VLSI Architecture Design Course (048853) Final Exam July 14th, 2004 Electrical engineering Department

Student name:		Student nu	ımber:
This exam contains TWO The exam duration is 2:30 Please fill the answers Ol	0 hours.	rms.	
Please explain or provide	a formula for e	each comput	ation!
TAKE YOUR TIME, READ THE CONTENT AND ONL			
Good luck!			
	Q1		7
	Q2		
	Total		

Question 1 (50%) Cache Power Management

Nowadays, cache power consumption mainly consists of leakage power, caused by relatively low activity for many transistors. Several mechanisms exist that reduce the leakage (static) power.

We define the *Live Time* of a block (cache line) as the time between its allocation until its last access before eviction.

We define the Dead Time of a block (cache line) as the time between its last access until the block is evicted from the cache.

These definitions are explained in Figure 1.1:



Figure 1.1

We define the *dead time ratio* as the dead time divided by the sum of the dead time and the live time of the block.

Cache blocks are usually dead rather than alive, as can be seen in Figure 1.2.

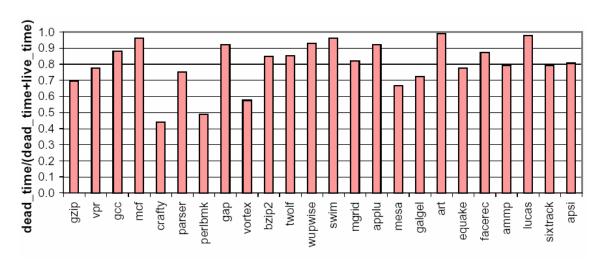


Figure 1.2 – Dead Time Ratio for various benchmarks

^{*}This question is based on the following:

^[1] S. Kaxiras et al., "Cache Decay: Exploiting Generational Behavior to Reduce Cache Leakage Power", Proc. ISCA 2001

^[2] M. Powell et al., "Gated Vdd: A Circuit Technique to Reduce Leakage in Deep-Submicron Cache Memories", Proc. ISLPED 2000

^[3] K. Fkautner et al., "Drowsy Caches: Simple Techniques for Reducing Leakage Power", Proc. ISCA 2002

QUESTIONS:

A. How does the miss rate affect the dead time ratio? Exp	olain.
---	--------

B. Describe how each of the following parameters affects the dead time ratio. Use the following table as a supporting tool.

	Miss rate (+=increase)	Replacement rate (+=increase)	Dead time ratio (+=increase)
Increasing cache size			
Increasing block size on the expense of the number of sets			

Explanations:

1. Increasing cache size

2. Increasing cache block size on the expense of the number of sets

C. In order to conserve power, "dead" cache blocks (cache lines) should be cut off from power (i.e. content is erased). This method is called "Cache Decay". An example of one bit of decay cache controlled by the gated-VDD control signal is shown in figure 1.3.

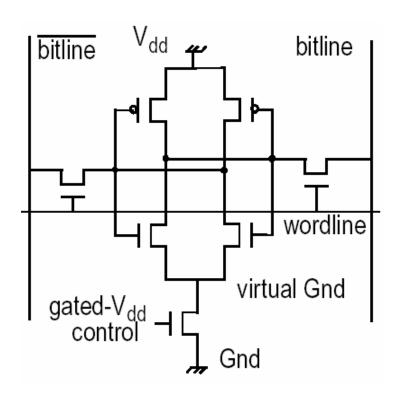


Figure 1.3 – Decay cache bit circuit

One of the ways to detect when a cache block is "dead" is by counting the number of cycles that have passed since the last access to that block.

1. How will you decide when to cut off a block from power?	
2. What are the disadvantages of using this method?	
3. What happens in case a block is not really "dead"? What are the implications?	
4. What can the software developer do in order to avoid eviction of liv blocks?	'e

5. Can you think of a way to reduce the cost of counting?

D. "Drowsy Cache" is a method that puts suspected dead blocks to sleep by lowering their supply voltage. When drowsed blocks are accessed, the cache needs to first wake up the drowsed blocks by increasing their supply voltage. This wake up process takes a few cycles. The number of cycles is called the wake-up penalty. An example of a drowsy line is shown in Figure 1.3.

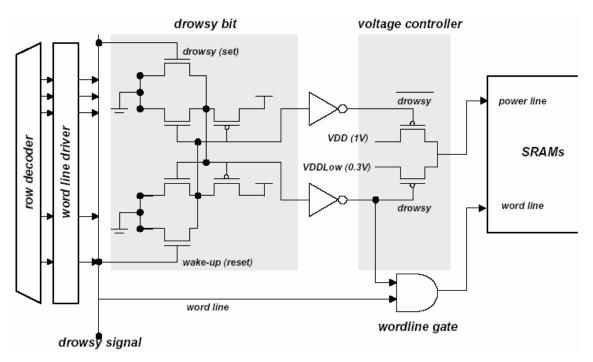


Figure 1.4 – Drowsy cache line circuit

1. What are the advantages of using Drowsy Cache over Cache Decay	y
2. What are the disadvantages of using Drowsy Cache over Cache Decay?	
3. Can you think of a way to use a drowsy cache without using a counter for each line?	

Question 2 (50%) Processor Core Performance

In this question deals with the execution of instructions within a processor core accounting for simple cache and branch prediction effects.

Assume the following 2 processor core configurations:

	In-order core	Out-of-order core
Fetch bandwidth	Restart a new fetch in the Correctly predicted taken	•
Execution bandwidth	Unlir	nited
L1 Instruction cache	Perfect (100% hit)
L1 data cache	32KB, 2 Ways, 64 Bytes	s lines, writeback cache
branch predictor	Static prediction: Forward	counter (bi-modal) l: strongly not taken, rd: strongly taken
Pipeline depth	4 (F/D/E/W) Fetch/Decode/ Execute/Writeback	6 (F/D/R/S/E/W) Fetch/Decode/Rename/ Schedule/Execute/Writeback
ALU instruction execution latency	1 cy	ycle
Load Instruction execution latency	16 outstanding cache/mer A memory element that is	available one cycle after the

We will examine the performance of our cores on the following 2 code fragments (A and B).

The programs sum a subset of array element based on some criteria. "int" is 4 bytes (32 bit element) in this code.

	(A) Simple "sum"	(B) Conditional "sum"
C code	int a[N], i, sum; sum = 0; for (i=0; i <n; i++)<br="">sum += a[i];</n;>	int a[N], i, sum; sum = 0; for (i=0; i <n; i++)<br="">if (a[i]>0) sum += a[i];</n;>
Assembly code	mov 0, Rsum mov 0, Ri L1: load a[Ri], R1 add R1, Rsum add 1, Ri, Ri cmp Ri, N jle L1 END:	mov 0, Rsum mov 0, Ri L1: load a[Ri], R1 cmp R1, 0 jle L2 add R1, Rsum L2: add 1, Ri, Ri cmp Ri, N jle L1 END:

QUESTIONS:

Assume that:

- I. L1 cache is empty before we start.
- II. N = 1,000,000
- III. For simplicity, ignore the 1st two assembly instructions (sum=0; i=0).
- IV. For simplicity, assume that there is an infinite instruction window.
- V. You can ignore edge effects it is OK if an answer is within 0.1% of the accurate result.
- A. Let us look at code portion (A).
 We follow its execution until it reaches the label END.
 - 1. How many cache misses are in this piece of code? Explain.

2. How many branch mispredictions are in this piece of code? Explain.

3. For the <u>in-order</u> core configuration: (The solution is provided for your convenience, please see also the instruction/cycle chart marked A.I that appears later.)

How many cycles will it take to execute? 5437500

What is the IPC? <u>0.92</u>

Explain.

Account for dependencies, cache misses and branch mispredictions.

4	Repeat for the Out-of-order core configuration:
	How many cycles will it take to execute?
	What is the IPC?
	Explain.
	Account for dependencies, cache misses and branch mispredictions
	For your convenience, you many use the instruction/cycle chart marked A.O that appears later.

- B. We now perform a similar exercise on the code portion (B). Assume:
 - I. No cache misses.
 - II. The array "a" is set so For all i, a[i]=1, except every 5th element starting from i=1, then it is -1. Or more formally: a[i]=-1 if i is of the form 5*k+1 (1, 6, 11...).

$$a[i] = \begin{cases} 1 & i \neq 5k+1 \\ -1 & i = 5k+1 \end{cases}$$

1. How many branches are taken in this code fragment? Explain.

JLE L1: Taken ______, Not Taken ______

JLE L2: Taken _____, Not Taken _____

2. How many instructions were committed (retired)? ______Explain.

3.	How many	branches were mispredicted in this piece of code? Explain.
	JLE L1:	Predicted Not Taken, actually Taken Predicted Taken, actually Not Taken
	JLE L2:	Predicted Not Taken, actually Taken Predicted Taken, actually Not Taken
4.	How many What is the Explain. Account for your co	order core configuration: cycles it will take to execute?e IPC?e or dependencies and branch mispredictions. convenience, you many use the instruction/cycle chart that appears later.
5.	How many What is the Explain. Account for your co	the Out-of-order core configuration: cycles it will take to execute? e IPC? or dependencies and branch mispredictions. convenience, you many use the instruction/cycle chart that appears later.

C.	What have you learnt from this example about the difference between in- order and out-of-order potential.
	1. Exploiting instruction level parallelism.
	2. Sensitivity to cache misses.
	3. Sensitivity to the quality of the branch predictor.

D. It was suggested to speed up this code by using an out-of-order Simultaneous Multi Threading (SMT) machine. The code is split into 2 threads, implemented by 2 distinct functions (f1 and f2). The first thread executes the function f1 that performs the first 500,000 iterations. The second thread executes the function f2 that performs the last 500,000 iterations. On each cycle the processor can fetch instructions from one thread only (up to 16 consecutive instructions). The processor can decide from which thread it can fetch next. 1. What is the advantage of using SMT in these cases: For code portion A? For code portion B? 2. For code portion B, what will be an ideal fetch policy (from which thread to fetch next) for this code, so to maximize performance? 3. For code portion B, estimate the performance benefit in such case.

A.I: Sheet for code portion (A) in-order

		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	3 2	1 25	26	2	7 28	B 2	9 3	0 3	1 3	2 3	3 3	4 3	5 3	6 3	37 3	8 3	9 4	10 4	1 42	2 43	44	4:	5 4	6 4	7 4	8 4	9 50	51	52	53	54	55	56	57	58	59	60
1 load	a[Ri], R1		D																							T			T			Ť							T	T								T	Ť				Т	T	Т			T	П	T	П
2 add														Е															T	T	T		T			T		T	T		T					T	T	T				Ť	T	T	Т				П	T	П
3 add		F	D											Е	w											Т			T		T	T	Ī			T			T	T						Ī	Ī	T					П	Г	Г				П	T	П
	Ri, N		D													w										Ī			T		Ī	T						Ť	T																				П	T	П
5 jle	L1	F	D													Е	w												T			T							T	T														T	Г				iΠ	T	П
	a[Ri], R1		F	D												Е	Е	Ε	w					Г							T		T	T				T			Ī					T	T	Ī							П				П	T	П
	R1, Rsum	, [F	D															Е	w						Т			T	Ī	T	T	T			T	T	T	T	T						Ī	Ī	T					Т	Г	Г			\Box	П	T	П
	1, Ri, Ri	Г	F	D															Е	w				Г			Г		T	T	T	T	T	T	T		T	T	T	T	T			T		Ī	T	T					Т	Т	Г			\Box	П	T	П
9 cmp	Ri, N	Г	F	D																Е	w					Т			T		Ī								T														П	Г					П	П	П
10 jle	L1	Г	F	D																	Ε	w				Т			Т		Ī	T							T									T					П	П	Г				П	П	П
11 load	a[Ri], R1			F	D																Ε	Е	Ε	W	,	Г			T			T							T	T								T					П	П	П			T	П	П	П
12 add	R1, Rsum 1, Ri, Ri	1		F	D																				W	1	Г				T																	T											П	П	П
13 add	1, Ri, Ri	Г		F	D																			E	W		Г				T		T	T				T			T					Г	T	T											ıΠ		
14 cmp	Ri, N	Г		F	D																			Г	E	w	Г				T		T					T			T							T											iΠ		
15 jle	L1			F	D																					E	w																																		
16 load	a[Ri], R1				F	D																				E	Е	E	w	/	Т																														
	R1, Rsum	۱ 🗆			F	D																							E	١	v																						\mathbb{L}								
18 add	1, Ri, Ri				F	D																							E	١	V																														
19 cmp	Ri, N				F	D																								E	١	v																													
20 jle	Ri, N L1				F	D																									_	ΕV	٧																												
21 load	a[Ri], R1																																																				\mathbb{L}								
22 add	R1, Rsum	ı																																																			Ш								
23 add	R1, Rsum 1, Ri, Ri Ri, N																																																				Ш	L	L				Ш		Ш
24 cmp	Ri, N	L																																																			L	L	L				Ш		Ш
25 jle	L1																																				╙																퇶	L	┖				Ш		Ш
26 load	a[Ri], R1 R1, Rsum 1, Ri, Ri	L																																			┸																丄	L	L				Ш		Ш
27 add	R1, Rsum	L																											┸							╙		\perp	\perp	\perp	_												丄	L	上		Ш		Ш	Ш	Ш
28 add	1, Ri, Ri	L																																		╙	╙												╧				丄	L	L				Ш		Ш
29 cmp	Ri, N																																					1	4	\perp	_												丄	L	L				Ш	Ш	Ш
30 jle	L1																													╙			_	\perp			_	1	_		_							L	\perp				丄	丄	丄			_	Ш	Ш	Ш
31 load	a[Ri], R1	L	1_	Ш		$oxed{L}$	Ц		L					Ц			L		L	L		L	L	L	\perp	L	L	L	L	L	┸	┸	\perp	\perp		╙	\perp	\perp	\perp	\perp	1	\perp		L	L	L	┸	┸	\perp	\perp	\perp	\perp	丄	$oldsymbol{\perp}$	丄	_	Ш	ightharpoonup	Ш	Ц	Ш
32 add	R1, Rsum	L	1_	Ш		Ш	Ц		Ш			<u> </u>		Ш			L		L	L		L	L	L		L	L	L	L	L	L	1	╙	\perp		ļ	\perp	1	1	\perp	4	1			L	L	L	L	1	\perp	\perp	$oldsymbol{\perp}$	丄	丄	丄	1	Ш	_	Ш	\sqcup	Ш
33 add	1, Ri, Ri Ri, N L1	L	1_	Ш			Ш		L			<u> </u>		Ш			L		L				L	L	L	L	L		L	L	L	1	ļ	\downarrow		ļ	\perp	1	1	\perp	4	1			L	L	L	Ļ	1		_		丄	丄	丄	1	Ш	_	Ш	_	Ш
34 cmp	Ri, N																																\perp				\perp	↓	4	╙	↓												丄	Ļ	L		Ш		Ш	Ш	Ш
35 jle	L1	┸		Ц		Ц	Ц		Ш	Щ		<u> </u>	L	Ш			L		L				L	L	L	L	L	L	Ļ	Ļ	Ļ	1	┸	┸	\perp	ļ	┸	퇶	1	┸	┸			L	L	L	Ļ	Ļ	┸	\downarrow	$oldsymbol{\perp}$	L	丄	丄	丄	<u> </u>	Ш		Ш	ᆜ	Ш
36 load	a[Ri], R1	L	1_	Ц		Ш	Ц		L			<u> </u>	L	Ц			L		L	L		L	L	L	\perp	L	L	L	L	L	┸	┸	\perp	\perp		┸	\perp	\perp	\perp	\perp	1	\perp		L	L	L	┸	┸	\perp	\perp	\perp	$oldsymbol{\perp}$	丄	丄	上	_	Ш	ightharpoonup	Ш	Ц	Ш
37 add	R1, Rsum 1, Ri, Ri	L	1_	Ш		L	Ц		L			<u> </u>		Ц			L		L	L		L	L	L	L	L	L	L	L	L	┸	┸	┸	\perp		┸	\perp	\perp	1	\perp	1	1	\perp		L	L	L	┸	\perp	\perp	\perp	$oldsymbol{\perp}$	丄	丄	丄	_	Ш	\dashv	Ш	\sqcup	Ш
38 add	1, Ri, Ri	L	1_	Ш		L	Ш		L			<u> </u>		Ш			L		L	L		L	L	L		L	L	L	L	L	L	ļ	╙	\perp		\downarrow	\perp	\downarrow	\perp	\perp	\downarrow	\perp	\perp		L	L	L	L	\downarrow	\perp	\perp	$oldsymbol{\perp}$	丄	丄	丄	_	Ш	Ц	Ш	Ш	Ш
39 cmp	Ri, N	L	1	Ш		$oxed{oxed}$	Ш		L					Ш			L		<u> </u>	L		L	L	L		L	L	L	L	L	┸	ļ	\perp	\perp		\downarrow	\perp	\perp	\perp	\perp	\perp	\perp			L	L	L	L	\perp	\perp	\perp		丄	\perp	上	_	Ш	Ц	Ш	Щ	Ш
40 jle	L1	Ļ	_			L	Ш							Ш					<u> </u>				L	L	L	L	L	L	L	L	L	ļ	ļ	┸		ļ	ļ	↓	┸	ᆚ	↓	1		L	L	L	L	L	퇶		L	L	丄	丄	丄	L	Ш	_	Ш	┙	Ш
41									L																																												丄	丄	上			丄	Ш	┙	Ш

A.O: Sheet for code portion (A) out-of—order

			1	2	3	4	5	6	7	8	9) 1	0 1	1 1	12	13	14	15	16	17	18	19 2	20 2	21 2	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	14 4	5 4	16 4	17 4	48 4	49 !	50	51 .	52 5	53 5	4 55	56	57	58	59	60
1 load	a[Ri],	l. R1	F	ь	R	s	Е	Е	Е	E	Ī	: 1	E E	: 1	Ε	Е	Ε	w	T	T	Ť		Ť	T	T	7													T		T	T				T	T		Ť	T	T	T	T	T	十	T	T	T	П	П	П	7
2 add		Rsum	F	Б	R	s			Ī	Ī	T	T	T	T	Ī	Ŧ		E	w	1	T		T	T	T	T												Ī	T		T	T				寸	T		T	T	T	十	T	T	十	T	T	T	П	П	П	_
		i, Ri	F	D	R	s	Е				T	T		T	T		1		w	1	T	T	T	1	T																							Ť	T	1	T	T		T	T			T	П	П	П	
_	Ri, N		F					Е		Ī	T	Ť	T	Ť	T	T	T	_	w	1	T		T	1	T	T												İ	T		T	T				T	T		T	1	T	十	T	T	十	T	T	Т	П	П	П	
5 jle	L1		F	D	R	s			Е		T	T		T	T		T	T	w	T	1		T	T	T	T																							1	T	T	T	T		T	T		П	П	П	П	
6 load	a[Ri],						s				х	,	x >	(x	х	х	Е	w				T																							T	T		T		T	T	T	T	T	T		Г	П	П	П	7
	R1, F											T		T	T		T		E	w	T																												1		T	T	T	T	T	T		П	П	П	П	
	1, Ri,		T	F	D	R	s	Е			Ī	T	T	T	T		T	T		w	ı		T	T		T												Π								T	T		T	T	T	T	T	T	T	T		Г	П	П	П	
	Ri, N			F	D	R	s		Е					T	T		T	T	-	w			П	T		T																							T	T		T			T				П	П	П	7
10 jle	L1			F	D	R	s			Е				Ī	T		T			w	Î																												Ī		T	T	T	T	T	T		П	П	П	П	
11 load	a[Ri],	, R1																																																	Т	Т	T	T	Т	Т		П	П		П	
12 add	R1, F	Rsum																																																		floor										
13 add	1, Ri,	i, Ri																																																		ightharpoons			\Box							
14 cmp	Ri, N	١																																																												
15 jle	L1																																																		⊥	Ш			Ш	Ш			Ш	Ш	Ш	
16 load	a[Ri],	, R1																																																	\perp	\perp	┙	\perp	Т	┵		L	Ш	Ш	Ш	
17 add	R1, F	Rsum			Ц																		┙																												\perp	\perp	Ц	\perp	Т	┙		L	Ш	Ш	Ш	
18 add	1, Ri,	i, Ri																																																	⊥	\perp			\perp	┵			Ш	Ш	Ш	
19 cmp	Ri, N	١																																																	┙	┙	┙	\perp	ᆚ	┙		L	Ш	Ц	Ш	
20 jle	L1			4																																															┵	┙	┙	┙	4	ᆚ		╚	Ш	Ш	Ш	_
21 load				4	_						L	1	4	1	_		_	_		_			4	_	_																					_			1	_	4	4	4	┵	4	┵		igspace	Ш	Ш	Ш	
22 add				_	_							1	4	1	_		_	_		_	4		4	_																							4		1	_	4	4	4	\bot	4	┵		╙	Ш	Ц	Ц	
23 add	1, Ri,	i, Ri		_	4	_				<u> </u>	L	1		1	4		4	4		4	1		4	4	_	_																							4	4	_	4	4	\perp	4	4		\perp	Ш	Ц	Ш	_
24 cmp	Ri, N	١			4	_						1		1	4		4	4		4			4	_	_	_												L									4		4	4	_	4	4	\perp	4	4		Ľ	Ш	Ш	Ы	_
25 jle	L1			4	4	_				L	L	1	4	1	4	_	4	4	_	4	4	_	4	1	4	_			_									L			_					_	4	_	4	1	4	4	4	4	4	4	_	₽	Ш	Ш	Ш	_
26 load			_	4	4	_					L	1	4	1	4	4	4	4	4	4	4	4	4	4	_	_	_		4	_	_							L	_	_	┡	_				_	4	_	4	4	+	4	4	\bot	4	_	_	₽	Ш	Ш	Ш	_
27 add					4						L	1	4	1	4		4	_		_	_		4	_		_																				4	4		4	_	4	4	4	4	4	4	4	₽	Ш	Ш	Ш	_
28 add			_	_	4	4				┡	╀	1	4	4	4	4	4	4	4	4	4	4	4	4	4	4			4	_								L	_		_	_				_	4	4	4	4	+	4	4	\bot	4	4	4	╨	ш	Н	Н	_
29 cmp		1	_	4	4	4				╀	╀	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4		4	4								L	_			_				_	4	_	4	4	+	4	4	+	4	4	_	₽	Ш	Н	Н	4
30 jle	L1		4	4	4	4				╄	Ļ	+	4	4	4	4	4	4	4	4	4	4	4	4	4	4	_		_	_	4							L	<u> </u>		1	<u> </u>				4	4	_	4	4	+	4	4	+	4	4	+	₽	Ш	Н	\dashv	_
31 load			4	4	4	4	4			1	╀	+	\bot	+	4	4	4	4	4	4	4	4	+	4	4	4	_		_	_	_		Н			_	_	L	_	1	_	_			_	4	+	+	+	4	+	4	+	+	+	+	+	↩	\sqcup	Н	Н	4
32 add	R1, F	Rsum	4	\dashv	+	4				1	╀	+	\bot	+	\dashv	4	+	\dashv	4	4	4	4	4	4	\dashv	4	4	_	4	4	_		L			_	_	L	┡	1	╄	┡				_	4	-	+	4	+	+	+	+	+	+	+	-	\sqcup	Н	Н	4
33 add			4	4	+	4	4		L	\vdash	╀	+	+	+	+	4	+	+	4	+	4	4	+	+	4	4	4	4	4	4	4		\vdash			_	-	\vdash	1	1	\vdash	1		Щ	_	\dashv	+	4	+	+	+	+	+	+	+	+	+	╨	\sqcup	Н	卅	4
34 cmp		١	_	_	4	4	_			₩	╀	+	4	+	4	4	4	4	4	4	4	4	4	4	4	4	_		4	4								_	<u> </u>	-	-	<u> </u>				4	4	4	+	4	+	+	+	+	+	+	+	₩	\sqcup	Н	Н	_
35 jle	L1		4	4	4	4			L	-	╄	+	+	+	4	4	4	4	4	4	4	4	4	4	4	4	4	_	4	4	_		H					L	┡	1	╄	┡				+	4	-	+	4	+	+	+	+	+	+	+	╨	\sqcup	Н	卅	4
36 load			\dashv	4	4	+	-			-	╀	+	+	+	4	\dashv	4	4	\dashv	+	+	+	+	+	4	4	-		4	-	-		H				_	H	1	1	+	1			-	\dashv	+	+	+	+	+	+	+	+	+	+	+	╀	\vdash	Н	\dashv	-
37 add			4	+	+	+	4		L	+	╄	+	+	+	+	+	+	+	+	+	+	+	+	+	4	4	-		-	-	4						-	H	1	1	1	1				+	+	+	+	+	+	+	+	+	+	+	+	╀	\vdash	Н	\dashv	-
38 add			\dashv	+	+	4	-		L	-	╀	+	+	+	+	+	+	+	+	+	4	+	+	+	\dashv	4	-	_	\dashv	4	-						-	H	1	1	+	1			-	\dashv	+	+	+	+	+	+	+	+	+	+	+	╀	\vdash	Н	\dashv	-
39 cmp		1	\dashv	+	+	-				-	\vdash	+	+	+	\dashv	4	+	+	4	4	-	4	+	4	\dashv	4	4	_	\dashv	\dashv			H					H	1	1	+	1				\dashv	+	-	+	4	+	+	+	+	+	+	+	₩'	\vdash	Н	\dashv	-
40 jle	L1		\dashv	+	+	+	_		L	\vdash	╄	+	+	+	+	+	+	+	+	+	4	+	+	+	\dashv	4	4	_	4	4	-		H					┝	-	1	+	-				+	+	+	+	+	+	+	+	+	+	+	+	₩	\vdash	Н	\dashv	-
41									_	<u> </u>	1_			L																			L					<u> </u>	<u> </u>		<u> </u>							L_			丄	ㅗ	丄	ㅗ	ㅗ	丄		Щ	ш	Ш	Ш	

B.I: Sheet for code portion (B) – In-order

		1	2	3	4	5	6	7	8	9	1	0 1	1 1	2 1	3 1	4 1	5	16	17	18	19	20	21	22	23	24	25	26	27	28	29	130	0 3	13	2 3	33 3	34 :	35	36	37	38	39	40	41 4	42	43 4	14	15 4	46	47	48	49	50	51	52	53	54	55	56	57	58	59 (60
1 load	a[Ri], R1	F							Ť	Ť	+	Ĭ.	Ť	+	Ĭ	+			Ï	. <u>.</u>				_		_	-		Ť	1		1	+	Ť	+	~	77	-	-	"	-	00	-	+	72	-	+			77	70	70	-	٠.	-	-	07	-	-	-	"	-	~
2 cmp		F		티	-	-		w	┢	t	+	+	+	$^{+}$	+	$^{+}$	$^{+}$	+	┪	\dashv	_				-	H	╁	H	H	╁	H	t	╁	$^{+}$	$^{+}$	+	+	$^{+}$	\dashv	+	\dashv	\dashv	+	+	+	+	+	+	+	-	-		H		-	H		H		\dashv	十	+	\dashv
				-			E		w	+	+	+	+	+	+	+	+	+	+	\dashv							1	H	┢	┢	┢	H	+	+	+	+	+	+	-	_	\dashv	\dashv	-	+	-	+	-	+	+											-	+	+	\dashv
3 jle	L2		D	-							┿	+	+	+	+	+	+	+	4	-						H	-	┝	┢	┢	┝	╀	╁	╀	+	+	+	+	4	+	4	4	4	+	+	+	4	+	+		_					H	-	-		\dashv	+	+	-
4 add	R1, Rsum		D	_					w		4	+	+	+	4	+	4	4	4	4						L	<u> </u>	L	<u> </u>	-	<u> </u>	╄	4	+	+	4	4	4	_	_	4	4	4	4	4	4	4	4	4								_			4	4	4	_
5 add		F		_				Е	w		1	_	┸	1	_	1	4	4	_	_									L	_		L	_	1	1	4	4	4	_	_	4	4	4	4	4	4		_	_	_			_							4	4	4	_
6 cmp			D						E	W				1		1	1																	L	1			_						4		_															4	_	
7 jle	L1	F	D							E	٧	V																																																	Ш		
8 load	a[Ri], R1		F							E	<u> </u>	E E																																																	\Box		
9 cmp	R1, 0		F	D									E	١	N	_	_																		_			Ц						4																	┙		
10 jle	L2		F	D				L		L	┸		ļ	Ш	ΕV	٧	4	_	_								L					L	Ļ	L	1		4	4		_			_		_	_				_											4	4	
	R1, Rsum		-	-	-	-		-	ŀ	<u> </u>	<u> </u>	-	<u> </u>	4	- -	4	4	_	_	_						el	eme	ente	· m	arke	ed v	vith	h "-	" d	end	ote	flus	she	d	Ц	_		_	_	_	4	_	_		_					<u> </u>					_	4	4	_
12 add	1, Ri, Ri		-	-	-	-	•	-	-	ļ.	<u>. .</u>	<u>. .</u>	<u> </u>	4	<u>- </u>	_	_	E '	_							Ŭ.	•						tior		•			0	-	Ц	_		_	4	_	4		_	4	_										_	4	4	_
13 cmp	Ri, N		-	-	-	-		-	_	<u> </u> -	. -	<u>. .</u>	<u>.</u>	<u>. L</u>	- 1	<u> </u>	D		_	W							_				Ė			_	_	_	_	_	_	1																					ightharpoonup		
14 jle	L1		-	-	-	-	-	١.	١.	-	. .	<u>. .</u>	- 1		D			Ε																																									Ш		
15 load	a[Ri], R1																F	D		Е	Ε	Ε	W																																								
16 cmp	R1, 0																																																														
17 jle	L2																																																												П		
18 add	R1, Rsum									Γ				T		T	T		T													Γ		Τ	T		T	T			T								T												Т		1
	1, Ri, Ri															T	T																	Ī	T			T			T					T															T		
20 cmp													T	T	T	T	T		T													Г	T		T		T	T			T		T			T														T	T	T	
21 jle	L1									T	T		T	T		T	T		T													T		T	T		T	7		T					T				T												T	T	
22 load	a[Ri], R1									Ī	T		Ī	T		T	T		T													Ī		Ī	T		T	T		T					T																Т	Т	
23 cmp	R1, 0																																																												\Box		
24 jle	L2																																																														
25 add	R1, Rsum																																																														
	1, Ri, Ri																																																														
27 cmp	Ri, N																																																														
28 jle	L1																																																														
29 load	a[Ri], R1																																																												\Box		
30 cmp																																																													П		
31 jle	L2		T									Τ	Ī	Τ	T	T	T	T	T															Ī	T	T	T	T	1	T	T			T	T	T	T													T	T	T	٦
	R1, Rsum	Ħ	T					T		Τ	T	T	T	T		T	T		T	T							Π		Π			T	T	T	T	T	T	T	T	T	T	T	T	T	T	T		Ì	T									Π		T	\top	T	7
33 add	1, Ri, Ri		T	T				Γ		T	Ť	T	T	T	T	T	T	T	┪	T						Γ	Π	Π	Π		Π	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T	Ì	T				Γ					Π		T	\top	T	7
34 cmp		П	寸	T				Γ		T	Ť	T	T	T	T	T	T	T	┪	T						Π	Γ	Π	Γ	1	Γ	T	T	T	Ť	T	T	T	T	T	寸	T	7	丁	T	丁	T	İ	T	T			Γ					İ		寸	\top	丁	7
35 ile	L1	Ħ	T	7				T	T	T	T	T	T	T	T	Ť	T	T	T	7			П			Г	T	Π	T	T	Π	T	T	T	Ť	T	T	寸		T	T	7	T	T	T	T	T	Ť	T	7			Г					Г		T	\top	\forall	7
-r	a[Ri], R1	Ħ	7	7				T		t	T	T	T	T	T	T	T	1	7	7						Г	T		T	T	T	t	T	t	T	T	T	7	7	7	寸	7	7	1	T	T	T		7	T								Г	П	7	\dashv	ヿ	٦
37 cmp		П	T	1				T	T	T	T	T	T	T	T	1	T	T	7							Г	T	Γ	T	T	T	T	T	T	1	T	1	7	1	T	7	1	7	T	t	1	T	t	7	7			Г			П				7	\top	\forall	7
38 ile	L2	Ħ	T	7				T	Ħ	t	Ť	\top	Ť	T	T	T	十	+	T	7						Г	Ħ		T	T	T	T	Ť	T	T	\dagger	T	7	1	T	寸	1	1	1	7	7	1	t	1	7			Г			П		Г	П	寸	\top	十	٦
39 add	R1, Rsum	H	7	7				T		t	T	\top	t	T	T	T	T	1	7	7						Г	T	Г	T	T	T	t	t	t	T	T	T	7	7	1	7	7	7	T	T	1	T		7	7								Г		寸	+	十	┪
	1, Ri, Ri	H	7	┪				T	H	t	t	\top	Ť	t	\top	†	†	\forall	┪	┪						Т	T	Т	T	T	T	t	╁	t	†	\top	1	7	\dashv	┪	\dashv	7	十	十	\forall	十	+	T	7	┪			Т							\dashv	十	十	1
		H	\dashv	1	-			H	H	t	t	+	T	+	+	\dagger	+	+	┪	+	-		Н			H	H	H	t	t	H	t	t	t	\dagger	\dagger	+	+	+	\dashv	\dashv	+	\dashv	+	+	\dashv	+	\dagger	\dashv	-			Н		H	Н			Н	\dashv	十	+	\dashv
42 jle	L1	\vdash	\dashv	\dashv				H	H	+	+	+	+	+	+	+	+	+	┪	\dashv						H	H	H	H	╁	H	t	+	+	+	+	+	\dashv	\dashv	\dashv	\dashv	\dashv	\dashv	\pm	\dashv	\dashv	+	+	\dashv	\dashv			<u> </u>					H		\dashv	十	+	\dashv
43	LI	H	┪	┪				H	H	t	\dagger	+	+	+	+	+	+	+	┪	7						H	H	H	H	t	H	٢	+	+	+	+	+	+	+	┪	\dashv	+	+	+	\dashv	+	+	+	+	╡			\vdash			H		H		\dashv	十	+	\dashv
			1					_	1	_			_				_		_1					_		_	_	_	_		_	_		_				_		1								_	_	!	!		_	_	<u> </u>	_		_			ㅗ	_	_

B.O: Sheet for code portion (B) – Out-of-order

		1	2	3	4	5	<u> </u>	T ₇	7 8	٠ ٦	م آ	10 1	11	12	13	14	15	16	17	18	10	20	1 21	22	22	1 2	1 2	12	6 2	7 29	8 2	ء اه	30 3	11	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56 5	7 5	8 5	9 60
1 load	a[Ri], R1								= v		Ť		7	-		•			Ë	<u> </u>	<u> </u>	1	Ŧ	f		1	Ŧ	1-	1	Ŧ	1	1	~	Ŧ	-	-	04	00	-	0.	-	-	70	H	72		+		70	7,	70	70	-	<u>.</u>	-	-	-	-	-	1	+	100
			Б				╀	+5		<u> </u>		+	\dashv	-	-	┪			H	H	+	+	╁	t	۲	t	+	t	+	+	+	$^{+}$	+	+	\dashv	-	+				H					_	\dashv	+			_						_	\dashv	+	+	+	+
_2 cmp							┢	+	+		E		+			-			H	┢		╁	╀	╁	╁	+	+	+	+	+	+	+	+	+	+	-	-							Н		\dashv	\dashv	-									-	+	+	+	+	+
3 jle	L2		D				┢	╀	+	_			+	+	-	_			H	┢	┢	╀	╄	╀	╄	╀	╁	+	+	╄	+	+	+	+	+	+	+				-	H				\dashv	4	+	_	_	_					\dashv	+	+	+	+	+	+
4 add	R1, Rsum						<u> </u>	╀	E	4	_	w	4		_	4				╄	╀	╀	╀	┡	╄	+	+	+	+	╀	╀	+	4	4	4	_	4				_					_	4	_								_	_	+	4	+	+	╄
5 add			D					1	4	1		w	4		_	_				L	┡	┡	╄	1	L	1	_	1	1	_	╀	1	4	4	4	4	4									_	4	_									_	4	_	4	4	╄
6 cmp			D				E	_	1	1	_	w	_										╙					L	┸		L	1			_		Ц																					_		╙	丄	Ļ
7 jle	L1	F	D	R	s			E	<u> </u>			w																																																	Щ	Ш
8 load	a[Ri], R1		F	D	R	s	E	E	E			w																																																		
9 cmp	R1, 0		F							<u> </u>	E		Ц																┸			_			Ц																							_		\perp	丄	L
10 jle	L2	Ш			R	s		┸		1	4	E١	w							L	╙	L	L		L	L	L	1	╀		┸	1		_	4		_							Ш													_	_		_	丰	丄
11 add	R1, Rsum		-	-	-	ŀ	Ŀ	ŀ	<u>. -</u>	4	-	_	_	-	-	-	-	-			L		L			Ш													<u></u>		<u>L</u>																				丄	Ļ
12 add	1, Ri, Ri	Ш	-	-	-	ļ.	Ŀ	↓-	- ↓-	4	-	_	_	_	R	_	Е			L	L	L	L	L	L	L	eie	me	11(\$	ma			vitn erat			erio	te I	ius	hed	1				Ш		\perp	_						Ш			Ц	_	\perp	1	\perp	丄	丄
13 cmp	Ri, N		-	-	-	<u> </u> -	<u> </u> -	_	. -		-	-	_	-	R	-		Е	w										_			оp.			<u>.</u>			_	_																						丄	丄
14 jle	L1		_]	-	Ŀ	Ŀ	<u>L-</u>	<u> </u>	.∐-		<u>.</u>	-			R					W		L	L		L	L	L	L			L													L																	L	┖
15 load	a[Ri], R1								I	I				F	D	R	s	Ε	Е	E	W							I				floor	I	floor																									floor	I	L	
16 cmp	R1, 0																																																												\perp	
17 jle	L2																																																												T	
18 add	R1, Rsum							T		T			T												Γ	T		T	T			T		T	T																										Т	
	1, Ri, Ri						Ī	T		T	T		7							Ī	T	T	T	T	T	T		T	T		T	1		T	7													T													T	T
20 cmp						Г	T	T	T	T	T	T	T			T				T	T	T	T	T	T	T	T	T	T	T	T	Ť	7	T	T		T									T		T										T	1	T	T	T
21 jle	L1		T					T	T	T	T	T	T							Ī	T		T			T	T	T	T		T	T	T	T	T											T	T										T	T		T	T	1
	a[Ri], R1							T		T			T													T	T	T	T		T	Ť		T	T												Ť										T	T		T	T	T
23 cmp												T	T															T			T	T	Ī	T	T																									T	T	T
24 jle	L2							T		T	T		T											Γ		Ī			T			T		T	T																										Т	
25 add	R1, Rsum																																																												Τ	
26 add	1, Ri, Ri																																																												Ι	
27 cmp	Ri, N																																																												T	
28 jle	L1										ı		T										Ī				T	Ī			Ī	T		T	T																										Т	Т
29 load	a[Ri], R1																																																											T	Т	Т
30 cmp								T		T			T												Γ	T		T	T			T		T	T																										Т	
31 jle	L2		\neg				Γ	T		T	T		7			ヿ				Γ	T	T	T	T	T	T	Ī	T	T		T	T		T	7		╗									T	ı	T									T				T	T
32 add	R1, Rsum					İ	T	T	1	T	T	T	T							T	T	T	T	T	T	T	Ī	T	Ť	Т	T	T	T	T	T	T	7		П		T			П		丁											T	T	T	T	T	1
33 add	1, Ri, Ri					Ī	T	T	Ť	T	T	T	T			7				T	T	T	T	T	T	T	Ť	T	T	T	T	T	T	T	T	T	7							П		┪											T	寸	T	十	T	T
34 cmp						T	T	T	1	T	1	寸	寸			┪			Г	T	T	T	T	T	T	T	T	T	T	Т	T	T	T	T	寸	T	7		П		T	Г		П		寸	1	1									寸	寸	T	十	T	1
35 ile	L1	П	7			T	T	T	T	T	T	\top	\forall			7			Г	T	T	T	T	T	T	T	T	Ť	T	T	T	T	+	T	\forall	T	7		П			Г		П		寸	1	\exists									T	十	+	T	T	T
	a[Ri], R1	П	7			T	T	T	T	T	†	T	7	7	7	7			Г	T	T	T	T	T	T	T	T	T	T	T	T	T	†	†	7	7	┪		H		T	T		П		寸	7	1	7				П	T	П		┪	寸	T	十	十	T
37 cmp			\dashv			T	T	T	T	T	T	T	1			7				T	T	T	t	T	t	t	t	t	T	╅	T	1	1	+	1	1	7				H			H		1	1	\dashv									1	十	1	\dagger	T	T
38 ile	L2	H	\dashv			H	T	t	\top	\dagger	†	\dashv	寸	7	7	┪			Н	T	t	t	t	t	t	t	t	t	t	T	t	T	+	\dagger	寸	1	寸	_	H		H	Г		Н		7	1	\dashv	7			_	Н				┪	\dashv	+	+	+	T
39 add	R1, Rsum	H	\dashv			t	t	t	†	+	+	\dagger	\forall	7	\dashv	ᅥ		H	H	t	t	t	t	t	t	t	t	t	t	T	t	\dagger	\dagger	\dagger	\forall	+	\dashv	_	H		H	Т		H		\dashv	+	\dashv	7				Н	Т		\exists	\dashv	\dashv	\dagger	+	+	T
	1, Ri, Ri	H	\dashv			t	t	t	+	$^{+}$	+	+	\forall	1	\dashv	┪		Н	H	t	t	t	t	t	t	t	╁	t	t	╁	t	\dagger	1	$^{+}$	\forall	+	\dashv		H		t	Н		H		1	7	\forall	\dashv				H	\vdash	H		┪	\dashv	1	+	十	+
40 add 41 cmp		H	\dashv		H	H	H	+	+	+	\pm	+	+	\dashv	\dashv	┪		H	H	H	t	t	t	t	t	t	+	t	+	+	+	+	+	+	+	\dashv	\dashv		\vdash		\vdash	H	-	H		\dashv	\dashv	+	\dashv				Н	 	Н	\dashv	\dashv	\dashv	+	+	+	+
41 cmp	HI, N L1	Н	\dashv	-	-	H	H	+	+	+	+	+	\dashv	┪	\dashv	-		H	H	H	╁	╁	╁	+	+	+	╁	+	+	+	+	+	+	+	\dashv	\dashv	\dashv	-	Н		┢	H		H		\dashv	┥	\dashv	\dashv	-	-	-	H	-	H	\vdash	\dashv	+	+	+	+	+
42 Jie 43	LI	H	\dashv		H	H	H	十	+	+	+	+	\dashv	┪	\dashv	┪		H	H	H	t	╁	t	十	╁	╁	╁	╁	+	+	╁	+	+	+	\dashv	+	┪	-	H		H	H		H		┪	\dashv	\dashv	┪			-	H	┢	H	H	┪	+	+	+	十	十
TU					_		_	_			_							_	_	1	1			_	_	_							_	_			1				_	_	I	ш					_				_	_	Ш						ㅗ	—