacia Borland C++ Builder, principalmente por las siguientes características:

- *Basado en C++*, un lenguaje de programación orientado a objetos muy robusto y eficiente.
- *Entorno de desarrollo (IDE) muy completo*, conteniendo componentes que facilitan la tarea de desarrollo.
- *Fácil de modular*, gracias a su naturaleza orientada a objetos.
- *Fácil de depurar*, gracias a las múltiples herramientas que incluye.

4.2. Características del simulador

Como se mencionó en el capítulo 2, son muchos los conceptos involucrados en una jerarquía de memoria, así que teniendo en cuenta que se debía implementar un simulador lo más didáctico y funcional posible, se optó por dividirlo en dos grandes partes:

- *Traducción de direcciones:* Aquí se contemplan todos los conceptos asociados a los mecanismos de traducción de direcciones virtuales a direcciones reales. En el simulador se implementaron 3 mecanismos:
 - *Traducción de direcciones por transformación directa*
 - *Traducción de direcciones por transformación asociativa-directa con TLB*
 - *Traducción de direcciones por transformación asociativa-directa con TLB dividida en datos e instrucciones*
- *Búsqueda de páginas:* Aquí se contemplan todos los conceptos asociados a la búsqueda de páginas y bloques en la jerarquía de memoria. En el simulador se implementa una jerarquía de memoria paginada con caché multinivel que incluye:
 - *Memoria secundaria*
 - *Memoria principal*
 - *Caché de nivel 3*
 - *Caché de nivel 2*
 - *Caché de nivel 1 conjunta o separada en datos e instrucciones*

En esta sección del simulador se contemplan las siguientes estrategias:

- Para el sistema de memoria virtual
 - *Búsqueda:* Paginación por demanda y paginación anticipada cargando la página siguiente a la referenciada.
 - *Colocación:* FIRST FIT
 - *Reemplazamiento:* FIFO (First In – First Out)
- Para los diferentes niveles de caché

- o *Colocación*: Mapeado directo, completamente asociativa y asociativa por conjuntos.
- o *Búsqueda*: División de bloques en índice (Index) y etiqueta (Tag).
- o *Reemplazamiento*: Azar, FIFO, LRU, Clock, LFU, NUR.
- o *Coherencia*: Escritura directa y escritura retardada.

Dado que en la búsqueda de páginas también es necesaria la traducción de direcciones, ésta se implementó como un módulo separado dentro del código de la aplicación. Así, cuando el usuario elige la *Búsqueda de páginas*, el simulador efectúa la traducción de direcciones de forma transparente al usuario proporcionándole directamente la dirección real, aunque si tiene dudas sobre el origen de ésta puede dirigirse a la *Traducción de direcciones*.

4.3. Control de la simulación

Otra de las decisiones importantes que se hubo de tomar fue determinar cómo proporcionar al usuario un control de la simulación que fuese funcional y cómodo a la vez.

La solución adoptada fue dividir la simulación en una serie de pasos que mostrasen el estado de la memoria en ese instante destacando las posiciones afectadas, la acción realizada o el suceso ocurrido, y una serie de estadísticas.

Además, se optó por incluir una novedad dentro de los simuladores, la posibilidad de ir tanto hacia adelante como hacia atrás. Este método permite que los usuarios, en el caso de que den un paso hacia adelante y no comprendan lo sucedido, puedan volver atrás uno o varios pasos, para tener una visión global del suceso.

Finalmente, dado que los ficheros de traza pueden ser bastante largos, se incluyó también un modo automático en el que el simulador da pasos a intervalos de tiempo fijos, que además el usuario puede modificar.



Captura 4.1. Control de la simulación

4.4. Asistente

Una de las características principales del simulador es la gran cantidad de parámetros configurables (tamaños de las diferentes memorias, bloques y páginas,

tiempos de acceso, caches habilitadas o deshabilitadas, algoritmos de colocación, reemplazamiento y coherencia, etc...) que le otorgan una elevada funcionalidad.

Sin embargo, esta gran cantidad de parámetros podría llegar a confundir al usuario. Por lo que se buscó una manera de guiar al usuario durante el proceso, optando por crear un asistente que lo ayudara en el proceso de configuración.

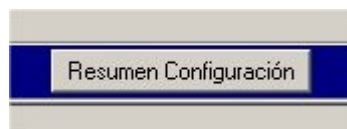
Dado que además una vez configurado el simulador, se debían de elegir los ficheros de traza entre múltiples opciones y posteriormente el tipo de simulación a realizar, se dividió al asistente en tres grandes fases:

- *Configurar*, a través de una serie de pasos, guía al usuario en el proceso de configurar todos los parámetros del simulador.
- *Cargar fichero*, aquí el usuario elige el fichero de traza entre los múltiples ejemplos incluidos en el simulador o bien, los propios ficheros creados por él mismo.
- *Simular*, aquí el usuario elige el tipo de simulación a realizar.



Captura 4.2. Asistente

Además se incluye el botón *Resumen configuración*, que muestra en cualquier momento un resumen de la configuración elegida durante el proceso de configuración.



Captura 4.3. Resumen de configuración

4.4.1. Asistente - Configurar

Para guiar al usuario en el proceso de configurar el simulador, se agruparon los parámetros según la siguiente secuencia:

- *Memoria virtual / secundaria*
- *Memoria principal*
- *Tamaño de página*

- *Mecanismo de traducción de direcciones*
- *Parámetros de paginación*
- *Tamaño del bloque de caché*
- *Caché de nivel 3* (Tamaño, organización, número de vías, política de reemplazamiento)
- *Caché de nivel 2* (Tamaño, organización, número de vías, política de reemplazamiento)
- *Caché de nivel 1* (Tamaño, organización, número de vías, política de reemplazamiento)

De esta manera, los grupos se van mostrando al usuario en una serie ordenada de pantallas, donde el usuario rellena los parámetros y puede:

- Pasar a la *Siguiente* pantalla: los datos serán validados, si son erróneos se mostrarán al usuario los errores para que los pueda resolver, y si son correctos, se mostrará la nueva pantalla con los datos a rellenar.
- Volver a la pantalla *Anterior*: si el usuario se percata de que ha cometido algún error o ha cambiado de parecer, puede volver a la pantalla anterior y cambiar aquellos parámetros que considere necesarios.
- Solicitar *Ayuda*: en todo momento el usuario tendrá disponible un botón de ayuda sobre el que pulsar para acceder a la sección de la ayuda correspondiente a la pantalla que está viendo en ese instante.



Captura 4.4. Resumen de configuración

Dado que rellenar a mano cada uno de los parámetros podría resultar una tarea tediosa, se incluyó la opción de cargar ficheros de configuración creados a mano por el usuario o incluidos como ejemplo y que contuviesen los parámetros necesarios para la simulación.

4.4.1.1. Ficheros de configuración

Los ficheros de configuración son ficheros de texto con extensión .cfg que contienen todos los parámetros necesarios para realizar la simulación. Estos ficheros pueden cargarse al inicio del asistente, de manera que los parámetros de las

diferentes pantallas son introducidos automáticamente, aunque el usuario tiene la posibilidad de cambiarlos.

La estructura de un fichero de configuración que contenga todos los posibles parámetros sería la siguiente:

- /Tamaño total de memoria secundaria (en Kbytes)
- 1:
- /Tiempo acceso a memoria secundaria (en ciclos)
- 2:
- /Ancho bus Memoria Secundaria-Memoria principal (en bits)
- 3:
- /Tiempo acceso bus Memoria Secundaria-Memoria principal (en ciclos)
- 4:
- /Tamaño total memoria principal (en Kbytes)
- 5:
- /Tamaño página (en Kbytes)
- 6:
- /Tiempo acceso memoria principal (en ciclos)
- 7:
- /Traducción direcciones
- /(1 - Directa, 2 - Asociativa directa con 1 TLB, 3 - Asociativa directa con 2 TLB)
- 8:
- /Número de entradas de la TLB
- 9:
- /Tiempo de acceso a la TLB (en ciclos)
- 10:
- /Tipo de paginación 1
- /(1 - Por demanda, 2 - Anticipada)
- 11:
- /Tipo de paginación 2
- /(1 - Escritura directa, 2 - Escritura retardada)
- 12:
- /Caché habilitada
- /(0 - NO, 1 - SI)
- 13:
- /Tamaño del bloque de caché (en bytes)
- 14:
- /Caché de nivel 3 habilitada?
- /(0 - NO, 1 - SI)
- 15:
- /Organización de la cache de nivel 3
- /(1 - Mapeado directo, 2 - Completamente asociativa, 3 - Asociativa por conjuntos)
- 16:
- /Política de reemplazamiento de cache de nivel 3
- /(1 - AZAR, 2 - FIFO, 3 - LRU, 4 - LFU, 5 - NUR, 6 - CLOCK)
- 17:
- /Tamaño de la caché de nivel 3 (en Kbytes)
- 18:
- /Tiempo de acceso a la caché de nivel 3 (en Ciclos)
- 19:
- /Ancho bus memoria caché nivel 3 a nivel superior (en Bits)
- 20:
- /Tiempo acceso bus memoria caché nivel 3 a nivel superior (en Ciclos)
- 21:
- /Número de vías de la caché de nivel 3 (si es asociativa por conjuntos)
- 22:
- /Caché de nivel 2 habilitada?

/(0 - NO, 1 - SI)
-23:
/Organización de la cache de nivel 2
/(1 - Mapeado directo, 2 - Completamente asociativa, 3 - Asociativa por conjuntos)
-24:
/Política de reemplazamiento de cache de nivel 2
/(1 - AZAR, 2 - FIFO, 3 - LRU, 4 - LFU, 5 - NUR, 6 - CLOCK)
-25:
/Tamaño de la caché de nivel 2 (en Kbytes)
-26:
/Tiempo de acceso a la caché de nivel 2 (en Ciclos)
-27:
/Ancho bus memoria caché nivel 2 a nivel superior (en Bits)
-28:
/Tiempo acceso bus memoria caché nivel 2 a nivel superior (en Ciclos)
-29:
/Número de vías de la caché de nivel 2 (si es asociativa por conjuntos)
-30:
/Caché de nivel 1 habilitada?
/(0 - NO, 1 - SI)
-31:
/Tipo de caché de nivel 1
/(1 - Conjunta, 2 - Separada)
-32:
/Organización de la cache de nivel 1
/(1 - Mapeado directo, 2 - Completamente asociativa, 3 - Asociativa por conjuntos)
-33:
/Política de reemplazamiento de cache de nivel 1
/(1 - AZAR, 2 - FIFO, 3 - LRU, 4 - LFU, 5 - NUR, 6 - CLOCK)
-34:
/Tamaño de la caché de nivel 1 (en Kbytes)
-35:
/Tiempo de acceso a la caché de nivel 1 (en Ciclos)
-36:
/Ancho bus memoria caché nivel 1 a nivel superior (en Bits)
-37:
/Tiempo acceso bus memoria caché nivel 1 a nivel superior (en Ciclos)
-38:
/Número de vías de la caché de nivel 1 (si es asociativa por conjuntos)
-39:

Ejemplo 4.1. Fichero de configuración

Las líneas que comienzan con ‘/’ se consideran líneas de comentario y no son necesarias para el correcto funcionamiento del fichero.

Cómo se puede observar, son muchos los parámetros que se pueden configurar para una simulación. Sin embargo, no todos son necesarios en algunos casos, ya que pueden existir ficheros de configuración más cortos.

Dentro del simulador se incluyeron una serie de ficheros de configuración de ejemplo para *Traducción de direcciones* y *Búsqueda de páginas*. El contenido de los ficheros puede consultarse en la ayuda del programa.

4.4.2. Asistente – Cargar fichero

Para realizar una buena simulación es necesario tener ficheros de traza que se correspondan con programas reales.

La solución adoptada fue obtenerlos a través del portal de distribución de trazas del PEL (Performance Evaluation Laboratory) de la Bringman Young University en Washington¹.

Este portal se actualiza frecuentemente con nuevas trazas de programas, en su mayor parte benchmarks, ejecutados sobre plataformas Windows y Linux que cubren los diferentes tipos de aplicaciones más utilizadas en computación: compiladores, editores, traductores, bases de datos, programas cad, diseño 3D, servidores web, de correo y ftp, etc...

4.4.2.1 Ficheros de traza

Dado que se trataba de efectuar simulaciones que mostraran las características de diferentes tipos de programas se optó por incluir dentro del simulador las trazas correspondientes a 6 tests del benchmark SPEC2000².

	<i>Descripción</i>
Crafty	Programa de ajedrez
Eon cook	Programa de trazado de rayos sobre una tetera
Facerec	Programa de reconocimiento de caras
Gcc	Compilador de C y C++
Mesa	Librería de gráficos 3D
Vortex	Base de datos

Tabla 4.1. Ficheros de traza de ejemplo

De cada uno de estos 6 ficheros se realizaron 5 versiones, para poder así contar con trazas en diferentes formatos de direcciones.

<i>Terminado en</i>	<i>Direcciones</i>	<i>Memoria virtual</i>
a	32 bits	Hasta 4GB
b	28 bits	Hasta 256MB
c	24 bits	Hasta 16MB
d	20 bits	Hasta 1 MB
e	16 bits	Hasta 64KB

Tabla 4.2. Formatos de direcciones

¹ El portal se encuentra en la dirección web <http://tds.cs.byu.edu/tds/>

² Más información sobre el benchmark puede encontrarse en la dirección web <http://www.spec.org> [10]

Estos ficheros, que no son más que ficheros de texto con extensión .trd, contienen 100 direcciones virtuales referenciadas por el programa, según la siguiente estructura:

- DIRECCION: Contiene una dirección virtual de X bit que representa la dirección generada por el programa en ejecución.
- TIPO DE ACCESO: Indica el tipo de acceso a memoria, que puede ser:
 - o FETCH - Búsqueda de instrucciones.
 - o MEMREAD - Lectura de memoria.
 - o MEMWRITE - Escritura en memoria.
- TIEMPO: Indica el tiempo (Delta) transcurrido desde la última petición de página en ciclos de reloj.

06709ea0	MEMWRITE	103
0776acf8	MEMREAD	263
07781770	FETCH	143
0776aa40	MEMREAD	119

Ejemplo 4.2. Extracto de un fichero de traza

Dado que no siempre es posible obtener buenos ficheros de traza que contengan la serie de propiedades que queremos estudiar, se optó por incluir además otro tipo de ficheros, los ficheros de página. Éstos, también son ficheros de texto, aunque con extensión .trp, que siguen la siguiente estructura:

- PÁGINA: Contiene la página referenciada por el programa en ejecución.
- TIPO DE ACCESO: Indica el tipo de acceso, que puede ser:
 - o FETCH - Búsqueda de instrucciones.
 - o MEMREAD - Lectura de memoria.
 - o MEMWRITE - Escritura en memoria.
- DIRECCION: Contiene una dirección virtual de X bit que representa la dirección generada por el programa en ejecución. En caso de no conocerla puede ser 0 siempre.

Estos ficheros pueden ser creados fácilmente por el usuario para estudiar el comportamiento, por ejemplo de páginas adyacentes.

0	FETCH	1821
1	MEMREAD	37
2	MEMREAD	1792
3	FETCH	29
4	MEMWRITE	15853
1	MEMREAD	37

Ejemplo 4.3. Extracto de un fichero de páginas

4.4.3. Asistente – Simular

Una vez realizada la configuración y elegido el fichero de traza es el momento de comenzar la simulación. En este paso del asistente elegimos el tipo de simulación a realizar.

4.5. Simulación

Cómo se describió en la sección 4.2 la simulación se divide en dos partes:

- *Traducción de direcciones.*
- *Búsqueda de páginas.*

A continuación vamos a pasar a detallar cada una de ellas.

4.5.1. Traducción de direcciones

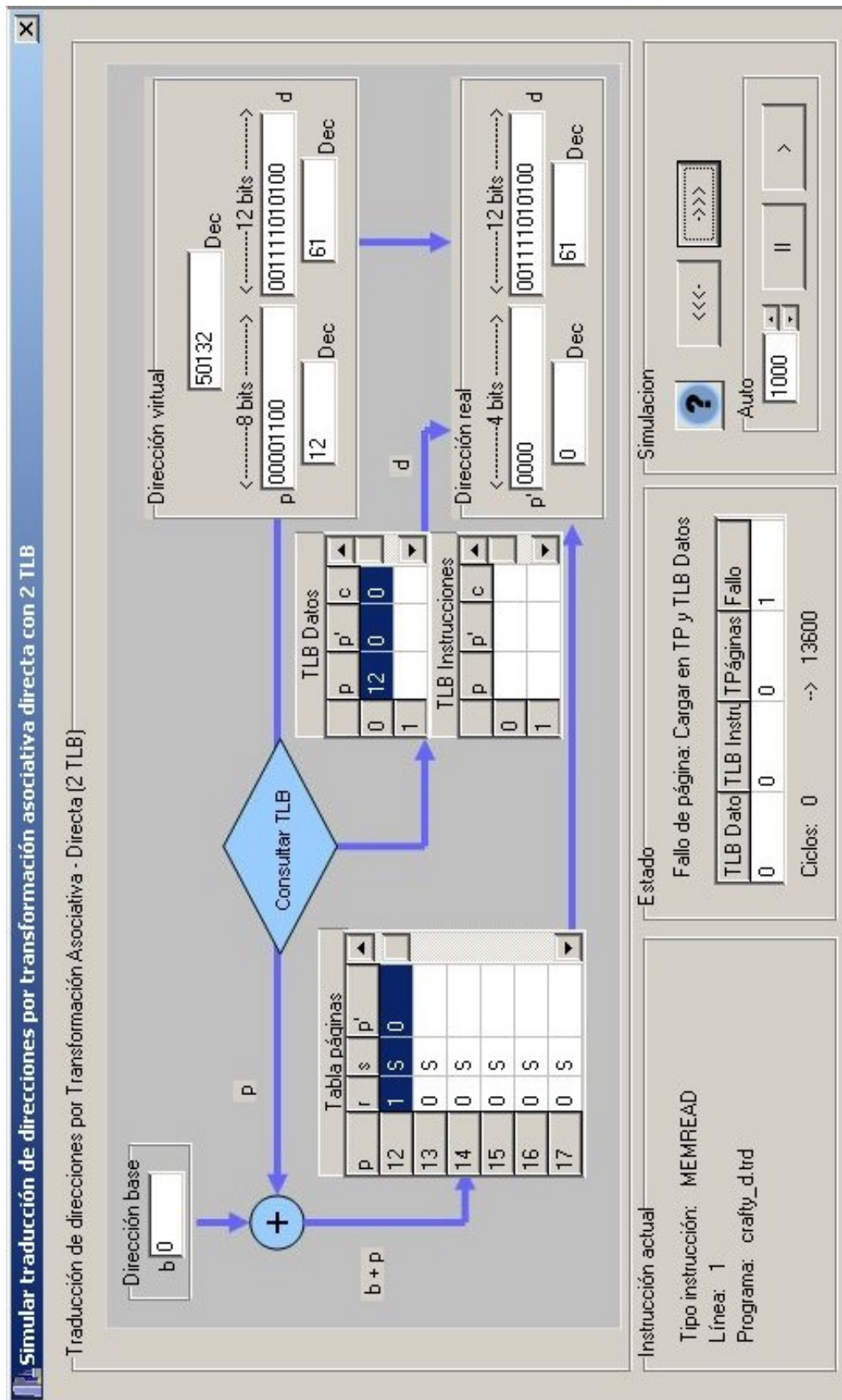
Dependiendo de la configuración elegida, se pueden mostrar tres interfaces diferentes, uno para cada tipo de mecanismo de traducción de direcciones:

- *Traducción de direcciones por transformación directa*
- *Traducción de direcciones por transformación asociativa-directa con TLB*
- *Traducción de direcciones por transformación asociativa-directa con TLB dividida en datos e instrucciones*

Todos ellos comparten los siguientes elementos:

- *Dirección virtual*, tanto en binario como en decimal, dividida en parte p (página dentro de la tabla de páginas) y parte d (desplazamiento dentro de la página).
- *Dirección real*, tanto en binario como en decimal, dividida en parte p' (página en memoria principal) y parte d (desplazamiento dentro de la página).
- *Dirección base* de la tabla de páginas.
- *Tabla de páginas*

A la que los dos últimos tipos de traducción añaden la *TLB* bien conjunta bien dividida en parte de datos y parte de instrucciones.



Captura 4.5. Traducción de direcciones

4.5.2. Búsqueda de páginas

La estructura de la jerarquía de memoria implementada se muestra en la siguiente figura:

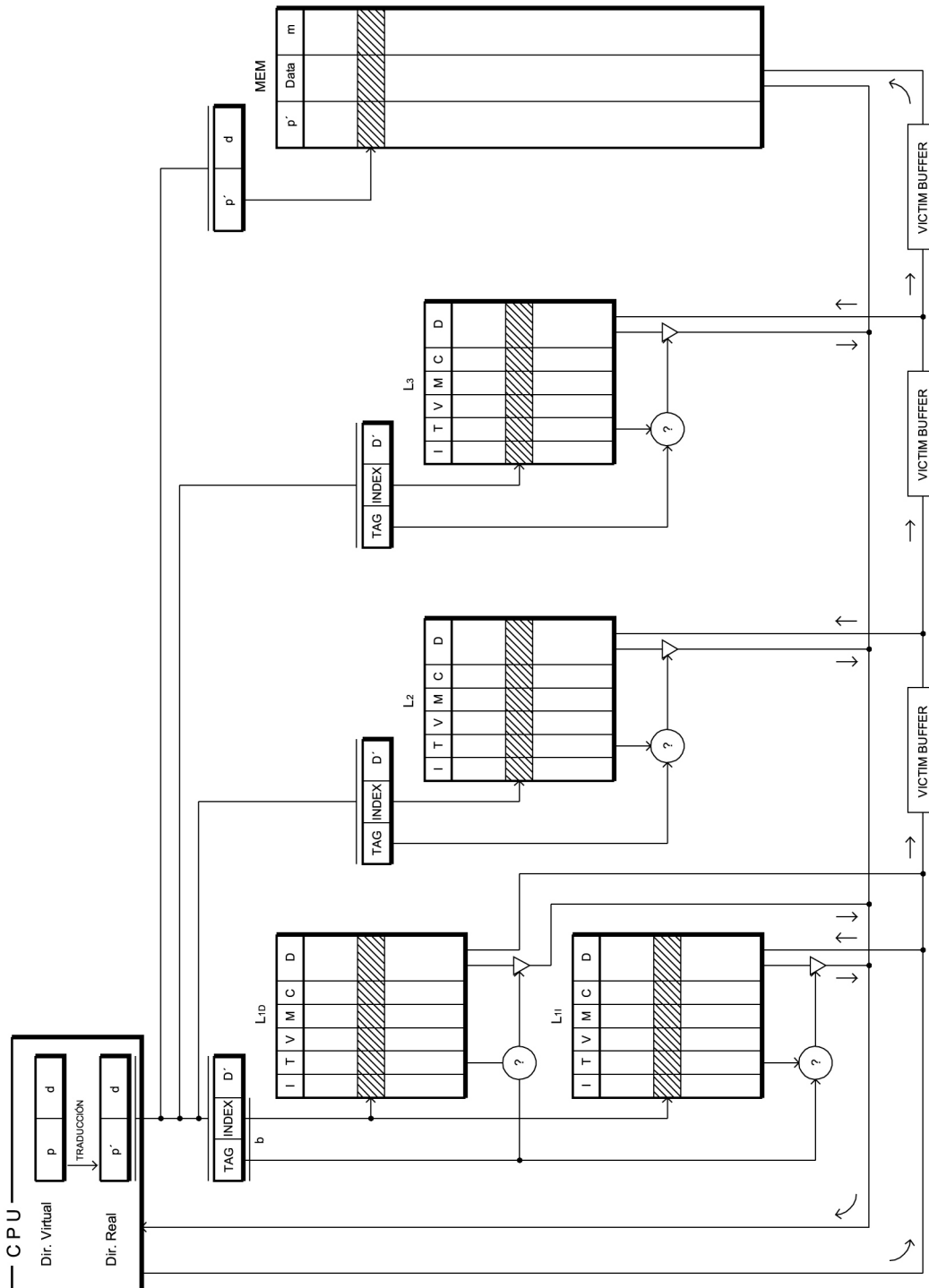


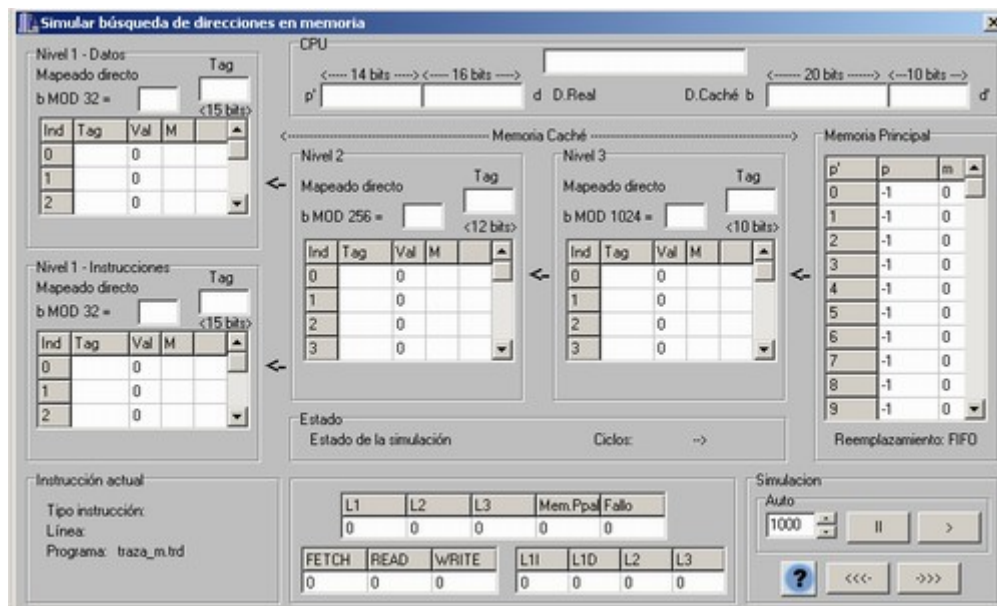
Figura 4.1. Estructura de la jerarquía de memoria implementada

Si elegimos la estrategia de coherencia por escritura retardada debemos tener en cuenta que el grupo de victim buffers de la figura no está presente.

Dependiendo de la configuración elegida, puede llegar a tener los siguientes componentes:

- o Memoria secundaria
- o Memoria principal
- o Caché de nivel 3
- o Caché de nivel 2
- o Caché de nivel 1 conjunta o separada en datos e instrucciones
- o CPU

Habilitando todos los niveles de caché la pantalla mostrada sería esta:



Captura 4.6. Búsqueda de páginas

Donde se puede observar la parte correspondiente a la CPU que incluye:

- Dirección real, en binario y en decimal dividida en parte p (página real) y parte d (desplazamiento dentro de la página).
- Dirección caché, en decimal dividida en parte b (bloque) y d' (desplazamiento dentro del bloque).

4.6. Estrategias

El simulador tiene implementadas múltiples estrategias para el sistema de memoria virtual y caché, a continuación vamos a detallar su implementación.

4.6.1. Estrategias de administración del sistema de memoria virtual

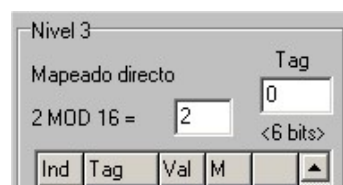
El simulador implementa las siguientes estrategias para el sistema de memoria virtual:

- *Estrategias de búsqueda:* Paginación por demanda y paginación anticipada cargando la página siguiente a la referenciada.
- *Estrategias de colocación:* FIRST FIT, la página nueva que entra a memoria principal se coloca en la primera posición libre.
- *Estrategias de reemplazamiento:* FIFO (First In – First Out), en caso de que todas las posiciones de memoria principal estén llenas se reemplaza la primera página que entró.

4.6.2. Estrategias de administración de la memoria caché

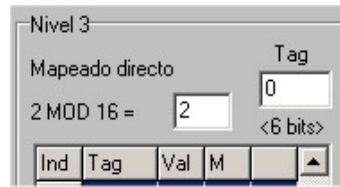
Para la memoria caché se implementan el siguiente conjunto de estrategias:

- *Estrategias de colocación:*
 - *Mapeado directo:* el cálculo de la posición se muestra sobre la memoria.
 - *Completamente asociativa*
 - *Asociativa por conjuntos:* el cálculo del conjunto se muestra sobre la memoria.



Captura 4.7. Cálculo de la posición o conjunto

- *Estrategias de búsqueda:* las direcciones de bloque se dividen en index y tag, mostrándose el tag buscado en un recuadro por encima de la memoria.



Captura 4.8. Etiqueta (Tag) de la caché

- *Estrategias de reemplazamiento:*
 - o *AZAR*: El bloque a reemplazar se elige al azar.
 - o *FIFO (First In – First Out)*: El bloque a reemplazar es aquel que lleva más tiempo en memoria.

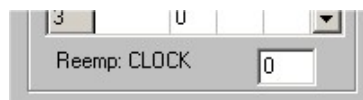
Se implementa usando un contador de 0 a Num bloques, de manera que los nuevos bloques entran a 0, aumentando el contador en 1 del resto de los bloques. Así el bloque con el contador más alto es el que lleva más tiempo en memoria, por lo que es el elegido para ser reemplazado.

- o *LRU (Least Recently Used)*: El bloque a reemplazar es aquel que lleva más tiempo en memoria sin ser usado.

Se implementa usando un contador de 0 a Num bloques, de manera que los nuevos bloques entran a 0, aumentando el contador en 1 del resto de los bloques. Además cuando un bloque es referenciado su contador se pone de nuevo a 0 y se incrementan en 1 aquellos contadores que sean menores. Así el bloque con el contador más alto es el que lleva más tiempo en memoria sin ser usado, por lo que es el elegido para ser reemplazado.

- o *CLOCK aproximación a LRU*: El bloque a reemplazar es aquel que lleva más tiempo en memoria sin ser usado.

Se implementa sin necesidad de contador, tan sólo se necesita un puntero que determina la posición siguiente a ser reemplazada. La posición actual de este puntero se muestra en un recuadro bajo la memoria.



Captura 4.9. Puntero de la política de reemplazamiento Clock

El puntero avanza una posición, y si esa posición tiene el bit de modificado a 1, indicando que ha sido modificada recientemente,

lo pone a 0 y avanza a la siguiente posición. Si tiene el bit a 0 esa es el bloque elegido para ser reemplazado.

- o *LFU (Least Frequently Used)*: El bloque a reemplazar es aquel que ha sido usado el menor número de veces.

Se implementa utilizando un contador que se inicializa a 0 cuando el bloque entra en memoria y se incrementa en 1 cada vez que el bloque es referenciado.

- o *NUR (Not Used Recently)*: El bloque a reemplazar es aquel que no ha sido usado recientemente. Para determinar esta situación se basa en dos factores: si el bloque ha sido referenciado y si ha sido modificado, dando lugar a cuatro posibles situaciones:

- 00 (0) - Página no referenciada y no modificada
- 01 (1) - Página no referenciada pero modificada
- 10 (2) - Página referenciada pero no modificada
- 11 (3) - Página referenciada y modificada

Para implementarlo se utiliza un contador que se inicializa a 0 cuando el bloque entra en memoria por un acceso a instrucciones FETCH o una lectura de memoria MEMREAD, y a 1 si entra por una escritura en memoria MEMWRITE. Posteriormente, si el bloque es referenciado pasa de 0 a 2 o de 1 a 3, y si es modificado, de 2 a 3. Así el bloque elegido para ser reemplazado es el que tiene el contador más bajo.

- *Estrategias de coherencia:*

- o *Write Through o Escritura directa*: Una escritura en memoria actualiza tanto la copia en memoria principal como la copia en memoria caché.

Cuando un bloque es referenciado por una escritura a memoria MEMWRITE se modifican los bits de modificado de los bloques de todos los niveles de caché.

- o *Write Back o Escritura retardada*: Una escritura en memoria actualiza sólo la copia en caché, mientras que la copia de la memoria principal se actualizará cuando el bloque de caché sea reemplazado.

Cuando un bloque es referenciado por una escritura MEMWRITE el bit de modificado de ese nivel de caché pasa a 1, y cuando sea reemplazado se modificará el de la memoria principal.

4.7. Estadísticas

Según el tipo de simulación elegido se muestran las siguientes estadísticas.

4.7.1. Estadísticas de traducción de direcciones

- Número de fallos de página.
- Número de aciertos en la tabla de páginas.
- Número de aciertos en la TLB conjunta o en la TLB de datos y la TLB de instrucciones.

Estado			
Fallo de página: Cargar en TP y TLB Datos			
TLB Dato	TLB Instru	TPáginas	Fallo
0	0	0	1
Ciclos: 0		--> 13510	

Captura 4.10. Estado, estadísticas y ciclos de traducción de direcciones

4.7.2. Estadísticas de búsqueda de páginas

- Número de aciertos en los diferentes niveles de la jerarquía.
- Número de reemplazamientos en los diferentes niveles de la jerarquía.
- Número de instrucciones de cada tipo.

Estado

Estado de la simulación

Ciclos: -->

L1	L2	L3	Mem.Ppal	Fallo
0	0	0	0	0

FETCH	READ	WRITE
0	0	0

L1I	L1D	L2	L3
0	0	0	0

Simu

Aut

10

Captura 4.11. Estado, estadísticas y ciclos de búsqueda de páginas

4.8. Ciclos

Junto a las estadísticas y el estado de la máquina se muestran los ciclos que transcurren en cada paso de la simulación, mostrando los ciclos transcurridos actualmente frente a los del último paso.

4.8.1. Ciclos en traducción de direcciones

- Traducción de direcciones por transformación directa

- o *Cargar a memoria principal*

$$\text{Ciclos} = T_{\text{acceso memoria principal (consulta tabla páginas)}} + T_{\text{acceso memoria secundaria}} + T_{\text{acceso bus}} + T_{\text{acceso memoria principal}}$$
- o *Acierto en tabla de páginas*

$$\text{Ciclos} = T_{\text{acceso memoria principal (consulta tabla páginas)}}$$
- *Traducción de direcciones por transformación asociativa-directa con TLB*
 - o *Cargar a memoria principal*

$$\text{Ciclos} = T_{\text{acceso TLB}} + T_{\text{acceso memoria principal}} + T_{\text{acceso memoria secundaria}} + T_{\text{acceso bus}} + T_{\text{acceso memoria principal}}$$
 - o *Acierto en tabla de páginas*

$$\text{Ciclos} = T_{\text{acceso TLB}} + T_{\text{acceso memoria principal (consulta tabla páginas)}}$$
 - o *Acierto en TLB*

$$\text{Ciclos} = T_{\text{acceso TLB}}$$

4.8.2. Ciclos búsqueda de páginas

- *Cargar a memoria principal*

$$\text{Ciclos} = T_{\text{acceso memoria principal}} + T_{\text{acceso memoria secundaria}} + T_{\text{acceso bus}}$$
- *Acierto en caché de nivel 1*

$$\text{Ciclos} = T_{\text{acceso cache L1}}$$
- *Acierto en caché de nivel 2*

$$\text{Ciclos} = T_{\text{acceso cache L1}} + T_{\text{acceso cache L2}} + T_{\text{acceso bus L1-L2}} + T_{\text{acceso cache L1}}$$
- *Acierto en caché de nivel 3*

$$\text{Ciclos} = T_{\text{acceso cache L1}} + T_{\text{acceso cache L2}} + T_{\text{acceso cache L3}} + T_{\text{acceso bus L3-L2}} + T_{\text{acceso cache L2}} + T_{\text{acceso bus L2-L1}} + T_{\text{acceso cache L1}}$$
- *Acierto en memoria principal*

$$\text{Ciclos} = T_{\text{acceso cache L1}} + T_{\text{acceso cache L2}} + T_{\text{acceso cache L3}} + T_{\text{acceso memoria principal}} + T_{\text{acceso bus MemPpal-L3}} + T_{\text{acceso cache L3}} + T_{\text{acceso bus L3-L2}} + T_{\text{acceso cache L2}} + T_{\text{acceso bus L2-L1}} + T_{\text{acceso cache L1}}$$

Además, cuando usamos la escritura retardada se añade una penalización por el tiempo necesario para actualizar los diferentes niveles de caché, que en el peor de los casos es:

$$\text{Ciclos} = T_{\text{acceso cache L1}} + T_{\text{acceso cache L2}} + T_{\text{acceso cache L3}} + T_{\text{acceso memoria principal}}$$

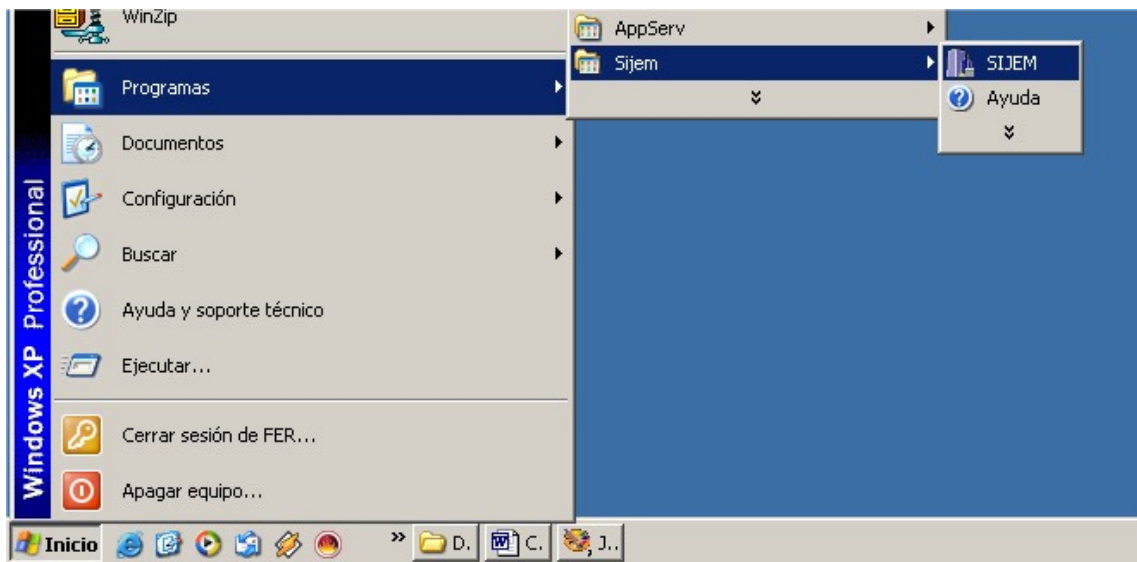
Capítulo 5.

Ayuda del programa

En este capítulo se describe el uso de la aplicación haciendo un recorrido sobre el fichero de ayuda incluido en el mismo.

5.1. Instalación y ejecución del programa

La distribución del simulador incluye un programa de instalación que tras su ejecución muestra dos nuevos accesos directos a la aplicación y su ayuda en el menú inicio.



Captura 5.1. Inicio del programa SIJEM y su Ayuda

Haciendo clic sobre SIJEM accedemos a la aplicación, mientras que haciendo clic sobre Ayuda accedemos al archivo de ayuda.

5.2. Archivo de ayuda

En las siguientes páginas se muestran las diferentes secciones del archivo de ayuda.

Al iniciar la ayuda nos encontramos con la siguiente pantalla, que nos muestra las diferentes secciones:

Bienvenido

- Introducción a SIJEM
- Interfaz de usuario
- Ficheros

5.2.1. Introducción a SIJEM

SIJEM es un simulador didáctico de jerarquías de memoria.

Su objetivo es servir de apoyo al aprendizaje de los conceptos de jerarquías de memoria, fundamentales en materias como Sistemas Operativos y Arquitectura de Computadores.

A través de un interfaz altamente visual trata de ilustrar los siguientes conceptos de jerarquías de memoria:

Memoria virtual

- Direcciones virtuales y direcciones reales.
- Paginación
- Mecanismos de traducción de direcciones.
- Uso de TLBs.
- Estrategias de búsqueda, colocación y reemplazamiento en memoria principal.

Niveles de memoria

- Memoria secundaria.
- Memoria principal.
- Memoria caché multinivel.
 - Nivel 3
 - Nivel 2
 - Nivel 1 Conjunta o separada en datos e instrucciones.
- Estrategias de colocación, reemplazamiento y coherencia entre los diferentes niveles.

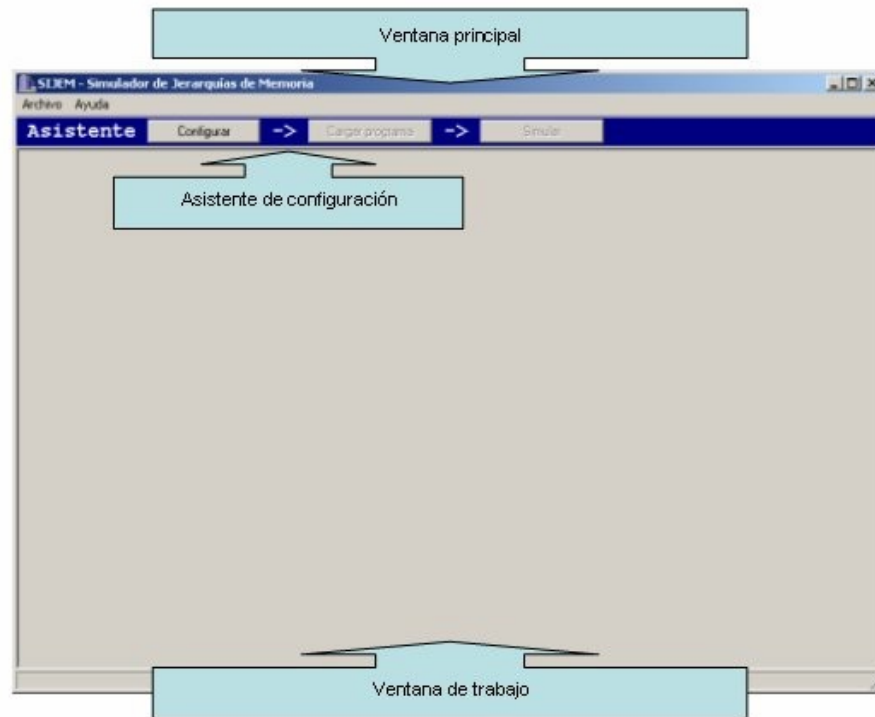
Y haciendo uso de ficheros de traza procedentes de la ejecución de programas reales o inventados por el usuario obtenemos resultados como:

- Tiempo de ejecución del programa en ciclos.
- Tiempo empleado en acceder a un dato/instrucción en ciclos.
- Tasa de aciertos y fallos en memoria principal y memoria caché.

- Número de instrucciones de cada tipo contenidas por un programa.
- Número de reemplazamientos en memoria principal y memoria caché.

5.2.2. Interfaz de usuario de SIJEM

La interfaz de usuario de SIJEM se compone de:



Captura 5.2. Interfaz de usuario

- Ventana principal
Es la ventana principal del programa dentro de la cual se ejecuta el mismo.
- Asistente de configuración
Se compone de tres botones
 - Configurar
 - Cargar programa
 - Simular

que ayudan en la ejecución del simulador.

Además existe el botón de "Resumen Configuración", activo en las etapas de cargar programa y simular, para obtener un resumen de la configuración efectuada.

- Ventana de trabajo
En ella se muestran las diferentes ventanas con las que trabaja el programa.

5.2.2.1. Asistente - Configurar

Al pulsar sobre el botón configurar del asistente se nos muestran una serie de ventanas que nos ayudarán en la tarea de ejecución del simulador.

Estas ventanas son:

- Bienvenido - Introducción a las páginas de configuración.
- Página 1 - Cargar fichero y configurar memoria principal y secundaria.
- Página 2 - Mecanismo de traducción de direcciones y paginación.
- Página 3 - Caché de nivel 3.
- Página 4 - Caché de nivel 2.
- Página 5 - Caché de nivel 1.
- Página 6 - Resumen de la configuración elegida.

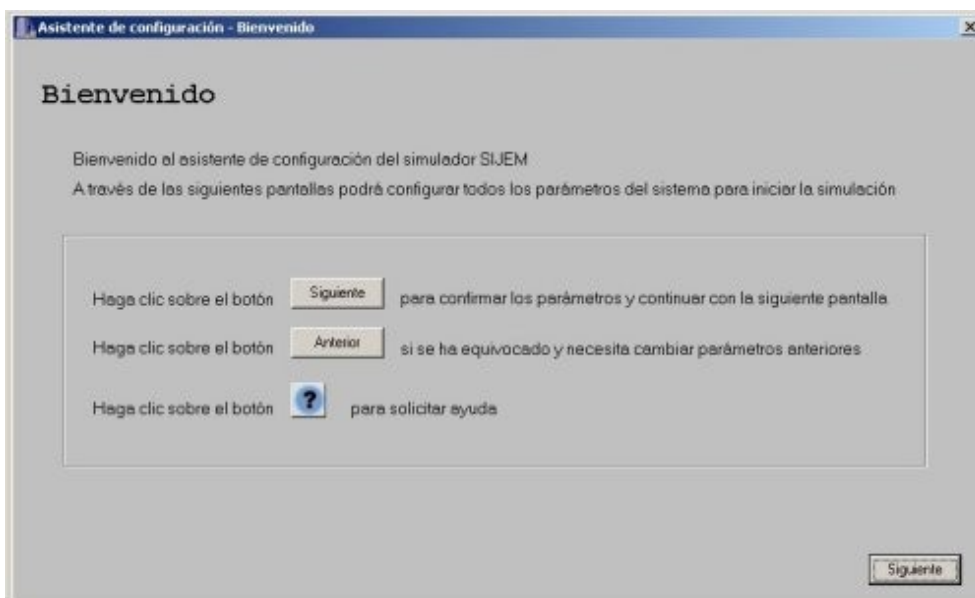
A través de estas páginas iremos configurando nuestro simulador para realizar la simulación elegida.

A. Asistente > Configurar > Bienvenido

Al hacer clic sobre el botón configurar del asistente se visualiza esta ventana que nos muestra la serie de pautas a seguir para configurar la ejecución del simulador.

Tales como:

- Hacer clic sobre el botón "Siguiente" para confirmar los parámetros y continuar con la siguiente pantalla.
- Hacer clic sobre el botón "Anterior" si se ha equivocado y necesita cambiar parámetros anteriores.
- Hacer clic sobre el botón "Ayuda" para solicitar ayuda.



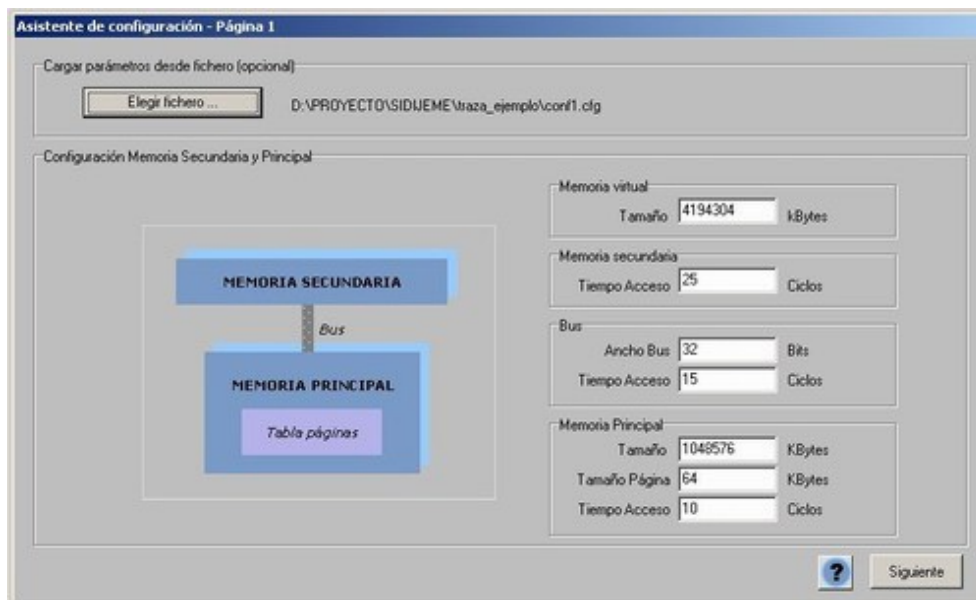
Captura 5.3. Asistente de configuración - Bienvenido

Al hacer clic sobre el botón "Siguiente" pasamos a la página 1 de la configuración.

B. Asistente > Configurar > Página 1

A través de esta ventana podemos:

- Cargar un fichero de configuración.
- Configurar los parámetros de la memoria principal y de la memoria secundaria.



Captura 5.4. Asistente de configuración – Página 1

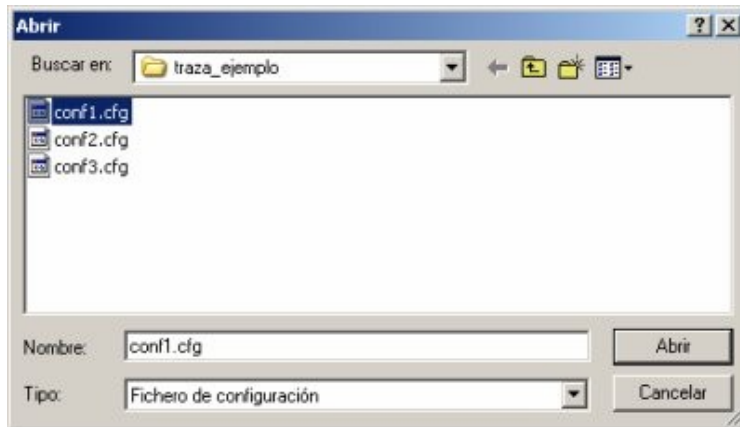
B.1. Asistente > Configurar > Página 1. Cargar un fichero de configuración

El simulador permite cargar todos los parámetros de configuración desde un fichero previamente realizado por el usuario, u obtenido de entre los múltiples ejemplos contenidos en el paquete.

La estructura de este fichero se define en la sección ficheros.

Para cargar un fichero de configuración haremos clic sobre el botón "Elegir fichero" contenido en el cuadro de texto "Cargar parámetros desde fichero".

A continuación se nos mostrará un cuadro de diálogo para elegir el fichero .cfg que deseamos cargar, hacemos clic sobre el botón "Aceptar" y si se trata de un fichero válido los parámetros serán volcados a los diferentes cuadros de configuración. Y junto al botón "Elegir fichero" se mostrará la ruta del fichero elegido.



Captura 5.5. Carga de un fichero de configuración

Al hacer clic sobre el botón "Siguiente" pasamos a la página 1 de la configuración.

B.2. Asistente > Configurar > Página 1. Configurar los parámetros de la memoria principal y de la memoria secundaria.

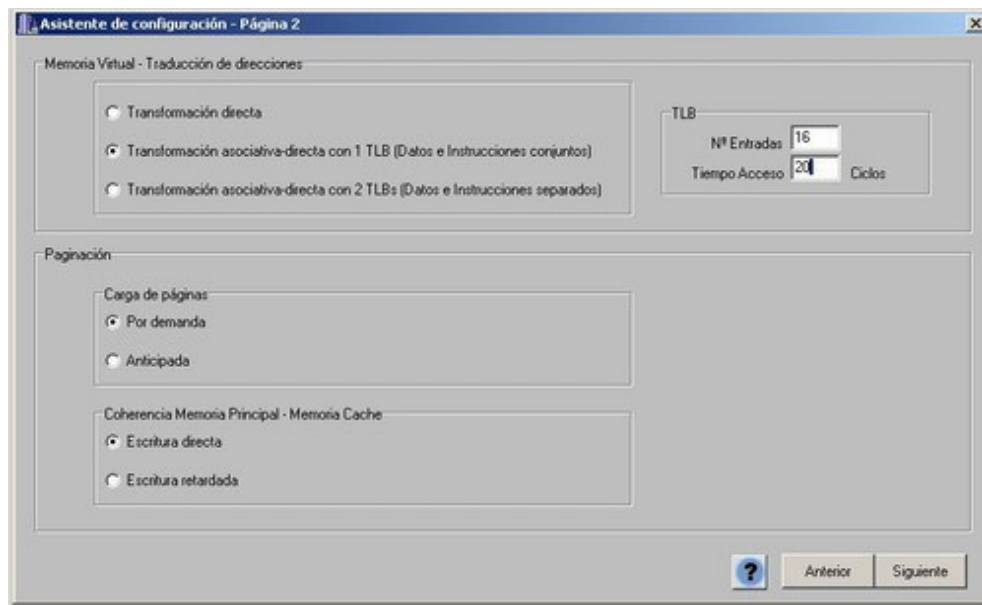
A través de esta página de configuración fijamos los siguientes parámetros:

- **MEMORIA VIRTUAL**
 - **Tamaño (en Kbytes):** Tamaño del espacio de direcciones que puede ser referenciado por el programa en ejecución.
- **MEMORIA SECUNDARIA**
 - **Tiempo de acceso (en ciclos):** Tiempo empleado para acceder a una página situada en la memoria secundaria.
- **BUS MEMORIA SECUNDARIA - MEMORIA PRINCIPAL**
 - **Ancho bus (en bits):** Ancho del bus situado entre memoria principal y memoria secundaria.
 - **Tiempo de acceso (en ciclos):** Tiempo empleado para trasladar una palabra entre la memoria secundaria y la memoria principal a través de este bus.
- **MEMORIA PRINCIPAL**
 - **Tamaño (en Kbytes):** Tamaño físico de la memoria principal.
 - **Tamaño de página (en Kbytes):** Tamaño de la página de memoria.
 - **Tiempo de acceso (en ciclos):** Tiempo empleado para acceder a una página situada en la memoria principal.

C. Asistente > Configurar > Página 2

A través de esta ventana podemos:

- Establecer el mecanismo de traducción de direcciones virtuales en direcciones reales.
- Definir los parámetros de la paginación.



Captura 5.6. Asistente de configuración – Página 2

C.1. Asistente > Configurar > Página 2. Mecanismo de traducción de direcciones

A través de este cuadro fijamos el mecanismo de traducción de direcciones de memoria virtual a direcciones reales de memoria.

Podemos elegir entre:

- TRANSFORMACIÓN DIRECTA

La tabla de mapa de páginas contiene una entrada para cada una de las páginas de la memoria virtual de este programa.

- TRANSFORMACIÓN ASOCIATIVA-DIRECTA CON 1 TLB

Junto con la tabla de páginas existe una memoria asociativa denominada TLB (Translation Lookaside buffer) que contiene las traducciones más recientes.

- TRANSFORMACIÓN ASOCIATIVA-DIRECTA CON 2 TLB

Es similar a la anterior, pero usando dos memorias asociativas, una para las traducciones más recientes de búsqueda de instrucciones (TLB de instrucciones) y otra para las traducciones más recientes de lectura o escritura de datos (TLB de datos).

Si elegimos cualquiera de estas dos últimas opciones debemos configurar los parámetros de la TLB:

- Número de entradas de la TLB: Número de entradas que tendrá la TLB. Si se selecciona 2 TLB será el número de entradas de cada TLB.
- Tiempo acceso: Tiempo empleado para acceder a una entrada de la TLB.

C.2. Asistente > Configurar > Página 2. Paginación

A través de este cuadro fijamos los siguientes parámetros:

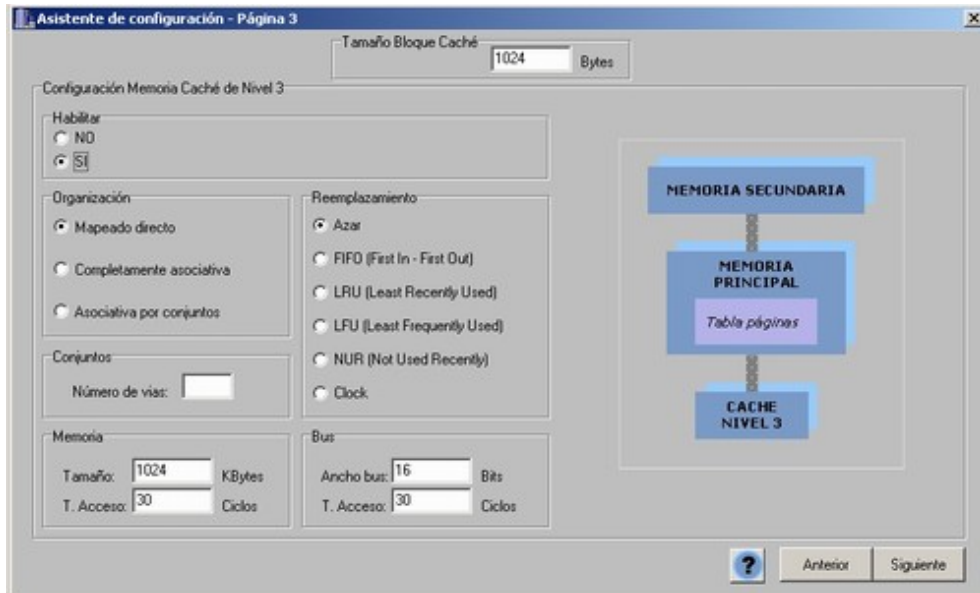
- MECANISMO DE CARGA DE PÁGINAS
 - *Por demanda:* Las páginas se van cargando en memoria a medida que van siendo solicitadas por el programa en ejecución.
 - *Anticipada:* Además de la página solicitada por el programa en ejecución se carga la siguiente página de memoria.
- COHERENCIA ENTRE MEMORIA CACHE Y MEMORIA PRINCIPAL
 - *Escritura directa:* La escritura de una página situada en memoria caché implica que la copia de memoria principal se actualice inmediatamente. Dado que se utiliza una estructura basada en Victim Buffer esta actualización no conlleva ciclos de CPU.
 - *Escritura retardada:* La copia en memoria principal sólo es actualizada cuando la página es reemplazada en memoria caché. En este caso, cuando una página es reemplazada, debemos esperar a que la copia en memoria principal sea actualizada, lo que implica que se tenga que esperar tantos ciclos como una escritura en memoria principal.

D. Asistente > Configurar > Página 3

A través de esta página configuramos el tamaño del bloque de caché y los siguientes parámetros de la caché de nivel 3:

- Habilitar o deshabilitar este nivel de memoria.
- Definir la organización.
- Establecer el número de conjuntos.

- Establecer la política de reemplazamiento.
- Definir los parámetros de la memoria.
- Definir los parámetros del bus.



Captura 5.7. Asistente de configuración – Página 3

D.1. Asistente > Configurar > Página 3. Caché de nivel 3. Habilitar

Especifica el tamaño del bloque de caché en bytes.

D.2. Asistente > Configurar > Página 3. Caché de nivel 3. Habilitar

A través de este cuadro habilitamos o deshabilitamos el uso de la memoria caché de nivel 3.

Si deshabilitamos la memoria caché de nivel 3 el resto de los parámetros de la página dejan de estar disponibles pues pierden su significado.

D.3. Asistente > Configurar > Página 3. Caché de nivel 3. Organización

A través de este cuadro fijamos la organización de este nivel de memoria.

Podemos elegir entre:

- **MAPEADO DIRECTO**

Cada bloque tiene un sólo lugar donde ser colocado en memoria caché según la fórmula

(Página de memoria principal) MOD (Número de bloques en caché)

- **COMPLETAMENTE ASOCIATIVA**

Cada bloque puede ser colocado en cualquier lugar de la memoria caché.

- **ASOCIATIVA POR CONJUNTOS**

A cada bloque le corresponde un conjunto dentro de memoria caché, de manera que puede ser colocada en cualquier posición dentro del conjunto, según la fórmula:

(Bloque de memoria caché) MOD (Número de conjuntos en caché)

Si un conjunto tiene n bloques se denomina Memoria Asociativa por Conjuntos de N -Vías.

D.4. Asistente > Configurar > Página 3. Caché de nivel 3. Número de conjuntos

Si hemos elegido una organización "Asociativa por conjuntos" usaremos este cuadro para definir el número de vías, es decir, el número de bloques de los que consta cada conjunto.

D.5. Asistente > Configurar > Página 3. Caché de nivel 3. Reemplazamiento

A través de este cuadro fijamos la política de reemplazamiento de este nivel de memoria.

Debemos tener en cuenta que la política de reemplazamiento no influye si elegimos una organización por Mapeado Directo.

Y que si elegimos una organización Asociativa por conjuntos sólo tiene influencia sobre el conjunto que corresponde al nuevo bloque

Podemos elegir entre:

- **AZAR**

Cuando la memoria caché se encuentra llena el bloque a reemplazar es elegido al azar.

- **FIFO (First In - First Out)**

Se reemplaza aquel bloque que lleva más tiempo en memoria.

Se implementa usando un contador de 0 a Num bloques, de manera que los nuevos bloques entran a 0, aumentando el contador en 1 del resto de los

bloques. Así el bloque con el contador más alto es el que lleva más tiempo en memoria, por lo que es el elegido para ser reemplazado.

- LRU (Least Recently Used) - Menos recientemente usada

Se reemplaza aquel bloque que lleva más tiempo en memoria sin ser usado.

Se implementa usando un contador de 0 a Num bloques, de manera que los nuevos bloques entran a 0, aumentando el contador en 1 del resto de los bloques.

Además cuando un bloque es referenciado su contador se pone de nuevo a 0 y se incrementan en 1 aquellos contadores que sean menores. Así el bloque con el contador más alto es el que lleva más tiempo en memoria sin ser usado, por lo que es el elegido para ser reemplazado.

- CLOCK aproximación a LRU

Se reemplaza aquel bloque que lleva más tiempo en memoria sin ser usado.

Se implementa sin necesidad de contador, tan sólo se necesita un puntero que determina la posición siguiente a ser reemplazada. La posición actual de este puntero se muestra en un recuadro bajo la memoria.

El puntero avanza una posición, y si esa posición tiene el bit de modificado a 1, indicando que ha sido modificada recientemente, lo pone a 0 y avanza a la siguiente posición. Si tiene el bit a 0 esa es el bloque elegido para ser reemplazado.

- LFU (Least Frequently Used) - Menos frecuentemente usada

Se reemplaza aquel bloque que ha sido usado un menor número de veces.

Se implementa utilizando un contador que se inicializa a 0 cuando el bloque entra en memoria y se incrementa en 1 cada vez que el bloque es referenciado.

- NUR (Not Used Recently) - No recientemente usada

Se reemplaza aquel bloque que no ha sido usado recientemente.

Para determinar esta situación se basa en dos factores: si el bloque ha sido referenciado y si el bloque ha sido modificado. Dando lugar a cuatro posibles situaciones:

- 00 (0) - Página no referenciada y no modificada
- 01 (1) - Página no referenciada pero modificada
- 10 (2) - Página referenciada pero no modificada

- 11 (3) - Página referenciada y modificada

Para implementarlo se utiliza un contador que se inicializa a 0 cuando el bloque entra en memoria por un acceso a instrucciones FETCH o una lectura de memoria MEMREAD, y a 1 si entra por una escritura en memoria MEMWRITE. Posteriormente, si el bloque es referenciado pasa de 0 a 2 o de 1 a 3, y si es modificado, de 2 a 3. Así el bloque elegido para ser reemplazado es el que tiene el contador más bajo.

D.6. Asistente > Configurar > Página 3. Caché de nivel 3. Memoria

A través de este cuadro fijamos los parámetros de la memoria caché de nivel 3:

- TAMAÑO (en Kbytes): Tamaño físico de la memoria caché de este nivel.
- TIEMPO DE ACCESO (en ciclos): Tiempo empleado para acceder a una página situada en este nivel de caché.

D.7. Asistente > Configurar > Página 3. Caché de nivel 3. Bus

A través de este cuadro fijamos los parámetros del bus entre la memoria caché de nivel 3 y la memoria principal:

- ANCHO (en bits): Ancho del bus situado entre memoria caché de nivel 3 y memoria principal.
- TIEMPO DE ACCESO (en ciclos): Tiempo empleado para trasladar una palabra entre la memoria caché de nivel 3 y la memoria principal a través de este bus.

E. Asistente > Configurar > Página 4

A través de esta ventana podemos configurar los siguientes parámetros de la caché de nivel 2:

- Habilitar o deshabilitar este nivel de memoria.
- Definir la organización.
- Establecer el número de conjuntos.
- Establecer la política de reemplazamiento.
- Definir los parámetros de la memoria.
- Definir los parámetros del bus.



Captura 5.8. Asistente de configuración – Página 4

E.1. Asistente > Configurar > Página 4. Caché de nivel 2. Habilitar

A través de este cuadro habilitamos o deshabilitamos el uso de la memoria caché de nivel 2.

Si deshabilitamos la memoria caché de este nivel el resto de los parámetros de la página dejan de estar disponibles pues pierden su significado.

E.2. Asistente > Configurar > Página 4. Caché de nivel 2. Organización

A través de este cuadro fijamos la organización de este nivel de memoria.

Podemos elegir entre:

- **MAPEADO DIRECTO**

Cada bloque tiene un sólo lugar donde ser colocado en memoria caché según la fórmula

$$(\text{Página de memoria principal}) \bmod (\text{Número de bloques en caché})$$

- **COMPLETAMENTE ASOCIATIVA**

Cada bloque puede ser colocado en cualquier lugar de la memoria caché.

- **ASOCIATIVA POR CONJUNTOS**

A cada bloque le corresponde un conjunto dentro de memoria caché, de manera que puede ser colocada en cualquier posición dentro del conjunto, según la formula:

(Bloque de memoria caché) MOD (Número de conjuntos en caché)

Si un conjunto tiene n bloques se denomina Memoria Asociativa por Conjuntos de N-Vias.

E.3. Asistente > Configurar > Página 4. Caché de nivel 2. Número de conjuntos

Si hemos elegido una organización "Asociativa por conjuntos" usaremos este cuadro para definir el número de vías, es decir, el número de bloques de los que consta cada conjunto.

E.4. Asistente > Configurar > Página 4. Caché de nivel 2. Reemplazamiento

A través de este cuadro fijamos la política de reemplazamiento de este nivel de memoria.

Debemos tener en cuenta que la política de reemplazamiento no influye si elegimos una organización por Mapeado Directo.

Y que si elegimos una organización Asociativa por conjuntos sólo tiene influencia sobre el conjunto que corresponde al nuevo bloque

Podemos elegir entre:

- **AZAR**

Cuando la memoria caché se encuentra llena el bloque a reemplazar es elegido al azar.

- **FIFO (First In - First Out)**

Se reemplaza aquel bloque que lleva más tiempo en memoria.

Se implementa usando un contador de 0 a Num bloques, de manera que los nuevos bloques entran a 0, aumentando el contador en 1 del resto de los bloques. Así el bloque con el contador más alto es el que lleva más tiempo en memoria, por lo que es el elegido para ser reemplazado.

- **LRU (Less Recently Used) - Menos recientemente usada**

Se reemplaza aquel bloque que lleva más tiempo en memoria sin ser usado.

Se implementa usando un contador de 0 a Num bloques, de manera que los nuevos bloques entran a 0, aumentando el contador en 1 del resto de

los bloques. Además cuando un bloque es referenciado su contador se pone de nuevo a 0 y se incrementan en 1 aquellos contadores que sean menores. Así el bloque con el contador más alto es el que lleva más tiempo en memoria sin ser usado, por lo que es el elegido para ser reemplazado.

- **CLOCK aproximación a LRU**

Se reemplaza aquel bloque que lleva más tiempo en memoria sin ser usado.

Se implementa sin necesidad de contador, tan sólo se necesita un puntero que determina la posición siguiente a ser reemplazada. La posición actual de este puntero se muestra en un recuadro bajo la memoria.

El puntero avanza una posición, y si esa posición tiene el bit de modificado a 1, indicando que ha sido modificada recientemente, lo pone a 0 y avanza a la siguiente posición. Si tiene el bit a 0 esa es el bloque elegido para ser reemplazado.

- **LFU (Least Frequently Used) - Menos frecuentemente usada**

Se reemplaza aquel bloque que ha sido usado un menor número de veces. Se implementa utilizando un contador que se inicializa a 0 cuando el bloque entra en memoria y se incrementa en 1 cada vez que el bloque es referenciado.

- **NUR (Not Used Recently) - No recientemente usada**

Se reemplaza aquel bloque que no ha sido usado recientemente.

Para determinar esta situación se basa en dos factores: si el bloque ha sido referenciado y si el bloque ha sido modificado. Dando lugar a cuatro posibles situaciones:

- 00 (0) - Página no referenciada y no modificada
- 01 (1) - Página no referenciada pero modificada
- 10 (2) - Página referenciada pero no modificada
- 11 (3) - Página referenciada y modificada

Para implementarlo se utiliza un contador que se inicializa a 0 cuando el bloque entra en memoria por un acceso a instrucciones FETCH o una lectura de memoria MEMREAD, y a 1 si entra por una escritura en memoria MEMWRITE. Posteriormente, si el bloque es referenciado pasa

de 0 a 2 o de 1 a 3, y si es modificado, de 2 a 3. Así el bloque elegido para ser reemplazado es el que tiene el contador más bajo.

E.5. Asistente > Configurar > Página 4. Caché de nivel 2. Memoria

A través de este cuadro fijamos los parámetros de la memoria caché de nivel 2:

- **TAMAÑO** (en Kbytes): Tamaño físico de la memoria caché de este nivel.
- **TIEMPO DE ACCESO** (en ciclos): Tiempo empleado para acceder a una página situada en este nivel de caché.

E.6. Asistente > Configurar > Página 4. Caché de nivel 2. Bus

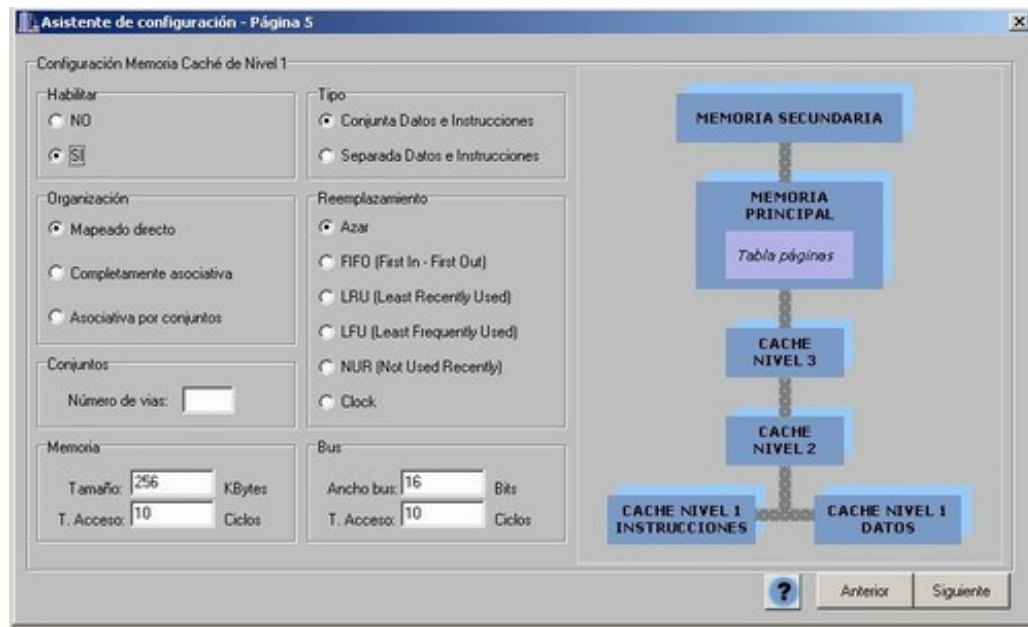
A través de este cuadro fijamos los parámetros del bus entre la memoria caché de nivel 3 y la caché de nivel 2, o bien entre la caché de nivel 2 y la memoria principal:

- **ANCHO** (en bits): Ancho del bus situado entre memoria caché de nivel 3 y la caché de nivel 2, o bien entre la caché de nivel 2 y la memoria principal.
- **TIEMPO DE ACCESO** (en ciclos): Tiempo empleado para trasladar una palabra entre la memoria caché de nivel 3 y la caché de nivel 2, o bien entre la caché de nivel 2 y la memoria principal a través de este bus.

F. Asistente > Configurar > Página 5

A través de esta ventana podemos configurar los siguientes parámetros de la caché de nivel 1:

- Habilitar o deshabilitar este nivel de memoria.
- Definir el tipo.
- Definir la organización.
- Establecer el número de conjuntos.
- Establecer la política de reemplazamiento.
- Definir los parámetros de la memoria.
- Definir los parámetros del bus.



Captura 5.9. Asistente de configuración – Página 5

F.1. Asistente > Configurar > Página 5. Caché de nivel 1. Habilitar

A través de este cuadro habilitamos o deshabilitamos el uso de la memoria caché de nivel 1.

Si deshabilitamos la memoria caché de este nivel el resto de los parámetros de la página dejan de estar disponibles pues pierden su significado.

F.2. Asistente > Configurar > Página 5. Caché de nivel 1. Tipo

A través de este cuadro definimos el tipo de memoria caché de nivel 1.

Podemos elegir entre:

- CONJUNTA

Datos e Instrucciones comparten la misma memoria.

- SEPARADA

Datos e Instrucciones utilizan memorias separadas.

F.3. Asistente > Configurar > Página 5. Caché de nivel 1. Organización

A través de este cuadro fijamos la organización de este nivel de memoria.

Podemos elegir entre:

- **MAPEADO DIRECTO**

Cada bloque tiene un sólo lugar donde ser colocado en memoria caché según la fórmula

$$(\text{Página de memoria principal}) \text{ MOD } (\text{Número de bloques en caché})$$

- **COMPLETAMENTE ASOCIATIVA**

Cada bloque puede ser colocado en cualquier lugar de la memoria caché.

- **ASOCIATIVA POR CONJUNTOS**

A cada bloque le corresponde un conjunto dentro de memoria caché, de manera que puede ser colocada en cualquier posición dentro del conjunto, según la fórmula:

$$(\text{Bloque de memoria caché}) \text{ MOD } (\text{Número de conjuntos en caché})$$

Si un conjunto tiene n bloques se denomina Memoria Asociativa por Conjuntos de N-Vías.

F.4. Asistente > Configurar > Página 5. Caché de nivel 1. Número de conjuntos

Si hemos elegido una organización "Asociativa por conjuntos" usaremos este cuadro para definir el número de vías, es decir, el número de bloques de los que consta cada conjunto.

F.5. Asistente > Configurar > Página 5. Caché de nivel 1. Reemplazamiento

A través de este cuadro fijamos la política de reemplazamiento de este nivel de memoria.

Debemos tener en cuenta que la política de reemplazamiento no influye si elegimos una organización por Mapeado Directo y que si elegimos una organización Asociativa por conjuntos, sólo tiene influencia sobre el conjunto que corresponde al nuevo bloque.

Podemos elegir entre:

- **AZAR**

Cuando la memoria caché se encuentra llena el bloque a reemplazar es elegido al azar.

- **FIFO (First In - First Out)**

Se reemplaza aquel bloque que lleva más tiempo en memoria.

Se implementa usando un contador de 0 a Num bloques, de manera que los nuevos bloques entran a 0, aumentando el contador en 1 del resto de los bloques. Así el bloque con el contador más alto es el que lleva más tiempo en memoria, por lo que es el elegido para ser reemplazado.

- LRU (Least Recently Used) - Menos recientemente usada

Se reemplaza aquel bloque que lleva más tiempo en memoria sin ser usado.

Se implementa usando un contador de 0 a Num bloques, de manera que los nuevos bloques entran a 0, aumentando el contador en 1 del resto de los bloques. Además cuando un bloque es referenciado su contador se pone de nuevo a 0 y se incrementan en 1 aquellos contadores que sean menores. Así el bloque con el contador más alto es el que lleva más tiempo en memoria sin ser usado, por lo que es el elegido para ser reemplazado.

- CLOCK aproximación a LRU

Se reemplaza aquel bloque que lleva más tiempo en memoria sin ser usado.

Se implementa sin necesidad de contador, tan sólo se necesita un puntero que determina la posición siguiente a ser reemplazada. La posición actual de este puntero se muestra en un recuadro bajo la memoria.

El puntero avanza una posición, y si esa posición tiene el bit de modificado a 1, indicando que ha sido modificada recientemente, lo pone a 0 y avanza a la siguiente posición. Si tiene el bit a 0 esa es el bloque elegido para ser reemplazado.

- LFU (Least Frequently Used) - Menos frecuentemente usada

Se reemplaza aquel bloque que ha sido usado un menor número de veces.

Se implementa utilizando un contador que se inicializa a 0 cuando el bloque entra en memoria y se incrementa en 1 cada vez que el bloque es referenciado.

- NUR (Not Used Recently) - No recientemente usada
Se reemplaza aquel bloque que no ha sido usado recientemente.

Para determinar esta situación se basa en dos factores: si el bloque ha sido referenciado y si el bloque ha sido modificado. Dando lugar a cuatro posibles situaciones:

- 00 (0) - Página no referenciada y no modificada
- 01 (1) - Página no referenciada pero modificada
- 10 (2) - Página referenciada pero no modificada
- 11 (3) - Página referenciada y modificada

Para implementarlo se utiliza un contador que se inicializa a 0 cuando el bloque entra en memoria por un acceso a instrucciones FETCH o una lectura de memoria MEMREAD, y a 1 si entra por una escritura en memoria MEMWRITE. Posteriormente, si el bloque es referenciado pasa de 0 a 2 o de 1 a 3, y si es modificado, de 2 a 3. Así el bloque elegido para ser reemplazado es el que tiene el contador más bajo.

F.6. Asistente > Configurar > Página 5. Caché de nivel 1. Memoria

A través de este cuadro fijamos los parámetros de la memoria caché de nivel 1:

- TAMAÑO (en Kbytes): Tamaño físico de la memoria caché de este nivel.
- TIEMPO DE ACCESO (en ciclos): Tiempo empleado para acceder a una página situada en este nivel de caché.

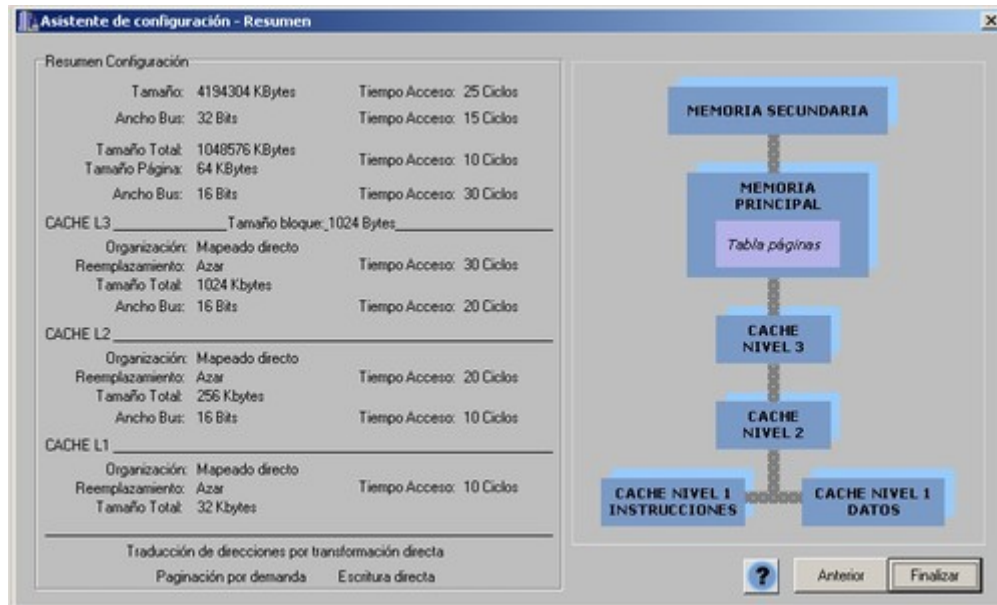
F.7. Asistente > Configurar > Página 5. Caché de nivel 1. Bus

A través de este cuadro fijamos los parámetros del bus entre la memoria caché de nivel 2 y la caché de nivel 1, o bien entre la caché de nivel 1 y la memoria principal:

- ANCHO (en bits): Ancho del bus situado entre memoria caché de nivel 2 y la caché de nivel 1, o bien entre la caché de nivel 1 y la memoria principal.
- TIEMPO DE ACCESO (en ciclos): Tiempo empleado para trasladar una palabra entre la memoria caché de nivel 2 y la caché de nivel 1, o bien entre la caché de nivel 1 y la memoria principal a través de este bus.

G. Asistente > Configurar > Resumen

En esta ventana se muestran los parámetros elegidos durante la configuración del sistema, para que el usuario pueda validarlos, o bien volver atrás a corregir aquellos que haya introducido de forma incorrecta.



Captura 5.10. Asistente de configuración – Resumen

Al hacer clic sobre el botón "Siguiente" pasamos a la siguiente fase del proceso de simulación, la carga de un fichero de traza de un programa.

5.2.2.2. Asistente > Cargar programa

A través de esta ventana podemos elegir el fichero de traza sobre el que vamos a realizar la simulación.

Existen dos posibles tipos de ficheros a cargar:

- Fichero de direcciones (extensión .trd)

Contiene las direcciones virtuales referenciadas por un programa, real o simulado.

Si elegimos este tipo de fichero podremos visualizar luego la traducción de direcciones o la búsqueda de páginas.

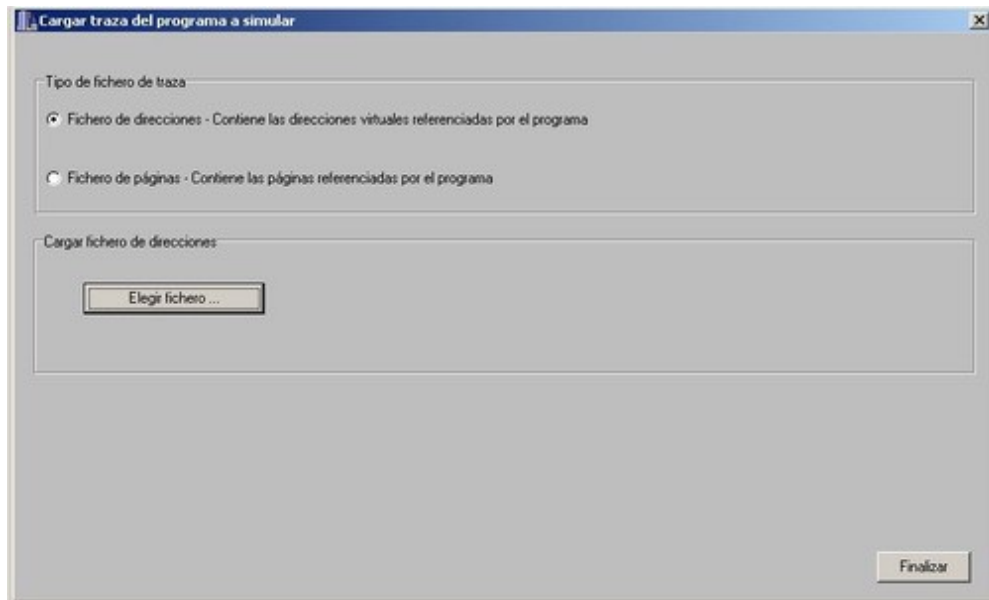
- Fichero de páginas (extensión .trp)

Contiene las páginas referenciadas por un programa, real o simulado.

Si elegimos este tipo de fichero podremos visualizar sólo la búsqueda de páginas.

La estructura de estos ficheros se define en la sección Ficheros.

Al pulsar sobre el botón configurar del asistente se nos muestra la siguiente ventana:



Captura 5.11. Asistente – Cargar programa

Si el fichero de traza ya se encuentra cargado se muestra la ruta en el cuadro inferior.

Elegimos el tipo de fichero de traza a cargar y pulsando sobre el botón "Elegir fichero" se nos muestra el cuadro de diálogo para acceder al fichero que deseamos cargar



Captura 5.12. Abrir fichero de traza

pulsamos sobre el botón abrir y una vez cargado el fichero se nos muestra la ruta del mismo en el cuadro inferior.

A continuación podemos pasar al siguiente paso del asistente pulsando el botón "Finalizar".

5.2.2.3. Asistente > Simular

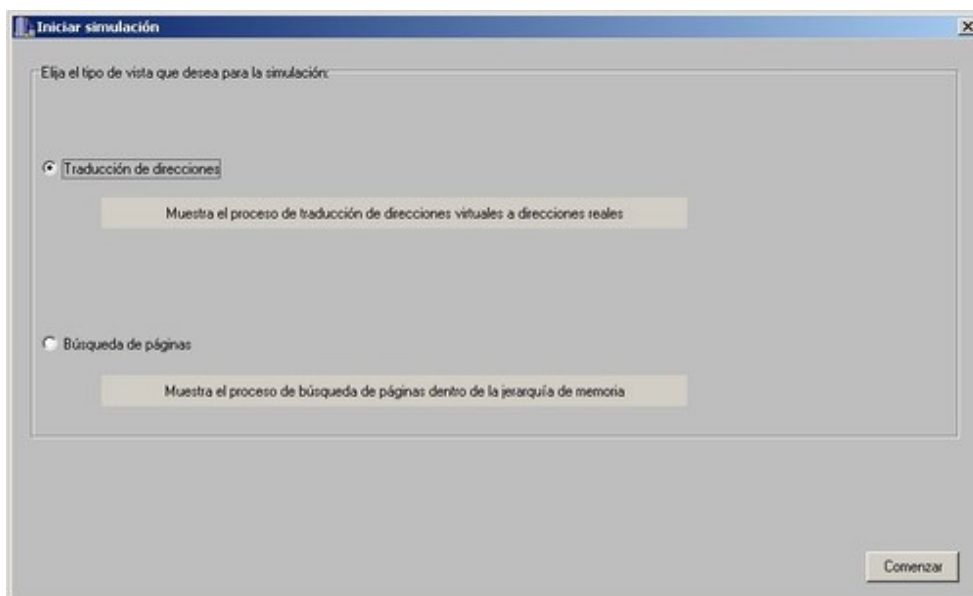
A través de esta ventana podemos elegir el tipo de simulación que deseamos realizar escogiendo entre las dos opciones:

- Traducción de direcciones

Muestra el proceso de traducción de direcciones virtuales a direcciones reales.

- Búsqueda de páginas

Muestra el proceso de búsqueda de páginas dentro de la jerarquía de memoria.



Captura 5.13 Asistente – Simular

y pulsando sobre el botón Comenzar.

5.2.2.3.1. Traducción de direcciones

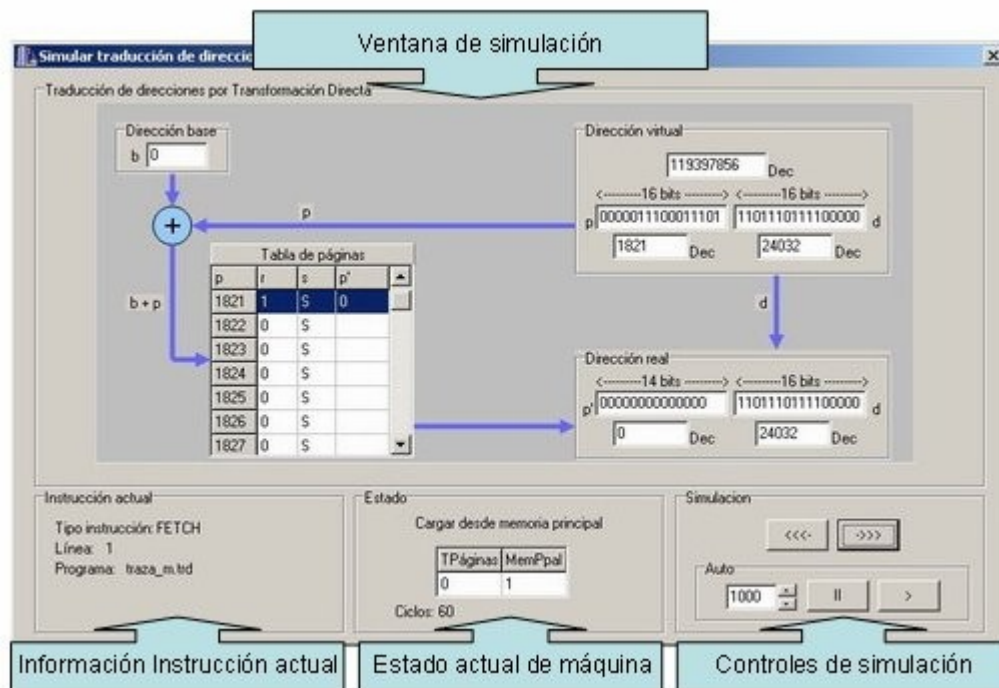
Muestra el proceso de traducción de direcciones virtuales a direcciones reales.

Según el método de traducción que hayamos elegido durante la configuración se nos muestran una de las siguientes ventanas:

- Simular traducción de direcciones por transformación directa.
- Simular traducción de direcciones por transformación asociativa-directa con 1 TLB.
- Simular traducción de direcciones por transformación asociativa-directa con 2 TLB.

A.1. Simular traducción de direcciones por transformación directa

Al iniciar la traducción de direcciones por transformación directa el programa le muestra la siguiente ventana.



Captura 5.14 Simular traducción de direcciones por transformación directa

La ventana está dividida en 4 secciones:

- Ventana de simulación
- Información de la instrucción actual
- Estado actual de la máquina
- Controles de simulación

A.1. Ventana de simulación

La ventana de simulación muestra gráficamente los pasos de cada instante de la simulación.

Se compone de:

- Dirección base

Dirección base de la tabla de páginas.

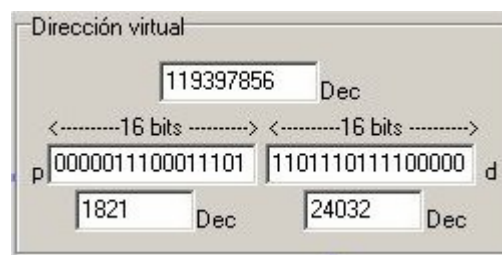


p	r	s	p'
1821	1	S	0
1822	0	S	
1823	0	S	
1824	0	S	
1825	0	S	
1826	0	S	
1827	0	S	

Captura 5.15. Tabla de páginas

Tabla de páginas que se compone de:

- **p**: Página dentro de la tabla de páginas.
 - **r**: Bit de residencia, indica si la página se encuentra cargada en la tabla de páginas.
 - **s**: Dirección de la página en memoria secundaria,
 - **p'**: Página en memoria principal.
- Dirección virtual



Dirección virtual

119397856 Dec

<-----16 bits-----> <-----16 bits----->

p 0000011100011101 1101110111100000 d

1821 Dec 24032 Dec

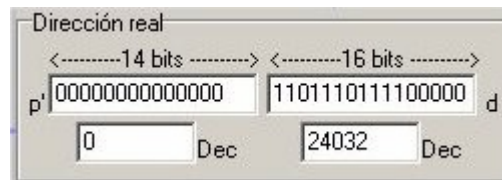
Captura 5.16. Dirección virtual

Dirección de la instrucción en la memoria secundaria.

Dentro del cuadro se muestran:

- Dirección virtual en decimal.
- p: Página dentro de la tabla de páginas, en binario y en decimal.
- Número de bits de p.
- d: Desplazamiento dentro de la página, en binario y en decimal.
- Número de bits de d.

- Dirección real



Captura 5.17. Dirección real

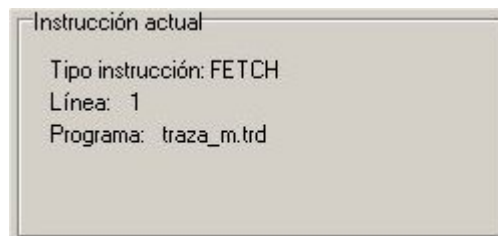
Dirección de la instrucción en la memoria principal

Dentro del cuadro se muestran:

- p': Página en memoria principal, en binario y en decimal.
- Número de bits de p'.
- d: Desplazamiento dentro de la página, en binario y en decimal.
- Número de bits de d.

A.2. Información de la instrucción actual

Muestra la siguiente información:



Captura 5.18. Información de la instrucción actual

- Tipo de acceso:

Tipo de acceso que se está tratando en ese instante.

- FETCH - Búsqueda de instrucciones.
- MEMREAD - Lectura de memoria.
- MEMWRITE - Escritura en memoria.

- Línea

Línea de la traza que se está ejecutando en ese instante.

- Programa

Fichero de traza del programa que se está ejecutando en ese instante

A.3. Estado actual de la máquina

Muestra la siguiente información:

The 'Estado' window displays the following information:

- Fallo de página: Cargar en TP
- A table with two columns: 'Acierto' and 'Fallo'. The 'Acierto' column contains the value '0' and the 'Fallo' column contains the value '1'.
- Ciclos: 0 --> 13500

Captura 5.19. Estado actual de la máquina

- **Estado:**

Estado del simulador en ese instante:

- Fallo de página: Cargar en TP (Tabla de páginas): Debe cargar la página desde memoria secundaria a memoria principal y apuntar en la tabla de páginas.
- Acierto en tabla de páginas - La página buscada se encuentra ya en memoria principal y apuntada en la tabla de páginas.
- Reemplazamiento en memoria principal - La página a cargar desde memoria secundaria no cabe en memoria principal y es necesario reemplazar una de las páginas y actualizar la tabla de páginas.

- **Estadística**

Tabla que contiene el número de aciertos/fallos:

- Fallo - Número de fallos, es decir, número de veces que la página ha tenido que ser cargada en memoria principal desde memoria secundaria y apuntada en la tabla de páginas.
- Acierto - Número de veces que la página se ha encontrado en memoria principal.

- **Ciclos**

Número de ciclos de CPU de la etapa anterior frente a número de ciclos de CPU consumidos en la etapa actual.

A.4. Control de la simulación

Controla la ejecución de la simulación de dos formas:

The 'Simulacion' window contains the following controls:

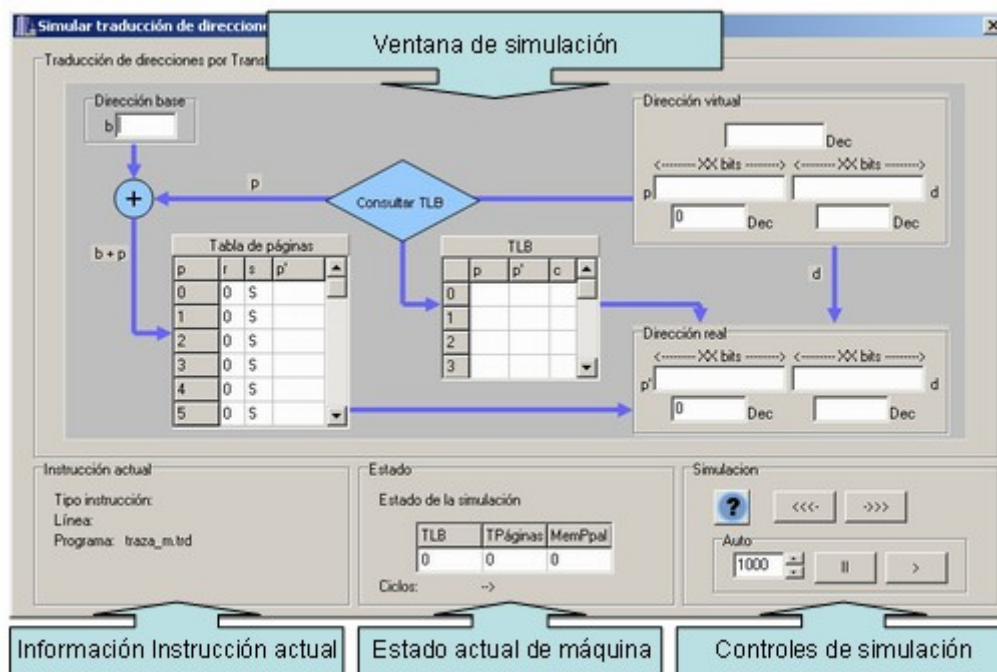
- Buttons for navigation: '<<<-', '>>>-', and '>>>'.
- An 'Auto' section with a counter set to '1000', a 'II' (stop) button, and a '>' (next) button.

Captura 5.20. Control de la simulación

- Manual:
Control manual de la simulación
 - Botón adelante (->>>) - Realiza un paso hacia delante en la simulación.
 - Botón atrás (<<<-) - Realiza un paso hacia delante en la simulación.
- Automático
Control automático de la simulación
 - Botón adelante (>)- Realiza pasos hacia delante en la simulación cada "intervalo" elegido.
 - Botón pausa (II) - Pausa la simulación automática. Intervalo - Intervalo de tiempo que transcurre entre los pasos.
 - Intervalo: Intervalo de tiempo entre pasos. Usar las flechas abajo y arriba para decrementar o incrementar el intervalo.

B. Simular traducción de direcciones por transformación asociativa-directa con TLB

Al iniciar la traducción de direcciones por transformación directa el programa le muestra la siguiente ventana.



Captura 5.21. Traducción de direcciones por transformación asociativa-directa con TLB

La ventana está dividida en 4 secciones:

- Ventana de simulación
- Información de la instrucción actual
- Estado actual de la máquina
- Controles de simulación

B.1. Ventana de simulación

La ventana de simulación muestra gráficamente los pasos de cada instante de la simulación. Se compone de:

- Dirección base

Dirección base de la tabla de páginas.

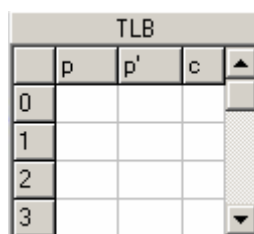


p	r	s	p'
1821	1	S	0
1822	0	S	
1823	0	S	
1824	0	S	
1825	0	S	
1826	0	S	
1827	0	S	

Captura 5.22. Tabla de páginas

Tabla de páginas que se compone de:

- **p**: Página dentro de la tabla de páginas.
 - **r**: Bit de residencia, indica si la página se encuentra cargada en la tabla de páginas.
 - **s**: Dirección de la página en memoria secundaria,
 - **p'**: Página en memoria principal.
- TLB (Translation Lookaside buffer)



	p	p'	c
0			
1			
2			
3			

Captura 5.23. TLB

Se compone de:

- ***n***: Número de entrada en la TLB.
- ***p***: Página dentro de la tabla de páginas.
- ***p'***: Página en memoria principal.
- ***c***: Contador para la política de reemplazamiento.

Implementa una política de reemplazamiento FIFO.

- Dirección virtual

Dirección virtual

119397856 Dec

<-----16 bits-----> <-----16 bits----->

p 0000011100011101 d 1101110111100000

1821 Dec 24032 Dec

Captura 5.24. Dirección virtual

Dirección de la instrucción en la memoria secundaria. Dentro del cuadro se muestran:

- Dirección virtual en decimal.
- ***p***: Página dentro de la tabla de páginas, en binario y en decimal.
- Número de bits de ***p***.
- ***d***: Desplazamiento dentro de la página, en binario y en decimal.
- Número de bits de ***d***.
- Dirección real

Dirección real

<-----14 bits-----> <-----16 bits----->

p' 0000000000000000 d 1101110111100000

0 Dec 24032 Dec

Captura 5.25. Dirección real

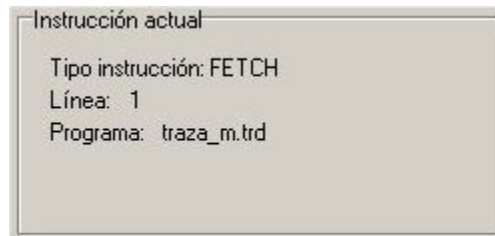
Dirección de la instrucción en la memoria principal

Dentro del cuadro se muestran:

- ***p'***: Página en memoria principal, en binario y en decimal.
- Número de bits de ***p'***.
- ***d***: Desplazamiento dentro de la página, en binario y en decimal.
- Número de bits de ***d***.

B.2. Información de la instrucción actual

Muestra la siguiente información:

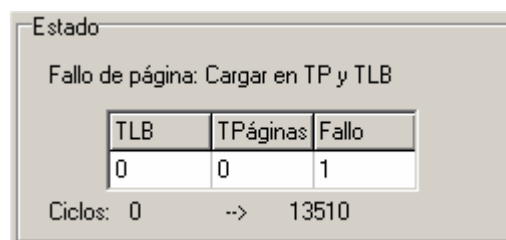


Captura 5.26. Información de la instrucción actual

- Tipo de acceso:
Tipo de acceso que se está tratando en ese instante.
 - FETCH - Búsqueda de instrucciones.
 - MEMREAD - Lectura de memoria.
 - MEMWRITE - Escritura en memoria.
- Línea
Línea de la traza que se está ejecutando en ese instante.
- Programa
Fichero de traza del programa que se está ejecutando en ese instante

B.3. Estado actual de la máquina

Muestra la siguiente información:



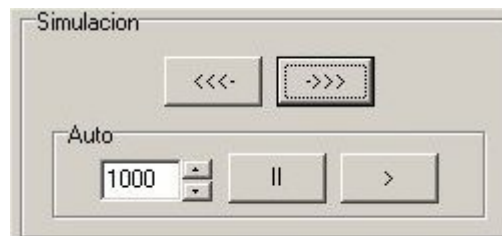
Captura 5.27. Estado actual de la máquina

- **Estado:**
Estado del simulador en ese instante:
 - Fallo de página: Cargar en TP (Tabla de páginas) y TLB - Debe cargar la página desde memoria secundaria a memoria principal y apuntar tanto en la tabla de páginas como en la TLB.
 - Acierto en TLB - La página buscada se encuentra ya en memoria principal y apuntada en la TLB y en la tabla de páginas.

- Acierto en tabla de páginas y cargar en TLB - La página buscada se encuentra ya en memoria principal y apuntada en la tabla de páginas pero no en la TLB, por lo que se actualiza la TLB.
- Reemplazamiento en memoria principal - La página a cargar desde memoria secundaria no cabe en memoria principal y es necesario reemplazar una de las páginas y actualizar tanto la tabla de páginas como la TLB.
- **Estadística**
Tabla que contiene el número de aciertos/fallos:
 - Fallo - Número de fallos, es decir, número de veces que la página ha tenido que ser cargada en memoria principal desde memoria secundaria y apuntada en la tabla de páginas.
 - TPáginas - Número de veces que la página se ha encontrado en memoria principal usando la tabla de páginas.
 - TLB - Número de veces que la página se ha encontrado en memoria principal usando la TLB.
- **Ciclos**
Número de ciclos de CPU de la etapa anterior frente a número de ciclos de CPU consumidos en la etapa actual.

B.4. Control de la simulación

Controla la ejecución de la simulación de dos formas:



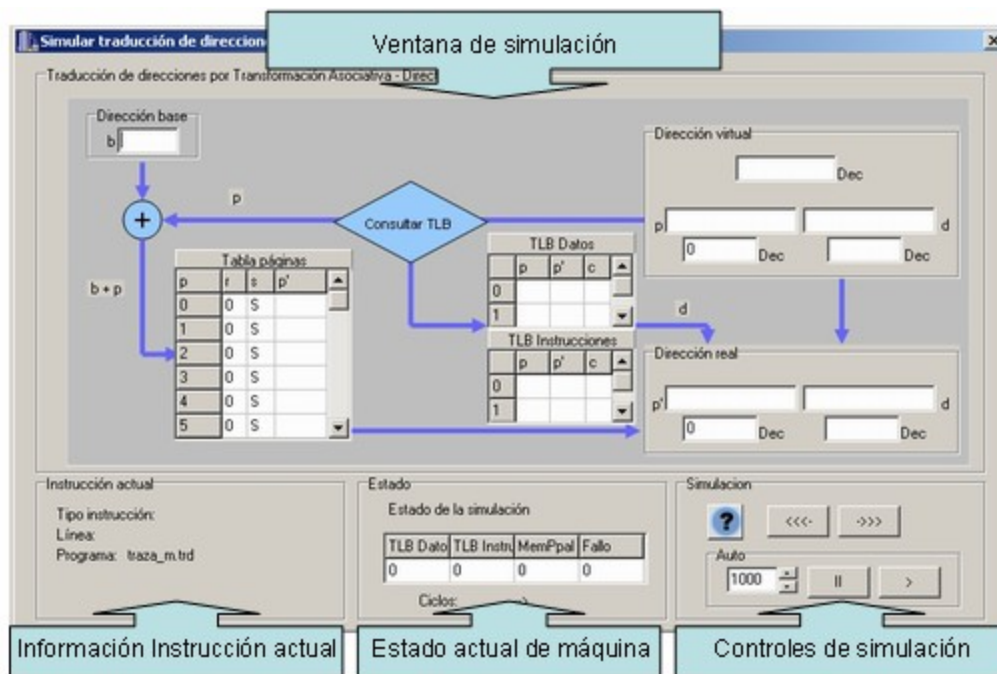
Captura 5.28. Control de la simulación

- Manual:
Control manual de la simulación
 - Botón adelante (->>>) - Realiza un paso hacia delante en la simulación.
 - Botón atrás (<<<-) - Realiza un paso hacia delante en la simulación.
- Automático
Control automático de la simulación
 - Botón adelante (>)- Realiza pasos hacia delante en la simulación cada "intervalo" elegido.

- Botón pausa (II) - Pausa la simulación automática. Intervalo - Intervalo de tiempo que transcurre entre los pasos.
- Intervalo: Intervalo de tiempo entre pasos. Usar las flechas abajo y arriba para decrementar o incrementar el intervalo.

C. Simular traducción de direcciones por transformación asociativa-directa con 2 TLB

Al iniciar la traducción de direcciones por transformación directa el programa le muestra la siguiente ventana.



Captura 5.29. Traducción de direcciones por transformación asociativa-directa con TLB dividida

La ventana está dividida en 4 secciones:

- Ventana de simulación
- Información de la instrucción actual
- Estado actual de la máquina
- Controles de simulación

C.1. Ventana de simulación

La ventana de simulación muestra gráficamente los pasos de cada instante de la simulación. Se compone de:

- Dirección base

Dirección base de la tabla de páginas.

Tabla de páginas				
p	r	s	p'	
1821	1	S	0	
1822	0	S		
1823	0	S		
1824	0	S		
1825	0	S		
1826	0	S		
1827	0	S		

Captura 5.30. Tabla de páginas

Tabla de páginas que se compone de:

- **p**: Página dentro de la tabla de páginas.
 - **r**: Bit de residencia, indica si la página se encuentra cargada en la tabla de páginas.
 - **s**: Dirección de la página en memoria secundaria,
 - **p'**: Página en memoria principal.
- TLB (Translation Lookaside buffer)

Se divide en TLB de datos y TLB de instrucciones.

TLB Datos				
	p	p'	c	
0				
1				

TLB Instrucciones				
	p	p'	c	
0				
1				

Captura 5.31. TLB Dividida

Cada una de las divisiones se compone de:

- **n**: Número de entrada en la TLB.
- **p**: Página dentro de la tabla de páginas.
- **p'**: Página en memoria principal.
- **c**: Contador para la política de reemplazamiento.

Implementa una política de reemplazamiento FIFO.

- Dirección virtual

Dirección virtual

119397856 Dec

<-----16 bits-----> <-----16 bits----->

p 0000011100011101 d 1101110111100000

1821 Dec 24032 Dec

Captura 5.32. Dirección virtual

Dirección de la instrucción en la memoria secundaria. Dentro del cuadro se muestran:

- Dirección virtual en decimal.
 - p: Página dentro de la tabla de páginas, en binario y en decimal.
 - Número de bits de p.
 - d: Desplazamiento dentro de la página, en binario y en decimal.
 - Número de bits de d.
- Dirección real

Dirección real

<-----14 bits-----> <-----16 bits----->

p' 00000000000000 d 1101110111100000

0 Dec 24032 Dec

Captura 5.33. Dirección real

Dirección de la instrucción en la memoria principal. Dentro del cuadro se muestran:

- p': Página en memoria principal, en binario y en decimal.
- Número de bits de p'.
- d: Desplazamiento dentro de la página, en binario y en decimal.
- Número de bits de d.

C.2. Información de la instrucción actual

Muestra la siguiente información:

Instrucción actual

Tipo instrucción: FETCH

Línea: 1

Programa: traza_m.trd

Captura 5.34. Información de la instrucción actual

- Tipo de acceso:

Tipo de acceso que se está tratando en ese instante.

- FETCH - Búsqueda de instrucciones.
- MEMREAD - Lectura de memoria.
- MEMWRITE - Escritura en memoria.

- Línea:

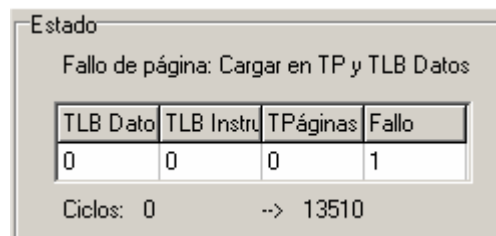
Línea de la traza que se está ejecutando en ese instante.

- Programa:

Fichero de traza del programa que se está ejecutando en ese instante

C.3. Estado actual de la máquina

Muestra la siguiente información:



TLB Dato	TLB Instru	TPáginas	Fallo
0	0	0	1

Ciclos: 0 --> 13510

Captura 5.35. Estado actual de la máquina

- **Estado:**

Estado del simulador en ese instante:

- Fallo de página: Cargar en TP (Tabla de páginas) y TLB (Datos o Instrucciones) - Debe cargar la página desde memoria secundaria a memoria principal y apuntar tanto en la tabla de páginas como en la TLB (Datos o Instrucciones).
- Acierto en TLB (Datos o Instrucciones) - La página buscada se encuentra ya en memoria principal y apuntada en la TLB y en la tabla de páginas.
- Acierto en tabla de páginas y cargar en TLB (Datos o Instrucciones) - La página buscada se encuentra ya en memoria principal y apuntada en la tabla de páginas pero no en la TLB, por lo que se actualiza la TLB (Datos o Instrucciones).
- Reemplazamiento en memoria principal - La página a cargar desde memoria secundaria no cabe en memoria principal y es necesario reemplazar una de las páginas y actualizar tanto la tabla de páginas como la TLB (Datos o Instrucciones).

- **Estadística**

Tabla que contiene el número de aciertos/fallos:

- Fallo - Número de fallos, es decir, número de veces que la página ha tenido que ser cargada en memoria principal desde memoria secundaria y apuntada en la tabla de páginas.
- TPáginas - Número de veces que la página se ha encontrado en memoria principal usando la tabla de páginas.

TLB (Datos o Instrucciones) - Número de veces que la página se ha encontrado en memoria principal usando la TLB (Datos o Instrucciones).

- **Ciclos**

Número de ciclos de CPU de la etapa anterior frente a número de ciclos de CPU consumidos en la etapa actual.

C.4. Control de la simulación

Controla la ejecución de la simulación de dos formas:



Captura 5.36. Control de la simulación

- **Manual:**

Control manual de la simulación

- Botón adelante (->>>) - Realiza un paso hacia delante en la simulación.
- Botón atrás (<<<-) - Realiza un paso hacia atrás en la simulación.

- **Automático**

Control automático de la simulación

- Botón adelante (>)- Realiza pasos hacia delante en la simulación cada "intervalo" elegido.
- Botón pausa (II) - Pausa la simulación automática. Intervalo - Intervalo de tiempo que transcurre entre los pasos.

- Intervalo: Intervalo de tiempo entre pasos. Usar las flechas abajo y arriba para decrementar o incrementar el intervalo.

5.2.2.3.2. Simular búsqueda de direcciones en memoria

Muestra el proceso de búsqueda de direcciones dentro de la jerarquía de memoria. La máquina simulada sigue la siguiente estructura:

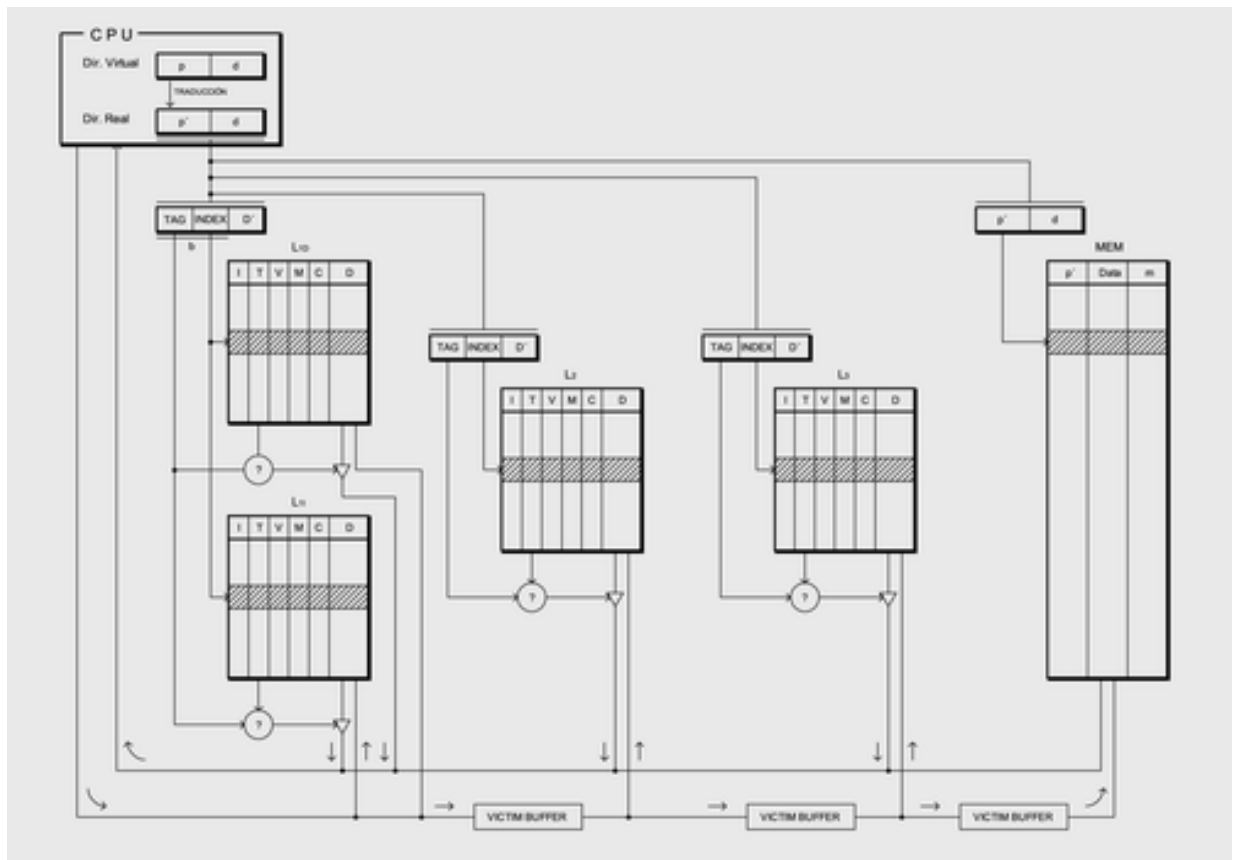


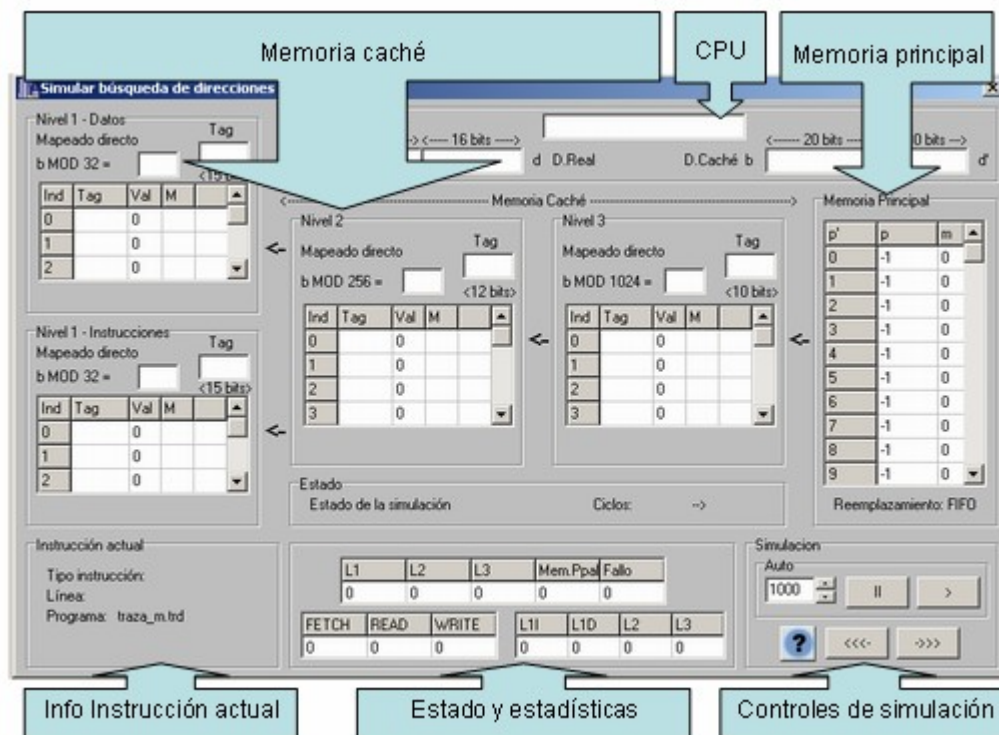
Figura 5.1. Estructura de la jerarquía de memoria implementada

NOTA: Los diferentes niveles de caché pueden o no estar activos, así como en escritura retardada el/los Victim Buffer/s no está/n presentes.

Al elegir esta opción se muestra la siguiente ventana que está dividida en 7 secciones:

- Memoria principal
- Memoria caché
- CPU
- Estado actual de la máquina

- Información de la instrucción actual
- Controles de simulación



Captura 5.37. Búsqueda de direcciones/páginas

A. Memoria principal

Muestra la memoria principal. Se compone de:

- p': Página en memoria principal.
- p: Página dentro de la tabla de páginas.
- m: Bit de modificación, indica si la página se ha modificado mientras se encontraba cargada en memoria principal.

Memoria Principal		
p'	p	m
0	-1	0
1	-1	0
2	-1	0
3	-1	0
4	-1	0
5	-1	0
6	-1	0
7	-1	0
8	-1	0
9	-1	0

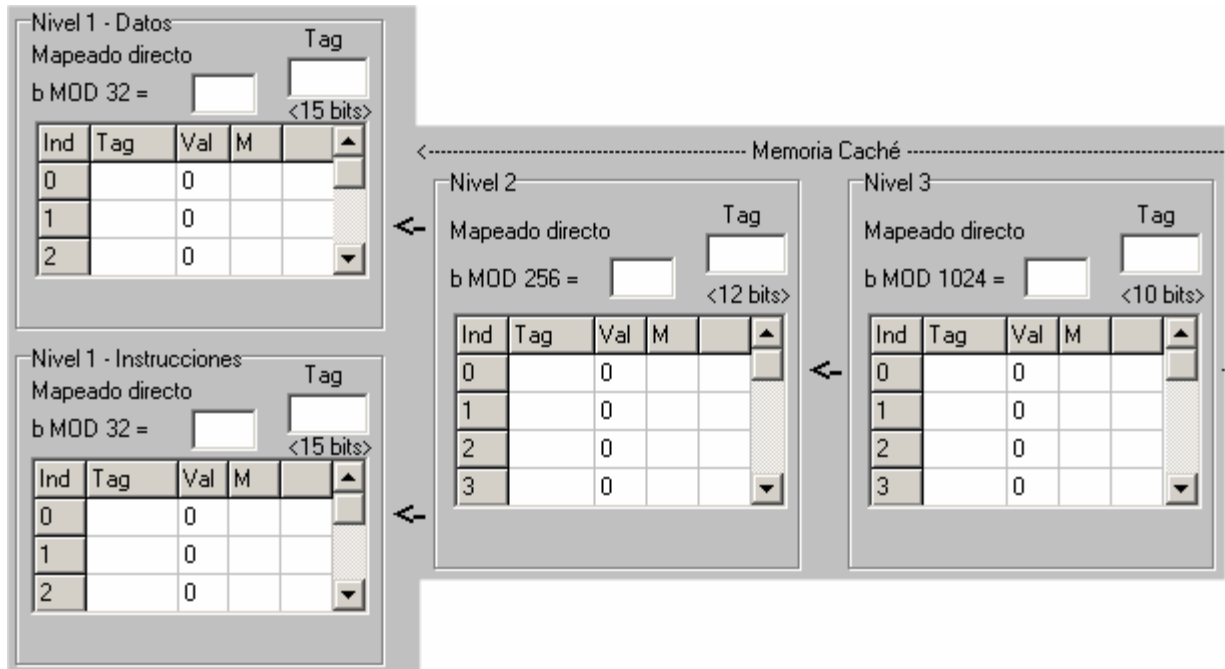
Reemplazamiento: FIFO

Captura 5.38. Memoria principal

La memoria principal implementa un algoritmo de reemplazamiento FIFO.

B. Memoria Caché

Muestra los diferentes niveles de caché según la configuración que hayamos elegido al configurar el simulador.



Captura 5.39. Memoria caché

Según la configuración elegida se nos mostrarán:

- Caché de nivel 3.
- Caché de nivel 2.
- Caché de nivel 1, bien conjunta o bien separada en Nivel 1 - Datos y Nivel 1 - Instrucciones.

Cada una de estos niveles se compone de:

- Organización elegida para ese nivel de caché.
- Cálculo de la posición del bloque en ese nivel de caché (sólo en "Mapeado directo" y "Asociativa por conjuntos").
- Tag: Etiqueta con el Tag buscado en ese instante en ese nivel de caché, así como su tamaño en bits.
- Política de reemplazamiento elegida para ese nivel de caché (no en "Mapeado directo").
- Contenido de ese nivel de caché:
 - Ind (Index): Índice del bloque de memoria caché.

- Tag: Etiqueta del bloque de memoria caché.
- Val (Valid): Bit válido del bloque de memoria caché.
- M (Modified): Bit de modificación, indica si el bloque se ha modificado mientras se encontraba cargado en ese nivel de caché.
- C (Counter): Bit contador, bit utilizado para aplicar las diferentes políticas de reemplazamiento (no aparece si elegimos mapeado directo o política de reemplazamiento clock o al azar).

Cuando elegimos CLOCK como política de reemplazamiento se nos muestra un cuadro con el contador en ese instante.

C. CPU

Muestra la dirección física solicitada por la CPU una vez ha sido obtenida de la dirección virtual.

Captura 5.40. CPU

Se compone de:

- Dirección real en binario: Muestra la dirección real en binario solicitada por la CPU.
- Dirección real en decimal compuesta por:
 - p': Página real.
 - d: Desplazamiento dentro de la página.

Sobre ella se muestran los bits que componen cada una de las divisiones.

- Dirección caché en decimal compuesta por:
 - b: Bloque.
 - d': Desplazamiento dentro del bloque.

Sobre ella se muestran los bits que componen cada una de las divisiones.

D. Estado actual de la máquina

Muestra la siguiente información:

Estado

Estado de la simulación

Ciclos: -->

L1	L2	L3	Mem.Ppal	Fallo
0	0	0	0	0

FETCH	READ	WRITE
0	0	0

L1I	L1D	L2	L3
0	0	0	0

Captura 5.41. Estado actual de la máquina

• Estado:

Estado del simulador en ese instante

- Cargar desde memoria principal - Debe cargar la instrucción/datos desde memoria secundaria a memoria principal y los diferentes niveles de cache.
- Acierto en memoria principal - La instrucción/datos se encuentran cargados en memoria principal, y deben ser cargados a los diferentes niveles de cache.
- Acierto en caché de nivel X (de datos y/o de instrucciones) - La instrucción/datos se encuentran cargados en el nivel X de caché y deben ser cargados a los niveles superiores.

• Ciclos:

Número de ciclos de CPU de la etapa anterior frente a número de ciclos de CPU consumidos en la etapa actual.

• Estadística

Tabla que contiene el número de aciertos/fallos

- L1 - Número de veces que la instrucción se ha encontrado en la cache de nivel 1.
- L2 - Número de veces que la instrucción se ha encontrado en la cache de nivel 2.
- L3 - Número de veces que la instrucción se ha encontrado en la cache de nivel 3.
- MemPpal - Número de veces que la instrucción se ha encontrado en memoria principal.
- Fallo - Número de fallos, es decir, número de veces que la instrucción ha tenido que ser cargada en memoria principal.

Número de instrucciones de cada tipo:

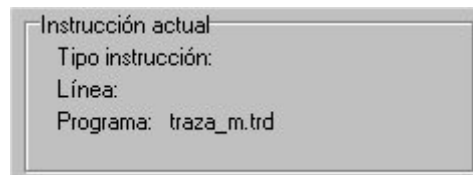
- FETCH - Búsqueda de instrucciones.
- MEMREAD - Lectura de memoria.
- MEMWRITE - Escritura en memoria.

Número de reemplazamientos:

- L1I - Número de reemplazamientos en la caché de instrucciones de nivel 1.
- L1D - Número de reemplazamientos en la caché de datos de nivel 1.
- L2 - Número de reemplazamientos en la caché de instrucciones de nivel 2.
- L3 - Número de reemplazamientos en la caché de instrucciones de nivel 3.

E. Información de la instrucción actual

Muestra la siguiente información:



Captura 5.42. Información de la instrucción actual

- Tipo de acceso:

Tipo de acceso que se está tratando en ese instante.

- FETCH - Búsqueda de instrucciones.
- MEMREAD - Lectura de memoria.
- MEMWRITE - Escritura en memoria.

- Línea

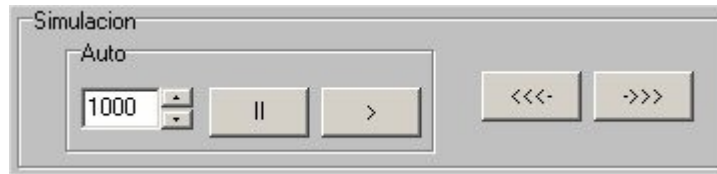
Línea de la traza que se está ejecutando en ese instante.

- Programa

Fichero de traza del programa que se está ejecutando en ese instante

F. Control de la simulación

Controla la ejecución de la simulación de dos formas:



Captura 5.43. Control de la simulación

- Manual:

Control manual de la simulación

- Botón adelante (->>>) - Realiza un paso hacia delante en la simulación.
- Botón atrás (<<<-) - Realiza un paso hacia delante en la simulación.

- Automático

Control automático de la simulación

- Botón adelante (>)- Realiza pasos hacia delante en la simulación cada "intervalo" elegido.
- Botón pausa (II) - Pausa la simulación automática. Intervalo - Intervalo de tiempo que transcurre entre los pasos.
- Intervalo: Intervalo de tiempo entre pasos. Usar las flechas abajo y arriba para decrementar o incrementar el intervalo.

5.2.3. Ficheros de SIJEM

- Tipos de ficheros
- Ficheros de ejemplo

5.2.3.1. Tipos de ficheros de SIJEM

Para la ejecución del simulador es necesario un fichero de traza, que puede ser de dos tipos:

- Fichero de direcciones (extensión .trd)
Contiene las direcciones virtuales referenciadas por un programa, real o simulado, según la siguiente estructura:
 - DIRECCION: Contiene una dirección virtual de X bit que representa la dirección generada por el programa en ejecución.
 - TIPO DE INSTRUCCIÓN: Indica el tipo de instrucción, que puede ser:

- FETCH - Búsqueda de instrucciones.
 - MEMREAD - Lectura de memoria.
 - MEMWRITE - Escritura en memoria.
 - TIEMPO: Indica el tiempo (Delta) transcurrido desde la última petición de página en ciclos de reloj.
- Fichero de páginas (extensión .trp)

Contiene las páginas referenciadas por un programa, real o simulado, según la siguiente estructura:

 - PÁGINA: Contiene la página referenciada por el programa en ejecución.
 - TIPO DE INSTRUCCIÓN: Indica el tipo de instrucción, que puede ser:
 - FETCH - Búsqueda de instrucciones.
 - MEMREAD - Lectura de memoria.
 - MEMWRITE - Escritura en memoria.
 - DIRECCION: Contiene una dirección virtual de X bit que representa la dirección generada por el programa en ejecución. En caso de no conocerla puede ser 0 siempre.

Y opcionalmente podemos guardar todos los parámetros de configuración en otro fichero:

- Fichero de configuración (extensión .cfg)

Puede contener todos o algunos de los parámetros de configuración, según la estructura:

```

/Tamaño total de memoria secundaria (en Kbytes)
-1:
/Tiempo acceso a memoria secundaria (en ciclos)
-2:
/Ancho bus Memoria Secundaria-Memoria principal (en bits)
-3:
/Tiempo acceso bus Memoria Secundaria-Memoria principal (en ciclos)
-4:
/Tamaño total memoria principal (en Kbytes)
-5:
/Tamaño página (en Kbytes)
-6:
/Tiempo acceso memoria principal (en ciclos)
-7:
/Traducción direcciones
/(1 - Directa, 2 - Asociativa directa con 1 TLB, 3 - Asociativa directa con 2 TLB)
-8:
/Número de entradas de la TLB
-9:
/Tiempo de acceso a la TLB (en ciclos)
-10:
/Tipo de paginación 1
/(1 - Por demanda, 2 - Anticipada)
-11:
/Tipo de paginación 2

```

/(1 - Escritura directa, 2 - Escritura retardada)
-12:
/Caché habilitada
/(0 - NO, 1 - SI)
-13:
/Tamaño del bloque de caché (en bytes)
-14:
/Caché de nivel 3 habilitada?
/(0 - NO, 1 - SI)
-15:
/Organización de la cache de nivel 3
/(1 - Mapeado directo, 2 - Completamente asociativa, 3 - Asociativa por conjuntos)
-16:
/Política de reemplazamiento de cache de nivel 3
/(1 - AZAR, 2 - FIFO, 3 - LRU, 4 - LFU, 5 - NUR, 6 - CLOCK)
-17:
/Tamaño de la caché de nivel 3 (en Kbytes)
-18:
/Tiempo de acceso a la caché de nivel 3 (en Ciclos)
-19:
/Ancho bus memoria caché nivel 3 a nivel superior (en Bits)
-20:
/Tiempo acceso bus memoria caché nivel 3 a nivel superior (en Ciclos)
-21:
/Número de vías de la caché de nivel 3 (si es asociativa por conjuntos)
-22:
/Caché de nivel 2 habilitada?
/(0 - NO, 1 - SI)
-23:
/Organización de la cache de nivel 2
/(1 - Mapeado directo, 2 - Completamente asociativa, 3 - Asociativa por conjuntos)
-24:
/Política de reemplazamiento de cache de nivel 2
/(1 - AZAR, 2 - FIFO, 3 - LRU, 4 - LFU, 5 - NUR, 6 - CLOCK)
-25:
/Tamaño de la caché de nivel 2 (en Kbytes)
-26:
/Tiempo de acceso a la caché de nivel 2 (en Ciclos)
-27:
/Ancho bus memoria caché nivel 2 a nivel superior (en Bits)
-28:
/Tiempo acceso bus memoria caché nivel 2 a nivel superior (en Ciclos)
-29:
/Número de vías de la caché de nivel 2 (si es asociativa por conjuntos)
-30:
/Caché de nivel 1 habilitada?
/(0 - NO, 1 - SI)
-31:
/Tipo de caché de nivel 1
/(1 - Conjunta, 2 - Separada)
-32:
/Organización de la cache de nivel 1
/(1 - Mapeado directo, 2 - Completamente asociativa, 3 - Asociativa por conjuntos)
-33:
/Política de reemplazamiento de cache de nivel 1
/(1 - AZAR, 2 - FIFO, 3 - LRU, 4 - LFU, 5 - NUR, 6 - CLOCK)
-34:
/Tamaño de la caché de nivel 1 (en Kbytes)

- 35:
/Tiempo de acceso a la caché de nivel 1 (en Ciclos)
- 36:
/Ancho bus memoria caché nivel 1 a nivel superior (en Bits)
- 37:
/Tiempo acceso bus memoria caché nivel 1 a nivel superior (en Ciclos)
- 38:
/Número de vías de la caché de nivel 1 (si es asociativa por conjuntos)
- 39:

Ejemplo 5.1. Ficheros de configuración

Para insertar comentarios se utiliza '/'.

5.2.3.2. Ficheros de ejemplo de SIJEM

En la carpeta "Ejemplos" del directorio donde hayamos instalado el simulador se proporcionan una serie de ejemplos para ejecutar pruebas:

- Fichero de configuración (extensión .cfg)
- Ficheros para el uso con traducción de direcciones

	<i>Ejem1.cfg</i>	<i>Ejem2.cfg</i>	<i>Ejem3.cfg</i>
Tamaño total memoria secundaria	1024 Kbytes		
Tiempo acceso memoria secundaria	10000 ciclos		
Ancho bus memoria principal- memoria secundaria	20 bits		
Tiempo acceso bus memoria principal-memoria secundaria	1500 ciclos		
Tamaño total memoria principal	64 Kbytes		
Tamaño página	4 Kbytes		
Tiempo acceso memoria principal	1000 ciclos		
Traducción de direcciones	Transformación directa	Transformación asociativa-directa con TLB	Transformación asociativa-directa con TLB dividida
Carga de páginas	Por demanda		
Coherencia	Escritura directa		
Tamaño bloque caché	0 bytes		
Caché nivel 3 habilitada	NO		
Caché nivel 2 habilitada	NO		
Caché nivel 1 habilitada	NO		

Tabla 5.1. Ficheros de configuración de ejemplo

- Ficheros para el uso con búsqueda de direcciones

	<i>Ejem4.cfg</i>	<i>Ejem5.cfg</i>	<i>Ejem6.cfg</i>	<i>Ejem7.cfg</i>
Tamaño memoria secundaria	4194304 Kbytes (4Gbytes)			
T. acceso memoria secundaria	10000 ciclos			
Ancho bus memoria principal	32 bits			
Tamaño total memoria principal	1500 ciclos			
Tamaño página	4 Kbytes			
T. acceso memoria principal	1000 ciclos			
Traducción de direcciones	Transformación directa			
Carga de páginas	Por demanda			
Coherencia	Escritura directa			
Tamaño bloque caché	1024 Kbytes (1 Mbyte)			
Caché nivel 3 habilitada	SI			
Organización caché nivel 3	Mapeado directo	Completamente asociativa		
Política reemplazamiento nivel 3	-	AZAR	FIFO	LRU
Tamaño caché nivel 3	16 Kbytes			
Tiempo acceso caché nivel 3	100 ciclos			
Ancho bus caché nivel 3	20 bits			
Tiempo acceso bus caché nivel 3	50 ciclos			
Caché nivel 2 habilitada	SI			
Organización caché nivel 2	Mapeado directo	Completamente asociativa		
Política reemplazamiento nivel 2	-	AZAR	FIFO	LRU
Tamaño caché nivel 2	8 Kbytes			
Tiempo acceso caché nivel 2	50 ciclos			
Ancho bus caché nivel 2	14 bits			
Tiempo acceso bus caché nivel 2	25 ciclos			
Caché nivel 1 habilitada	SI			
Tipo caché nivel 1	Separada			
Organización caché nivel 1	Mapeado directo	Completamente asociativa		
Política reemplazamiento nivel 1	-	AZAR	FIFO	LRU
Tamaño caché nivel 1	4 Kbytes (Datos) + 4 Kb (Instrucciones)			
Tiempo acceso caché nivel 1	10 ciclos			
Ancho bus caché nivel 1	13 bits			
Tiempo acceso bus caché nivel 1	5 ciclos			

Tabla 5.2. Ficheros de configuración de ejemplo

	<i>Ejem8.cfg</i>	<i>Ejem9.cfg</i>	<i>Ejem10.cfg</i>	<i>Ejem11.cfg</i>
Tamaño memoria secundaria	4194304 Kbytes (4Gbytes)			
T. acceso memoria secundaria	10000 ciclos			
Ancho bus memoria principal	32 bits			
Tamaño total memoria principal	1500 ciclos			
Tamaño página	4 Kbytes			
T. acceso memoria principal	1000 ciclos			
Traducción de direcciones	Transformación directa			
Carga de páginas	Por demanda			
Coherencia	Escritura directa			
Tamaño bloque caché	1024 Kbytes (1 Mbyte)			
Caché nivel 3 habilitada	SI			
Organización caché nivel 3	Completamente asociativa			Asociativa por cjtos de 2 vías
Política reemplazamiento nivel 3	LFU	NUR	CLOCK	AZAR
Tamaño caché nivel 3	16 Kbytes			
Tiempo acceso caché nivel 3	100 ciclos			
Ancho bus caché nivel 3	20 bits			
Tiempo acceso bus caché nivel 3	50 ciclos			
Caché nivel 2 habilitada	SI			
Organización caché nivel 2	Completamente asociativa			Asociativa por cjtos de 2 vías
Política reemplazamiento nivel 2	LFU	NUR	CLOCK	AZAR
Tamaño caché nivel 2	8 Kbytes			
Tiempo acceso caché nivel 2	50 ciclos			
Ancho bus caché nivel 2	14 bits			
Tiempo acceso bus caché nivel 2	25 ciclos			
Caché nivel 1 habilitada	SI			
Tipo caché nivel 1	Separada			
Organización caché nivel 1	Completamente asociativa			Asociativa por cjtos de 2 vías
Política reemplazamiento nivel 1	LFU	NUR	CLOCK	AZAR
Tamaño caché nivel 1	4 Kbytes (Datos) + 4 Kb (Instrucciones)			
Tiempo acceso caché nivel 1	10 ciclos			
Ancho bus caché nivel 1	13 bits			
Tiempo acceso bus caché nivel 1	5 ciclos			

Tabla 5.3. Ficheros de configuración de ejemplo

- Fichero de direcciones (extensión .trd)

Para las simulaciones se incluyen una serie de ficheros de traza de programas reales extraídos de ejecuciones del benchmark SPEC2000.

Su tipo se detalla en la siguiente tabla:

	Descripción
Crafty	Programa de ajedrez
Eon_cook	Programa de trazado de rayos sobre una tetera
Facerec	Programa de reconocimiento de caras
Gcc	Compilador de C y C++
Mesa	Librería de gráficos 3D
Vortex	Base de datos

Tabla 5.4. Ficheros de direcciones de ejemplo

Existen 5 ficheros de cada tipo, cada uno de ellos con un tipo de direcciones:

- Terminado en a: Direcciones de 32 bits (Ejemplos del 5 al 11) - Memoria virtual hasta 4GB.
- Terminado en b: Direcciones de 28 bits - Memoria virtual hasta 256MB.
- Terminado en c: Direcciones de 24 bits - Memoria virtual hasta 16MB.
- Terminado en d: Direcciones de 20 bits (Ejemplos del 1 al 3) - Memoria virtual hasta 1MB.
- Terminado en e: Direcciones de 16 bits - Memoria virtual hasta 64kB.

Capítulo 6.

Tecnología educativa

Ya que el objetivo de este proyecto era realizar un programa de simulación que sirviese como herramienta de apoyo a la docencia, durante la etapa de desarrollo y posteriormente de pruebas, debía seguirse una metodología eficaz que garantizara la capacidad didáctica del mismo. Para ello se optó por guiarse según las pautas que nos propone el campo de estudio conocido como *Tecnología Educativa*.

6.1. ¿Qué es tecnología educativa?

La *Tecnología Educativa* es una disciplina que aborda los nuevos retos surgidos de aplicar las tecnologías de la información y la comunicación a la enseñanza.

Su origen se remonta a la formación militar norteamericana en los años cuarenta, cuando nació un enfoque de la enseñanza caracterizado por la búsqueda de procesos eficaces de formación utilizando medios y recursos técnicos sofisticados.

6.2. Tecnologías de la información y la comunicación (TIC)

Se denominan Tecnologías de la Información y comunicación, en adelante TIC, al conjunto de tecnologías que permiten la adquisición, almacenamiento y presentación de informaciones en forma de voz, imágenes y datos.

Las TIC se fundamentan en el uso de los ordenadores y las comunicaciones como el canal para transmitir la información.

6.2.1. Las TIC en la enseñanza

Más importante que aprender tecnología es aprender con la tecnología, conocer cómo utilizar y procesar la información y cómo trabajar en el nuevo entorno que ofrece la sociedad de la información.

En este nuevo contexto, el profesorado ha de detectar qué nuevos métodos de alfabetización requiere el medio tecnológico y adaptarlos a la enseñanza de las diferentes materias.

6.2.1.1. Uso de las TIC en la enseñanza

El uso de las TIC potencia:

- *La habilidad para recopilar, sistematizar y usar información* gracias a las diferentes formas de acceso a ella.
- *El fortalecimiento del pensamiento visual.* Proveyendo una nueva forma de conocimiento basada en leer imágenes que implican intuición en el estudiante.
- *Nuevas formas de interacción.* La interacción hombre-máquina alcanza una nueva dimensión gracias a las comunicaciones, convirtiendo al ordenador en un miembro de una comunidad en las que tanto los estudiantes como los profesores, enseñan y aprenden.

6.2.1.2. Objetivo de las TIC en la enseñanza

Los objetivos que se pretenden para un profesor con el uso de las TIC son:

- Explorar cómo las tecnologías ayudan a crear un ambiente de aprendizaje en el que se favorecen procesos de construcción del pensamiento.
- Desarrollar habilidades necesarias para manipular con precisión herramientas, objetos y sistemas tecnológicos.
- Encontrar situaciones reales centradas en la resolución de problemas, que sirvan para que los propios estudiantes sean pequeños investigadores.
- Desarrollar interés y curiosidad hacia la actividad tecnológica, generando iniciativas de investigación, así como de búsqueda y elaboración de nuevas realidades tecnológicas.
- Utilizar Internet para localizar información en diversos soportes, contenida en diferentes fuentes (páginas Web, imágenes, sonidos, programas de libre uso).
- Intercambiar y comunicar ideas utilizando las posibilidades de Internet (correo electrónico, Chat, videoconferencias, etc.).

6.3. Los medios didácticos

Dentro del proceso de enseñanza y aprendizaje se utilizan una serie de medios de información y comunicación elaborados específicamente para realizar funciones didácticas. En este sentido se distinguen tres funciones básicas:

- Función informativa: relacionada con la adquisición de conocimientos.
- Función motivadora: relacionada con la transmisión de emociones, sensaciones y la estimulación de la imaginación.

- Función instructiva: enfocada a la organización del conocimiento y al desarrollo de destrezas.

6.3.1. Tipos de medios

Existen múltiples clasificaciones de los posibles medios a utilizar en el proceso de enseñanza y aprendizaje, aunque la más común suele ser clasificarlos en:

- *Medios manipulativos*, tales como juguetes, pelotas, cuerdas, regletas, ...
- *Medios impresos*, tales como libros de texto, fotocopias, guías didácticas, apuntes, ...
- *Medios audiovisuales*, tales como diapositivas, transparencias, vídeos, cds audio, ...
- *Medios digitales*, que posibilitan utilizar y combinar cualquier tipo de representación de la información

Dentro de ésta última clasificación se incluye al software educativo, que es un subconjunto de las TIC, definido como programas de ordenador creados con la finalidad de ser utilizados como medio didáctico.

6.4. Teorías de enseñanza y aprendizaje aplicadas al diseño de software educativo

La mayor parte de los modelos de elaboración de programas de software educativo se basan en modelos didácticos ya existentes que incorporan conceptos de teorías sobre el aprendizaje (hacen referencia a las teorías que intentan explicar cómo aprendemos) y la enseñanza (teorías que pretenden determinar las condiciones óptimas para enseñar).

En esta sección se hace un repaso de las diferentes teorías existentes, pues ofrecen aspectos interesantes que siempre deben ser tenidas en cuenta a la hora del diseño de un software educativo. Para su elaboración se ha tomado como base el módulo VI del Master Duria [8] y el libro Los medios y las tecnologías en la educación [7], por lo que se recomienda su lectura de cara a ampliar los conocimientos en este campo.

6.4.1. Aproximación conductista al diseño de software educativo

El impulso más importante de las teorías del aprendizaje con relación a la enseñanza y su aplicación posterior a los primeros programas informáticos educativos se debe, sobre todo, a las aportaciones de Skinner y al desarrollo de la enseñanza programada.

Según Skinner “*el sujeto no absorbe pasivamente el conocimiento del mundo sino que debe jugar un papel activo*”. En su obra explica cómo las personas aprenden haciendo, experimentando y ensayando, para lo que estableció una serie de leyes de aprendizaje cuyo objetivo es explicar las relaciones estímulo-respuesta-refuerzo que se pueden producir en las diferentes situaciones de aprendizaje.

Dos de las aportaciones educativas de Skinner son:

- La enseñanza programada
- La máquina de enseñar.

6.4.1.1. La enseñanza programada

Las bases para los procesos de programación educativa y la enseñanza programada se fundamentan en:

- *La formulación del objetivo* en términos lo más descriptivos posibles.
- *Establecer una jerarquía de aprendizaje* determinando las tareas y subtarear que constituirán el proceso de enseñanza.
- *El análisis de las tareas*, estableciendo aquellos aspectos que deben ser aprendidos con esa tarea
- *La evaluación del programa en función de los objetivos propuestos*, de forma constante e individualizada según se van concluyendo las tareas.

6.4.1.2. La máquina de enseñar

En 1954, Skinner publicó un artículo titulado *La ciencia del aprendizaje y el arte de la enseñanza* en el cual señalaba las deficiencias de las técnicas educativas tradicionales e indicaba que la utilización de máquinas de enseñar podía ayudar a solucionar muchos de los problemas de la educación.

La máquina de enseñar diseñada por Skinner estaba formada por una pantalla y un carrete que servía para correr el rollo de papel, el cual contenía una serie de preguntas que el alumno debía responder accionando una palanca. Si la respuesta era correcta, se pasaba a la siguiente pregunta, en caso contrario el papel quedaba fijo.

El objetivo fundamental de la máquina de enseñar era, según Skinner, asegurar que el refuerzo fuera inmediato y obligar al alumno a emitir una respuesta que pudiera ser reforzada a continuación.

La idea de base de la máquina de enseñar era la misma que en la enseñanza programada, pero utilizando un instrumento que permitiera hacer el proceso de forma individualizada y más rápidamente que mediante la intervención del profesor. Como puede suponerse, la influencia de este artilugio ha sido decisiva en el desarrollo de la enseñanza asistida por ordenador.

6.4.2. Aproximación cognitivista al diseño de software educativo

Junto a la teoría conductista surge la teoría cognitivista, que aprovecha del conductismo la importancia de los refuerzos y el análisis de las tareas, y añade los principios de dos nuevas teorías, el aprendizaje significativo de Ausubel y el procesamiento de la información.

6.4.2.1. Aprendizaje significativo de Ausubel

El aprendizaje significativo se opone al aprendizaje “memorístico” o “mecánico” incluyendo técnicas que relacionen los nuevos conocimientos con los previos.

Para implementar estas técnicas destaca las posibilidades de los ordenadores en la enseñanza en tanto posibilitan el control de muchas variables de forma simultánea, si bien considera necesario que su utilización en este ámbito venga respaldada por “una teoría validada empíricamente de la recepción significativa y el aprendizaje por descubrimiento”.

6.4.2.2. Teorías del procesamiento de la información

Las teorías del procesamiento de la información nos dicen que, para que se puedan alcanzar los objetivos de aprendizaje, deberán darse una serie de *condiciones internas* (requisitos previos, aprendizajes anteriores) y *condiciones externas* (influencia del medio en el sujeto).

Las diferentes combinaciones de estas condiciones darán lugar a diferentes tipos de aprendizaje: habilidades intelectuales, estrategias cognitivas, información verbal, destrezas motrices y actitudes.

Uno de los aspectos más importantes de las teorías del procesamiento de la información es la serie de fases por las que pasa el sujeto durante su aprendizaje:

- *Fase de motivación*: el sujeto debe estar motivado para conseguir un cierto objetivo, y tiene que recibir una recompensa citando lo alcanza.
- *Fase de comprensión*: cuando se presenta algún estímulo externo, el sujeto no lo percibe en su totalidad, sino que selecciona algunos aspectos de dicho estímulo.
- *Fase de adquisición*: una vez percibido el estímulo se entra en la fase de adquisición, durante la cual el individuo reconstruye la información recibida para almacenarla en la memoria.
- *Fase de retención*: la información ya codificada, llega al almacén de la memoria a largo plazo donde será organizada para poder ser recuperada.
- *Fase de recuerdo*: cuando la información es retenida, hemos de comprobar que puede ser recuperada cuando la necesitemos.
- *Fase de generalización*: uno de los objetivos más importantes del aprendizaje son la transferencia y la generalización, consistentes en aplicar los conocimientos aprendidos y recordados a nuevas situaciones.
- *Fase de ejecución*: en el proceso de aprendizaje la única fase que puede ser observada es la de la actuación, en la que el sujeto ejecuta una respuesta, de modo que pone en práctica aquello que ha aprendido.

- *Fase de realimentación:* el profesor comprueba que el alumno ha adquirido cierto conocimiento o habilidad, y lo que es más importante, el propio alumno lo percibe.

Para que el alumno experimente estas fases, establece una serie de requerimientos al profesor, que dejarán de ser aplicados cuando sea el mismo alumno el que los aporte:

- *Informar al alumno del objetivo a conseguir.*
- *Dirigir la atención a las partes a enseñar.*
- *Estimular el recuerdo.*
- *Presentar el estímulo.*
- *Guiar el aprendizaje.*
- *Producir la actuación.*
- *Valorar la actuación.*
- *Proporcionar realimentación.*
- *Promover la retención y fomentar la transferencia.*

6.4.3. Aproximación constructivista al diseño de software educativo

Las teorías constructivistas se caracterizan por retomar algunos postulados de diferentes teorías:

- de la *teoría genética* comparten el concepto de actividad mental constructiva.
- de la *teoría del procesamiento de la información* toman la idea de la organización de los conocimientos
- de la *teoría del aprendizaje significativo de Ausubel*, la idea de la construcción de esquemas de conocimiento
- de la *teoría sociocultural de Vygotski*, la importancia de la interacción social en el aprendizaje.

Para la teoría constructivista los conocimientos deben construirse y no reproducirse, de manera que los alumnos deben participar activamente en la construcción de las estructuras del conocimiento. Todo lo que se aprende depende del conocimiento previo y de cómo la nueva información es interpretada por el

alumno. Lo que somos capaces de aprender en un momento determinado depende tanto del nivel de competencia cognitiva como de los conocimientos que han podido construirse en el transcurso de las experiencias previas. Estos dos aspectos configuran los esquemas del conocimiento que el alumno aporta a la situación de aprendizaje y que le permitirán elaborar el nuevo contenido de aprendizaje.

La diferencia fundamental con los enfoques conductista y cognitivista radica en que el conocimiento ya no es una entidad verdadera o falsa, por tanto, la meta de la instrucción no termina cuando los alumnos adquieren el conocimiento sino que éste puede seguir creciendo. Son los propios alumnos los que desarrollan sus propias estrategias de aprendizaje y señalan sus objetivos y metas, al mismo tiempo que se responsabilizan de qué y cómo aprender mientras que la función del profesor es apoyar las decisiones del alumno. Estos diseños enfatizan la habilidad de los alumnos para crear interpretaciones por sí mismos y para manipular las cosas hasta que las conozcan. De esta manera, el alumno aprende a valorar sus propias habilidades y a aprender por sí mismo.

6.4.4. Consideraciones en la aplicación de las teorías de enseñanza y aprendizaje

A la hora de llevar a la práctica cualquiera de las teorías de enseñanza y aprendizaje debemos tener en cuenta que cualquiera de ellas resulta insuficiente para fundamentar todas las situaciones de aprendizaje, y que la utilización de una u otra dependerá de criterios tanto técnicos como personales.

Además la aplicación de cualquiera de ellas se encuentra condicionada a varios factores tales como:

- *Diseño inicial del software*
- *Contexto de utilización del software:*
 - *Aprendizaje individualizado o colaborativo*
 - *Aprendizaje en el aula o fuera de ella*
 - *Profesor activo o pasivo*
- *Papel del sujeto:*
 - *Activo o pasivo*

En la actualidad, las posibilidades del uso de la tecnología en la enseñanza son muy amplias, de ahí la importancia de continuar investigando en la aplicación del ordenador como medio didáctico.

6.5. Características esenciales del software educativo

El software educativo puede tratar diferentes materias de formas muy diversas (a partir de cuestionarios, facilitando una información estructurada al alumno, mediante la simulación de fenómenos,...) y ofrecer un entorno de trabajo

más o menos rico en posibilidades de interacción, pero siempre cuenta con las siguientes características:

- Se desarrolla con una *finalidad didáctica*.
- *Utiliza el ordenador* como soporte en el que los alumnos realizan las actividades propuestas.
- *Es interactivo*, permitiendo un intercambio de información entre el ordenador y los alumnos.
- *Individualiza el trabajo*, adaptándolo al ritmo de trabajo de cada alumno.
- *Resulta fácil de usar*, pues los conocimientos informáticos necesarios para utilizar la mayoría de los programas es mínima.

6.6. Estructura básica del software educativo

La mayoría de los programas de software educativo, igual que muchos de los programas informáticos generales, tienen tres módulos principales bien definidos:

- *La interfase* o módulo que gestiona la comunicación con el usuario (sistema de entradas y salidas).
- *La base de datos* o módulo que contiene debidamente organizados los contenidos informativos del programa.
- *El motor* o módulo que gestiona las actuaciones del ordenador y sus respuestas a las acciones de los usuarios.

6.7. Funciones de los programas educativos

El software educativo, según la forma en que sea aplicado, puede realizar funciones de muy diversa índole, tales como:

- *Función informativa*
- *Función instructiva*
- *Función motivadora*
- *Función evaluadora*
- *Función investigadora*
- *Función expresiva*

- *Función metalingüística*
- *Función lúdica*
- *Función innovadora*

6.8. Ventajas potenciales del software educativo

Los programas didácticos, bien utilizados, pueden aportar muchas ventajas a los procesos de enseñanza y aprendizaje. Se pueden destacar las siguientes:

- *Motivan al alumno.*
- *Liberan al profesor de trabajos repetitivos.*
- *Fomentan la continua actividad intelectual gracias a la interacción.*
- *Desarrollan la iniciativa.*
- *Permiten el aprendizaje a partir de los errores.*
- *Facilitan las actividades cooperativas.*
- *Alto grado de interdisciplinariedad.*
- *Permiten la individualización.*
- *Simulan fenómenos de difícil observación práctica.*

6.9. Inconvenientes potenciales del software educativo

Aunque presenta muchas ventajas, el software educativo, también tiene sus límites, y su uso puede comportar inconvenientes como:

- *Cansancio y monotonía por un uso excesivo.*
- *Necesidad de formalización previa de la materia a enseñar.*
- *Aprendizajes incompletos y superficiales.*
- *Desarrollo de estrategias de mínimo esfuerzo.*
- *Ansiedad.*
- *Aislamiento.*

6.10. Criterios de evaluación del software educativo

Para medir la calidad de un software educativo debemos someterlo a un proceso de evaluación. El objetivo buscado será determinar la facilidad de uso del sistema y su efectividad. Para ello, según la fase de desarrollo, establecemos dos tipos de evaluación:

- *Formativa*: durante el desarrollo del sistema.
- *Sumativa*: se realiza al finalizar el sistema.

En las que aplicamos métodos muy variados como:

- Observación
- Entrevistas
- Cuestionarios

Orientados a obtener información de tipo:

- *Cualitativa*
- *Cuantitativa*

Con la información obtenida, y teniendo en cuenta otros factores como el propósito del proyecto, sus limitaciones de tiempo y el costo, podremos hacer un estudio de la calidad del software educativo desarrollado.

6.11. La tecnología educativa en SIJEM

Para el desarrollo de SIJEM, concebida como software educativo, se optó por seguir el enfoque constructivista.

Se partió de la base de que los posibles usuarios de la herramienta serían alumnos que estudiaran materias relacionadas con la informática, con conocimientos previos del tema tratado, las jerarquías de memoria. Éste hecho simplificaba la aplicación, pues los alumnos ya poseían conocimientos de manejo de software y se sentirían motivados por probar nuevos programas.

Se trataba de proporcionar al alumno un medio adicional al libro escrito, aprovechando las posibilidades que brindan los lenguajes visuales, donde el alumno pudiera consolidar sus conocimientos a través de la interpretación de las acciones llevadas a cabo por el programa, y proporcionándole una base que le estimulara a realizar experimentos. De esta manera, serían los propios alumnos los que establecieran sus propias estrategias de aprendizaje, detectando sus puntos débiles y reforzando los conocimientos ya aprendidos.

A su vez, se trataba de proporcionar una guía útil al alumno a través de la ayuda, incluyendo en ella explicaciones de los conceptos incluidos.

Por último, no se quiso olvidar que el software educativo presenta ventajas e inconvenientes, y que estas deben de ser tenidas muy en cuenta, por ello se sometió a la herramienta a un profundo proceso de evaluación, como veremos en el siguiente capítulo.

Capítulo 7.

Evaluación de la herramienta

Una vez finalizado el simulador, y siguiendo las pautas propuestas por la Tecnología Educativa, era necesario realizar una serie de pruebas que demostraran la capacidad didáctica del mismo así como poner de manifiesto posibles errores que se hubieran pasado por alto durante la etapa de desarrollo.

7.1. Pruebas

Para la realización de las pruebas se escogieron alumnos de último año de la Ingeniería Superior en Informática, matriculados en la asignatura de Arquitectura de Computadores.

A estos alumnos se les realizó una presentación del programa en el aula, mostrando primero un repaso de los conceptos teóricos incluidos en el mismo y luego procediendo a mostrar las diferentes capacidades del programa utilizando los ejemplos contenidos en el mismo.

Tras la presentación, la copia del simulador así como un guión con una serie de instrucciones y posibles ejercicios (ANEXO A) les fueron entregados para que procedieran a usar la herramienta por ellos mismos.

Pasadas unas semanas se realizaron nuevas pruebas a estos alumnos:

- Una encuesta personal en la que se realizaban preguntas relacionadas con el perfil del alumno, el aspecto educativo, la funcionalidad y los aspectos técnicos del software así como otras cuestiones como comparaciones frente a programas similares y posibles errores, mejoras y sugerencias.
- Un test adaptativo por parejas basado en un conjunto de preguntas combinadas con ejercicios del simulador, para determinar el nivel de los alumnos y su capacidad de manejo del programa.
- Una encuesta para cada grupo en la que se realizaban de nuevo preguntas relacionadas con el perfil del alumno, aspecto educativo y funcionalidad, pero centrándose ésta vez en la combinación del test adaptativo y el simulador.

7.2. Encuesta personal

Esta encuesta fue entregada a los alumnos dos semanas después de que les fuera entregado el simulador.

SIJEM. Simulador didáctico de Jerarquías de memoria

Encuesta de validación para el alumnado

I. PERFIL DEL ALUMNO

1. Nombre y Apellidos:
2. Viene de:
☐ Gestión ☐ Sistemas ☐ Otro:
3. ¿Es la primera vez que cursa esta asignatura?
☐ Sí ☐ No
4. ¿Cuánto tiempo ha dedicado a probar este software?
☐ Nada ☐ Menos de 3 horas ☐ 3 - 6 horas ☐ Más de 6 horas
5. ¿Qué interés tiene para usted esta parte de la asignatura (Jerarquías de memoria) con respecto al resto de los temas vistos en clase?
☐ Más interesante ☐ Igual de interesante ☐ Menos interesante

II. ASPECTO EDUCATIVO DEL SOFTWARE

1. Este software ayuda a comprender los contenidos de la asignatura.
☐ Muy de acuerdo ☐ De acuerdo ☐ En desacuerdo ☐ Muy en desacuerdo
2. He aprendido usando este software.
☐ Muy de acuerdo ☐ De acuerdo ☐ En desacuerdo ☐ Muy en desacuerdo
3. Este software es de mi interés.
☐ Muy de acuerdo ☐ De acuerdo ☐ En desacuerdo ☐ Muy en desacuerdo
4. Este software me ha ayudado a comprender mejor las jerarquías de memoria.
☐ Muy de acuerdo ☐ De acuerdo ☐ En desacuerdo ☐ Muy en desacuerdo
5. La información suministrada en la ayuda sobre jerarquías de memoria es clara y concisa.
☐ Muy de acuerdo ☐ De acuerdo ☐ En desacuerdo ☐ Muy en desacuerdo

III. FUNCIONALIDAD DEL SOFTWARE

1. Las opciones de configuración son suficientes.
☐ Muy de acuerdo ☐ De acuerdo ☐ En desacuerdo ☐ Muy en desacuerdo
2. La configuración de los parámetros de la simulación es sencilla.
☐ Muy de acuerdo ☐ De acuerdo ☐ En desacuerdo ☐ Muy en desacuerdo
3. La traducción de direcciones se muestra con suficiente detalle y claridad.
☐ Muy de acuerdo ☐ De acuerdo ☐ En desacuerdo ☐ Muy en desacuerdo
4. La traducción de direcciones funciona correctamente.
☐ Muy de acuerdo ☐ De acuerdo ☐ En desacuerdo ☐ Muy en desacuerdo
5. La búsqueda de direcciones en la jerarquía de memoria se muestra con suficiente detalle y claridad.
☐ Muy de acuerdo ☐ De acuerdo ☐ En desacuerdo ☐ Muy en desacuerdo

6. La búsqueda de direcciones en la jerarquía de memoria funciona correctamente.
☐ Muy de acuerdo ☐ De acuerdo ☐ En desacuerdo ☐ Muy en desacuerdo
7. La realización de nuevos ficheros de traza ofrece muchas posibilidades y es lo suficientemente adaptable.
☐ Muy de acuerdo ☐ De acuerdo ☐ En desacuerdo ☐ Muy en desacuerdo

IV. ASPECTOS TÉCNICOS DEL SOFTWARE

1. El programa se cuelga a menudo.
☐ Muy de acuerdo ☐ De acuerdo ☐ En desacuerdo ☐ Muy en desacuerdo
2. El programa funciona bien en distintos PCs.
☐ Muy de acuerdo ☐ De acuerdo ☐ En desacuerdo ☐ Muy en desacuerdo
3. Los mensajes de error son claros.
☐ Muy de acuerdo ☐ De acuerdo ☐ En desacuerdo ☐ Muy en desacuerdo
4. Los mensajes de error ayudan a resolver el problema.
☐ Muy de acuerdo ☐ De acuerdo ☐ En desacuerdo ☐ Muy en desacuerdo
5. La interfaz es suficientemente clara.
☐ Muy de acuerdo ☐ De acuerdo ☐ En desacuerdo ☐ Muy en desacuerdo
6. El programa es fácil de utilizar.
☐ Muy de acuerdo ☐ De acuerdo ☐ En desacuerdo ☐ Muy en desacuerdo
7. La ayuda suministrada es suficiente.
☐ Muy de acuerdo ☐ De acuerdo ☐ En desacuerdo ☐ Muy en desacuerdo

V. OTRAS CUESTIONES

1. ¿Conoce algunos programas similares?

☐ Sí ☐ No

¿Cuáles?

2. Si conoce otros programas, valore este software en relación con ellos:

- Interfaz	☐ Mejor	☐ Igual	☐ Peor
- Ayuda	☐ Mejor	☐ Igual	☐ Peor
- Adecuación de los contenidos	☐ Mejor	☐ Igual	☐ Peor
- Interacción con el usuario	☐ Mejor	☐ Igual	☐ Peor
- Estabilidad	☐ Mejor	☐ Igual	☐ Peor
- Potencia	☐ Mejor	☐ Igual	☐ Peor

3. ¿Cree que la interfaz del programa podría mejorarse?

☐ Sí ☐ No

Indique las posibles mejoras que aplicaría al interfaz del programa

4. ¿Cree que el programa debería mejorarse con nuevas capacidades o cambiando alguna de las disponibles?

☐ Sí ☐ No

Indique los cambios que cree que harían más interesante o más útil este programa.

5. Si detectó algún fallo al ejecutar el programa le ruego que me lo indique de la

forma más detallada posible para poder corregirlo inmediatamente.

6. En este espacio escriba cualquier comentario adicional que crea que pueda aportar algo interesante al proyecto.

El objetivo de la encuesta era realizar una evaluación de la capacidad didáctica de la herramienta tras dejarla en manos de alumnos con conocimientos previos de la materia.

Además, al tratarse de alumnos de la carrera de Ingeniería Superior en Informática se daba la situación de que podían intuir el funcionamiento interno de la herramienta y comprender la problemática de la implementación de la simulación de ciertos mecanismos. De esta manera, podrían actuar como analistas que proporcionaran información muy valiosa para depurar los posibles errores de la herramienta.

7.2.1. Resultados de la encuesta

A continuación vamos a realizar un estudio de los resultados obtenidos con la encuesta personal.

7.2.1.1. Muestra

Los resultados se obtuvieron sobre un conjunto de 25 alumnos, de los que 14 procedían de la Ingeniería Técnica en Informática de Gestión y 11 de la Ingeniería Técnica en Informática de Sistemas.

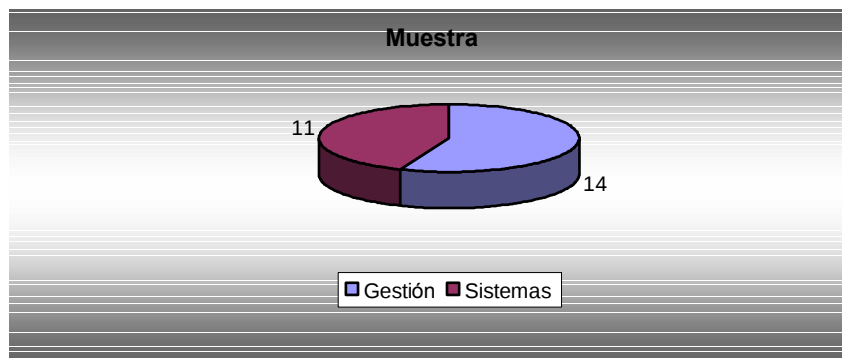


Gráfico 7.1. Muestra de la encuesta inicial

Resaltar que entre la muestra no se encontraba ningún alumno repetidor.

7.2.1.2. Dedicación al uso del programa

Los alumnos dedicaron una media de 4 horas al uso del programa, aunque una parte importante dedicó menos de 3 horas.



Gráfico 7.2. Resultados de ¿Cuánto tiempo ha dedicado a usar este software?

7.2.1.3. Interés por las jerarquías de memoria

Se quería conocer el interés que despertaba en los alumnos el tema de jerarquías de memoria, pues ello influiría también en el interés por la herramienta de simulación.

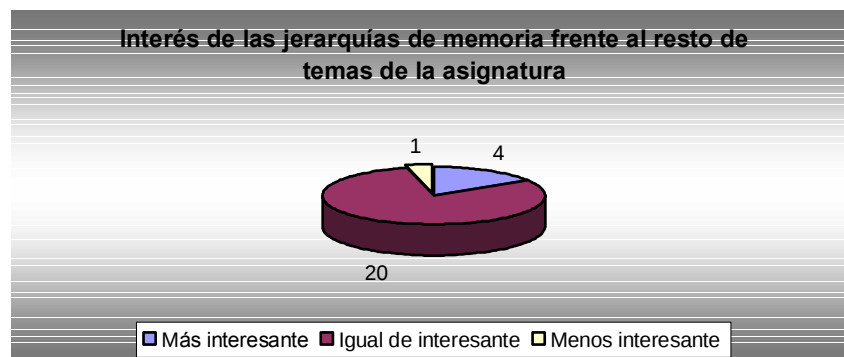


Gráfico 7.3. Interés de las jerarquías de memoria frente al resto de temas de la asignatura

De los resultados se desprende que el interés por el tema es alto, lo que proporciona una buena base para que el alumno utilice con motivación la herramienta y mejore su conocimiento.

7.2.1.4. Aspecto educativo del software

Una vez conocido el perfil de los alumnos sometidos al estudio, era necesario conocer si el software desempeñaba correctamente su función educativa. Para ello, se realizó a los alumnos una serie de preguntas sobre el interés y contenido de la herramienta.

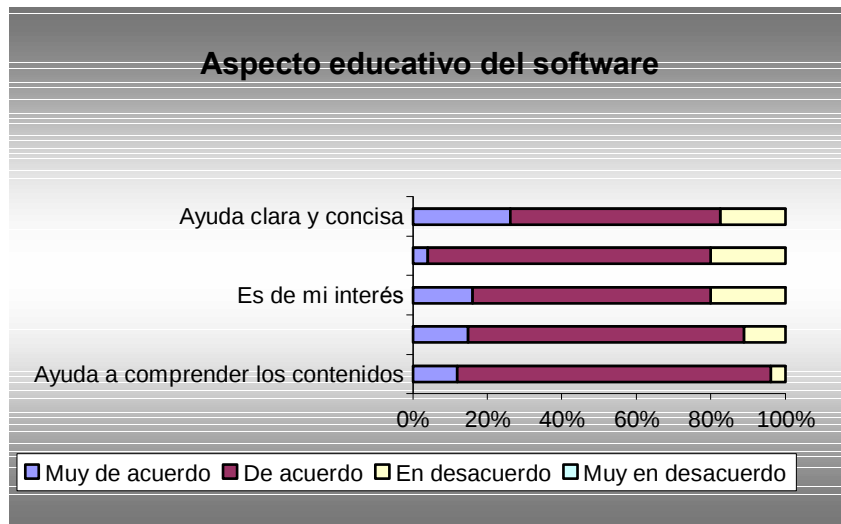


Gráfico 7.4. Aspecto educativo del software

Los resultados son muy positivos, pues la mayor parte de los alumnos muestra interés por el simulador y considera que ha aprendido con el mismo. Además destacan que ayuda a comprender mejor las jerarquías de memoria.

La ayuda se ha mostrado correcta, ayudando a los alumnos a resolver sus dudas en la mayoría de los casos. Sin embargo, como algunos alumnos resaltan, todavía tiene deficiencias que se han de resolver.

7.2.1.5. Funcionalidad del software

Una vez comprobado que el software cumplía su función educativa, se debía comprobar si las funcionalidades ofrecidas eran suficientes.

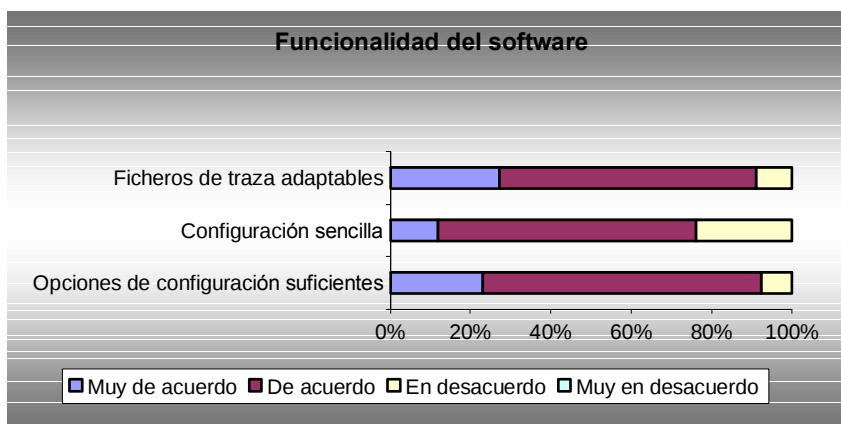


Gráfico 7.5. Funcionalidad del software

Los alumnos se mostraron de acuerdo en que la configuración era sencilla y las opciones de configuración suficientes. Además muchos de ellos manifestaron que las posibilidades de los ficheros de traza eran bastante buenas.

Uno de las características más importantes del simulador es su división entre traducción de direcciones y búsqueda de páginas, ya que en la encuesta se incluyeron preguntas para conocer la funcionalidad de la misma.

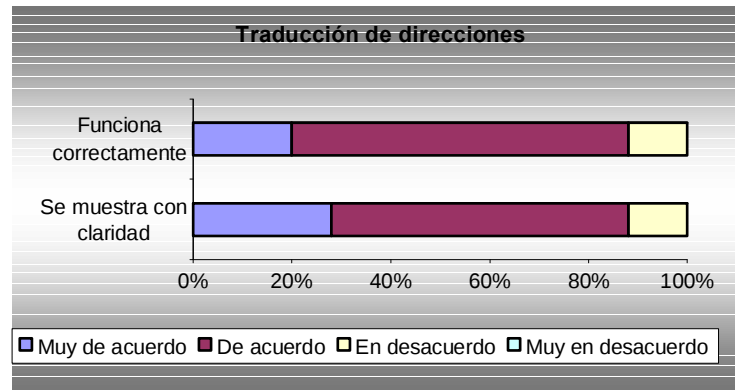


Gráfico 7.6. Traducción de direcciones

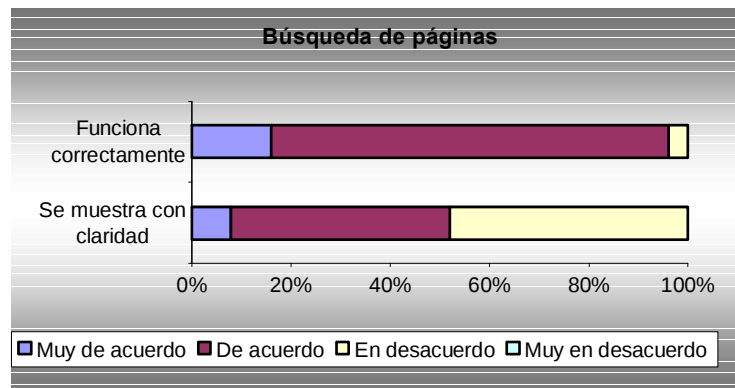


Gráfico 7.7. Búsqueda de páginas

Cómo se puede observar en los gráficos, mientras que la mayoría se muestra de acuerdo en que la traducción de direcciones se muestra con claridad, en el caso de la búsqueda de páginas, hay un porcentaje elevado de alumnos a los que les cuesta entender esta parte del simulador. Este hecho demuestra que el software presenta carencias en este sentido, por lo que, como se verá más adelante, tras el proceso de evaluación se llevaron a cabo modificaciones en la herramienta.

7.2.1.6. Funcionamiento del software

Uno de los factores que se debe comprobar a la hora de ejecutar cualquier software es si este funciona correctamente.

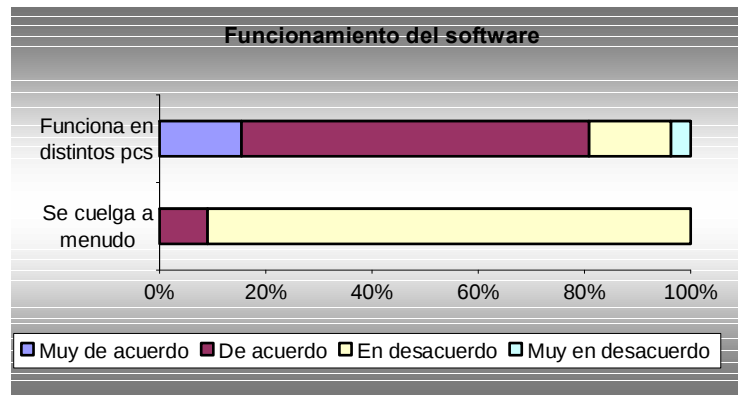


Gráfico 7.8. Funcionamiento del software

Cómo cualquier software, el simulador sufrió de problemas en sistemas concretos aunque funcionó correctamente en la gran mayoría de ellos. Sin embargo, como se puede comprobar funcionó correctamente sin quedarse colgado.

7.2.1.7. Mensajes de error

Una de las mayores fuentes de orientación para el usuario cuando se produce un error son los mensajes mostrados por pantalla. La aplicación contiene varios dentro de su código, y para comprobar su eficiencia, se incluyeron preguntas sobre ellos dentro de la encuesta.

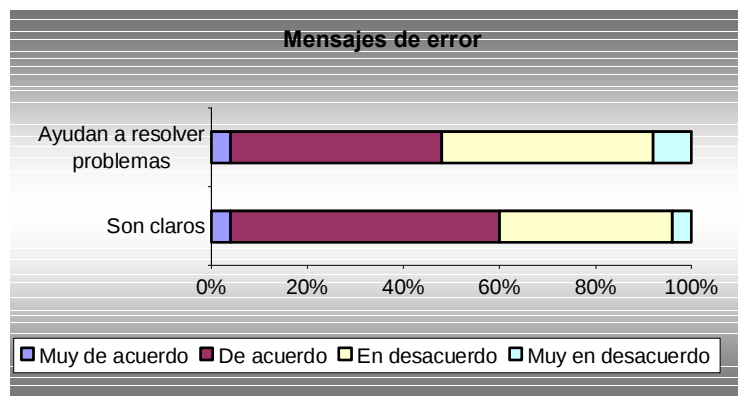


Gráfico 7.9. Funcionamiento del software

Del gráfico se extrae que los mensajes de error, aunque bastante claros, no son suficientes en algunos casos. En concreto, como se verá más adelante, hubo un error que se produjo en varios casos cuyo mensaje de error confundió a varios alumnos, por lo que también se realizaron cambios en este sentido.

7.2.1.8. Usabilidad del programa

Finalmente, se incluyeron preguntas sobre el uso del programa, para comprobar que su manejo no resultaba complicado para el alumno.

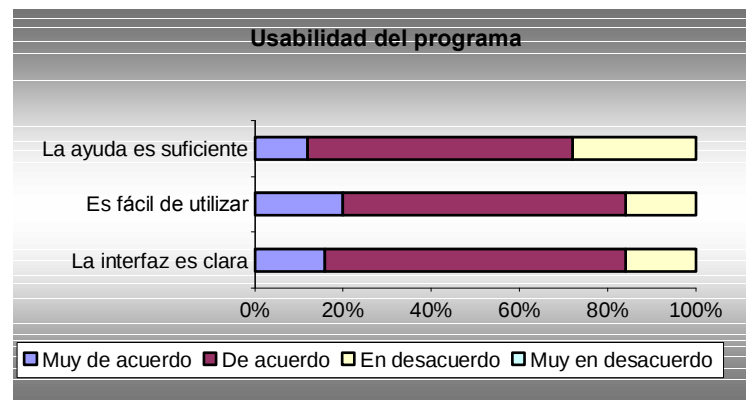


Gráfico 7.10. Usabilidad del programa

Los alumnos se mostraron de acuerdo en que el programa era fácil de utilizar, su interfaz clara, y la ayuda suficiente. Demostrando que las características de control que se han añadido, los accesos directos a la ayuda y los elementos gráficos de la interfaz han desempeñado correctamente su papel en la usabilidad del simulador.

7.2.1.9. Otras cuestiones

Dentro del simulador se incluía la pregunta de que si habían utilizado un software similar, pero la respuesta fue negativa en todos los casos. Sin embargo, dados los resultados positivos de SIJEM, el profesorado implicado intentará fomentar el uso de otras herramientas de simulación como las citadas en la introducción de esta memoria.

7.2.1.10. Críticas y posibles mejoras

Uno de los mayores valores de la encuesta fue la aportación escrita de los alumnos. En los últimos apartados, los alumnos se mostraron críticos con los fallos del programa y a su vez muy constructivos, proponiendo soluciones para los mismos.

A continuación se tratará de ofrecer un pequeño resumen de los aspectos destacados, las críticas y las posibles mejoras propuestas por los alumnos.

- Aspectos destacados
 - El asistente de configuración facilita el uso del programa.
 - Carga de ficheros fácil e intuitiva.
 - Posibilidad de volver atrás.
- Críticas
 - El uso de un fichero de traza equivocado puede provocar errores en el programa.
 - En la búsqueda de páginas, la simulación paso a paso realiza demasiadas acciones en un solo paso.

- Posibles mejoras
 - o Incluir un botón que muestre una lista de todas las acciones ocurridas en un paso.
 - o Posibilidad de grabar en un fichero la configuración realizada a mano con el asistente.
 - o Posibilidad de ver una gráfica del comportamiento de una estrategia, y comparaciones con otras estrategias.

7.2.2. Conclusiones

Cómo se puede deducir de los resultados de la encuesta, el simulador ha demostrado ser una herramienta muy apreciada por los alumnos, y que ha cumplido sobradamente su objetivo de complementar la enseñanza de las jerarquías de memoria.

Sin embargo, también deja patente que existen fallos que se deben de solventar para contar con un herramienta de alta calidad. Por ello, como se verá más adelante, tras la evaluación se introdujeron algunos cambios orientados a tratar se solucionar los problemas surgidos.

7.3. Test adaptativo

Aunque la encuesta proporcionaba información bastante relevante de la capacidad y características didácticas del simulador, era necesario realizar alguna prueba que incorporase la herramienta al aprendizaje en el aula.

De entre las muchas maneras posibles de hacerlo, se buscaba un método que fuera a su vez sencillo, eficaz y automatizado, permitiendo así obtener resultados fiables de un nutrido grupo de alumnos con poco esfuerzo. Además, debía proporcionar una manera de evaluar los conocimientos del alumno para determinar su nivel y medir la capacidad de ayuda que le proporcionaba el simulador.

La respuesta a esa búsqueda se encontró en los test adaptativos y el portal PORTAD desarrollado como proyecto de Fin de Carrera por los alumnos D.Zebenzui Pérez Ramos, D.Francisco Javier Medina Santos y D.Daniel Jacobo Díaz González bajo la dirección de Carina Soledad González González y que tiene como título ‘PORTAD y HEVAH: Interfaz Web y Personaje 3D para la evaluación de test adaptativos en XML’. Más información sobre el proyecto puede ser encontrada en la memoria del mismo [].

7.3.1. ¿Qué es un test adaptativo?

Un test adaptativo es un test en el cuál la presentación de cada elemento y la decisión de finalizar el test son dinámicamente adaptados según el rendimiento del estudiante. Si usamos los ordenadores para administrarlos tenemos test adaptativos informatizados, que poseen las siguientes características:

- Los tests pueden ser administrados individualmente, almacenando los resultados para, una vez tengamos la muestra completa, analizar los resultados, todo ello de forma rápida y cómoda.

- Los elementos que conforman el test son elegidos dinámicamente, a medida que el alumno lo va realizando, según su nivel de conocimientos. Esto significa que dos alumnos que estén realizando el mismo test no tienen por qué contestar a las mismas preguntas.
- El test no acaba por agotamiento de los elementos, sino que acaba en el momento en que el ordenador localiza el nivel de conocimiento del alumno que está siendo evaluado.

Los beneficios que aportan este tipo de tests respecto a los tests tradicionales son los siguientes:

- El test puede realizarse las veces que se desee.
- El tiempo que ocupa el test es típicamente más corto.
- El test es administrado y puntuado de manera automática, emitiendo un informe sobre la valoración global del alumno de manera casi inmediata.
- Las preguntas presentadas al alumno, corresponden al nivel de éste: ni demasiado fáciles, ni demasiado difíciles.

El algoritmo general que siguen los tests adaptativos es este:

- El ordenador establece el nivel inicial con el que empezarán los elementos del test en función del conocimiento estimado del alumno. Hay diversas formas para estimar este conocimiento inicial: aleatoriamente, a través de tests anteriores realizados por el alumno, mediante entrada directa (se le pregunta que nivel cree que tiene, o un profesor lo introduce...).
- Una vez llevada a cabo esta estimación inicial, se procede a intentar determinar el nivel exacto del estudiante, para lo cual el ordenador selecciona una pregunta del nivel adecuado entre los elementos del banco de preguntas.
- A medida que el alumno va dando respuestas, el ordenador va seleccionando la siguiente pregunta entre las existentes en el banco; el nivel de dificultad de la nueva pregunta irá dependiendo de las respuestas que ha dado hasta ese momento el alumno.
- Una vez el ordenador determina que ha estimado de manera relativamente exacta el nivel del alumno, informa de su valoración y da por concluido el test. Dentro de esta valoración pueden ir recomendaciones orientadas a la mejora del rendimiento por parte del estudiante, tales como determinación de puntos flojos, inconsistencias en el dominio de los conceptos....

Para que los tests adaptativos sean realmente funcionales, tenemos que tener en cuenta una serie de puntos básicos:

- Lo primero que se debe hacer es enseñar a utilizar el ordenador, ya que de otra manera, si el alumno no está familiarizado con los ordenadores, el test pierde bastante eficacia.
- A la hora de seleccionar la siguiente pregunta, el algoritmo implementado debe contemplar la posibilidad de que, en el caso de que la respuesta a la pregunta anterior sea correcta, el alumno la haya adivinado. Esto significa que el algoritmo no debe ser simplemente aumentar el nivel en caso de respuesta correcta y disminuirlo en el caso de respuesta errónea, sino que además se deben contemplar otros factores.
- Se debe seguir el orden que establezca el ordenador, es decir, no se permitirá pasar a una nueva pregunta sin haber contestado la actual, y no se podrá volver a una ya formulada.

7.3.2. Características del test adaptativo de PORTAD

El test adaptativo implementado dentro de PORTAD utiliza técnicas de inferencia bayesiana para realizar la evaluación del alumno de forma dinámica.

Así la red bayesiana se compone de test que está dividido en temas, a su vez estos temas están divididos en conceptos, éstos en preguntas y cada pregunta tiene asociada un número variable de respuestas.

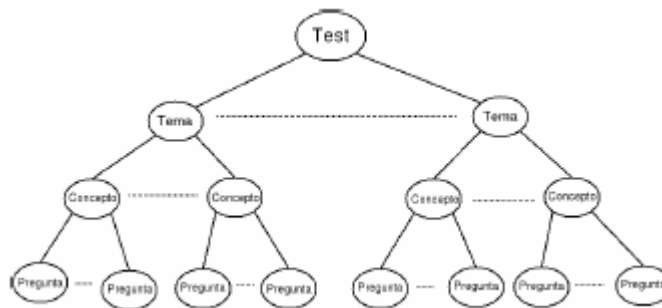


Figura 7.1. Red bayesiana

Para los conceptos distinguimos cuatro niveles de complejidad: simple, *intermedio*, *complejo* y *avanzado*. Así mismo, cada pregunta lleva asociada dos probabilidades, la probabilidad de saber la respuesta a dicha pregunta y la probabilidad de adivinarla. Convertimos estas probabilidades en conjuntos de inferencia bayesiana similares a los de los conceptos. Para ello sumamos ambas probabilidades obteniendo la probabilidad total de saber dicha pregunta. Suponiendo que dichas probabilidades son independientes entre sí, tendríamos una primera fórmula:

$$P(\text{acertar}) = P(\text{saber}) + P(\text{adivinar})$$

Agrupando el resultado distinguimos tres niveles de dificultad para las preguntas: *baja*, *media* y *alta*.

Con respecto al usuario también distinguimos distintos conjuntos bayesianos en función de sus conocimientos. Los conjuntos son: *novel*, *intermedio*, *avanzado* y *experto*. Este nivel varía a medida que el usuario va realizando los test. A los usuarios que no han realizado ningún test se les asigna un nivel por defecto de *novel*.

Para la evaluación en primer lugar se establecen las probabilidades iniciales, que determinan el conocimiento que tienen los distintos conjuntos de usuarios sobre los distintos conjuntos de dificultad de los conceptos, es decir, para cada par 'nivel de usuario'-'dificultad del concepto' tendremos una probabilidad inicial.

Una vez establecidas estas probabilidades pasamos a seleccionar el primer concepto del test que va a ser evaluado. Los conceptos se encuentran agrupados según su nivel de dificultad. El concepto que será presentado al usuario está en relación directa con su nivel de la siguiente forma:

<i>Nivel usuario</i>	<i>Nivel concepto</i>
Novel	Simple
Avanzado	Intermedio
Intermedio	Avanzado
Complejo	Complejo

Tabla 7.1. Niveles de usuario y concepto

De esta forma obtenemos el nivel del primer concepto. Del conjunto de conceptos de este nivel se elige uno al azar.

Para evaluar este concepto debemos saber, a priori, la probabilidad que tiene el usuario de saberlo. Esta es una de las probabilidades inicialmente establecidas.

A continuación debemos seleccionar el nivel de dificultad de la pregunta que se le va a presentar al usuario de las disponibles para ese concepto. Por defecto se establece dificultad 'Media'. De las preguntas de ese nivel, al igual que los conceptos, seleccionamos una al azar.

Durante la selección de preguntas puede suceder que un concepto no disponga de más preguntas del nivel requerido, en cuyo caso, se pasaría a un nuevo concepto de la misma dificultad que el anterior con el fin de conseguir una pregunta del nivel deseado.

Para valorar los resultados fijamos una serie de probabilidades a priori:

	<i>Simple</i>	<i>Intermedio</i>	<i>Avanzado</i>	<i>Complejo</i>
<i>Novel</i>	0 a 6	0 a 4	0 a 2	0 a 2
<i>Intermedio</i>	0 a 4	0 a 4	0 a 2	0 a 2
<i>Avanzado</i>	0 a 2	0 a 3	0 a 4	0 a 1
<i>Experto</i>	0	0 a 2	0 a 4	0 a 4

Tabla 7.2. Probabilidades a priori

Ahora debemos determinar cual es la puntuación para cada usuario, para lo que usamos el siguiente baremo:

<i>Dificultad pregunta</i>	<i>Fallos</i>	<i>Aciertos</i>	<i>Valoración</i>
<i>Baja</i>	X1	Y1	20%
<i>Media</i>	X2	Y2	30%
<i>Alta</i>	X3	Y3	50%

Tabla 7.3. Baremo de puntuación

Cómo se puede observar, las preguntas de dificultad baja supondrán el 20% de la nota, las de dificultad media el 30% y las de dificultad alta el 50% restante.

7.3.3. Test adaptativo para la evaluación de SIJEM

Las siguientes líneas muestran la implementación del test adaptativo que fue realizada para la evaluación del simulador SIJEM.

- Se asumió que todos los alumnos tenían la categoría de usuario novel.
- Se eligió un conjunto de 62 preguntas (ANEXO B) con diferentes niveles de dificultad: baja, media y alta.
- Cada pregunta estaba asociada a una de las divisiones de la aplicación (Traducción de direcciones o Búsqueda de páginas) y a un conjunto de fichero de configuración – fichero de traza, de manera que si el alumno no era capaz de contestar directamente la pregunta pudiera dirigirse al simulador para encontrar la respuesta a la misma.
- Cada pregunta tenía 3 posibles respuestas: A, B y C.
- Las preguntas se agruparon dentro de un tema, las jerarquías de memoria y 5 conceptos, de niveles simple, intermedio, avanzado y complejo.
- La red bayesiana resultante se muestra en la siguiente figura:

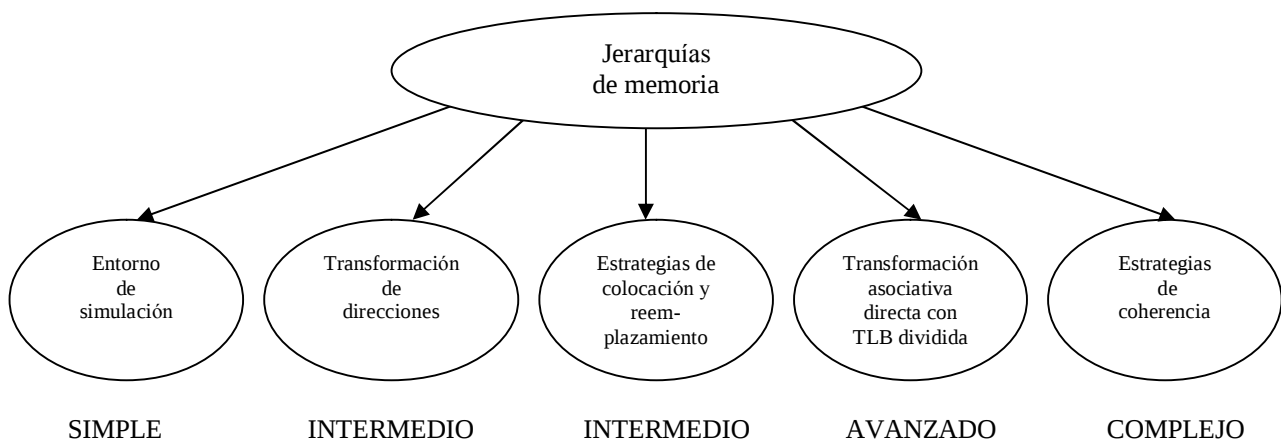


Figura 7.2. Red bayesiana del test adaptativo

7.3.4. Realización del test adaptativo

Para la realización del test adaptativo se reservó una hora en un aula del Centro de Cálculo de la ETSII y se agrupó a los alumnos de dos en dos, para fomentar ésta vez el trabajo cooperativo de la herramienta en lugar del individual.

Aunque el test proporcionaba una nota final, se optó por que esta no tuviera influencia sobre la nota global de la asignatura, pues se trataba de una prueba piloto en la que se quería validar el conjunto de dos proyectos: PORTAD y SIJEM. Y aunque no valorar el ejercicio podría derivar en un menor esfuerzo por parte del alumno, se quería evitar que los problemas que pudieran surgir influyeran sobre su evaluación académica.

Y lo cierto es que los problemas técnicos fueron bastante acusados. El uso simultáneo de múltiples puestos provocaba que, en ciertos momentos, PORTAD no fuera capaz de evaluar la respuesta a la pregunta y provocara errores que se mostraban en pantalla al alumno. Entonces éste tenía que volver atrás y contestar de nuevo la pregunta, que aunque en esta ocasión si fuera evaluada, influía sobre su motivación.

7.3.5. Resultados del test adaptativo

A continuación vamos a realizar una valoración de los resultados obtenidos con el test adaptativo.

7.3.5.1. Muestra

Los problemas técnicos también afectaron a los resultados, pues de los dieciséis grupos, la información de los test de tres de ellos no se guardó en el servidor. El estudio ha sido realizado por tanto con una muestra de 13 grupos.

7.3.5.2. Nota final

La nota media final que obtuvieron los alumnos fue de un 3,9. Éste es un dato a considerar, pues habrá que valorar si es que las preguntas han resultado difíciles o si los alumnos no habían practicado lo suficiente con la herramienta.

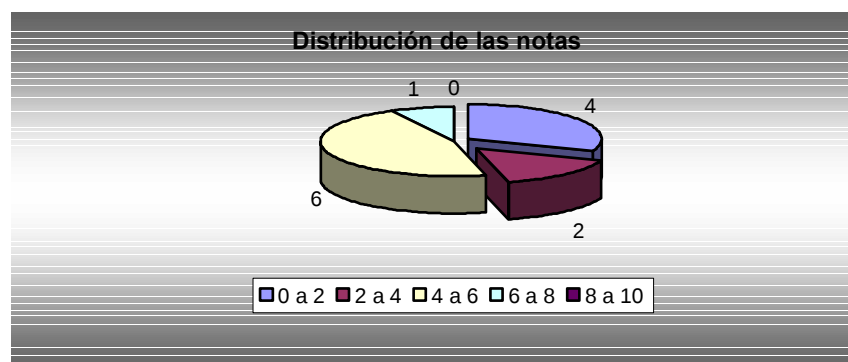


Gráfico 7.11. Distribución de las notas del test

Aunque si observamos el gráfico de distribución de las notas vemos que más de la mitad han superado el 4, ninguno ha superado el 8, y la máxima nota ha sido un 7,2. Esto puede considerarse positivo, teniendo en cuenta que los alumnos se enfrentaban por primera vez a un test adaptativo junto con una herramienta de simulación, y que el portal presentó numerosos fallos técnicos.

7.3.5.3. Número de preguntas

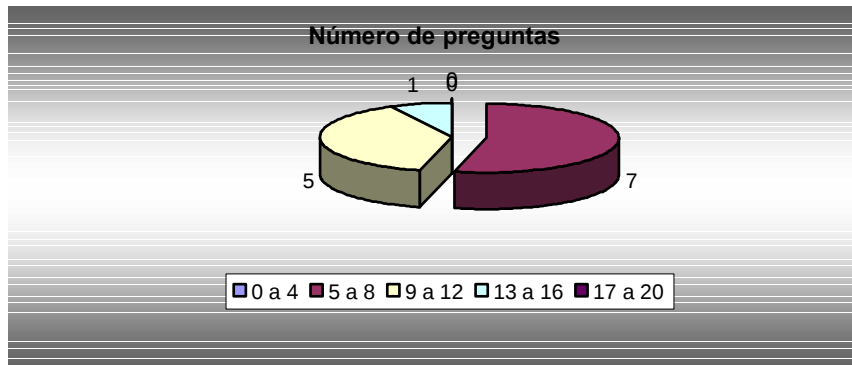
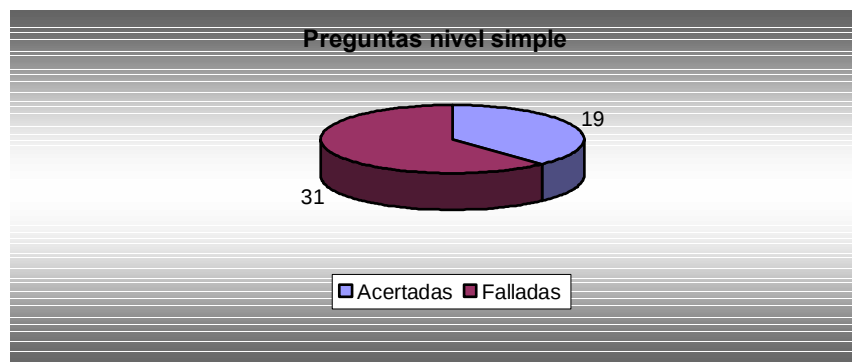


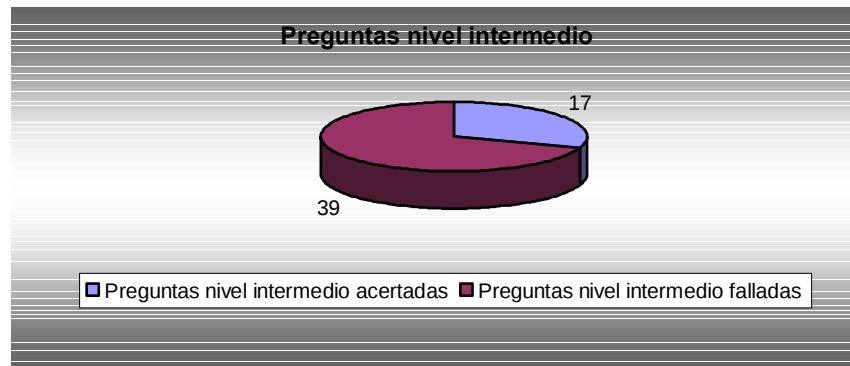
Gráfico 7.12. Número de preguntas del test

Uno de los factores más importantes de un test adaptativo de cara a evaluar al alumno es el número de preguntas formuladas, que se situó en una media de 10.

7.3.5.4. Aciertos y fallos

Habiendo observado que la nota media final de los alumnos es bastante baja, se debía comprobar cuál había sido el ratio de aciertos y fallos en las preguntas contestadas.





Gráficos 7.13 y 7.14. Aciertos y fallos de preguntas

Cómo se puede observar en los gráficos, un importante número de preguntas de nivel simple e intermedio fueron contestadas erróneamente por los alumnos.

7.3.6. Conclusiones

Los malos resultados obtenidos en la evaluación a través de PORTAD pueden llevar a pensar que la experiencia ha sido mala, sin embargo, nada más lejos de la realidad. Ha quedado demostrado que el uso de herramientas TIC en el aula no está exenta de problemas, aunque debe tratarse de evitarlos en la medida de lo posible, pues su influencia en la motivación del alumno es un factor muy importante. Es de esperar que una vez solventados los problemas los resultados mejoren.

7.4. Encuesta de grupo

A la finalización del test, se entregó a cada grupo la siguiente encuesta:

SIJEM. Simulador didáctico de Jerarquías de memoria ***PORTAD. Portal para la realización de test adaptativos***

Encuesta de validación para el alumnado

I. PERFIL DEL ALUMNO

1. Nombre y Apellidos:
2. Viene de:

☐ Gestión
 ☐ Sistemas
 ☐ Otro:
3. ¿Es la primera vez que cursa esta asignatura?

☐ Sí
 ☐ No

II. ASPECTO EDUCATIVO DEL TEST

1. El test ha despertado mi interés

- ☐ Muy de acuerdo ☐ De acuerdo ☐ En desacuerdo ☐ Muy en desacuerdo
 2. La información suministrada al inicio del test me ha parecido suficiente
☐ Muy de acuerdo ☐ De acuerdo ☐ En desacuerdo ☐ Muy en desacuerdo
 3. El test ayuda a comprender mejor el simulador
☐ Muy de acuerdo ☐ De acuerdo ☐ En desacuerdo ☐ Muy en desacuerdo
 4. El test ayuda a comprender mejor los contenidos de la asignatura
☐ Muy de acuerdo ☐ De acuerdo ☐ En desacuerdo ☐ Muy en desacuerdo

III. CONTENIDO DEL TEST

1. Las preguntas formuladas en el test son suficientemente claras
☐ Muy de acuerdo ☐ De acuerdo ☐ En desacuerdo ☐ Muy en desacuerdo
 2. Los ejemplos aportados junto a las preguntas me han ayudado a contestarlas
☐ Muy de acuerdo ☐ De acuerdo ☐ En desacuerdo ☐ Muy en desacuerdo

IV. FUNCIONALIDAD DEL TEST

8. La interfaz del portal es clara.
☐ Muy de acuerdo ☐ De acuerdo ☐ En desacuerdo ☐ Muy en desacuerdo
 9. La interacción con el portal es sencilla.
☐ Muy de acuerdo ☐ De acuerdo ☐ En desacuerdo ☐ Muy en desacuerdo
 10. La realización de un test con el portal es sencilla.
☐ Muy de acuerdo ☐ De acuerdo ☐ En desacuerdo ☐ Muy en desacuerdo

V. OTRAS CUESTIONES

1. Anote cualquier sugerencia que pueda resultar interesante para una próxima experiencia.

El objetivo de esta encuesta era realizar una evaluación de la capacidad didáctica del test resultante de combinar PORTAD y SIJEM, para poder así realizar un estudio del valor del simulador tanto sólo como ayudado por otro medio didáctico.

7.4.1. Resultados de la encuesta de grupo

A continuación vamos a realizar una valoración de los resultados obtenidos con la encuesta de grupo.

7.4.1.1. Muestra

La encuesta de grupo se realizó sobre 16 grupos de 2 personas. De estos grupos, 6 se componían únicamente de alumnos de gestión y 6 únicamente de alumnos de sistemas, mientras que los 4 grupos restantes estaban formados por un alumno de sistemas junto a otro de gestión.

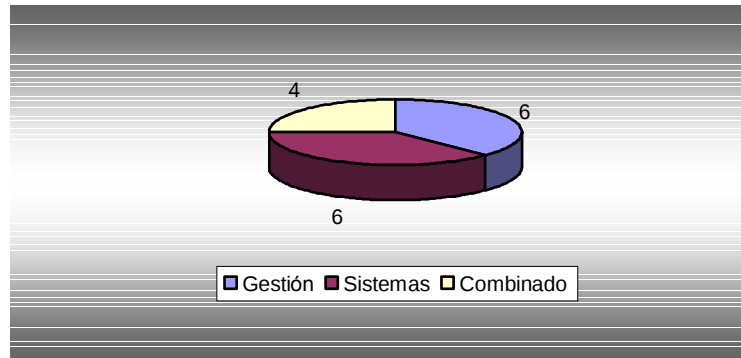


Gráfico 7.15. Muestra de la encuesta de grupo

Señalar también que entre la muestra no se encontraba ningún alumno repetidor.

7.4.1.2. Interés mostrado por el test

A la hora de valorar el resultado del test era importante conocer si realmente el test había generado interés entre los alumnos.

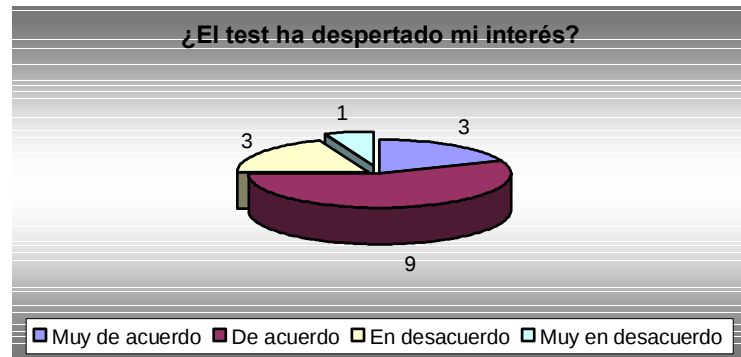


Gráfico 7.16. Resultados de ¿El test ha despertado mi interés?

De las respuestas a la pregunta “El test ha despertado mi interés” se desprende que en la mayoría de los casos los alumnos sentían curiosidad por conocer el portal y realizar el test. Ya que se puede considerar que el portal es una buena forma de ofrecer al alumno una forma atractiva de mejorar su conocimiento.

7.4.1.3. Aspecto educativo del test

También era importante averiguar si el test había realizado correctamente su función de enseñar, ya que se preguntó a los alumnos si consideraban haber aprendido con el test.

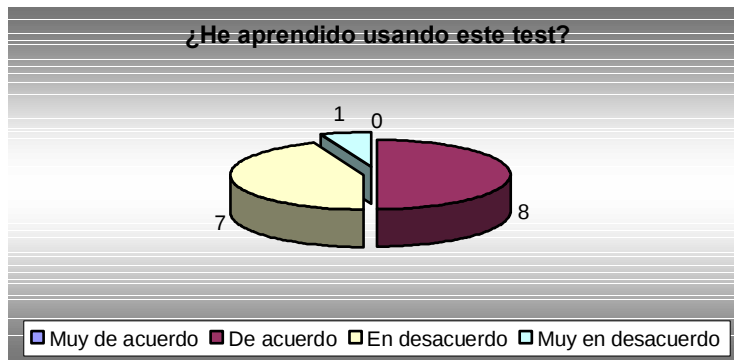


Gráfico 7.17. Resultados de ¿He aprendido usando este test?

Con las respuestas llegamos a la conclusión de que el test ha cumplido su función en la mitad de los casos, lo que es un dato positivo, pero lleva a preguntarse porqué no ha enseñado en el resto de los casos y deja patente que el test es mejorable.

También debemos tener en cuenta que los fallos técnicos ocurridos con el portal afectaron a la motivación del alumno y le impidieron centrarse plenamente en adquirir conocimientos, lo que puede haber influido en este resultado.

7.4.1.4. Información sobre el test

Al inicio del test se proporcionó al alumno información de manejo del test, de las características de un test adaptativo y de la forma de utilizar los ejemplos asociados a las preguntas para resolver el test. Por tanto, era necesario evaluar si esa información era suficiente.

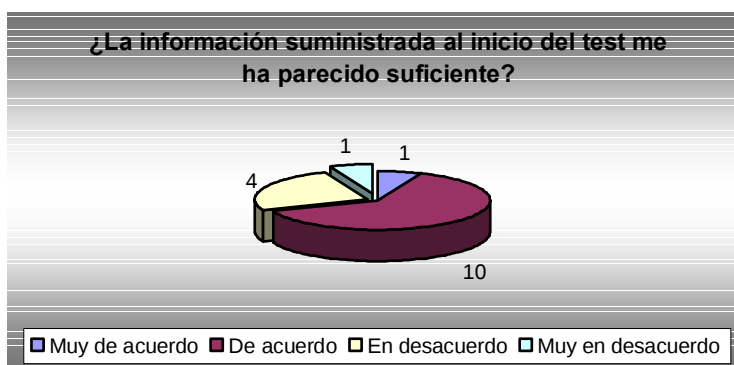


Gráfico 7.18. Resultados de ¿La información proporcionada al inicio del test me ha parecido suficiente?

En las respuestas a la pregunta observamos que la información fue suficiente para la mayoría de los casos, pero que también hubo alumnos que quedaron con dudas tras la misma. En el futuro se debe tratar de mejorar este apartado.

7.4.1.5. Comprensión del simulador y las jerarquías de memoria

Otra de las características importantes que se debía evaluar era si la combinación de PORTAD y SIJEM ayudada a comprender mejor tanto al simulador como a los contenidos de la asignatura tratados, las jerarquías de memoria.

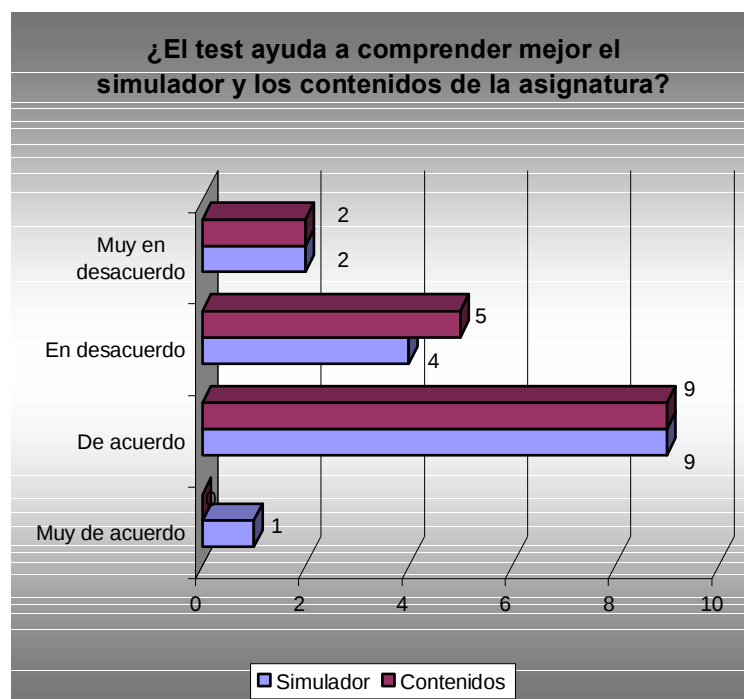


Gráfico 7.19. Resultados de ¿El test ayuda a comprender mejor el simulador? y ¿El test ayuda a comprender mejor los contenidos de la asignatura?

Los resultados obtenidos en este sentido son bastante buenos, pues la mayoría de alumnos han encontrado que el test les ha proporcionado nuevos conocimientos tanto de SIJEM como de las jerarquías de memoria.

Debemos destacar también que en el test los alumnos trabajaban por parejas, por lo que un miembro del grupo podía traspasar un conocimiento al compañero, resolver alguna de sus dudas o percatarse de errores en su aprendizaje pasado, aumentando así el conocimiento de ambos.

Del análisis de todos los resultados de esta sección podemos concluir que la experiencia de combinar de PORTAD y SIJEM ha resultado muy positiva, lo que demuestra que el uso de dos herramientas didácticas trabajando juntas puede proporcionar mayores ventajas que las de una sola herramienta y que el trabajo cooperativo con las mismas aporta conocimientos extra al alumno.

7.4.1.6. Preguntas

Una vez evaluados los aspectos educativos, se debió considerar también el contenido de las preguntas, pues constituyen la base de cualquier test.

Por ello se preguntó al alumno “¿ Las preguntas formuladas son suficientemente claras?”

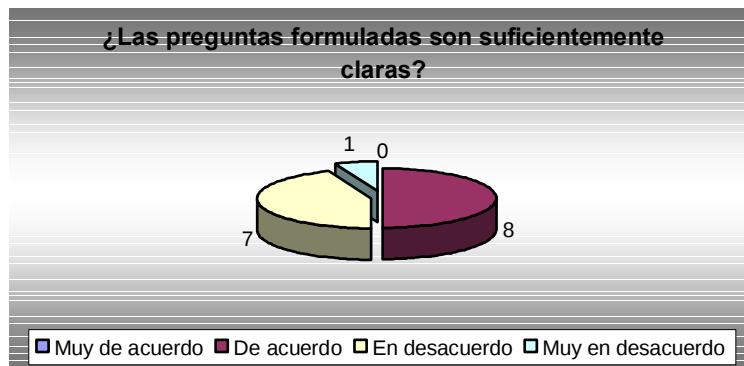


Gráfico 7.20. Resultados de ¿Las preguntas formuladas son suficientemente claras?

Cómo se puede observar en el gráfico, hubo bastantes alumnos que consideraron que existían preguntas que no estaban suficientemente claras. En todo test la claridad de las preguntas es una cuestión que debe ser estudiada con esmero, pues puede afectar mucho a la efectividad de la prueba. Sin embargo, debemos tener en cuenta que las preguntas son elaboradas a mano por un profesor o especialista, y que por tanto son susceptibles de contener errores gramaticales que lleven a dobles interpretaciones. Además, en ocasiones, la falta de conocimiento de un alumno puede también llevarle a confusión en la lectura de la preguntas.

Cada una de las preguntas llevaba asociado un ejercicio dentro del simulador que ayudaba a descubrir la respuesta, se quería así conseguir que ambas herramientas trabajaran juntas para orientar al alumno en su proceso de aprendizaje. Dentro de la encuesta se pregunto a los alumnos “¿Los ejemplos aportados junto a las preguntas han ayudado a contestarlas?”



Gráfico 7.21. Resultados de ¿Los ejemplos aportados junto a las preguntas me han ayudado a contestarlas?

Observando el gráfico vemos que el objetivo buscado con los ejemplos asociados a las preguntas se cumplió con creces, existiendo tan sólo un grupo que

consideró que no le ayudaron. Por tanto, se demuestra que puede resultar un método muy efectivo para combinar los test con cualquier tipo de software educativo.

7.4.1.7. Funcionalidad del test

Aunque el objetivo buscado era la mejora del conocimiento de las jerarquías de memoria por parte del alumno, no podíamos obviar que en el aprendizaje con herramientas didácticas la funcionalidad de la herramienta es un factor determinante, que influye en la motivación y la capacidad de aprender del alumno.

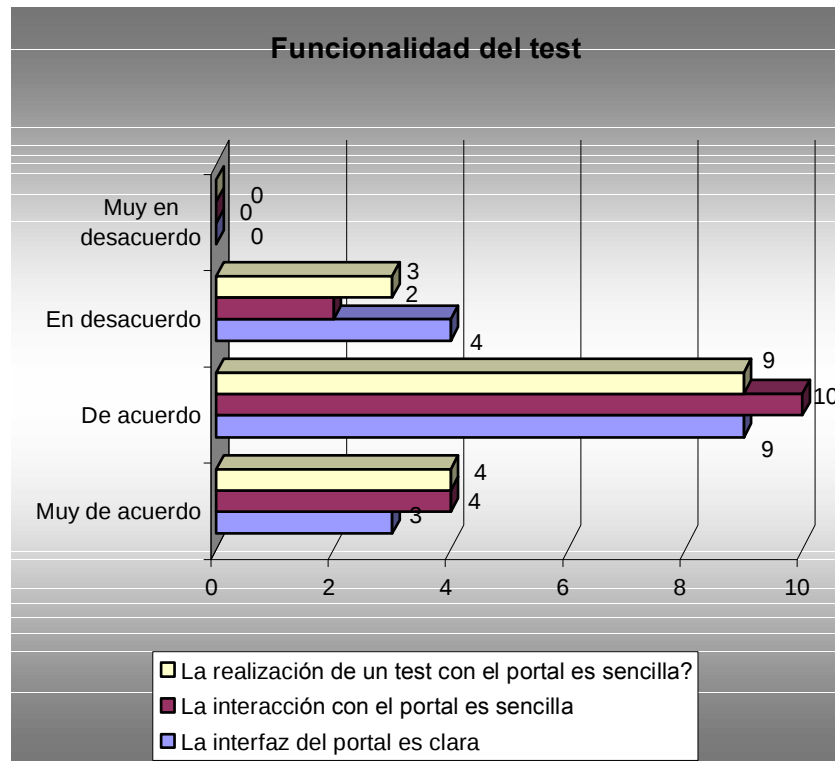


Gráfico 7.22. Resultados de Funcionalidad del test

Del gráfico se desprende que la valoración general del portal es muy positiva, resultando su interfaz y manejo simples, constituyendo una herramienta muy útil para futuros proyectos.

7.4.1.8. Otras cuestiones

La última de las preguntas del test era un espacio abierto para que los alumnos pudieran mostrar sugerencias y comentar su experiencia de la prueba.

La principal crítica se dirigió hacia los fallos técnicos que ocurrieron durante el test, que en ocasiones provocó que algunos alumnos tuvieran que responder algunas preguntas de nuevo, dejando constancia de ello en la encuesta.

Otro de los factores que influyó en la percepción de los alumnos del test es que, en ocasiones, el número de preguntas que contestaron fue pequeño; por una parte debido a que el banco de preguntas no es lo suficientemente grande para

algunos casos y por otra debido a que el algoritmo era capaz de evaluar bastante rápido el nivel de un alumno. Esto provocaba que el alumno se encontraba con una nota final que consideraba que no tenía suficientes argumentos para ser emitida.

La sugerencia más importante que aportaron los alumnos fue el conocer la respuesta correcta después de emitir la suya, pues consideraban que eso les ayudaría a mejorar sus conocimientos.

7.5. Modificaciones

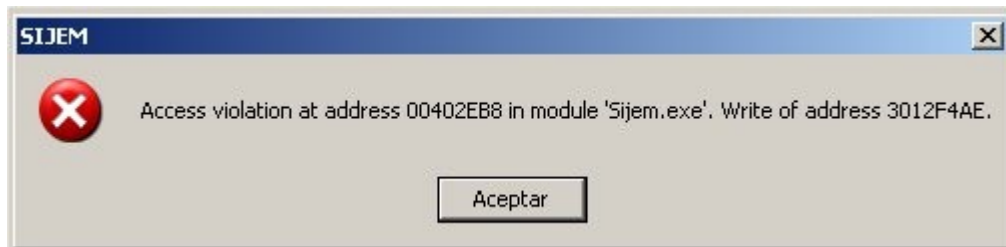
Durante el proceso de evaluación se detectaron dos errores importantes:

- La carga de un fichero de traza con un formato de direcciones no adecuado provocaba un error fatal en el programa, y el mensaje de error que se producía no aclaraba la situación.
- En la búsqueda de páginas, la simulación paso a paso realiza demasiadas acciones en un solo paso.

Para solucionarlos se llevaron a cabo dos modificaciones importantes en el simulador.

7.5.1. Ficheros de traza

Frente al antiguo error que se producía en el programa:



Captura 7.1. Error inicial del programa

Se incluyó una rutina de gestión de las excepciones para controlar el error y ahora, cuando se produce, se muestra este otro mensaje:

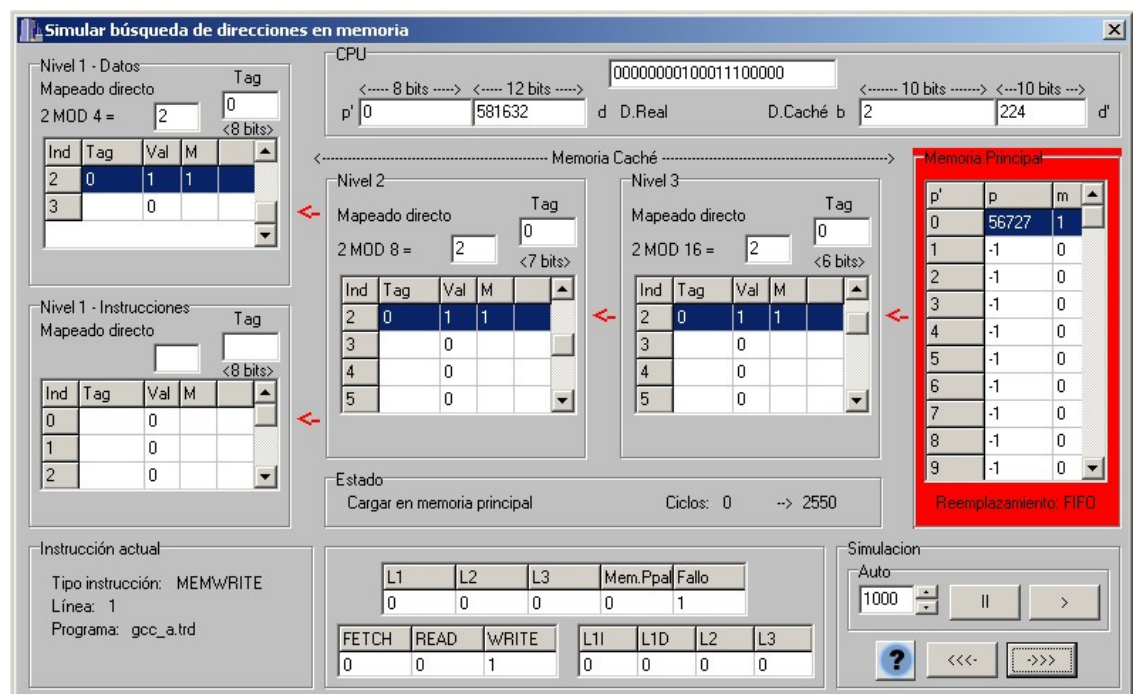


Captura 7.2. Error final del programa

Con este mensaje se aclara el problema al alumno, y se le indica que puede dirigirse a la sección Ficheros de la ayuda para encontrar información sobre la carga correcta de ficheros.

7.5.2. Búsqueda de direcciones

En la búsqueda de direcciones los alumnos destacaron que la simulación paso a paso realizaba demasiadas acciones en un solo paso. Aprovechando una de las posibles soluciones aportadas por un alumno, la adición de colores, se modificó la simulación para que destacara en verde el acierto en un nivel de caché o memoria principal, y en rojo un fallo en memoria principal. De esta manera se destaca al alumno donde debe fijar su atención.



Captura 7.3. Búsqueda de direcciones

Capítulo 8.

Conclusiones

Han sido muchas las horas dedicadas a la realización del simulador, a la codificación de sus más de 648000 líneas, al estudio de la implementación de los diferentes conceptos de las jerarquías de memoria y a la forma de integrarlos en un entorno usable y eficiente, a la depuración de posibles errores y a la realización de pruebas.

8.1. Tecnología educativa

La utilización de las pautas propuestas para el desarrollo de software educativo por este campo de estudio aportó muchas ideas nuevas al proyecto, y propició que pasara de ser un simple simulador a convertirse en una auténtica herramienta para el apoyo docente.

8.2. Proceso de evaluación

Tras el profundo proceso de evaluación al que fue sometida la herramienta contamos con un simulador funcional y robusto que desempeña su función educativa de forma eficiente. Ello no hace sino destacar la importancia de realizar pruebas para lograr un software educativo de calidad.

8.2.1. Participación de los alumnos

Uno de los aspectos importantes del proyecto fue la participación de los alumnos en el proceso de evaluación. Los alumnos, lejos de quedarse en meros usuarios del simulador, detallaron con precisión errores de funcionamiento, destacaron aquellas cualidades que consideraban efectivas e incluso propusieron múltiples mejoras.

8.2.2. Integración con otros proyectos

Una de las grandes novedades aportadas en el proyecto fue la combinación de dos herramientas didácticas para la enseñanza en el aula. El experimento realizado al combinar PORTAD con SIJEM resultó ser de gran utilidad para demostrar que los resultados de dos grandes proyectos pueden trabajar juntos para convertirse en otro

mayor. Los alumnos se sintieron muy motivados para trabajar con ambas herramientas y manifestaron haber mejorado sus conocimientos después de hacerlo.

8.3. Líneas abiertas

Aunque el resultado final puede considerarse muy bueno, existen muchas posibles mejoras y ampliaciones que se pueden aplicar a futuras versiones de la herramienta, siendo gran parte de ellas una aportación de los propios alumnos.

De las encuestas realizadas se extrajo que el simulador debería mejorar en los siguientes aspectos:

- Posibilidad de *visualizar gráficas* que permitan comparar el comportamiento de diferentes estrategias.
- Posibilidad de *grabar en un fichero* la configuración realizada a mano con el asistente.

De cara a ampliar las características del simulador, durante el desarrollo del proyecto se dejaron a un lado la implementación de:

- *Segmentación*, su mayor complejidad frente a la paginación hizo que se desechara frente a la paginación.
- *Multiprogramación*, de cara a la enseñanza de sistemas operativos.

ANEXO A.

Guión de prácticas con el simulador

Para introducir a los alumnos en el uso del simulador, se entregó a los alumnos un guión con instrucciones y posibles ejercicios con el mismo. Esta información se muestra a continuación en este anexo.

SIJEM. Simulador didáctico de Jerarquías de memoria

En el ftp de la asignatura se encuentra el fichero Sijem.zip que incluye:

- Carpeta INSTALL -Instalador para windows de la aplicación y de su archivo de ayuda.

Los ficheros de configuración de traza y ejemplo se incluyen dentro de la carpeta <Directorio de instalación del programa>\ejemplos.

El archivo de ayuda se encuentra en <Directorio de instalación del programa>\sijem.hlp

- Encuesta.doc – Fichero de encuesta.

Es posible responderlo sobre el mismo fichero .doc usando la opción “Proteger formulario” de la barra de herramientas “Formularios” de MS Word.

- Leeme.pdf – Este mismo fichero.

El alumno deberá realizar pruebas del simulador usando los ficheros incluidos, o bien alguno propio, y responder las preguntas formuladas en la encuesta.

En las próximas semanas se llevará a cabo un test adaptativo informatizado en las salas del centro de cálculo de la ETSII a través del portal PORTAD.

El test adaptativo consistirá en una batería de 15 a 20 preguntas, de las cuales algunas serán de respuesta inmediata y otras requerirán del uso del simulador, (siempre con los ejemplos incluidos) y cuyo nivel de dificultad se adaptará al nivel

mostrado por el usuario, emitiendo al final una valoración de los conocimientos del mismo.

El objetivo de la encuesta y el test adaptativo es realizar una valoración de la capacidad didáctica de la aplicación SIJEM así como servir para que el propio alumno valore sus conocimientos.

La encuesta y el test adaptativo no influyen en la nota global de la asignatura pero si es necesaria su realización.

Cualquier sugerencia, consulta o aclaración puede ser enviada por correo (Asunto: Consulta SIJEM) y se tratará de responder a la mayor brevedad posible.

A continuación se muestran una serie de posibles ejercicios para realizar prácticas con el simulador:

Traducción de direcciones

1. Probar los ejemplos Ejem1.cfg (Transformación directa), Ejem2.cfg (Transformación asociativa-directa con TLB) y Ejem3.cfg (Transformación asociativa-directa con TLB dividida) y usando algunos de los ficheros de traza de ejemplo acabados en d (20bits de direcciones – 1024Kb memoria) observar las diferencias (funcionamiento, tasa de aciertos/fallos, nº ciclos, etc...) entre los diferentes mecanismos de traducción de direcciones.

Búsqueda de direcciones en la jerarquía de memoria

2. Probar los ejemplos Ejem4.cfg, Ejem5.cfg y Ejem11.cfg con algunos de los ficheros de traza acabados en a (32bits de direcciones – 4GB memoria) y observar las diferencias entre los diferentes algoritmos de colocación.
3. Probar los ejemplos Ejem5.cfg, Ejem6.cfg, Ejem7.cfg, Ejem8.cfg, Ejem9.cfg y Ejem10.cfg con algunos de los ficheros de traza acabados en a (32bits de direcciones – 4GB memoria) y observar las diferencias entre los diferentes algoritmos de reemplazamiento.

ANEXO B.

Preguntas del test adaptativo

Para la realización de la prueba con el test adaptativo se elaboraron una serie de preguntas que se muestran a continuación.

A.1. Concepto 1 – Nivel simple – Entorno de simulación

A.1.1. Preguntas de nivel Bajo

- *Páginas*

1. (Ejem1.cfg y crafty_d.trd) Si el tamaño de la memoria principal es de 64KB y el tamaño de página es de 4KB, ¿Cuántas páginas puede almacenar la memoria principal?
 - a. 8
 - b. 16
 - c. 32

Solución b: $64/4 = 16$

NIVEL BAJO

- *Tabla de páginas*

2. (Ejem1.cfg y crafty_d.trd) Teniendo en cuenta que el tamaño de la memoria secundaria/virtual es de 1024KB, el tamaño de la memoria principal de 64KB y el tamaño de página es de 4KB. ¿Cuál es el número de entradas de la tabla de páginas?
 - a. 256
 - b. 1024
 - c. 128

Solución a: $1024/4 = 256$

NIVEL BAJO

- **Páginas**

3. (Ejem4.cfg y crafty_d.trd) Teniendo en cuenta que el tamaño de la memoria principal es de 1024KB y el tamaño de página es de 4KB. ¿De cuantas páginas se compone la memoria principal?
- a. 64
 - b. 128
 - c. 256

Solución c: $1024/4 = 256$

NIVEL BAJO

- **Memorias caché**

4. (Ejem4.cfg y crafty_d.trd) Si el tamaño de la memoria caché de nivel 3 es de 256KB y el tamaño de bloque es de 1KB, ¿Cuántas bloques puede almacenar la caché de nivel 3?
- a. 32
 - b. 64
 - c. 256

Solución c: $256/1 = 256$

NIVEL BAJO

5. (Ejem4.cfg y crafty_d.trd) Si el tamaño de la memoria caché de nivel 2 es de 64KB y el tamaño de bloque es de 1KB, ¿Cuántos bloques puede almacenar la caché de nivel 2?
- a. 16
 - b. 32
 - c. 64

Solución a: $64/1 = 64$

NIVEL BAJO

A.1.2. Preguntas de nivel intermedio

- **Dirección virtual y real**

6. (Ejem1.cfg y crafty_d.trd) Sabiendo que el tamaño de página es de 4KB, ¿Cuántos bits hacen falta para representar el desplazamiento dentro de la página?
- a. 4
 - b. 8
 - c. 12

Solución c: $2^{\exp 12} = 4096$

NIVEL INTERMEDIO

- **Dirección virtual**

7. (Ejem1.cfg y crafty_d.trd) Si el tamaño de la memoria virtual es de 1024KB y el tamaño de página es de 4KB, ¿De cuantos bits se compone la dirección virtual?
- a. 4
 - b. 16
 - c. 20

Solución c: $2^{\exp 20} = 1048576 - 1024kb$

NIVEL INTERMEDIO

8. (Ejem1.cfg y crafty_d.trd) ¿Cuántos bits hacen falta para direccionar una página dentro de la tabla de páginas? (Parte p de la dirección virtual)
- a. 2
 - b. 4
 - c. 8

Solución c: $2^{\exp 8} = 256$ páginas

NIVEL INTERMEDIO

- **Dirección real**

9. (Ejem1.cfg y crafty_d.trd) Si el tamaño de la memoria principal es de 64KB y el tamaño de página es de 4KB, ¿De cuantos bits se compone la dirección real?
- a. 4
 - b. 16
 - c. 20

Solución b: $2^{\exp 16} = 65536 - 64Kb$

NIVEL INTERMEDIO

10. (Ejem1.cfg y crafty_d.trd) ¿Cuántos bits hacen falta para direccionar una página dentro de memoria principal? (Parte p' de la dirección real)
- a. 2
 - b. 4
 - c. 8

Solución b: $2^{\exp 4} = 16$ páginas

NIVEL INTERMEDIO

• *Memorias caché*

11. (Ejem4.cfg y crafty_a.trd) Si el tamaño de la memoria caché de nivel 1 es de 16KB y el tamaño de bloque es de 1KB, y teniendo en cuenta que está dividida en instrucciones y datos, ¿Cuántos bloques puede almacenar cada una de las divisiones de la caché de nivel 1?
- 4
 - 8
 - 16

Solución a: $16/1 = 16$

NIVEL INTERMEDIO

12. (Ejem4.cfg y crafty_a.trd) Si el tamaño del bloque de memoria caché es de 1024Bytes ¿Cuántos bits son necesarios para representar el desplazamiento dentro del bloque?
- 4
 - 8
 - 10

Solución c: $2^{10} = 1024$

NIVEL INTERMEDIO

A.1.3. Preguntas de nivel alto• *Traducción de direcciones*

13. (TD - Ejem1.cfg y crafty_d.trd) El bit r de la tabla de páginas indica:
- Si la página se encuentra cargada en la memoria principal.
 - Si la página se encuentra cargada en la memoria secundaria.
 - Que la página ha sido reemplazada en memoria principal.

Solución a

NIVEL ALTO

14. (TD - Ejem1.cfg y crafty_d.trd) La tabla de páginas tiene tantas entradas como:
- Páginas reales.
 - Páginas virtuales.
 - Entradas tiene la TLB.

Solución a

NIVEL ALTO

• ***Búsqueda de páginas***

15. (BD – Ejem4.cfg y crafty_a.trd) ¿En qué campos se divide la parte correspondiente a bloque (b) de la dirección de memoria caché?
- Índice (Ind) y desplazamiento (d).
 - Índice (Ind) y página virtual (p).
 - Índice (Ind) y Etiqueta (Tag).

Solución c

NIVEL ALTO

16. (BD – Ejem4.cfg y crafty_a.trd) ¿Qué ventajas aporta que la memoria caché de nivel 1 esté dividida en dos?
- No aporta ventaja, sólo permite ordenar mejor las páginas.
 - Aumenta el tamaño de la caché de nivel 1.
 - Aprovecha mejor el principio de localidad.

Solución c

NIVEL ALTO

17. (BD – Ejem4.cfg y crafty_a.trd) Ejecuta 2 o 3 pasos de la simulación y di cual de las siguientes afirmaciones es correcta:
- Las cachés de nivel 2 y de nivel 3 son inclusivas.
 - Todas las cachés son inclusivas.
 - Las cachés de nivel 1 y de nivel 2 son inclusivas.

Solución a

NIVEL ALTO

18. (BD – Ejem4.cfg y crafty_a.trd) ¿Aumenta la caché la capacidad de direccionamiento de la memoria principal?
- Si
 - No
 - No tiene porqué.

Solución b

NIVEL ALTO

A.2. Concepto 2 – Nivel intermedio – Traducción de direcciones

A.2.1. Preguntas de nivel bajo

- **Funciones de la TLB**

19. (TD – Ejem2.cfg y crafty_d.trd) ¿Cuál es la función de la TLB?
- a. Acelerar la traducción almacenando las últimas direcciones traducidas.
 - b. Aumentar la capacidad de almacenamiento de la tabla de páginas.
 - c. Aumentar la capacidad de direccionamiento.

Solución a

NIVEL BAJO

20. (TD – Ejem2.cfg y crafty_d.trd) El número de entradas de la TLB es:
- a. Mayor que la tabla de páginas.
 - b. Menor o igual al de la tabla de páginas.
 - c. Menor que la tabla de páginas.

Solución b

NIVEL BAJO

A.2.2. Preguntas de nivel medio

21. (TD – Ejem1.cfg y crafty_d.trd) Ciclos. Ejecuta algunas líneas del programa y contesta. ¿Cuántos ciclos transcurren cuando hay que traer una página desde memoria secundaria hasta memoria principal?
- a. 12500. Consultar tabla de páginas (1000), leer desde memoria secundaria (10000), enviar a través del bus (1500).
 - b. 13500. Consultar tabla de páginas (1000), leer desde memoria secundaria (10000), enviar a través del bus (1500) y escribir en memoria principal (1000).
 - c. 14500. Consultar tabla de páginas (1000), leer desde memoria secundaria (10000), enviar a través del bus (1500), escribir en memoria principal (1000) y volver a consultar la tabla de páginas (1000).

Solución b

NIVEL MEDIO

22. (TD – Ejem1.cfg y crafty_d.trd) Ciclos. Ejecuta algunas líneas del programa y contesta. ¿Cuántos ciclos transcurren cuando la página ya se encuentra en memoria principal?
- a. 1000. Consultar la tabla de páginas (1000).
 - b. 11000. Consultar la tabla de páginas (1000) y leer desde memoria secundaria (10000).

- c. 13500. Consultar tabla de páginas (1000), leer desde memoria secundaria (10000), enviar a través del bus (1500) y escribir en memoria principal (1000).

Solución a

NIVEL MEDIO

A.2.3. Preguntas de nivel alto

- 23. Simula las configuraciones de traducción de direcciones Ejem1.cfg y Ejem2.cfg con el fichero de traza crafty_d.trd y contesta. El número de ciclos usando la TLB
 - a. Ha disminuido a la mitad.
 - b. Ha disminuido a menos de la mitad de tiempo.
 - c. Ha aumentado al doble.

Solución b

NIVEL ALTO

A.3. Concepto 3 – Nivel avanzado – Traducción de direcciones con TLB dividida

A.3.1. Preguntas de nivel bajo

- 24. (TD – Ejem3.cfg y crafty_d.trd) ¿Qué tipo de páginas se almacenan en la TLB de datos?
 - a. Páginas correspondientes a lecturas y/o escrituras en memoria.
 - b. Páginas correspondientes a búsquedas de instrucciones.
 - c. Páginas correspondientes a cualquier tipo de instrucciones.

Solución a

NIVEL BAJO

- 25. (TD – Ejem3.cfg y crafty_d.trd) ¿Qué tipo de páginas se almacenan en la TLB de instrucciones?
 - a. Páginas correspondientes a lecturas y/o escrituras en memoria.
 - b. Páginas correspondientes a búsquedas de instrucciones.
 - c. Páginas correspondientes a cualquier tipo de instrucciones.

Solución b

NIVEL BAJO

26. (TD – Ejem3.cfg y crafty_d.trd) ¿Qué ventajas aporta la TLB dividida en dos frente a la no dividida?
- No aporta ventaja, sólo permite ordenar mejor las páginas.
 - Aumenta el tamaño de la TLB
 - Aprovecha mejor el principio de localidad.

Solución c

NIVEL BAJO

A.3.2. Preguntas de nivel medio

27. (TD – Ejem3.cfg y crafty_d.trd) Ciclos. Ejecuta algunas líneas del programa y contesta. ¿Cuántos ciclos transcurren cuando hay que traer una página desde memoria secundaria hasta memoria principal?
11600. Consultar TLB (100), consultar tabla de páginas (1000), leer desde memoria secundaria (10000), enviar a través del bus (1500).
 13500. Consultar tabla de páginas (1000), leer desde memoria secundaria (10000), enviar a través del bus (1500) y escribir en memoria principal (1000).
 13600. Consultar TLB(100), consultar tabla de páginas (1000), leer desde memoria secundaria (10000), enviar a través del bus (1500) y escribir en memoria principal (1000).

Solución c

NIVEL MEDIO

28. (TD – Ejem3.cfg y crafty_d.trd) Ciclos. Ejecuta algunas líneas del programa y contesta. ¿Cuántos ciclos transcurren cuando la página buscada ya se encuentra en la TLB?
1100. Consultar TLB (100) y consultar tabla de páginas (1000).
 11100. Consultar TLB (100), consultar la tabla de páginas (1000) y leer desde memoria secundaria (10000).
 100. Consultar TLB(100).

Solución c

NIVEL MEDIO

29. (TD – Ejem3.cfg y crafty_d.trd) Ciclos. Ejecuta algunas líneas del programa y contesta. ¿Cuántos ciclos transcurren cuando la página buscada no se encuentra en la TLB pero si en la tabla de páginas?
1000. Consultar tabla de páginas (1000).
 1100. Consultar TLB (100) y consultar tabla de páginas (1000).
 11100. Consultar TLB (100), consultar la tabla de páginas (1000) y leer desde memoria secundaria (10000).

Solución b

NIVEL MEDIO

A.3.3. Preguntas de nivel alto

30. Simula las configuraciones Ejem2.cfg y Ejem3.cfg de traducción de direcciones con el fichero de traza crafty_d.trd y contesta ¿Cuál de los dos tipos de traducción es más rápida?
- a. La traducción por transformación directa sin TLB.
 - b. La traducción por transformación asociativa-directa con TLB conjunta.
 - c. Las dos traducciones por transformación asociativa son casi equivalentes.

Solución c

NIVEL ALTO

31. Simula las configuraciones Ejem2.cfg y Ejem3.cfg de traducción de direcciones con el fichero de traza crafty_d.trd y contesta ¿Qué propiedad manifiesta el programa de la traza de ejemplo?
- a. La localidad de las instrucciones es mayor que la de los datos.
 - b. La localidad de los datos es mayor que la de las instrucciones.
 - c. Las localidades de datos e instrucciones son prácticamente iguales.

Solución b

NIVEL ALTO

32. Simula las configuraciones Ejem1.cfg, Ejem2.cfg y Ejem3.cfg de traducción de direcciones con el fichero de traza crafty_d.trd y contesta ¿Cuál de los tres tipos de traducción es más rápida?
- a. La traducción por transformación directa sin TLB.
 - b. La traducción por transformación asociativa-directa con TLB conjunta.
 - c. Las dos traducciones por transformación asociativa son casi equivalentes.

Solución c

NIVEL ALTO

A.4. Concepto 4 – Nivel intermedio – Estrategias de colocación y reemplazamiento

A.4.1. Preguntas de nivel bajo

- **Mapeado directo**

33. (BD – Ejem4.cfg y crafty_a.trd) ¿En qué posición se coloca un nuevo bloque dentro de la memoria caché sabiendo que estamos utilizando una estrategia de colocación de mapeado directo?
- a. En la posición dada por la fórmula $(\text{Index}) \bmod (\text{Número de bloques de caché})$
 - b. En la posición dada por la fórmula $(\text{Tag}) \bmod (\text{Número de bloques de caché})$
 - c. En la posición dada por la fórmula $(\text{Bloque}) \bmod (\text{Número de bloques de caché})$

Solución a

NIVEL BAJO

- **Completamente asociativa**

34. (BD – Ejem5.cfg y crafty_a.trd) ¿En qué posición se coloca un nuevo bloque dentro de la memoria caché sabiendo que estamos utilizando una estrategia de colocación con una memoria completamente asociativa?
- a. En la posición dada por la fórmula $(\text{Index}) \bmod (\text{Número de bloques de caché})$
 - b. En la posición dada por la fórmula $(\text{Tag}) \bmod (\text{Número de bloques de caché})$
 - c. En cualquier posición libre que se encuentre.

Solución c

NIVEL BAJO

- **Asociativa por conjuntos**

35. (BD – Ejem11.cfg y crafty_a.trd) ¿En qué posición se coloca un nuevo bloque dentro de la memoria caché sabiendo que estamos utilizando una estrategia de colocación con una memoria asociativa por conjuntos?
- a. En la posición dada por la fórmula $(\text{Index}) \bmod (\text{Número de bloques de caché})$
 - b. En la posición dada por la fórmula $(\text{Tag}) \bmod (\text{Número de bloques de caché})$
 - c. En cualquier posición del conjunto dado por la fórmula $(\text{Index}) \bmod (\text{Número de conjuntos en caché})$.

Solución c

NIVEL BAJO

36. (BD – Ejem11.cfg y crafty_a.trd) ¿Cuál es el tamaño del conjunto de una memoria asociativa de por conjuntos de 2 vías?
- 1
 - 2
 - 4

Solución b

NIVEL BAJO

- **Reemplazamiento al AZAR**

37. (BD – Ejem5.cfg y crafty_a.trd) ¿Qué ocurre cuando no existe ninguna posición en la memoria caché a la que copiar un nuevo bloque, y hemos elegido una política de reemplazamiento al azar?
- Se elige un bloque cualquiera para sustituir con el nuevo bloque.
 - El nuevo bloque no se escribe en la memoria caché.
 - Se elige el primer bloque de la memoria para sustituirlo con el nuevo bloque.

Solución a

NIVEL BAJO

- **Reemplazamiento FIFO**

38. (BD – Ejem6.cfg y crafty_a.trd) ¿Qué bloque de caché se reemplaza cuando elegimos una política de reemplazamiento FIFO?
- Un bloque cualquiera elegido al azar.
 - El bloque que lleva más tiempo en memoria.
 - Se reemplaza aquel bloque que lleva más tiempo en memoria sin ser usado.

Solución b

NIVEL BAJO

- **Reemplazamiento LRU**

39. (BD – Ejem7.cfg y crafty_a.trd) ¿Qué bloque se reemplaza cuando elegimos una política de reemplazamiento LRU?
- Un bloque cualquiera elegido al azar.
 - El bloque que lleva más tiempo en memoria.
 - Se reemplaza aquel bloque que lleva más tiempo en memoria sin ser usado.

Solución c

NIVEL BAJO

• **Reemplazamiento CLOCK**

40. (BD – Ejem8.cfg y crafty_a.trd) ¿Qué bloque se reemplaza cuando elegimos una política de reemplazamiento CLOCK?
- Un bloque cualquiera elegido al azar.
 - El bloque que lleva más tiempo en memoria.
 - Se reemplaza aquel bloque que lleva más tiempo en memoria sin ser usado.

Solución c

NIVEL BAJO

• **Reemplazamiento LFU**

41. (BD – Ejem9.cfg y crafty_a.trd) ¿Qué bloque se reemplaza cuando elegimos una política de reemplazamiento LFU?
- Un bloque cualquiera elegido al azar.
 - El bloque que lleva más tiempo en memoria.
 - Se reemplaza aquel bloque que haya sido usado el menor número de veces.

Solución c

NIVEL BAJO

• **Reemplazamiento NUR**

42. (BD – Ejem10.cfg y crafty_a.trd) ¿Qué bloque se reemplaza cuando elegimos una política de reemplazamiento NUR?
- Un bloque que no haya sido usado (referenciado y/o modificado) recientemente.
 - El bloque que lleva más tiempo en memoria.
 - Se reemplaza aquel bloque que lleva más tiempo en memoria sin ser usado.

Solución a

NIVEL BAJO

A.4.2. Preguntas de nivel medio• **Mapeado directo**

43. (BD – Ejem4.cfg y crafty_a.trd) ¿Por qué no es necesaria una política de reemplazamiento utilizando mapeado directo?

- a. Porque nunca se reemplaza una página.
- b. Porque la página a reemplazar ya está determinada.
- c. Porque la página a reemplazar siempre se elige al azar.

Solución b

NIVEL MEDIO

44. (BD – Ejem4.cfg y crafty_a.trd) ¿Qué ocurre cuando la posición de la memoria caché a la que debemos copiar la nueva página procedente de memoria principal no está vacía sabiendo que estamos utilizando una estrategia de mapeado directo?
- a. La página de esa posición de memoria caché se sobrescribe con la nueva.
 - b. Se busca una nueva posición que esté libre.
 - c. La nueva página no se escribe en la memoria caché.

Solución a

NIVEL MEDIO

- ***Completamente asociativa***

45. (BD – Ejem5.cfg y crafty_a.trd) ¿Por qué no es necesaria la parte de índice en la dirección del bloque utilizando una estrategia de colocación con una memoria completamente asociativa?
- a. Porque la posición dentro de memoria caché no está definida, el bloque puede colocarse en cualquier posición.
 - b. Porque el tag determina la posición dentro de memoria caché.
 - c. Porque la memoria caché nunca se llena.

Solución a

NIVEL MEDIO

- ***Asociativa por conjuntos***

46. (BD – Ejem11.cfg y crafty_a.trd) ¿Para qué se utiliza la parte de índice en la dirección del bloque utilizando una estrategia de colocación con una memoria asociativa por conjuntos?
- a. Para obtener el conjunto donde colocar el bloque.
 - b. Para obtener la posición donde colocar el bloque.
 - c. Para obtener la posición del conjunto donde colocar el bloque.

Solución a

NIVEL MEDIO

- **Reemplazamiento FIFO**

47. (BD – Ejem6.cfg y crafty_a.trd) Teniendo en cuenta que hemos elegido una política de reemplazamiento FIFO, ¿Qué contador se asigna a un nuevo bloque que entra en memoria?
- a. 0
 - b. Esta política no utiliza contador.
 - c. Número de línea.

Solución a

NIVEL MEDIO

- **Reemplazamiento LRU**

48. (BD – Ejem7.cfg y crafty_a.trd) Teniendo en cuenta que hemos elegido una política de reemplazamiento LRU, ¿Qué contador se asigna a un nuevo bloque que entra en memoria?
- a. 0
 - b. Esta política no utiliza contador.
 - c. Número de línea.

Solución a

NIVEL MEDIO

- **Reemplazamiento CLOCK**

49. (BD – Ejem8.cfg y crafty_a.trd) Teniendo en cuenta que hemos elegido una política de reemplazamiento CLOCK, ¿Qué contador se asigna a un nuevo bloque que entra en memoria?
- a. 0
 - b. Esta política no utiliza contador.
 - c. Número de línea.

Solución b

NIVEL MEDIO

- **Reemplazamiento LFU**

50. (BD – Ejem9.cfg y crafty_a.trd) Teniendo en cuenta que hemos elegido una política de reemplazamiento LFU, ¿Qué contador se asigna a un nuevo bloque que entra en memoria?
- a. 0
 - b. Esta política no utiliza contador.
 - c. Número de línea.

Solución a

NIVEL MEDIO

- **Reemplazamiento NUR**

51. (BD – Ejem10.cfg y crafty_a.trd) Teniendo en cuenta que hemos elegido una política de reemplazamiento NUR, ¿Qué contador se asigna a un nuevo bloque que entra en memoria?
- a. 0(00) si es una escritura y 1(01) si es una lectura.
 - b. 1(01) si es una escritura y 0(00) si es una lectura.
 - c. Esta política no utiliza contador.

Solución b

NIVEL MEDIO

A.4.3. Preguntas de nivel alto

52. (BD – Ejem7.cfg y crafty_a.trd) En el algoritmo de reemplazamiento LRU la función de éxitos es:
- a. Monótona creciente
 - b. Monótona no decreciente
 - c. Ninguna de las tres anteriores

Solución b

NIVEL ALTO

53. (BD – Ejem6.cfg y crafty_a.trd) En el algoritmo de reemplazamiento FIFO la función de éxitos es:
- a. Monótona decreciente
 - b. Monótona no creciente
 - c. Ninguna de las tres anteriores

Solución c

NIVEL ALTO

A.5. Concepto 5 – Nivel complejo – Estrategias de coherencia

A.5.1. Preguntas de nivel bajo

54. (BD – Ejem4.cfg y crafty_a.trd) El victim buffer se utiliza en:
- a. La escritura directa
 - b. La escritura retardada
 - c. Tanto en escritura directa como en escritura retardada

Solución a

NIVEL BAJO

55. (BD – Ejem4.cfg y crafty_a.trd) Cuando se utiliza escritura directa:
- La información se escribe en la caché y en la memoria principal
 - La información se escribe sólo en la caché.
 - La información se escribe en la caché, en la memoria principal y en la memoria secundaria.

Solución a

NIVEL BAJO

56. (BD – Ejem4.cfg y crafty_a.trd) Cuando se utiliza escritura retardada:
- La información se escribe en la caché y en la memoria principal
 - La información se escribe sólo en la caché.
 - La información se escribe en la caché, en la memoria principal y en la memoria secundaria.

Solución b

NIVEL BAJO

A.5.2. Preguntas de nivel medio

57. (BD – Ejem4.cfg y crafty_a.trd) ¿Qué es más rápido, la escritura directa o la escritura retardada?:
- La escritura directa
 - La escritura retardada
 - Si utilizamos el victim buffer, ambas son prácticamente iguales.

Solución c

NIVEL MEDIO

58. (BD – Ejem4.cfg y crafty_a.trd) ¿Se puede implementar la escritura directa sin victim buffer?
- No
 - Si
 - Si, siempre que las caches sean pequeñas.

Solución b

NIVEL MEDIO

59. (BD – Ejem4.cfg y crafty_a.trd) ¿Para que se utiliza el bit de modificación de la memoria caché?

- a. Indica si el bloque se ha modificado mientras se encontraba cargado en ese nivel de caché.
- b. Indica si el bloque se ha modificado mientras se encontraba cargado en cualquier nivel de caché.
- c. Indica que la página correspondiente a ese bloque en memoria principal se ha modificado.

Solución a

NIVEL MEDIO

A.5.3. Preguntas de nivel alto

60. (Ejem4.cfg y Crafty_a.trd) Ejecuta algunas líneas del programa y contesta, ¿Cuándo se actualiza el bit de modificación del bloque?
- a. Cuando se produce una escritura en memoria.
 - b. Cuando se produce una lectura de memoria.
 - c. Cuando hay una instrucción.

Solución a

NIVEL ALTO

61. (BD – Ejem4.cfg y crafty_a.trd) Una página de memoria principal se corresponde con:
- a. Varios bloques de memoria caché en cada nivel.
 - b. Varios bloques de memoria caché en el nivel 1.
 - c. Un bloque de memoria caché en cada nivel.

Solución a

NIVEL ALTO

62. (BD – Ejem4.cfg y crafty_a.trd) ¿Se pueden tener más de un victim buffer en una jerarquía de memoria multinivel?
- a. No, en ningún caso.
 - b. Sí, uno por nivel.
 - c. Sí, pero dos como máximo.

Solución b

NIVEL ALTO

ANEXO B.

Preguntas del test adaptativo

Para la realización de la prueba con el test adaptativo se elaboraron una serie de preguntas que se muestran a continuación.

A.1. Concepto 1 – Nivel simple – Entorno de simulación

A.1.1. Preguntas de nivel Bajo

- *Páginas*

1. (Ejem1.cfg y crafty_d.trd) Si el tamaño de la memoria principal es de 64KB y el tamaño de página es de 4KB, ¿Cuántas páginas puede almacenar la memoria principal?
 - a. 8
 - b. 16
 - c. 32

Solución b: $64/4 = 16$

NIVEL BAJO

- *Tabla de páginas*

2. (Ejem1.cfg y crafty_d.trd) Teniendo en cuenta que el tamaño de la memoria secundaria/virtual es de 1024KB, el tamaño de la memoria principal de 64KB y el tamaño de página es de 4KB. ¿Cuál es el número de entradas de la tabla de páginas?
 - a. 256
 - b. 1024
 - c. 128

Solución a: $1024/4 = 256$

NIVEL BAJO

- **Páginas**

3. (Ejem4.cfg y crafty_d.trd) Teniendo en cuenta que el tamaño de la memoria principal es de 1024KB y el tamaño de página es de 4KB. ¿De cuantas páginas se compone la memoria principal?
- a. 64
 - b. 128
 - c. 256

Solución c: $1024/4 = 256$

NIVEL BAJO

- **Memorias caché**

4. (Ejem4.cfg y crafty_d.trd) Si el tamaño de la memoria caché de nivel 3 es de 256KB y el tamaño de bloque es de 1KB, ¿Cuántas bloques puede almacenar la caché de nivel 3?
- a. 32
 - b. 64
 - c. 256

Solución c: $256/1 = 256$

NIVEL BAJO

5. (Ejem4.cfg y crafty_d.trd) Si el tamaño de la memoria caché de nivel 2 es de 64KB y el tamaño de bloque es de 1KB, ¿Cuántos bloques puede almacenar la caché de nivel 2?
- a. 16
 - b. 32
 - c. 64

Solución a: $64/1 = 64$

NIVEL BAJO

A.1.2. Preguntas de nivel intermedio

- **Dirección virtual y real**

6. (Ejem1.cfg y crafty_d.trd) Sabiendo que el tamaño de página es de 4KB, ¿Cuántos bits hacen falta para representar el desplazamiento dentro de la página?
- a. 4
 - b. 8
 - c. 12

Solución c: $2^{\exp 12} = 4096$

NIVEL INTERMEDIO

- **Dirección virtual**

7. (Ejem1.cfg y crafty_d.trd) Si el tamaño de la memoria virtual es de 1024KB y el tamaño de página es de 4KB, ¿De cuantos bits se compone la dirección virtual?
- a. 4
 - b. 16
 - c. 20

Solución c: $2^{\exp 20} = 1048576 - 1024kb$

NIVEL INTERMEDIO

8. (Ejem1.cfg y crafty_d.trd) ¿Cuántos bits hacen falta para direccionar una página dentro de la tabla de páginas? (Parte p de la dirección virtual)
- a. 2
 - b. 4
 - c. 8

Solución c: $2^{\exp 8} = 256$ páginas

NIVEL INTERMEDIO

- **Dirección real**

9. (Ejem1.cfg y crafty_d.trd) Si el tamaño de la memoria principal es de 64KB y el tamaño de página es de 4KB, ¿De cuantos bits se compone la dirección real?
- a. 4
 - b. 16
 - c. 20

Solución b: $2^{\exp 16} = 65536 - 64Kb$

NIVEL INTERMEDIO

10. (Ejem1.cfg y crafty_d.trd) ¿Cuántos bits hacen falta para direccionar una página dentro de memoria principal? (Parte p' de la dirección real)
- a. 2
 - b. 4
 - c. 8

Solución b: $2^{\exp 4} = 16$ páginas

NIVEL INTERMEDIO

• *Memorias caché*

11. (Ejem4.cfg y crafty_a.trd) Si el tamaño de la memoria caché de nivel 1 es de 16KB y el tamaño de bloque es de 1KB, y teniendo en cuenta que está dividida en instrucciones y datos, ¿Cuántos bloques puede almacenar cada una de las divisiones de la caché de nivel 1?
- 4
 - 8
 - 16

Solución a: $16/1 = 16$

NIVEL INTERMEDIO

12. (Ejem4.cfg y crafty_a.trd) Si el tamaño del bloque de memoria caché es de 1024Bytes ¿Cuántos bits son necesarios para representar el desplazamiento dentro del bloque?
- 4
 - 8
 - 10

Solución c: $2^{10} = 1024$

NIVEL INTERMEDIO

A.1.3. Preguntas de nivel alto• *Traducción de direcciones*

13. (TD - Ejem1.cfg y crafty_d.trd) El bit r de la tabla de páginas indica:
- Si la página se encuentra cargada en la memoria principal.
 - Si la página se encuentra cargada en la memoria secundaria.
 - Que la página ha sido reemplazada en memoria principal.

Solución a

NIVEL ALTO

14. (TD - Ejem1.cfg y crafty_d.trd) La tabla de páginas tiene tantas entradas como:
- Páginas reales.
 - Páginas virtuales.
 - Entradas tiene la TLB.

Solución a

NIVEL ALTO

• **Búsqueda de páginas**

15. (BD – Ejem4.cfg y crafty_a.trd) ¿En qué campos se divide la parte correspondiente a bloque (b) de la dirección de memoria caché?
- Índice (Ind) y desplazamiento (d).
 - Índice (Ind) y página virtual (p).
 - Índice (Ind) y Etiqueta (Tag).

Solución c

NIVEL ALTO

16. (BD – Ejem4.cfg y crafty_a.trd) ¿Qué ventajas aporta que la memoria caché de nivel 1 esté dividida en dos?
- No aporta ventaja, sólo permite ordenar mejor las páginas.
 - Aumenta el tamaño de la caché de nivel 1.
 - Aprovecha mejor el principio de localidad.

Solución c

NIVEL ALTO

17. (BD – Ejem4.cfg y crafty_a.trd) Ejecuta 2 o 3 pasos de la simulación y di cual de las siguientes afirmaciones es correcta:
- Las cachés de nivel 2 y de nivel 3 son inclusivas.
 - Todas las cachés son inclusivas.
 - Las cachés de nivel 1 y de nivel 2 son inclusivas.

Solución a

NIVEL ALTO

18. (BD – Ejem4.cfg y crafty_a.trd) ¿Aumenta la caché la capacidad de direccionamiento de la memoria principal?
- Si
 - No
 - No tiene porqué.

Solución b

NIVEL ALTO

A.2. Concepto 2 – Nivel intermedio – Traducción de direcciones**A.2.1. Preguntas de nivel bajo**

- **Funciones de la TLB**

19. (TD – Ejem2.cfg y crafty_d.trd) ¿Cuál es la función de la TLB?
- a. Acelerar la traducción almacenando las últimas direcciones traducidas.
 - b. Aumentar la capacidad de almacenamiento de la tabla de páginas.
 - c. Aumentar la capacidad de direccionamiento.

Solución a

NIVEL BAJO

20. (TD – Ejem2.cfg y crafty_d.trd) El número de entradas de la TLB es:
- a. Mayor que la tabla de páginas.
 - b. Menor o igual al de la tabla de páginas.
 - c. Menor que la tabla de páginas.

Solución b

NIVEL BAJO

A.2.2. Preguntas de nivel medio

21. (TD – Ejem1.cfg y crafty_d.trd) Ciclos. Ejecuta algunas líneas del programa y contesta. ¿Cuántos ciclos transcurren cuando hay que traer una página desde memoria secundaria hasta memoria principal?
- a. 12500. Consultar tabla de páginas (1000), leer desde memoria secundaria (10000), enviar a través del bus (1500).
 - b. 13500. Consultar tabla de páginas (1000), leer desde memoria secundaria (10000), enviar a través del bus (1500) y escribir en memoria principal (1000).
 - c. 14500. Consultar tabla de páginas (1000), leer desde memoria secundaria (10000), enviar a través del bus (1500), escribir en memoria principal (1000) y volver a consultar la tabla de páginas (1000).

Solución b

NIVEL MEDIO

22. (TD – Ejem1.cfg y crafty_d.trd) Ciclos. Ejecuta algunas líneas del programa y contesta. ¿Cuántos ciclos transcurren cuando la página ya se encuentra en memoria principal?
- a. 1000. Consultar la tabla de páginas (1000).
 - b. 11000. Consultar la tabla de páginas (1000) y leer desde memoria secundaria (10000).

- c. 13500. Consultar tabla de páginas (1000), leer desde memoria secundaria (10000), enviar a través del bus (1500) y escribir en memoria principal (1000).

Solución a

NIVEL MEDIO

A.2.3. Preguntas de nivel alto

- 23. Simula las configuraciones de traducción de direcciones Ejem1.cfg y Ejem2.cfg con el fichero de traza crafty_d.trd y contesta. El número de ciclos usando la TLB
 - a. Ha disminuido a la mitad.
 - b. Ha disminuido a menos de la mitad de tiempo.
 - c. Ha aumentado al doble.

Solución b

NIVEL ALTO

A.3. Concepto 3 – Nivel avanzado – Traducción de direcciones con TLB dividida

A.3.1. Preguntas de nivel bajo

- 24. (TD – Ejem3.cfg y crafty_d.trd) ¿Qué tipo de páginas se almacenan en la TLB de datos?
 - a. Páginas correspondientes a lecturas y/o escrituras en memoria.
 - b. Páginas correspondientes a búsquedas de instrucciones.
 - c. Páginas correspondientes a cualquier tipo de instrucciones.

Solución a

NIVEL BAJO

- 25. (TD – Ejem3.cfg y crafty_d.trd) ¿Qué tipo de páginas se almacenan en la TLB de instrucciones?
 - a. Páginas correspondientes a lecturas y/o escrituras en memoria.
 - b. Páginas correspondientes a búsquedas de instrucciones.
 - c. Páginas correspondientes a cualquier tipo de instrucciones.

Solución b

NIVEL BAJO

26. (TD – Ejem3.cfg y crafty_d.trd) ¿Qué ventajas aporta la TLB dividida en dos frente a la no dividida?
- No aporta ventaja, sólo permite ordenar mejor las páginas.
 - Aumenta el tamaño de la TLB
 - Aprovecha mejor el principio de localidad.

Solución c

NIVEL BAJO

A.3.2. Preguntas de nivel medio

27. (TD – Ejem3.cfg y crafty_d.trd) Ciclos. Ejecuta algunas líneas del programa y contesta. ¿Cuántos ciclos transcurren cuando hay que traer una página desde memoria secundaria hasta memoria principal?
11600. Consultar TLB (100), consultar tabla de páginas (1000), leer desde memoria secundaria (10000), enviar a través del bus (1500).
 13500. Consultar tabla de páginas (1000), leer desde memoria secundaria (10000), enviar a través del bus (1500) y escribir en memoria principal (1000).
 13600. Consultar TLB(100), consultar tabla de páginas (1000), leer desde memoria secundaria (10000), enviar a través del bus (1500) y escribir en memoria principal (1000).

Solución c

NIVEL MEDIO

28. (TD – Ejem3.cfg y crafty_d.trd) Ciclos. Ejecuta algunas líneas del programa y contesta. ¿Cuántos ciclos transcurren cuando la página buscada ya se encuentra en la TLB?
1100. Consultar TLB (100) y consultar tabla de páginas (1000).
 11100. Consultar TLB (100), consultar la tabla de páginas (1000) y leer desde memoria secundaria (10000).
 100. Consultar TLB(100).

Solución c

NIVEL MEDIO

29. (TD – Ejem3.cfg y crafty_d.trd) Ciclos. Ejecuta algunas líneas del programa y contesta. ¿Cuántos ciclos transcurren cuando la página buscada no se encuentra en la TLB pero si en la tabla de páginas?
1000. Consultar tabla de páginas (1000).
 1100. Consultar TLB (100) y consultar tabla de páginas (1000).
 11100. Consultar TLB (100), consultar la tabla de páginas (1000) y leer desde memoria secundaria (10000).

Solución b

NIVEL MEDIO

A.3.3. Preguntas de nivel alto

30. Simula las configuraciones Ejem2.cfg y Ejem3.cfg de traducción de direcciones con el fichero de traza crafty_d.trd y contesta ¿Cuál de los dos tipos de traducción es más rápida?
- a. La traducción por transformación directa sin TLB.
 - b. La traducción por transformación asociativa-directa con TLB conjunta.
 - c. Las dos traducciones por transformación asociativa son casi equivalentes.

Solución c

NIVEL ALTO

31. Simula las configuraciones Ejem2.cfg y Ejem3.cfg de traducción de direcciones con el fichero de traza crafty_d.trd y contesta ¿Qué propiedad manifiesta el programa de la traza de ejemplo?
- a. La localidad de las instrucciones es mayor que la de los datos.
 - b. La localidad de los datos es mayor que la de las instrucciones.
 - c. Las localidades de datos e instrucciones son prácticamente iguales.

Solución b

NIVEL ALTO

32. Simula las configuraciones Ejem1.cfg, Ejem2.cfg y Ejem3.cfg de traducción de direcciones con el fichero de traza crafty_d.trd y contesta ¿Cuál de los tres tipos de traducción es más rápida?
- a. La traducción por transformación directa sin TLB.
 - b. La traducción por transformación asociativa-directa con TLB conjunta.
 - c. Las dos traducciones por transformación asociativa son casi equivalentes.

Solución c

NIVEL ALTO

A.4. Concepto 4 – Nivel intermedio – Estrategias de colocación y reemplazamiento

A.4.1. Preguntas de nivel bajo

- **Mapeado directo**

33. (BD – Ejem4.cfg y crafty_a.trd) ¿En qué posición se coloca un nuevo bloque dentro de la memoria caché sabiendo que estamos utilizando una estrategia de colocación de mapeado directo?
- a. En la posición dada por la fórmula $(\text{Index}) \bmod (\text{Número de bloques de caché})$
 - b. En la posición dada por la fórmula $(\text{Tag}) \bmod (\text{Número de bloques de caché})$
 - c. En la posición dada por la fórmula $(\text{Bloque}) \bmod (\text{Número de bloques de caché})$

Solución a

NIVEL BAJO

- **Completamente asociativa**

34. (BD – Ejem5.cfg y crafty_a.trd) ¿En qué posición se coloca un nuevo bloque dentro de la memoria caché sabiendo que estamos utilizando una estrategia de colocación con una memoria completamente asociativa?
- a. En la posición dada por la fórmula $(\text{Index}) \bmod (\text{Número de bloques de caché})$
 - b. En la posición dada por la fórmula $(\text{Tag}) \bmod (\text{Número de bloques de caché})$
 - c. En cualquier posición libre que se encuentre.

Solución c

NIVEL BAJO

- **Asociativa por conjuntos**

35. (BD – Ejem11.cfg y crafty_a.trd) ¿En qué posición se coloca un nuevo bloque dentro de la memoria caché sabiendo que estamos utilizando una estrategia de colocación con una memoria asociativa por conjuntos?
- a. En la posición dada por la fórmula $(\text{Index}) \bmod (\text{Número de bloques de caché})$
 - b. En la posición dada por la fórmula $(\text{Tag}) \bmod (\text{Número de bloques de caché})$
 - c. En cualquier posición del conjunto dado por la fórmula $(\text{Index}) \bmod (\text{Número de conjuntos en caché})$.

Solución c

NIVEL BAJO

36. (BD – Ejem11.cfg y crafty_a.trd) ¿Cuál es el tamaño del conjunto de una memoria asociativa de por conjuntos de 2 vías?
- 1
 - 2
 - 4

Solución b

NIVEL BAJO

- **Reemplazamiento al AZAR**

37. (BD – Ejem5.cfg y crafty_a.trd) ¿Qué ocurre cuando no existe ninguna posición en la memoria caché a la que copiar un nuevo bloque, y hemos elegido una política de reemplazamiento al azar?
- Se elige un bloque cualquiera para sustituir con el nuevo bloque.
 - El nuevo bloque no se escribe en la memoria caché.
 - Se elige el primer bloque de la memoria para sustituirlo con el nuevo bloque.

Solución a

NIVEL BAJO

- **Reemplazamiento FIFO**

38. (BD – Ejem6.cfg y crafty_a.trd) ¿Qué bloque de caché se reemplaza cuando elegimos una política de reemplazamiento FIFO?
- Un bloque cualquiera elegido al azar.
 - El bloque que lleva más tiempo en memoria.
 - Se reemplaza aquel bloque que lleva más tiempo en memoria sin ser usado.

Solución b

NIVEL BAJO

- **Reemplazamiento LRU**

39. (BD – Ejem7.cfg y crafty_a.trd) ¿Qué bloque se reemplaza cuando elegimos una política de reemplazamiento LRU?
- Un bloque cualquiera elegido al azar.
 - El bloque que lleva más tiempo en memoria.
 - Se reemplaza aquel bloque que lleva más tiempo en memoria sin ser usado.

Solución c

NIVEL BAJO

• **Reemplazamiento CLOCK**

40. (BD – Ejem8.cfg y crafty_a.trd) ¿Qué bloque se reemplaza cuando elegimos una política de reemplazamiento CLOCK?
- Un bloque cualquiera elegido al azar.
 - El bloque que lleva más tiempo en memoria.
 - Se reemplaza aquel bloque que lleva más tiempo en memoria sin ser usado.

Solución c

NIVEL BAJO

• **Reemplazamiento LFU**

41. (BD – Ejem9.cfg y crafty_a.trd) ¿Qué bloque se reemplaza cuando elegimos una política de reemplazamiento LFU?
- Un bloque cualquiera elegido al azar.
 - El bloque que lleva más tiempo en memoria.
 - Se reemplaza aquel bloque que haya sido usado el menor número de veces.

Solución c

NIVEL BAJO

• **Reemplazamiento NUR**

42. (BD – Ejem10.cfg y crafty_a.trd) ¿Qué bloque se reemplaza cuando elegimos una política de reemplazamiento NUR?
- Un bloque que no haya sido usado (referenciado y/o modificado) recientemente.
 - El bloque que lleva más tiempo en memoria.
 - Se reemplaza aquel bloque que lleva más tiempo en memoria sin ser usado.

Solución a

NIVEL BAJO

A.4.2. Preguntas de nivel medio• **Mapeado directo**

43. (BD – Ejem4.cfg y crafty_a.trd) ¿Por qué no es necesaria una política de reemplazamiento utilizando mapeado directo?

- a. Porque nunca se reemplaza una página.
- b. Porque la página a reemplazar ya está determinada.
- c. Porque la página a reemplazar siempre se elige al azar.

Solución b

NIVEL MEDIO

44. (BD – Ejem4.cfg y crafty_a.trd) ¿Qué ocurre cuando la posición de la memoria caché a la que debemos copiar la nueva página procedente de memoria principal no está vacía sabiendo que estamos utilizando una estrategia de mapeado directo?
- a. La página de esa posición de memoria caché se sobrescribe con la nueva.
 - b. Se busca una nueva posición que esté libre.
 - c. La nueva página no se escribe en la memoria caché.

Solución a

NIVEL MEDIO

- ***Completamente asociativa***

45. (BD – Ejem5.cfg y crafty_a.trd) ¿Por qué no es necesaria la parte de índice en la dirección del bloque utilizando una estrategia de colocación con una memoria completamente asociativa?
- a. Porque la posición dentro de memoria caché no está definida, el bloque puede colocarse en cualquier posición.
 - b. Porque el tag determina la posición dentro de memoria caché.
 - c. Porque la memoria caché nunca se llena.

Solución a

NIVEL MEDIO

- ***Asociativa por conjuntos***

46. (BD – Ejem11.cfg y crafty_a.trd) ¿Para qué se utiliza la parte de índice en la dirección del bloque utilizando una estrategia de colocación con una memoria asociativa por conjuntos?
- a. Para obtener el conjunto donde colocar el bloque.
 - b. Para obtener la posición donde colocar el bloque.
 - c. Para obtener la posición del conjunto donde colocar el bloque.

Solución a

NIVEL MEDIO

- **Reemplazamiento FIFO**

47. (BD – Ejem6.cfg y crafty_a.trd) Teniendo en cuenta que hemos elegido una política de reemplazamiento FIFO, ¿Qué contador se asigna a un nuevo bloque que entra en memoria?
- a. 0
 - b. Esta política no utiliza contador.
 - c. Número de línea.

Solución a

NIVEL MEDIO

- **Reemplazamiento LRU**

48. (BD – Ejem7.cfg y crafty_a.trd) Teniendo en cuenta que hemos elegido una política de reemplazamiento LRU, ¿Qué contador se asigna a un nuevo bloque que entra en memoria?
- a. 0
 - b. Esta política no utiliza contador.
 - c. Número de línea.

Solución a

NIVEL MEDIO

- **Reemplazamiento CLOCK**

49. (BD – Ejem8.cfg y crafty_a.trd) Teniendo en cuenta que hemos elegido una política de reemplazamiento CLOCK, ¿Qué contador se asigna a un nuevo bloque que entra en memoria?
- a. 0
 - b. Esta política no utiliza contador.
 - c. Número de línea.

Solución b

NIVEL MEDIO

- **Reemplazamiento LFU**

50. (BD – Ejem9.cfg y crafty_a.trd) Teniendo en cuenta que hemos elegido una política de reemplazamiento LFU, ¿Qué contador se asigna a un nuevo bloque que entra en memoria?
- a. 0
 - b. Esta política no utiliza contador.
 - c. Número de línea.

Solución a

NIVEL MEDIO

- **Reemplazamiento NUR**

51. (BD – Ejem10.cfg y crafty_a.trd) Teniendo en cuenta que hemos elegido una política de reemplazamiento NUR, ¿Qué contador se asigna a un nuevo bloque que entra en memoria?
- a. 0(00) si es una escritura y 1(01) si es una lectura.
 - b. 1(01) si es una escritura y 0(00) si es una lectura.
 - c. Esta política no utiliza contador.

Solución b

NIVEL MEDIO

A.4.3. Preguntas de nivel alto

52. (BD – Ejem7.cfg y crafty_a.trd) En el algoritmo de reemplazamiento LRU la función de éxitos es:
- a. Monótona creciente
 - b. Monótona no decreciente
 - c. Ninguna de las tres anteriores

Solución b

NIVEL ALTO

53. (BD – Ejem6.cfg y crafty_a.trd) En el algoritmo de reemplazamiento FIFO la función de éxitos es:
- a. Monótona decreciente
 - b. Monótona no creciente
 - c. Ninguna de las tres anteriores

Solución c

NIVEL ALTO

A.5. Concepto 5 – Nivel complejo – Estrategias de coherencia

A.5.1. Preguntas de nivel bajo

54. (BD – Ejem4.cfg y crafty_a.trd) El victim buffer se utiliza en:
- a. La escritura directa
 - b. La escritura retardada
 - c. Tanto en escritura directa como en escritura retardada

Solución a

NIVEL BAJO

55. (BD – Ejem4.cfg y crafty_a.trd) Cuando se utiliza escritura directa:
- La información se escribe en la caché y en la memoria principal
 - La información se escribe sólo en la caché.
 - La información se escribe en la caché, en la memoria principal y en la memoria secundaria.

Solución a

NIVEL BAJO

56. (BD – Ejem4.cfg y crafty_a.trd) Cuando se utiliza escritura retardada:
- La información se escribe en la caché y en la memoria principal
 - La información se escribe sólo en la caché.
 - La información se escribe en la caché, en la memoria principal y en la memoria secundaria.

Solución b

NIVEL BAJO

A.5.2. Preguntas de nivel medio

57. (BD – Ejem4.cfg y crafty_a.trd) ¿Qué es más rápido, la escritura directa o la escritura retardada?:
- La escritura directa
 - La escritura retardada
 - Si utilizamos el victim buffer, ambas son prácticamente iguales.

Solución c

NIVEL MEDIO

58. (BD – Ejem4.cfg y crafty_a.trd) ¿Se puede implementar la escritura directa sin victim buffer?
- No
 - Si
 - Si, siempre que las caches sean pequeñas.

Solución b

NIVEL MEDIO

59. (BD – Ejem4.cfg y crafty_a.trd) ¿Para que se utiliza el bit de modificación de la memoria caché?

- a. Indica si el bloque se ha modificado mientras se encontraba cargado en ese nivel de caché.
- b. Indica si el bloque se ha modificado mientras se encontraba cargado en cualquier nivel de caché.
- c. Indica que la página correspondiente a ese bloque en memoria principal se ha modificado.

Solución a

NIVEL MEDIO

A.5.3. Preguntas de nivel alto

60. (Ejem4.cfg y Crafty_a.trd) Ejecuta algunas líneas del programa y contesta, ¿Cuándo se actualiza el bit de modificación del bloque?
- a. Cuando se produce una escritura en memoria.
 - b. Cuando se produce una lectura de memoria.
 - c. Cuando hay una instrucción.

Solución a

NIVEL ALTO

61. (BD – Ejem4.cfg y crafty_a.trd) Una página de memoria principal se corresponde con:
- a. Varios bloques de memoria caché en cada nivel.
 - b. Varios bloques de memoria caché en el nivel 1.
 - c. Un bloque de memoria caché en cada nivel.

Solución a

NIVEL ALTO

62. (BD – Ejem4.cfg y crafty_a.trd) ¿Se pueden tener más de un victim buffer en una jerarquía de memoria multinivel?
- a. No, en ningún caso.
 - b. Sí, uno por nivel.
 - c. Sí, pero dos como máximo.

Solución b

NIVEL ALTO

Bibliografía.

- [1] *John L. Hennesy & David A. Patterson, Computer Architecture: A Quantitative Approach 3rd Edition*, Morgan Kauffman Publishers, 1990 – Third edition 2003, ISBN 1-55860-596-7.
- [2] *John L. Hennesy & David A. Patterson, Arquitectura de Computadores: Un enfoque cuantitativo*, Mc-Graw Hill, año1993, ISBN 84-7615-912-9.
- [3] *David A. Patterson & John L. Hennesy, Estructura y diseño de computadores: Interface circuitería / programación (3 volúmenes)*, Editorial Reverté S.A., año 2000, ISBN 84-291-2619-8.
- [4] *Carl Hamacher, Zvonko Vranesic, Safwat Zacky, Organización de computadores*, Mc-Graw Hill Inc., año 2002 – Quinta edición, ISBN 84-481-3951-8.
- [5] *Andrew S. Tannenbaun, Organización de computadores: Un enfoque estructurado*, Prentice-Hall Hispanoamericana S.A., año 1986 – 2ª Edición, ISBN 0-13-854489-1.
- [6] *Harvey M. Deitel, Introducción a los sistemas operativos*, Addison-Wesley Iberoamericana, año 1987, ISBN 0-201-640279.
- [7] *Manuel Area Moreira, Los medios y las tecnologías en la educación*, Ediciones Pirámide Grupo Añaya S.A., año 2004, ISBN 84-368-1895-4.
- [8] *Master Duria (Diseño y utilización de recursos informáticos en el aula) 1ª Edición 2003, Módulo VI, Las Tic en la enseñanza*, <http://duria.cyc.ull.es>, año 2003, Departamento de Computadoras y Control de la Universidad de La Laguna
- [9] *Zebenzui Pérez Ramos, Francisco Javier Medina Santos, Daniel Jacobo Díaz González, Memoria del Proyecto de Fin de Carrera “PORTAD y HEVAH, interfaz web y personaje 3D para la evaluación de test adaptativos en XML”*, Septiembre 2004, Escuela Técnica Superior de Ingeniería Informática de la Universidad de La Laguna.
- [10] *SPEC CPU2000 Documentation*, <http://www.spec.org/cpu2000/docs/>, Standard Performance Evaluation Cooperation
- [11] *Computer Architecture & Assembly Language HomePage Simulators* <http://www.sosresearch.org/caale/caalesimulators.html> Sos Research Group