Performance de Computadoras SIMD Implementadas en FPGAs

Lic. Eduardo A. Sanchez[1], Dr. Pablo A. Ferreyra[1],
Ing. Carlos A. Marqués[1], Dr. Raoul Velazco[2]

1: Universidad Nacional de Córdoba Córdoba, Argentina
2: Laboratorie TIMA, Grenoble, France

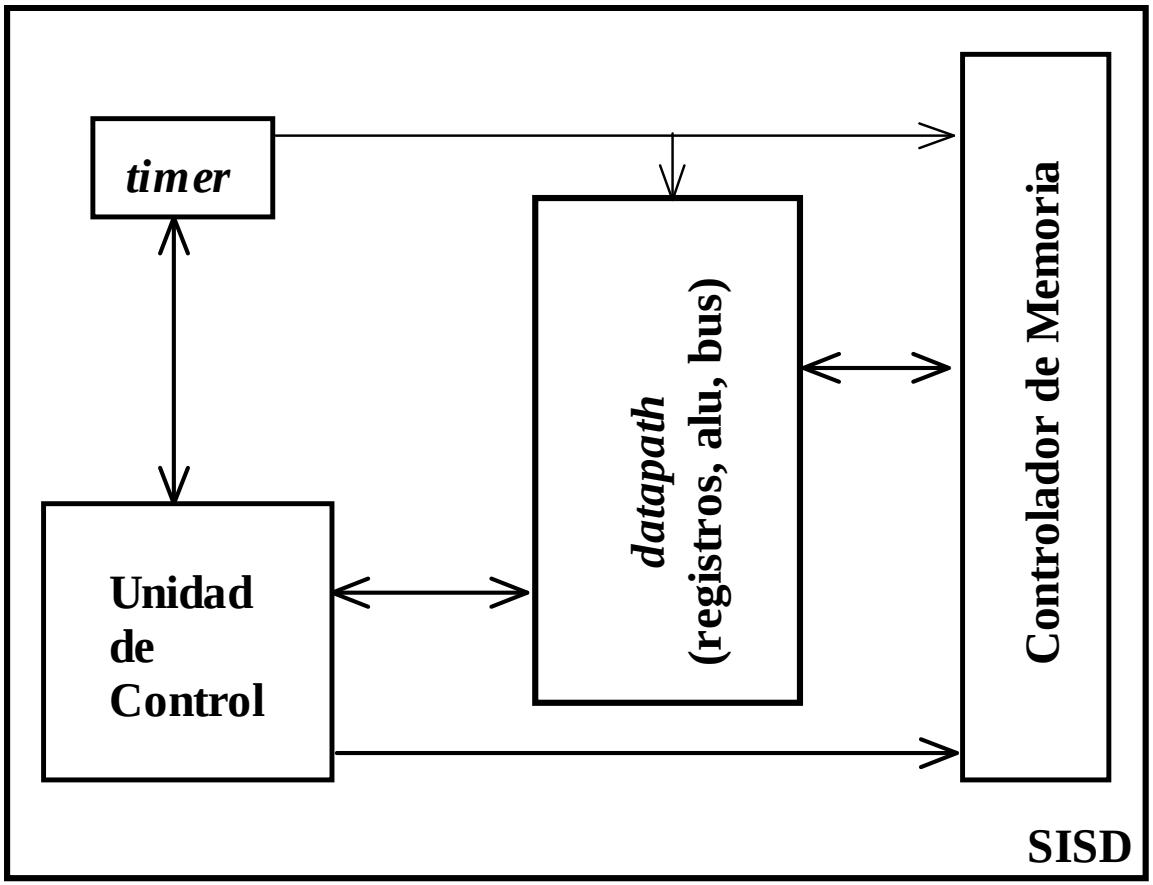
Agenda

- Introducción
- Computadora SISD Básica
- Algoritmo sobre SISD
- Rendimiento en SISD
- Computadora SIMD
- Algoritmo sobre SIMD
- Rendimiento sobre SIMD
- Algoritmo sobre SIMD Mejorado
- Rendimiento en SIMD Mejorado
- Conclusiones

Introducción

- Ventajas de desarrollo con FPGAs
  - Eliminación de los problemas de obsolescencia.
  - Reutilización de arquitecturas conocidas.
  - Permite crear adaptaciones e ir introduciendo conceptos nuevos
  - Verificar las ganancias de performance en cada una de las arquitecturas.

Computadora SISD Básica



Computadora SISD Básica (Cont.)

Instrucción	Ciclos
Limpiar acumulador	5
Sumar el acumulador con una posición de memoria	7
Cargar una posición de memoria en el acumulador	7
Guardar el acumulador en una posición de memoria	6
Salto incondicional	6
Cargar un literal de 10 bits en el acumulador	5
Decrementar el contenido de una posición de memoria y saltar la próxima instrucción si el resultado es 0.	8
Incrementar el contenido de una posición de memoria.	8

Tabla de Instrucciones mínima

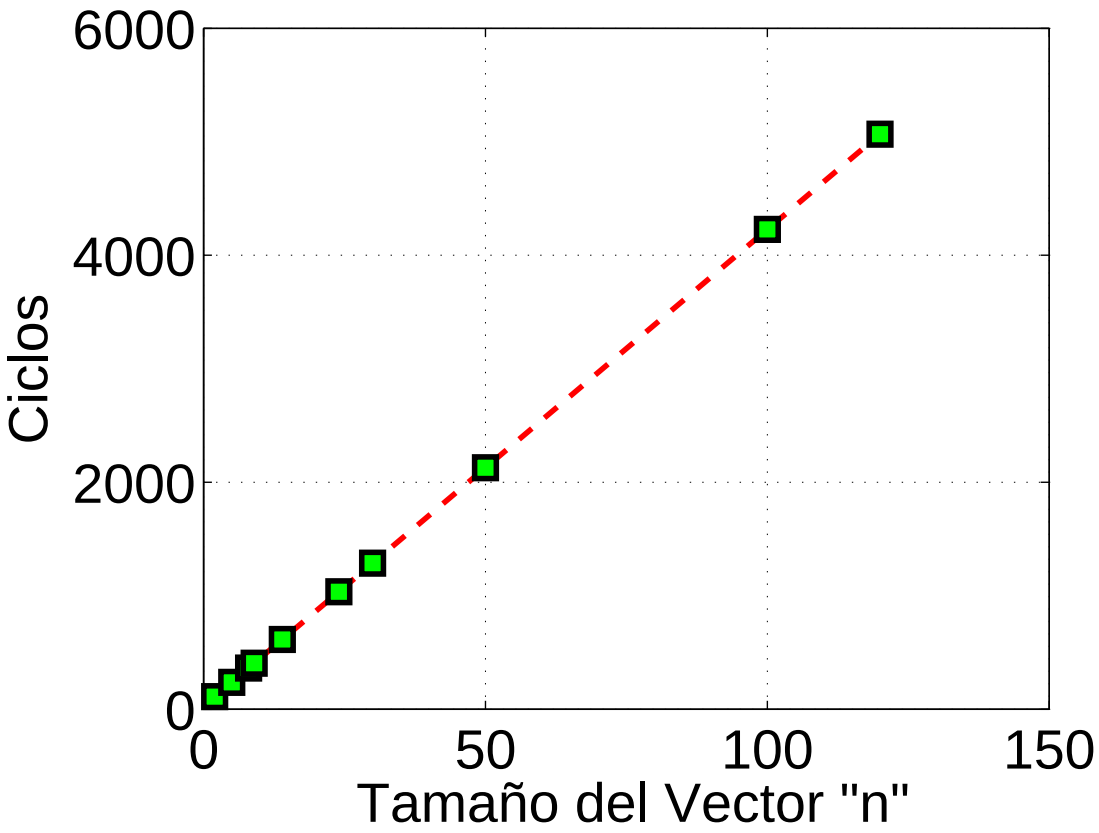
Algoritmo sobre SISD

1. Cargar la dirección base de la ROM donde se encuentran los datos a sumar.
2. Guardarlo en una dirección de RAM disponible (llamaremos a esta posición "*addr_ptr*").
3. Cargar el valor 8 en el acumulador. (se quiere hacer la sumatoria de 1 a 8)
4. Guardar en una dirección de RAM disponible (llamaremos a esta posición "*cantidad*")
5. Limpiar el acumulador.
6. Guardar en una dirección de RAM disponible (llamaremos a esta posición "*resultado_parcial*").

Aquí termina la inicialización.

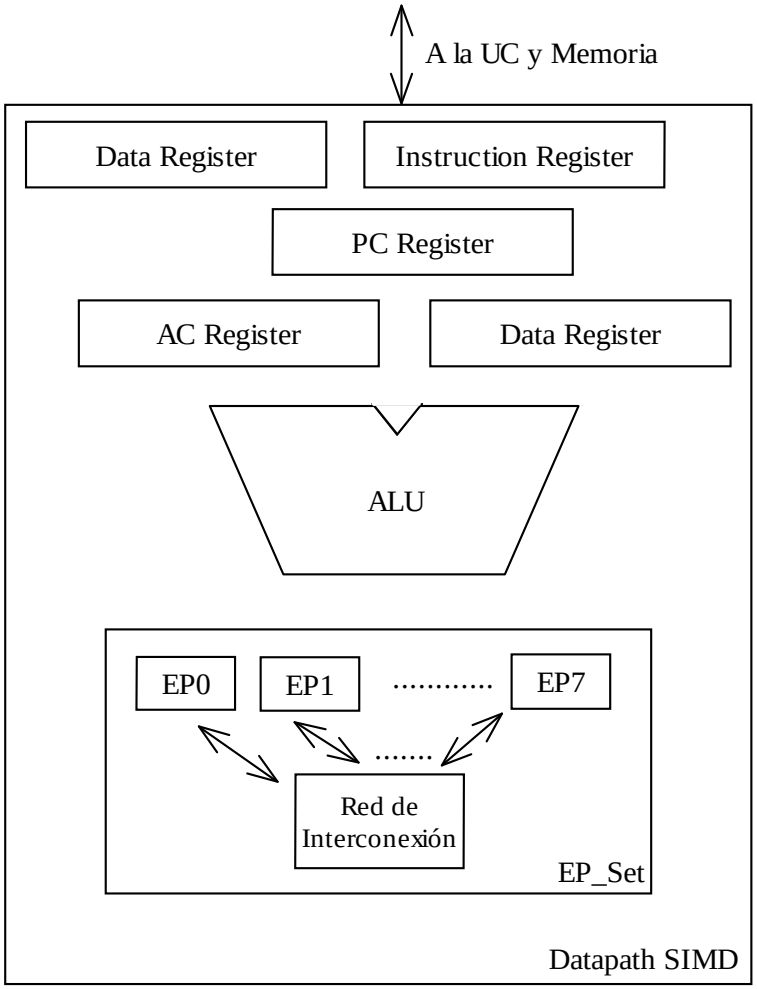
7. Cargar "*resultado_parcial*" en el acumulador.
8. Sumar el registro acumulador con lo apuntado por "*addr_ptr*" (suma indirecta)
9. Guardar el resultado en "*resultado_parcial*"
10. Incrementar "*addr_ptr*"
11. Decrementar cantidad y saltar la próxima instrucción si es cero.
12. Salto incondicional a 7
13. Terminar el programa.

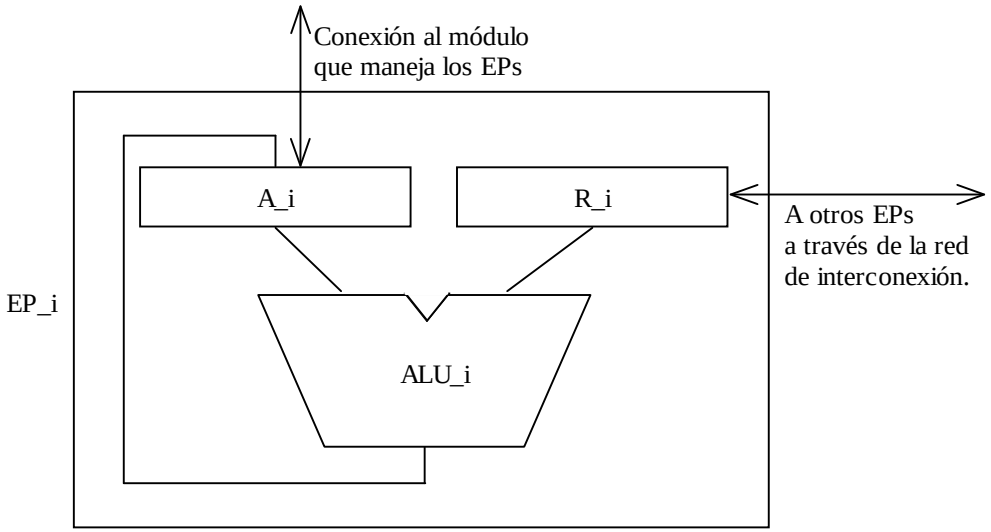
Rendimiento en SISD



Tamaño Vector	Ciclos necesarios
2	111
5	237
8	363
9	405
14	615
24	1035
30	1287
50	2127
100	4227
120	5067

Computadora SIMD





Algoritmo

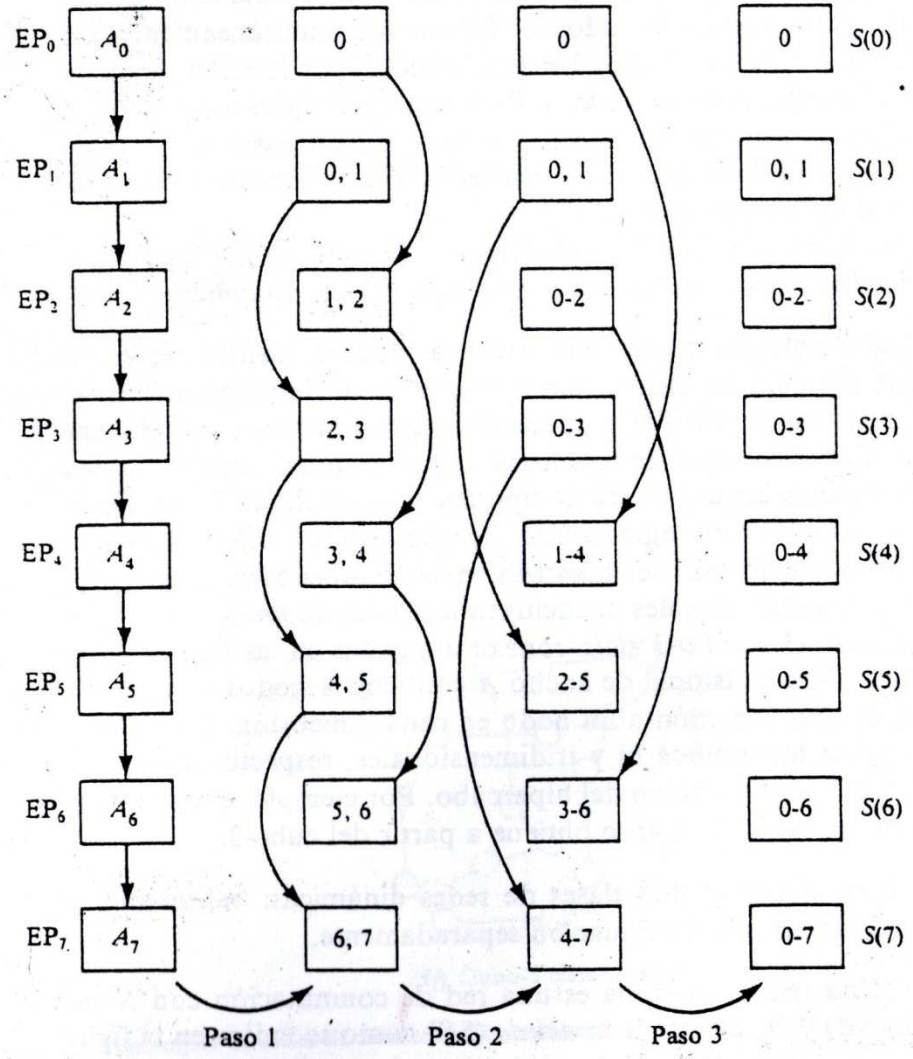


Figura 5.3 Cálculo de las sumas $S(k) = \sum_{i=0}^{k} A_i$, $k = 0, 1, \ldots, 7$ en una máquina SIMD.

Algoritmo sobre SIMD

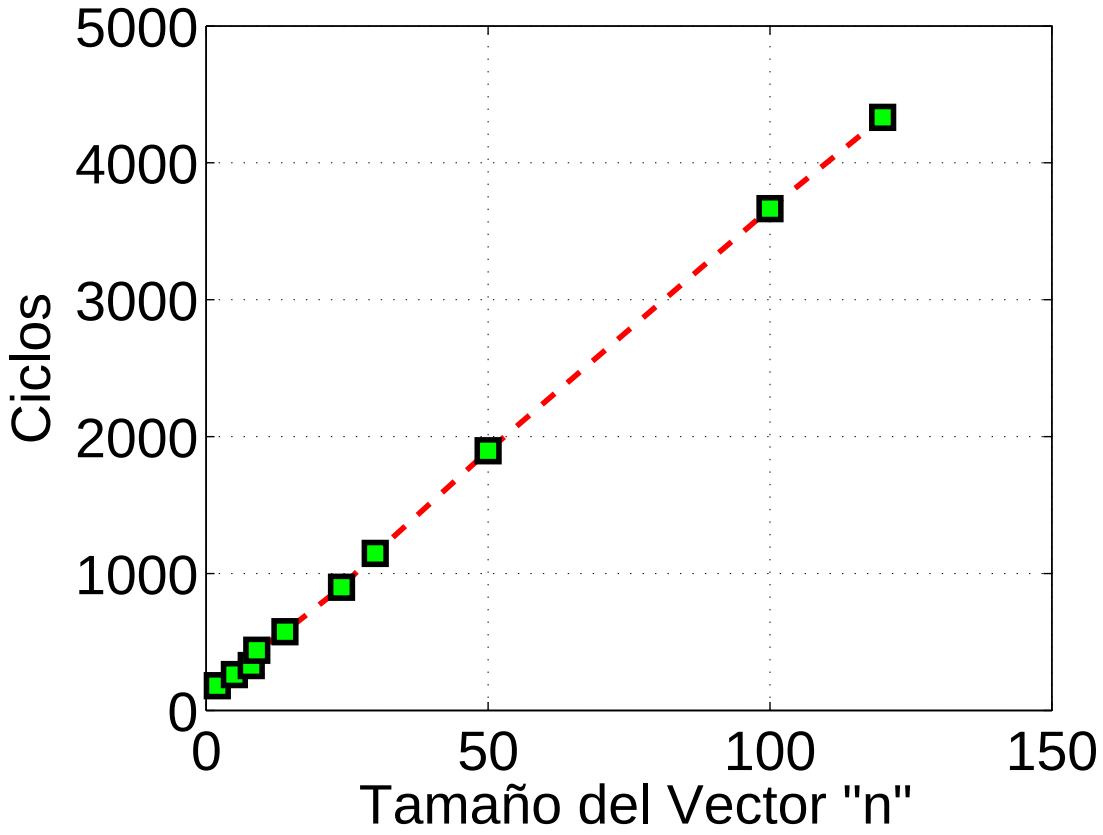
1. Limpiar el acumulador
2. Guardar en una dirección de RAM disponible (llamaremos a esta posición “resultado_parcial”).
3. Cargar la dirección base de la ROM donde se encuentran los datos a sumar.
4. Guardarlo en una dirección de RAM disponible (llamaremos a esta posición “addr_ptr”)
5. Cargar el valor 9 en el acumulador (cantidad de operandos +1)
6. Guardar en una dirección de RAM disponible (llamaremos a esta posición “cantidad”)
7. Decrementar “cantidad” y saltar la próxima instrucción si es <= 0.
8. Halt, termino de computar.
9. Cargar en el acumulador lo apuntado por la dirección “addr_ptr”
10. Guardarlo en EP0
11. Incrementar “addr_ptr”
12. Decrementar “cantidad” y saltar la próxima instrucción si es <= 0.
13. salto incondicional al paso 47 (principio del algoritmo).
14. Cargar en el acumulador lo apuntado por la dirección “addr_ptr”
15. Guardarlo en EP1. (Los mismos pasos son necesarios para EP2 a EP6)
41. Incrementar “addr_ptr”
42. Decrementar “cantidad” y saltar la próxima instrucción si es <= 0.
43. salto incondicional al principio del algoritmo.
44. Cargar en el acumulador lo apuntado por la dirección “addr_ptr”
45. Guardarlo en EP7
46. Incrementar “addr_ptr”.

FIN DE INICIALIZACION DE EP’s.

Algoritmo sobre SIMD (Cont.)

47. Copiar el registro A en el Registro R de todos los EPs
48. Hacer el primer movimiento por la red de interconexión de EP0 a EP7.
49. Sumar en EP1 a EP7.
50. Copiar el registro A en el Registro R en EP1 a EP7
51. Hacer el segundo movimiento por la red de interconexión de EP0 a EP7
52. Sumar en EP2 a EP7.
53. Copiar el registro A en el Registro R en EP2 a EP7
54. Hacer el tercer movimiento por la red de interconexión de EP0 a EP7
55. Sumar en EP4 a EP7.
56. Copiar el resultado de EP7 en el acumulador.
57. Guardar el resultado en “resultado_parcial”.
58. Limpiar el acumulador.
59. Cargar el acumulador en todos los EP’s (esto es para resetearlos)
60. Salto incondicional al paso 7 para buscar más datos si los hubiera.

Rendimiento sobre SIMD



Tamaño Vector	Ciclos necesarios
2	180
5	261
8	328
9	439
14	574
24	900
30	1146
50	1896
100	3666
120	4332

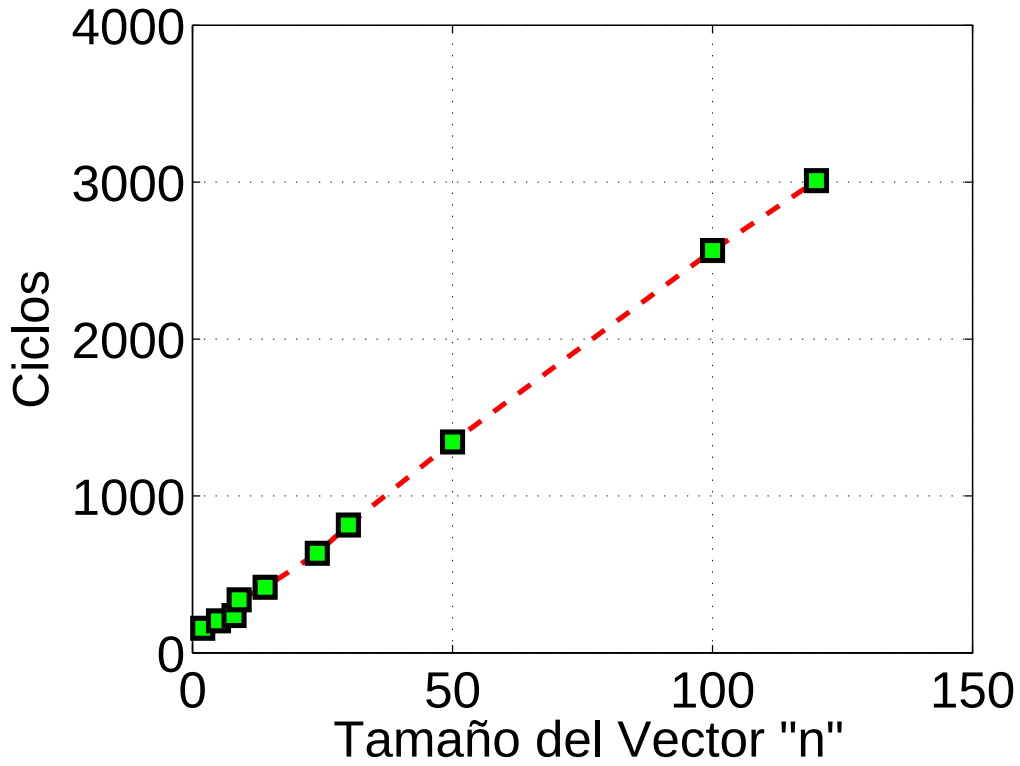
Algoritmo sobre SIMD Mejorado

Pasos 1 a 8 igual que en el algoritmo anterior.
9. Guardar en EP0, lo apuntado por *addr_ptr* e incrementarlo.
10. Decrementar "*cantidad*" y saltar la próxima instrucción si es <= 0.
11. Salto incondicional al paso 31 (principio del algoritmo).
12. Guardar en EP1, lo apuntado por *addr_ptr* e incrementarlo.
13. Decrementar "*cantidad*" y saltar la próxima instrucción si es <= 0.
14. Salto incondicional al paso 31.
    (Los pasos 15 a 29 son similares para EP2 a EP6)
30. Guardar en EP7, lo apuntado por *addr_ptr* e incrementarlo.

FIN DE INICIALIZACION DE EP's.

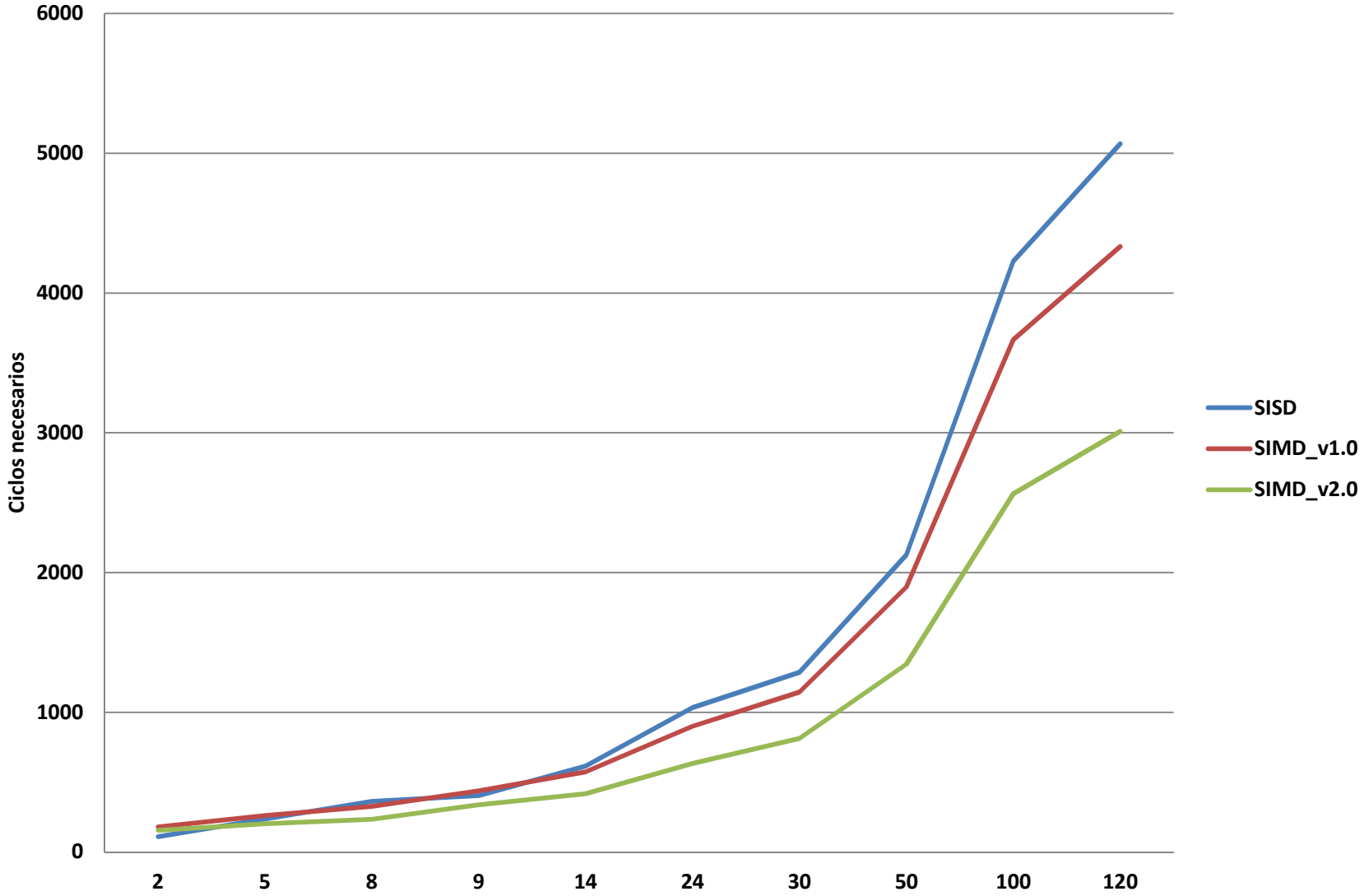
Los pasos 31 a 44 son los mismos que los correspondientes al algoritmo generalizado descripto previamente.

Rendimiento en SIMD Mejorado



Tamaño Vector	Ciclos necesarios
2	156
5	204
8	238
9	338
14	418
24	634
30	814
50	1344
100	2564
120	3010

Comparación de Rendimiento



Conclusiones

- Se desarrolló una computadora de tipo SISD.
- Se demostró la factibilidad de crear una computadora SIMD a partir de una SISD.
- Se detalló cada uno de los algoritmos utilizados.
- Se demostraron los aumentos de performance en cada una de las versiones.

Preguntas?