Appendix G
Vector Processors in More Depth

Processor (year)	Vector clock rate (MHz)	Vector registers	Elements per register (64-bit elements)	Vector arithmetic units	Vector load-store units	Lanes
Cray-1 (1976)	80	8	64	6: FP add, FP multiply, FP reciprocal, integer add, logical, shift	1	1
Cray X-MP (1983) Cray Y-MP (1988)	118 166	8	64	8: FP add, FP multiply, FP reciprocal, integer add, 2 logical, shift, population count/parity	2 loads 1 store	1
Cray-2 (1985)	244	8	64	5: FP add, FP multiply, FP reciprocal/sqrt, integer add/shift/population count, logical	1	1
Fujitsu VP100/ VP200 (1982)	133	8–256	32–1024	3: FP or integer add/logical, multiply, divide	2	1 (VP100) 2 (VP200)
Hitachi S810/S820 (1983)	71	32	256	4: FP multiply-add, FP multiply/divide-add unit, 2 integer add/logical	3 loads 1 store	1 (S810) 2 (S820)
Convex C-1 (1985)	10	8	128	2: FP or integer multiply/divide, add/logical	1	1 (64 bit) 2 (32 bit)
NEC SX/2 (1985)	167	8+32	256	4: FP multiply/divide, FP add, integer add/ logical, shift	1	4
Cray C90 (1991) Cray T90 (1995)	240 460	8	128	8: FP add, FP multiply, FP reciprocal, integer add, 2 logical, shift, population count/parity	2 loads 1 store	2
NEC SX/5 (1998)	312	8+64	512	4: FP or integer add/shift, multiply, divide, logical	1	16
Fujitsu VPP5000 (1999)	300	8–256	128–4096	3: FP or integer multiply, add/logical, divide	1 load 1 store	16
Cray SV1 (1998) SV1ex (2001)	300 500	8	64 (MSP)	8: FP add, FP multiply, FP reciprocal, integer add, 2 logical, shift, population count/parity	1 load-store 1 load	2 8 (MSP)
VMIPS (2001)	500	8	64	5: FP multiply, FP divide, FP add, integer add/ shift, logical	1 load-store	1
NEC SX/6 (2001)	500	8+64	256	4: FP or integer add/shift, multiply, divide, logical	1	8
NEC SX/8 (2004)	2000	8+64	256	4: FP or integer add/shift, multiply, divide, logical	1	4
Cray X1 (2002) Cray XIE (2005)	800 1130	32	64 256 (MSP)	3: FP or integer, add/logical, multiply/shift, divide/square root/logical	1 load 1 store	2 8 (MSP)

Figure G.1 Characteristics of several vector-register architectures. If the machine is a multiprocessor, the entries correspond to the characteristics of one processor. Several of the machines have different clock rates in the vector and scalar units; the clock rates shown are for the vector units. The Fujitsu machines' vector registers are configurable: The size and count of the 8K 64-bit entries may be varied inversely to one another (e.g., on the VP200, from eight registers each 1K elements long to 256 registers each 32 elements long). The NEC machines have eight foreground vector registers connected to the arithmetic units plus 32 to 64 background vector registers connected between the memory system and the foreground vector registers. Add pipelines perform add and subtract. The multiply/divide-add unit on the Hitachi S810/820 performs an FP multiply or divide followed by an add or subtract (while the multiply-add unit performs a multiply followed by an add or subtract). Note that most processors use the vector FP multiply and divide units for vector integer multiply and divide, and several of the processors use the same units for FP scalar and FP vector operations. Each vector load-store unit represents the ability to do an independent, overlapped transfer to or from the vector registers. The number of lanes is the number of parallel pipelines in each of the functional units as described in Section G.4. For example, the NEC SX/5 can complete 16 multiplies per cycle in the multiply functional unit. Several machines can split a 64-bit lane into two 32-bit lanes to increase performance for applications that require only reduced precision. The Cray SV1 and Cray X1 can group four CPUs with two lanes each to act in unison as a single larger CPU with eight lanes, which Cray calls a Multi-Streaming Processor (MSP).

Unit	Start-up overhead (cycles)
Load and store unit	12
Multiply unit	7
Add unit	6

Figure G.2 Start-up overhead.

Convoy	Starting time	First-result time	Last-result time
1. LV	0	12	$11+n$
2. MULVS.D LV	$12+n$	$12+n+12$	$23+2n$
3. ADDV.D	$24+2n$	$24+2n+6$	$29+3n$
4. SV	$30+3n$	$30+3n+12$	$41+4n$

Figure G.3 Starting times and first- and last-result times for convoys 1 through 4. The vector length is *n*.

Operation	Start-up penalty
Vector add	6
Vector multiply	7
Vector divide	20
Vector load	12

Figure G.4 Start-up penalties on VMIPS. These are the start-up penalties in clock cycles for VMIPS vector operations.

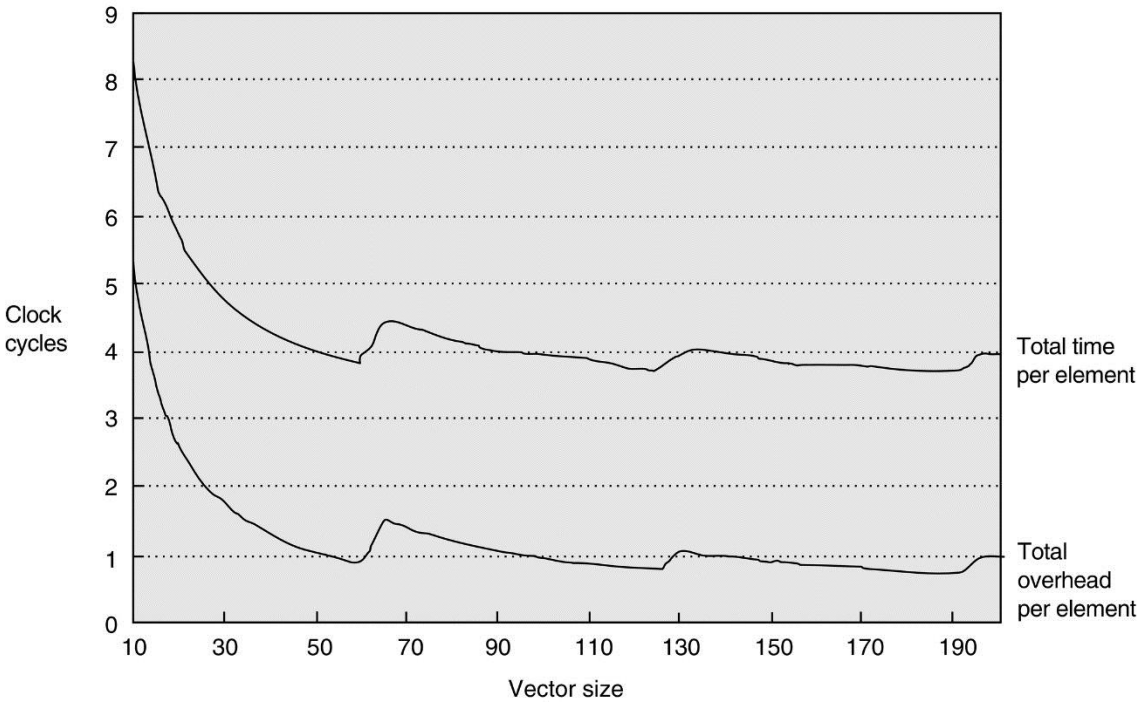

Figure G.5 The total execution time per element and the total overhead time per element versus the vector length for the example on page F-6. For short vectors, the total start-up time is more than one-half of the total time, while for long vectors it reduces to about one-third of the total time. The sudden jumps occur when the vector length crosses a multiple of 64, forcing another iteration of the strip-mining code and execution of a set of vector instructions. These operations increase T_n by $T_{loop} + T_{start}$.

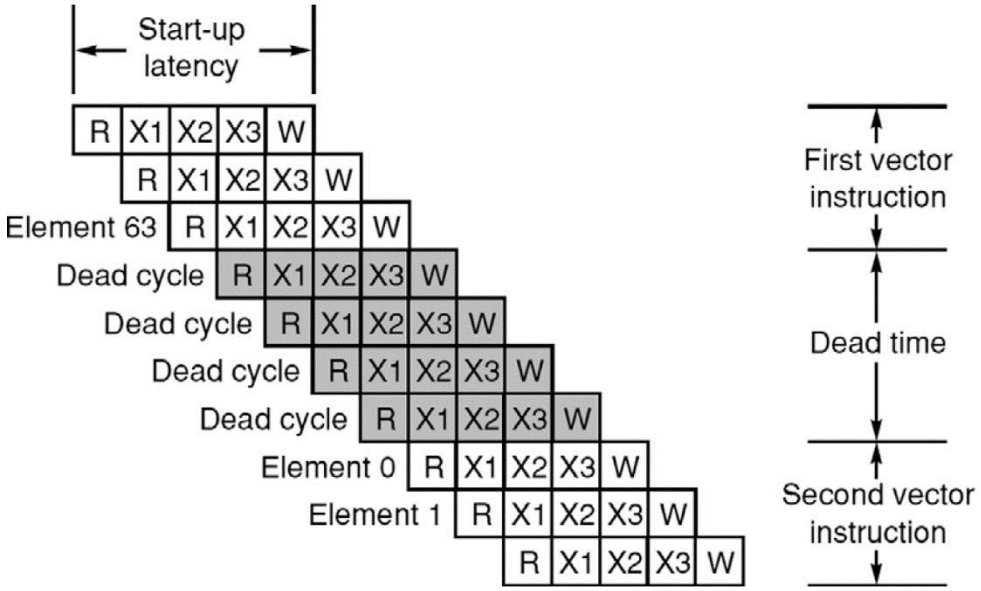

Figure G.6 Start-up latency and dead time for a single vector pipeline. Each element has a 5-cycle latency: 1 cycle to read the vector-register file, 3 cycles in execution, then 1 cycle to write the vector-register file. Elements from the same vector instruction can follow each other down the pipeline, but this machine inserts 4 cycles of dead time between two different vector instructions. The dead time can be eliminated with more complex control logic. (Reproduced with permission from Asanovic [1998].)

	Bank							
Cycle no.	0	1	2	3	4	5	6	7
0		136						
1		Busy	144					
2		Busy	Busy	152				
3		Busy	Busy	Busy	160			
4		Busy	Busy	Busy	Busy	168		
5		Busy	Busy	Busy	Busy	Busy	176	
6			Busy	Busy	Busy	Busy	Busy	184
7	192			Busy	Busy	Busy	Busy	Busy
8	Busy	200			Busy	Busy	Busy	Busy
9	Busy	Busy	208			Busy	Busy	Busy
10	Busy	Busy	Busy	216			Busy	Busy
11	Busy	Busy	Busy	Busy	224			Busy
12	Busy	Busy	Busy	Busy	Busy	232		
13		Busy	Busy	Busy	Busy	Busy	240	
14			Busy	Busy	Busy	Busy	Busy	248
15	256			Busy	Busy	Busy	Busy	Busy
16	Busy	264			Busy	Busy	Busy	Busy

Figure G.7 Memory addresses (in bytes) by bank number and time slot at which access begins. Each memory bank latches the element address at the start of an access and is then busy for 6 clock cycles before returning a value to the CPU. Note that the CPU cannot keep all 8 banks busy all the time because it is limited to supplying one new address and receiving one data item each cycle.

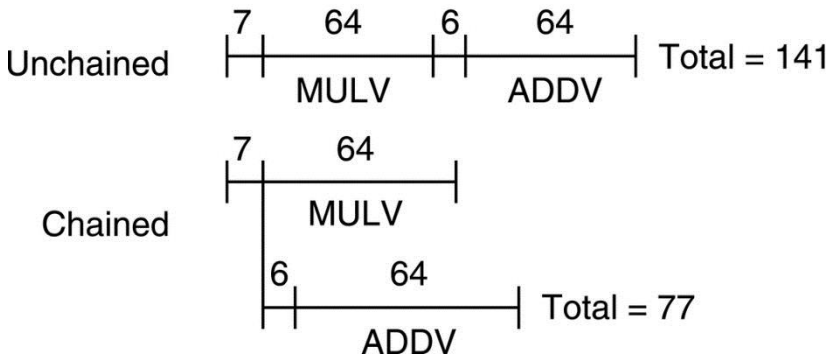

Figure G.8 Timings for a sequence of dependent vector operations `ADDV` and `MULV`, both unchained and chained. The 6- and 7-clock-cycle delays are the latency of the adder and multiplier.

Processor	Compiler	Completely vectorized	Partially vectorized	Not vectorized
CDC CYBER 205	VAST-2 V2.21	62	5	33
Convex C-series	FC5.0	69	5	26
Cray X-MP	CFT77 V3.0	69	3	28
Cray X-MP	CFT V1.15	50	1	49
Cray-2	CFT2 V3.1a	27	1	72
ETA-10	FTN 77 V1.0	62	7	31
Hitachi S810/820	FORT77/HAP V20-2B	67	4	29
IBM 3090/VF	VS FORTRAN V2.4	52	4	44
NEC SX/2	FORTRAN77 / SX V.040	66	5	29

Figure G.9 Result of applying vectorizing compilers to the 100 FORTRAN test kernels. For each processor we indicate how many loops were completely vectorized, partially vectorized, and unvectorized. These loops were collected by Callahan, Dongarra, and Levine [1988]. Two different compilers for the Cray X-MP show the large dependence on compiler technology.

LV V1,Rx	MULVS.D V2,V1,F0	Convoy 1: chained load and multiply
LV V3,Ry	ADDV.D V4,V2,V3	Convoy 2: second load and add, chained
SV Ry,V4		Convoy 3: store the result

Figure G.10 The inner loop of the DAXPY code in chained convoys.

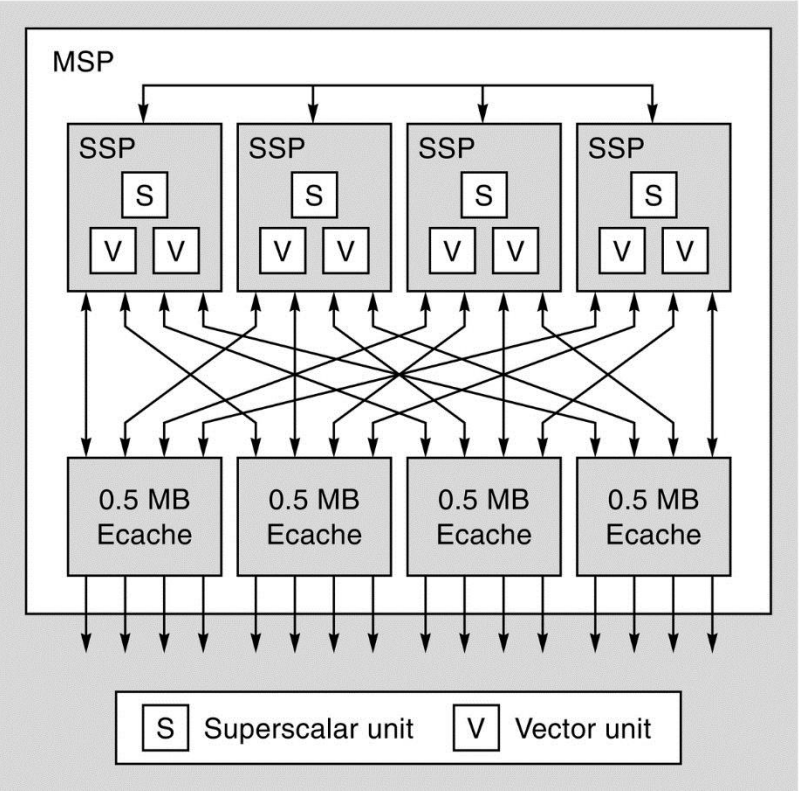

Figure G.11 Cray MSP module. (From Dunnigan et al. [2005].)

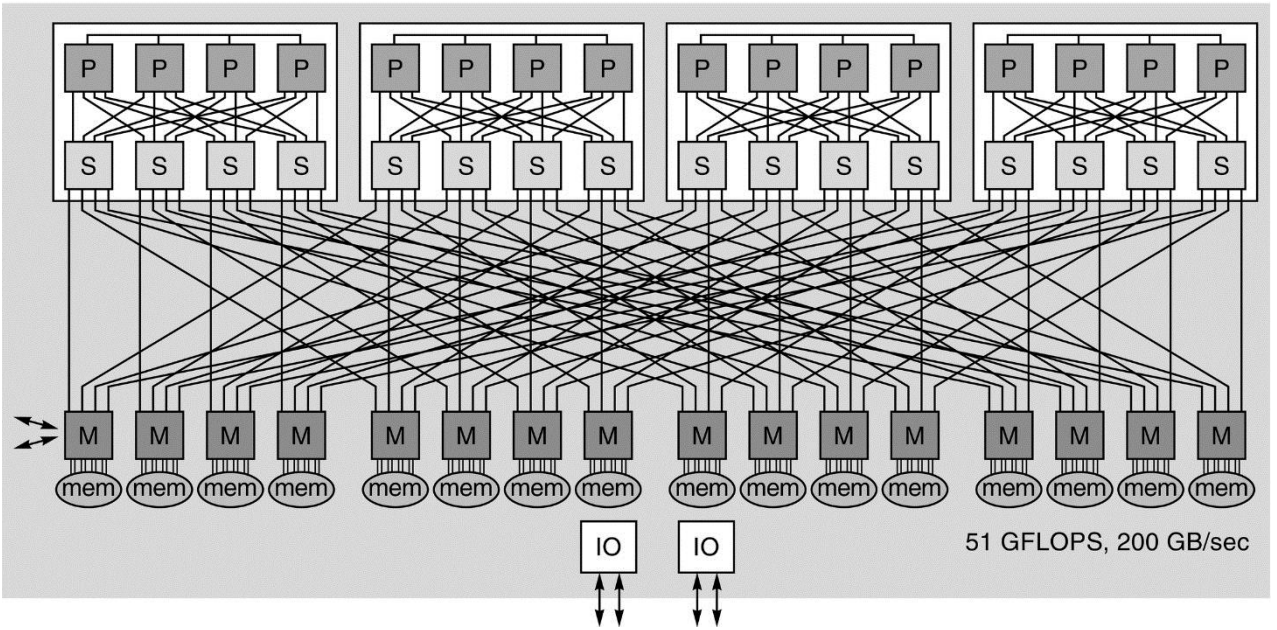

Figure G.12 Cray X1 node. (From Tanqueray [2002].)