

## SIMD:

Arquitectura escalar (SISD)	Arquitectura vectorial (SIMD)
No tiene registros vectoriales (solo tiene registros escalares (o de propósito general))	Sí tiene registros vectoriales (también tiene registros escalares)
No tienen instrucciones vectoriales <b>Ejemplo:</b> $Y = a \cdot X + Y$ // X e Y son vectores de 64 números en punto flotante doble precisión. <pre> loop:  l.d    \$f0,a(\$sp)      :load scalar a         addiu \$t0,\$s0,#512   :upper bound of what to load         l.d    \$f2,0(\$s0)    :load x(i)         mul.d   \$f2,\$f2,\$f0   :a x x(i)         l.d    \$f4,0(\$s1)    :load y(i)         add.d   \$f4,\$f4,\$f2   :a x x(i) + y(i)         s.d    \$f4,0(\$s1)    :store into y(i)         addiu  \$s0,\$s0,#8     :increment index to x         addiu  \$s1,\$s1,#8     :increment index to y         subu   \$t1,\$t0,\$s0    :compute bound         bne    \$t1,\$zero,loop :check if done           </pre> Se ejecutarían cerca de 600 instrucciones	Sí tienen instrucciones vectoriales <b>Ejemplo:</b> $Y = a \cdot X + Y$ // X e Y son vectores de 64 números en punto flotante doble precisión. <pre> loop:  l.d    \$f0,a(\$sp)      :load scalar a         lv     \$v1,0(\$s0)     :load vector x         mulvs.d \$v2,\$v1,\$f0   :vector-scalar multiply         lv     \$v3,0(\$s1)     :load vector y         addv.d \$v4,\$v2,\$v3    :add y to product         sv     \$v4,0(\$s1)     :store the result           </pre> Se ejecutan 6 instrucciones
No hay paralelismo a nivel de datos	Sí hay paralelismo a nivel de datos - Extensiones SIMD (ALU particionada o subword) - Procesadores vectoriales (unidad funcional pipelined (lane))

# C code	# Scalar Code	# Vector Code
<pre> for (i=0; i&lt;64; i++)   C[i] = A[i] + B[i];           </pre>	<pre> LI R4, 64 loop:   L.D F0, 0(R1)   L.D F2, 0(R2)   ADD.D F4, F2, F0   S.D F4, 0(R3)   DADDIU R1, 8   DADDIU R2, 8   DADDIU R3, 8   DSUBIU R4, 1   BNEZ R4, loop           </pre>	<pre> LI VLR, 64 LV V1, R1 LV V2, R2 ADDV.D V3, V1, V2 SV V3, R3           </pre>

## SIMD VS VECTORIAL

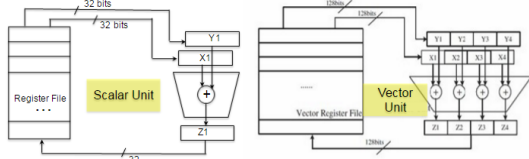
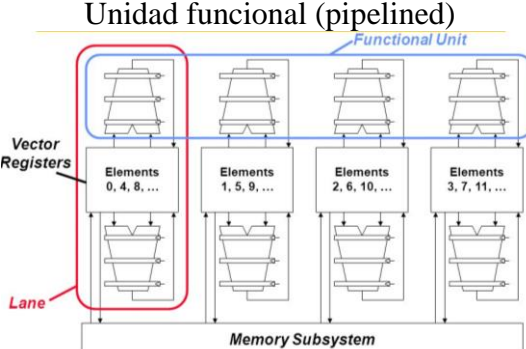
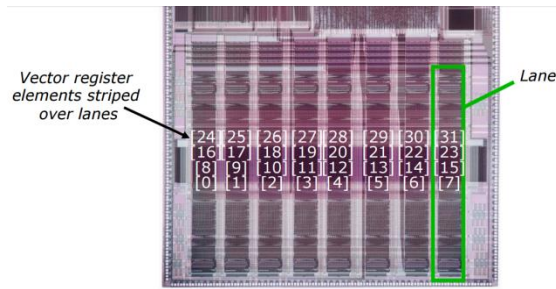
Procesadores con extensiones SIMD	Procesadores vectoriales
Registros vectoriales	
Instrucciones vectoriales	
ALU particionada o paralelismo subword 	Unidad funcional (pipelined) 
Opcode (código de la instrucción)	

Figure 1 illustrates a 4x4 mesh network structure. The network is composed of four stages, each represented by a trapezoidal block. The stages are labeled C[8], C[9], C[10], and C[11] on the top, and C[4], C[5], C[6], and C[7] on the bottom. Arrows indicate the flow of data from top to bottom. Below each stage, a label C[0], C[1], C[2], and C[3] respectively, with arrows pointing to them.



Año/computador	Ext. SIMD	Nº inst. añad.	Nº reg. Vect.	P. Subword		
				Nº op	Bit dato	Bit reg.
1996 Pentium MMX	MMX	57	8 MM0 – MM7	8	8	64
				4	16	64
				2	32	64
1999 Pentium III	SSE	70	8 XMM0 – XMM7	4	32	128
2001 Pentium IV	SSE2	144	8 XMM0 – XMM7	16	8	128
				8	16	128
				4	32	128
				2	64	128
2003 Opteron	AMD64	Base	16 XMM0 – XMM15	4	32	128
				2	64	128
2004 Pentium IV Prescott	EM64T	Base	16 XMM0 – XMM15	4	32	128 (1)
				2	64	128 (2)
2011 Sandy Bridge	AVX	128	16 YMM0 – YMM15	8	64	256 (1)
				4	64	256 (2)

(2). VADDPD: Suma vectorial de doble precisión.

[illegible]