	Convert the MIDC assembly sade below to C. Assume. For the following C statement units a minimal					
Store Registers	Convert the MIPS assembly code below to C. Assume that <i>a, b, c, i,</i> and <i>j,</i> are assigned to registers \$50, \$51, sequence of MIPS assembly instructions that does the					
Mach Lang 16 17 18 19 20 21 22 23 8 9 10 11 12 13 14 15	\$s2, $$s3$ , and $$s4$ , respectively. Assume that the base identical operation. Assume $$t1 = A$ , $$t2 = B$ , and $$s1$					
Decimal         0         1         2         3         4         5         6         7         8         9         10         11         12         13         14         15	addresses of arrays M and K are assigned to registers is the base address of C.  \$56 and \$57, respectively. $A = C[0] << 4;$					
Hex 0 1 2 3 4 5 6 7 8 9 A B C D E F	a b c i j M(0) K(0) A B C(0)					
Binary   0000   0001   0010   0011   0100   0101   0110   0111   1000   1001   1010   1011   1100   1011   1100   1101   1110   1111   1110   1111   add \$rd. \$rs. \$rt   add \$so. \$sr. \$t5   \$so = 16. \$sr = 23. \$t5 = 13	\$50 \$s1 \$s2 \$s3 \$s4 \$s6 \$s7 \$t1 \$t2 \$s1					
add \$rd, \$rs, \$rt         add \$s0, \$s7, \$t5         \$s0 = 16, \$s7 = 23, \$t5 = 13           op         rs         rt         rd         shamt         funct	sll \$t0, \$s3, 2 # \$t0 = i x 4 A = C[0]; lw \$t1, 0(\$s1) add \$t1, \$s7, \$t0 # \$t1 = &K[0] + 4i A = C[0] << 4; sll \$t1, \$t1, \$t1, \$t2, \$t3, \$t4, \$t5, \$t5, \$t5, \$t5, \$t5, \$t5, \$t5, \$t5					
0 23 13 16 0 32	lw \$t0, 12(\$t1) #\$t0 = K[i + 3]					
0 0 0 0 0 0 1 0 1 1 1 0 1 1 0 0 0 0 0 0	sub \$t1, \$s1, \$t0 # t1 = b - K[i + 3]					
0 2 14 13 8 0 2 0 0 2 E D 8 0 2 0	add \$s2, \$s0, \$t1					
sub \$rd, \$rs, \$rt sub \$s1, \$t7, \$t0 \$s1 = 17, \$t7 = 15, \$t0 = 8	Given the C statement: $g = h + Z[j]$ Given the C statement: $A[12] = h + A[5]$ What is the corresponding MIPS assembly language					
op rs rt rd shamt funct	code? Assume the variables $g$ , $h$ , and $j$ are assigned to code? Assume the variable $h$ is assigned to \$s2 and					
0 15 8 17 0 34	registers \$51, \$52, and \$53. Z is an m-word array, and the base address of array A is in \$53.					
0 0 0 0 0 0 0 1 1 1 1 1 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 1 0	the starting or base address of array Z is zero.  q h j Z[0] h A[0]					
0 2 E 8 8 8 2 2	\$s1 \$s2 \$s3 \$s2 \$s3					
sll \$rd, \$rt, shamt sll \$s3, \$t3, 3 \$s3 = 19, \$t3 = 11, shamt = 2^3	sll \$t0, \$s3, 2 # \$t0 = j * 4   lw \$t0, 20(\$s3) # \$t0 contains value in A[5]					
op rs rt rd shamt funct	add \$t0, \$s3, \$t0  # \$t0 = &Z[j]  add \$t0, \$s2, \$t0  # \$t0 = h + A[5]     w \$s3, 0(\$t0)  # \$s3 = Z[j]  sw \$t0, 48(\$s3)  # store \$t0 in A[12]					
0 0 11 19 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	add \$s1, \$s2, \$s3  #g = h + Z[j]					
0 0 11 9 8 12 0	CPU Time Clock Cycle Time Instruction Count Frequency Spec Ratio Speedup					
0 0 0 B 9 8 C 0	t $\tau$ IC $f$ Ref time (Exec time) old $\overline{Exec\ time}$ Exec time $\overline{Exec\ time}$ (Exec time) new					
addi \$rt, \$rs, const addi \$s4, \$t4, 128 \$s4 = 20, \$t4 = 12, const = 128	(Clock Cycle Time) Exec time (Exec time) new The results of the SPEC CPU2006 bzip2 benchmark running an AMD Barcelona has an instruction count of 2.389e12, an					
op         rs         rt         constant / address offset           8         12         20         128	execution time of 750s, and a reference time of 9650s.					
0 0 1 0 0 0 0 1 1 0 0 1 0 1 0 1 0 0 0 0	a) Find the change in CPU time if the number of instructions of the benchmark is increased by 10% and $t = 1.10(IC) * 1.05(CPI) \Rightarrow t = 1.155 \text{ or } 15.5\%$					
2 1 9 4 0 0 8 0	the CPI is increased by 5%.					
2 1 9 4 0 0 8 0	$SPECratio = \frac{9650}{750} \Rightarrow S.R. = 12.87$					
lw \$rt, offset(\$rs)	b) Find the change in the SPECratio if the number of instructions of the benchmark is increased by 10% and $SPECratio = \frac{9650}{750*1.155} \Rightarrow S.R. = 11.14$					
op         rs         rt         constant / address offset           35         19         9         100						
	the CPI is increased by 5%. $Change = \frac{12.87 - 11.14}{12.87} \Rightarrow Change = 13.44\%$					
8 13 6 9 0 0 6 4	Assume the following for a single 4 GHz processor running a program:					
8 D 6 9 0 0 6 4	Operation # of instructions CPI (Cycles per Instruction)					
sw \$rt, offset(\$rs)         sw \$t1, 40(\$s3)         \$t1 = 9, \$s3 = 19, offset = 40           op         rs         rt         constant / address offset	Arithmetic         1.28e9         3           Load/Store         2.56e9         8					
43 19 9 40	Branch 64 million (=64e6)(=.064e9) 5					
1 0 1 0 1 1 1 0 0 1 1 0 0 0 0 0 0 0 0 0	When the program is parallelized to run on parallel processors, the number of arithmetic and load/store instructions per processor is divided by 0.5p (where p is the number of processors) but the number of branch instructions per processor					
10 13 6 9 0 0 2 8 A D 6 9 0 0 2 8	remains the same. Find the <b>execution time</b> and <b>relative speedup</b> when the program is parallelized to run on 16					
	A D O O O O O O O O O O O O O O O O O O					
	$\nabla (IC * CPI) = (1.28e9 * 3) \pm (2.56e9 * 8) \pm (64e6 * 5) = 24.64e9$					
A Branch => op=4 (same as  w) Jump => op=2 (6.26 bits)	$(Exec\ time)old \Rightarrow t = \frac{\sum (IC * CPI)}{f} \Rightarrow \frac{(1.28e9 * 3) + (2.56e9 * 8) + (64e6 * 5)}{4e9} = \frac{24.64e9}{4e9} = 6.16s$					
	$(Exec\ time)old \Rightarrow t = \frac{\sum (IC * CPI)}{f} \Rightarrow \frac{(1.28e9 * 3) + (2.56e9 * 8) + (64e6 * 5)}{4e9} = \frac{24.64e9}{4e9} = 6.16s$ $(Exec\ time)old \Rightarrow t = \frac{\sum (IC * CPI)}{f} \Rightarrow \frac{(1.28e9 * 3) + (2.56e9 * 8) + (64e6 * 5)}{(0.5 * 16)} = \frac{24.64e9}{4e9} = 6.16s$ $(Exec\ time)old \Rightarrow t = \frac{\sum (IC * CPI)}{f} \Rightarrow \frac{(1.28e9 * 3) + (2.56e9 * 8) + (64e6 * 5)}{(0.5 * 16)} = \frac{3.36e9}{4e9} = 6.16s$					
Pranch ⇒ $Op = 4$ (same as  w) Jump ⇒ $Op = 2(b.26 \text{ br}^4\text{ s})$ X=2 and Y=3. If X and Y are represented by signed 4-bit 0 0 1 0 0 1 0 0 0 1 0 integers, calculate X-Y using saturating arithmetic.	$(Exec\ time)old \Rightarrow t = \frac{\sum (IC * CPI)}{f} \Rightarrow \frac{(1.28e9 * 3) + (2.56e9 * 8) + (64e6 * 5)}{4e9} = \frac{24.64e9}{4e9} = 6.16s$ $(Exec\ time)new \Rightarrow t = \frac{\sum \left(\frac{IC}{0.5p} * CPI\right)}{f} \Rightarrow \frac{(1.28e9}{(0.5 * 16)} * 3) + \left(\frac{2.56e9}{(0.5 * 16)} * 8\right) + (64e6 * 5)}{4e9} = \frac{3.36e9}{4e9} = 840e - 3s$					
↑ Branch => $Op = 4$ (same as $I\omega$ ) Jump => $Op = 2(6.26 \text{ bits})$ X=2 and Y=3. If X and Y are represented by signed 4-bit 0 0 1 0 + 0 0 1 0	$(Exec\ time)old \Rightarrow t = \frac{\sum(IC * CPI)}{f} \Rightarrow \frac{(1.28e9 * 3) + (2.56e9 * 8) + (64e6 * 5)}{4e9} = \frac{24.64e9}{4e9} = 6.16s$ $(Exec\ time)new \Rightarrow t = \frac{\sum\left(\frac{IC}{0.5p} * CPI\right)}{f} \Rightarrow \frac{(1.28e9)}{(0.5*16)} * 3 + \left(\frac{2.56e9}{0.5*16} * 8\right) + (64e6 * 5)}{4e9} = \frac{3.36e9}{4e9} = 840e - 3s$ $speedup = \frac{(Exec\ time)new}{(Exec\ time)new} \Rightarrow \frac{6.16}{840e - 3} = 7.33s$					
Branch $\Rightarrow$ $Op = 4$ (some $as$   $\omega$ ) Jump $\Rightarrow$ $Op = 2$ (b. 26 br s)  X=2 and Y=3. If X and Y are represented by signed 4-bit integers, calculate X-Y using saturating arithmetic. Express result in hexadecimal format.  Show the best way to calculate $(0x72^{\circ}0xCC)$ using shifts and adds.	$(Exec\ time)old \Rightarrow t = \frac{\sum (IC * CPI)}{f} \Rightarrow \frac{(1.28e^9 * 3) + (2.56e^9 * 8) + (64e^6 * 5)}{4e^9} = \frac{24.64e^9}{4e^9} = 6.16s$ $(Exec\ time)new \Rightarrow t = \frac{\sum \left(\frac{IC}{0.5p} * CPI\right)}{\left(\frac{1}{0.5p} * CPI\right)} \Rightarrow \frac{\left(\frac{1.28e^9}{(0.5 * 16)} * 3\right) + \left(\frac{2.56e^9}{(0.5 * 16)} * 8\right) + (64e^6 * 5)}{4e^9} = \frac{3.36e^9}{4e^9} = 840e - 3s$ $Speedup = \frac{(Exec\ time)new}{\left(\frac{Exec\ time)new}{Exec\ time)new}} \Rightarrow \frac{6.16}{840e - 3} = 7.33s$ The following table shows results for SPEC benchmark programs running on a processor. Find the geometric mean.					
Branch $\Rightarrow$ $Op = 4$ (same as   $\omega$ ) Jump $\Rightarrow$ $Op = 2$ (b. 26 bras)  X=2 and Y=3. If X and Y are represented by signed 4-bit integers, calculate X-Y using saturating arithmetic.  Express result in hexadecimal format.  Show the best way to calculate ( $0x72^{\circ}0xCC$ ) using shifts and adds. $0x72 = 0111 0010 \Rightarrow 2^{6} + 2^{5} + 2^{4} + 2^{1} \Rightarrow 6 + 5 + 4 + 1 = 16 \Rightarrow 16$ shifts and 3 adds	$(Exec\ time)\ old\ \Rightarrow\ t = \frac{\sum(IC\ \circ\ CFI)}{(0.5p\ \circ\ COS^p)} \Rightarrow \frac{(1.28e^9\ 3) + (2.56e^9\ 8) + (64e^6\ 5)}{4e^9} = \frac{24.64e^9}{4e^9} = 6.16s$ $(Exec\ time)\ new\ \Rightarrow\ t = \frac{\sum(IC\ \circ\ CFI)}{(0.5p\ \circ\ COS^p)} \Rightarrow \frac{(1.28e^9\ \circ\ (0.5^{\circ}\ 16)^{\circ}\ 3)}{(0.5^{\circ}\ 16)^{\circ}\ 3)} + \frac{(2.56e^9\ \circ\ 8) + (64e^6\ 5)}{(4e^9\ \circ\ COS^p)} = \frac{3.36e^9}{4e^9} = 840e - 3s$ $\frac{f}{f} = \frac{(Exec\ time)\ new\ }{f} = \frac{(Exec\ time)\ new\ }{f} = \frac{840e - 3}{840e - 3} = 7.33s$ The following table shows results for SPEC benchmark programs running on a processor. Find the geometric mean. $\frac{10^{\circ}\ \circ\ OS^p\ }{f} = \frac{10^{\circ}\ }{f} = \frac{10^{\circ}\ \circ\ OS^p\ }{f} = \frac{10^{\circ}\ $					
Branch $\Rightarrow$ $Op = 4$ (some as  w) Jump $\Rightarrow$ $Op = 2$ (b. 26 bras)  X=2 and Y=3. If X and Y are represented by signed 4-bit integers, calculate X-Y using saturating arithmetic.  Express result in hexadecimal format.  Show the best way to calculate (0x72*0xCC) using shifts and adds.	$(Exec\ time)old \Rightarrow t = \frac{\sum(IC * CPI)}{f} \Rightarrow \frac{(1.28e9 * 3) + (2.56e9 * 8) + (64e6 * 5)}{4e9} = \frac{24.64e9}{4e9} = 6.16s$ $(Exec\ time)new \Rightarrow t = \frac{\sum(\frac{IC}{0.5p} * CPI)}{f} \Rightarrow \frac{(1.28e9)}{(0.5*16)} * 3) + \frac{(2.56e9)}{(0.5*16)} * 8) + (64e6 * 5)}{4e9} = \frac{3.36e9}{4e9} = 840e - 3s$ $Exec\ time)new \Rightarrow t = \frac{\sum(\frac{IC}{0.5p} * CPI)}{f} \Rightarrow \frac{(Exec\ time)old}{(Exec\ time)new} \Rightarrow \frac{6.16}{840e - 3} = 7.33s$ The following table shows results for SPEC benchmark programs running on a processor. Find the geometric mean.  Name IC * 10e9   Execution Time (s)   Reference Time (s)   Ratio \frac{8eItme}{Exec\ time} \frac{8993}{1085} = \frac{8993}{8.288} P1 336   1085   8993   \frac{8993}{1085} = 8.288					
Branch $\Rightarrow$ $Op = 4$ (some as  w) Jump $\Rightarrow$ $Op = 2$ (b. 26 br4 s)  X=2 and Y=3. If X and Y are represented by signed 4-bit integers, calculate X-Y using saturating arithmetic.  Express result in hexadecimal format.  Show the best way to calculate (0x72*0xCC) using shifts and adds. $0x72 = 01111 0010 \Rightarrow 2^6 + 2^5 + 2^4 + 2^1 \Rightarrow 6 + 5 + 4 + 1 = 16 \Rightarrow 16 \text{ shifts and 3 adds}$ $0xCC = 1100 1100 \Rightarrow 2^7 + 2^6 + 2^3 + 2^2 \Rightarrow 7 + 6 + 3 + 2 = 18 \Rightarrow 18 \text{ shifts and 3 adds}$ $0xCC = 1100 1100 \Rightarrow 2^7 + 2^6 + 2^3 + 2^2 \Rightarrow 7 + 6 + 3 + 2 = 18 \Rightarrow 18 \text{ shifts and 3 adds}$ $0xCC = 1100 1100 \Rightarrow 2^7 + 2^6 + 2^3 + 2^2 \Rightarrow 7 + 6 + 3 + 2 = 18 \Rightarrow 18 \text{ shifts and 3 adds}$ $0xCC = 1100 1100 \Rightarrow 2^7 + 2^6 + 2^3 + 2^3 \Rightarrow 2^3 \Rightarrow 7 + 6 + 3 + 2 = 18 \Rightarrow 18 \text{ shifts and 3 adds}$ $0xCC = 1100 1100 \Rightarrow 2^7 + 2^6 + 2^3 + 2^3 \Rightarrow 2^7 + 6 + 3 + 2 = 18 \Rightarrow 18 \text{ shifts and 3 adds}$ $0xCC = 1100 1100 \Rightarrow 2^7 + 2^6 + 2^3 + 2^3 \Rightarrow 2^7 + 6 + 3 + 2 = 18 \Rightarrow 18 \text{ shifts and 3 adds}$	$(Exec\ time)old \Rightarrow t = \frac{\sum(IC * CPI)}{f} \Rightarrow \frac{(1.28e9 * 3) + (2.56e9 * 8) + (64e6 * 5)}{4e9} = \frac{24.64e9}{4e9} = 6.16s$ $(Exec\ time)new \Rightarrow t = \frac{\sum(\frac{IC}{0.5p} * CPI)}{f} \Rightarrow \frac{(1.28e9}{(0.5* 16)} * 3) + (\frac{2.56e9}{(0.5* 16)} * 8) + (64e6 * 5)}{4e9} = \frac{3.36e9}{4e9} = 840e - 3s$ $\frac{eedup}{ee} = \frac{(Exec\ time)old}{(Exec\ time)new} \Rightarrow \frac{6.16}{840e - 3} = 7.33s$ The following table shows results for SPEC benchmark programs running on a processor. Find the geometric mean.  Name $   C * 10e9   Execution\ Time (s)   Reference Time (s)   Ratio \frac{Ref time}{Exec\ time} $ P1 $   336   1085   8993   \frac{8993}{1085} = \frac{8.288}{1085} $ P2 $   1525   700   4960   \frac{4960}{700} = 7.086$					
Branch $\Rightarrow$ $Op = 4$ (some as  w) Jump $\Rightarrow$ $Op = 2$ (b. 26 br. s)  X=2 and Y=3. If X and Y are represented by signed 4-bit integers, calculate X-Y using saturating arithmetic.  Express result in hexadecimal format.  Show the best way to calculate (0x72*0xCC) using shifts and adds.  0x72 = 0111 0010 $\Rightarrow$ $2^6 + 2^5 + 2^4 + 2^1 \Rightarrow 6 + 5 + 4 + 1 = 16 \Rightarrow 16$ shifts and 3 adds  0xCC = 1100 1100 $\Rightarrow$ $2^7 + 2^6 + 2^3 + 2^2 \Rightarrow 7 + 6 + 3 + 2 = 18 \Rightarrow 18$ shifts and 3 adds  Best Way $\Rightarrow$ $(2^6 + 2^5 + 2^4 + 2^1) * 0xCC \Rightarrow (2^6 * 0xCC) + (2^5 * 0xCC) + (2^4 * 0xCC)$ Operation A B Overflow when result is	$ (Exec \ time) \ old \Rightarrow \ t = \frac{\sum (IC * CPI)}{(0.5p * CPI)} \Rightarrow \frac{(1.28e^9 * 3) + (2.56e^9 * 8) + (64e^6 * 5)}{4e^9} = \frac{24.64e^9}{4e^9} = 6.16s $ $ (Exec \ time) \ new \Rightarrow \ t = \frac{\sum (\frac{IC}{0.5p} * CPI)}{f} \Rightarrow \frac{(1.28e^9)}{(0.5 * 16)} * 3) + (\frac{2.56e^9}{(0.5 * 16)} * 8) + (64e^6 * 5)}{4e^9} = \frac{3.6e^9}{4e^9} = 840e - 3s $ $ F = \frac{(Exec \ time) \ new}{f} \Rightarrow \frac{(Exec \ time) \ new}{f} \Rightarrow \frac{6.16}{4e^9} = \frac{3.6e^9}{4e^9} = \frac{3.6e^9}{4e^9} = \frac{840e - 3s}{4e^9} = \frac{3.6e^9}{4e^9} = \frac{840e - 3s}{4e^9} = \frac{3.6e^9}{4e^9} = \frac{1.6e^9}{4e^9} = 1.$					
Branch $\Rightarrow$ $Op = 4$ (so me as  w) Jump $\Rightarrow$ $Op = 2$ (b. 26 brts)  X=2 and Y=3. If X and Y are represented by signed 4-bit integers, calculate X-Y using saturating arithmetic.  Express result in hexadecimal format.  Show the best way to calculate $(0\times72^*0\times CC)$ using shifts and adds. $0\times72 = 01111 \ 0010 \Rightarrow 2^6 + 2^5 + 2^4 + 2^1 \Rightarrow 6 + 5 + 4 + 1 = 16 \Rightarrow 16 \text{ shifts and 3 adds}$ $0\times CC = 1100 \ 1100 \Rightarrow 2^7 + 2^6 + 2^3 + 2^2 \Rightarrow 7 + 6 + 3 + 2 = 18 \Rightarrow 18 \text{ shifts and 3 adds}$ $0\times CC = 1100 \ 1100 \Rightarrow 2^7 + 2^6 + 2^5 + 2^4 + 2^1 \Rightarrow 0\times CC \Rightarrow 2^6 * 0\times CC) + (2^5 * 0\times CC) + (2^4 * 0\times CC)$ Operation A  B  Overflow when result is   IEEE Floating Point   bias   sign   exponent   fraction   A+B   $\geq 0 \geq 0 \leq 0 \leq 0$   Single Precision 32 bits   127   1   8   23	$(Exec \ time) old \Rightarrow t = \frac{\sum (IC \circ CPI)}{f} \Rightarrow \frac{(1.28e^9) \Rightarrow (1.28e^9) \Rightarrow (2.56e^9) \Rightarrow (4.64e^6 \times 5)}{4e^9} = \frac{24.64e^9}{4e^9} = 6.16s$ $(Exec \ time) new \Rightarrow t = \frac{\sum (\frac{IC}{0.5p} \circ CPI)}{f} \Rightarrow \frac{(1.28e^9) \Rightarrow (0.5 \times 16)^3 \Rightarrow (\frac{2.56e^9}{(0.5 \times 16)^3} \times 8) + (64e^6 \times 5)}{(\frac{2.56e^9}{0.5 \times 16)^3} \times 8) + (64e^6 \times 5)} = \frac{3.36e^9}{4e^9} = 840e - 3s$ $\frac{f}{f} = \frac{(Exec \ time) new}{f} \Rightarrow \frac{6.16}{4e^9} = \frac{840e - 3}{4e^9} = \frac{3.36e^9}{4e^9} = \frac{840e - 3s}{4e^9} = \frac{840e - 3s}{4e^$					
Branch ⇒ Op = 4 (some os   ω) Jump ⇒ Op = 2 (b.26 br4 s)  X=2 and Y=3. If X and Y are represented by signed 4-bit integers, calculate X-Y using saturating arithmetic.  Express result in hexadecimal format.  Show the best way to calculate $(0x72^*0xCC)$ using shifts and adds.  0x72 = 0111 0010 → 2 <sup>6</sup> + 2 <sup>5</sup> + 2 <sup>4</sup> + 2 <sup>1</sup> ⇒ 6 + 5 + 4 + 1 = 16 ⇒ 16 shifts and 3 adds  0xCC = 1100 1100 → 2 <sup>7</sup> + 2 <sup>6</sup> + 2 <sup>3</sup> + 2 <sup>2</sup> ⇒ 7 + 6 + 3 + 2 = 18 ⇒ 18 shifts and 3 adds  Best Way ⇒ $(2^6 + 2^5 + 2^4 + 2^1) \circ 0xCC \Rightarrow (2^6 \circ 0xCC) + (2^5 \circ 0xCC) + (2^4 \circ 0xCC)$ Operation A B Overflow when result is when result is A+B ≥ 0 ≥ 0 < 0 Single Precision 32 bits 127 1 8 23  A+B < 0 < 0 > 0 Double Precision 64 bits 1023 1 111 52	$ (Exec \ time) \ old \Rightarrow \ t = \frac{\sum (IC * CPI)}{(0.5p * CPI)} \Rightarrow \frac{(1.28e^9 * 3) + (2.56e^9 * 8) + (64e^6 * 5)}{4e^9} = \frac{24.64e^9}{4e^9} = 6.16s $ $ (Exec \ time) \ new \Rightarrow \ t = \frac{\sum (\frac{IC}{0.5p} * CPI)}{f} \Rightarrow \frac{(1.28e^9)}{(0.5 * 16)} * 3) + (\frac{2.56e^9}{(0.5 * 16)} * 8) + (64e^6 * 5)}{4e^9} = \frac{3.6e^9}{4e^9} = 840e - 3s $ $ F = \frac{(Exec \ time) \ new}{f} \Rightarrow \frac{(Exec \ time) \ new}{f} \Rightarrow \frac{6.16}{4e^9} = \frac{3.6e^9}{4e^9} = \frac{3.6e^9}{4e^9} = \frac{840e - 3s}{4e^9} = \frac{3.6e^9}{4e^9} = \frac{840e - 3s}{4e^9} = \frac{3.6e^9}{4e^9} = \frac{1.6e^9}{4e^9} = 1.$					
Branch ⇒ $Op = 4$ (same as  w) Jump ⇒ $Op = 2$ (b. 26 br4 s)  X=2 and Y=3. If X and Y are represented by signed 4-bit integers, calculate X-Y using saturating arithmetic.  Express result in hexadecimal format.  Show the best way to calculate ( $0x72^*0xCC$ ) using shifts and adds.  0x72 = 0111 0010 → $2^6 + 2^5 + 2^4 + 2^1 \Rightarrow 6 + 5 + 4 + 1 = 16 \Rightarrow 16$ shifts and 3 adds  0xCC = 1100 1100 → $2^7 + 2^6 + 2^3 + 2^2 \Rightarrow 7 + 6 + 3 + 2 = 18 \Rightarrow 18$ shifts and 3 adds  Best Way ⇒ ( $2^6 + 2^5 + 2^4 + 2^1$ ) *0xCC ⇒ ( $2^6 * 0xCC$ ) + ( $2^5 * 0xCC$ ) + ( $2^4 * 0xCC$ )  Operation   A   B   Overflow when result is   IEEE Floating Point   bias   sign   exponent   fraction    A+B   ≥ 0   ≥ 0   < 0   Single Precision 32 bits   127   1   8   23    A+B   < 0   < 0   > 0   Double Precision 64 bits   1023   1   11   52    Box Way = ( $2^6 + 2^5 + 2^4 + 2$	$ (Exec\ time)\ old\ \Rightarrow\ t = \frac{\sum(IC * CPI)}{(0.5p^{+}\ CPI)} \Rightarrow \frac{(1.28e^{9}\ 3) + (2.56e^{9}\ 8) + (64e^{6}\ 5)}{4e^{9}} = \frac{24.64e^{9}}{4e^{9}} = 6.16s $ $ (Exec\ time)\ new\ \Rightarrow\ t = \frac{\sum\left(\frac{IC}{0.5p^{+}\ CPI}\right)}{f} \Rightarrow \frac{\left(\frac{1.28e^{9}}{(0.5 \times 16)^{+}}^{3}\right) + \left(\frac{2.56e^{9}}{(0.5 \times 16)^{+}}^{8}\right) + (64e^{6}\ 5)}{\left(\frac{2.56e^{9}}{(0.5 \times 16)^{+}}^{8}\right) + (64e^{6}\ 5)} = \frac{3.6e^{9}}{4e^{9}} = 840e - 3s $ $ = \frac{3.6e^{9}}{f} = \frac{840e - 3s}{f} = \frac{3.6e^{9}}{4e^{9}} = \frac{840e - 3s}{4e^{9}} = \frac{3.6e^{9}}{4e^{9}} = \frac{840e - 3s}{4e^{9}} = \frac{3.6e^{9}}{4e^{9}} = \frac{840e - 3s}{4e^{9}} = \frac{3.6e^{9}}{4e^{9}} $					
	$ (Exec \ time) old \Rightarrow t = \frac{\sum (IC * CPI)}{f} \Rightarrow \frac{(1.28e9 * 3) + (2.56e9 * 8) + (64e6 * 5)}{4e9} = \frac{24.64e9}{4e9} = 6.16s $ $ (Exec \ time) new \Rightarrow t = \frac{\sum (\frac{IC}{0.5p} * CPI)}{f} \Rightarrow \frac{(1.28e9)}{(0.5p * 16)^{*}} \Rightarrow \frac{(\frac{1.28e9}{(0.5s * 16)} * 8) + (64e6 * 5)}{((0.5s * 16)} * 8) + (64e6 * 5)} = \frac{24.64e9}{4e9} = 6.16s $ $ (Exec \ time) new \Rightarrow t = \frac{\sum (\frac{IC}{0.5p} * CPI)}{f} \Rightarrow \frac{(\frac{1.28e9}{(0.5s * 16)} * 3) + (\frac{2.56e9}{(0.5s * 16)} * 8) + (64e6 * 5)}{4e9} = \frac{3.36e9}{4e9} = 840e - 3s $ $ (Exec \ time) new \Rightarrow \frac{1}{s} = \frac{1.28e9}{4e9} = \frac{1.36e9}{4e9} $					
Branch ⇒ $Op = 4$ (some as  w) Jump ⇒ $Op = 2$ (b. 26 br4 s)  X=2 and Y=3. If X and Y are represented by signed 4-bit integers, calculate X-Y using saturating arithmetic.  Express result in hexadecimal format.  Show the best way to calculate ( $0x72^*0xCC$ ) using shifts and adds.  Ox72 = 0111 0010 → $2^6 + 2^5 + 2^4 + 2^1 \Rightarrow 6 + 5 + 4 + 1 = 16 \Rightarrow 16$ shifts and 3 adds  OxCC = 1100 1100 → $2^7 + 2^6 + 2^3 + 2^2 \Rightarrow 7 + 6 + 3 + 2 = 18 \Rightarrow 18$ shifts and 3 adds  Best Way ⇒ $(2^6 + 2^5 + 2^4 + 2^1) * 0xCC \Rightarrow (2^6 * 0xCC) + (2^5 * 0xCC) + (2^4 * 0xCC)$ Operation A B Overflow when result is   IEEE Floating Point   Dias   Sign   Exponent   Fraction    A+B ≥ 0 ≥ 0 < 0   Single Precision 32 bits   127   1   8   23    A+B < 0 < 0 > 0 < 0   Single Precision 64 bits   1023   1   11   52    A-B ≥ 0 ≥ 0 < 0   Single min-max   -1022   1023    Exponent Field   Fraction Field   Represents   Denormalized number	$ (Exec \ time) \ old \Rightarrow \ t = \frac{\sum (IC \circ \ CPI)}{f} \Rightarrow \frac{(1.28e^9 \ s) + (2.56e^9 \ s) + (64e^6 \ s)}{4e^9} = \frac{24.64e^9}{4e^9} = 6.16s $ $ (Exec \ time) \ new \Rightarrow \ t = \frac{\sum (IC \ Sp^* \ CPI)}{f} \Rightarrow \frac{(1.28e^9 \ s) + (0.5 + 16)^* \ s}{(0.5 + 16)^* \ s} + \frac{(2.56e^9)}{(0.5 + 16)^* \ s} + (64e^6 \ s)} = \frac{24.64e^9}{4e^9} = 6.16s $ $ (Exec \ time) \ new \Rightarrow \ t = \frac{\sum (IC \ Sp^* \ CPI)}{f} \Rightarrow \frac{(1.28e^9)}{(0.5 + 16)^* \ s} + (1.28e^9)$					
Branch ⇒ $Op = 4$ (some as  w) Jump ⇒ $Op = 2$ (b. 26 br4 s)  X=2 and Y=3. If X and Y are represented by signed 4-bit integers, calculate X-Y using saturating arithmetic.  Express result in hexadecimal format.  Show the best way to calculate ( $0x72^*0xCC$ ) using shifts and adds.  Ox72 = 0111 0010 → $2^6 + 2^5 + 2^4 + 2^1 \Rightarrow 6 + 5 + 4 + 1 = 16 \Rightarrow 16$ shifts and 3 adds  OxCC = 1100 1100 → $2^7 + 2^6 + 2^3 + 2^2 \Rightarrow 7 + 6 + 3 + 2 = 18 \Rightarrow 18$ shifts and 3 adds  Best Way ⇒ $(2^6 + 2^5 + 2^4 + 2^1) \circ 0xCC \Rightarrow (2^6 \circ 0xCC) + (2^5 \circ 0xCC) + (2^4 \circ 0xCC)$ Operation A  B  Overflow when result is when result is Single Precision 32 bits 127  A+B < 0 < 0 < 0 Single Precision 32 bits 127  A-B < 0 < 0 < 0 Single min-max -126 127  A-B < 0 ≥ 0 < 0 Double Precision 64 bits 1023 1 11 52  Exponent Field Represents Denormalized number	$ (Exec \ time) \ old \Rightarrow \ t = \frac{\sum (IC * CPI)}{(0.5p * CPI)} \Rightarrow \frac{(1.28e9 * 3) + (2.56e9 * 8) + (64e6 * 5)}{4e9} = \frac{24.64e9}{4e9} = 6.16s $ $ (Exec \ time) \ new \Rightarrow \ t = \frac{\sum (IC * CPI)}{(0.5p * CPI)} \Rightarrow \frac{(1.28e9 * 3) + (2.56e9 * 8) + (64e6 * 5)}{(0.5 * 16) * 3} + \frac{24.64e9}{(0.5 * 16) * 8} = \frac{3.36e9}{4e9} = 840e - 3s $ $ = \frac{3.36e9}{4e9} = \frac{840e - 3s}{4e9} = \frac{3.36e9}{4e9} = \frac{3.36e9}{4e9} = \frac{840e - 3s}{4e9} = \frac{3.36e9}{4e9} = \frac{9.36e9}{4e9} = \frac{3.36e9}{4e9} = $					
Branch ⇒ $Op = 4$ (some os  w) Jump ⇒ $Op = 2$ (b. 26 br4 s)  X=2 and Y=3. If X and Y are represented by signed 4-bit integers, calculate X-Y using saturating arithmetic.  Express result in hexadecimal format.  Show the best way to calculate (0x72*0xCC) using shifts and adds.  0x72 = 0111 0010 → $2^6 + 2^5 + 2^4 + 2^1 \Rightarrow 6 + 5 + 4 + 1 = 16 \Rightarrow 16$ shifts and 3 adds  0xCC = 1100 1100 → $2^7 + 2^6 + 2^3 + 2^2 \Rightarrow 7 + 6 + 3 + 2 = 18 \Rightarrow 18$ shifts and 3 adds  Best Way ⇒ $(2^6 + 2^5 + 2^4 + 2^1) \approx 0xCC \Rightarrow (2^6 * oxCC) + (2^5 * oxCC) + (2^4 * oxCC)$ Operation A  B  Overflow when result is when result is   IEEE Floating Point   bias   sign   exponent   fraction    A+B ≥ 0 ≥ 0 < 0   Single Precision 32 bits   127   1   8   23    A+B < 0 < 0 < 0   O   Single min-max   -126   127    A-B   ≥ 0 ≥ 0   O   Double Prin-max   -1022   1023    Exponent Field   Fraction Field   Represents   Denormalized number    All 0s   All 0s   Denormalized number    All 0s   All 0s   Denormalized number    40   0   1   0   + 0   0   1   0    10   0   1   0   + 0   0    11   0   0   1   0    12   0   0   1   0    13   0   1   1   0   1    14   0   1   0   1    15   0   0   1   0    16   0   1   0   1    17   0   0   1    18   23    28   23    29   20   20   Double min-max   -126   127    20   0   0   0    20   0   0   0    20   0   0   0    Exponent Field   Fraction Field   Represents   Denormalized number    20   20   20   20   20   20    20   20	$ (Exec \ time) \ old \Rightarrow \ t = \frac{\sum (IC \circ \ CPI)}{f} \Rightarrow \frac{(1.28e^9 \ s) + (2.56e^9 \ s) + (64e^6 \ s)}{4e^9} = \frac{24.64e^9}{4e^9} = 6.16s $ $ (Exec \ time) \ new \Rightarrow \ t = \frac{\sum (IC \ Sp^* \ CPI)}{f} \Rightarrow \frac{(1.28e^9 \ s) + (0.5 + 16)^* \ s}{(0.5 + 16)^* \ s} + \frac{(2.56e^9)}{(0.5 + 16)^* \ s} + (64e^6 \ s)} = \frac{24.64e^9}{4e^9} = 6.16s $ $ (Exec \ time) \ new \Rightarrow \ t = \frac{\sum (IC \ Sp^* \ CPI)}{f} \Rightarrow \frac{(1.28e^9)}{(0.5 + 16)^* \ s} + (1.28e^9)$					
Spanch	$ (Exec \ time) \ old \Rightarrow \ t = \frac{\sum (IC + CPI)}{(0.5p^+ CPI)} \Rightarrow \frac{(1.28e^9 + 3) + (2.56e^9 + 8) + (64e^6 + 5)}{4e^9} = \frac{24.64e^9}{4e^9} = 6.16s $ $ (Exec \ time) \ new \Rightarrow \ t = \frac{\sum (IC + CPI)}{(0.5p^+ CPI)} \Rightarrow \frac{(1.28e^9 + 3) + (2.56e^9 + 8) + (64e^6 + 5)}{(0.5 + 16)^+ 3} + \frac{24.64e^9}{(0.5 + 16)^+ 8} = \frac{3.36e^9}{4e^9} = 840e - 3s $ $ = \frac{3.36e^9}{4e^9} = 840e - 3s $ $ = \frac{3.36e^9}{4e^9} = \frac$					
Real Property   Response   Re	$ (Exec \ time) \ old \Rightarrow \ t = \frac{\sum (IC * CPI)}{(0.5p * CPI)} \Rightarrow \frac{(1.28e9 * 3) + (2.56e9 * 8) + (64e6 * 5)}{4e9} = \frac{24.64e9}{4e9} = 6.16s $ $ (Exec \ time) \ new \Rightarrow \ t = \frac{\sum (IC * CPI)}{(0.5p * CPI)} \Rightarrow \frac{(1.28e9 * 3) + (2.56e9 * 8) + (64e6 * 5)}{(0.5 * 16) * 3} + \frac{24.64e9}{(0.5 * 16) * 8} = \frac{3.36e9}{4e9} = 840e - 3s $ $ = \frac{3.36e9}{4e9} = \frac{840e - 3s}{4e9} = \frac{3.36e9}{4e9} = \frac$					
Pranch ⇒ Op = 4 (same as  w) Jump ⇒ Op = 2 (b. 26 br4 s)  X=2 and Y=3. If X and Y are represented by signed 4-bit integers, calculate X-Y using saturating arithmetic. Express result in hexadecimal format.  Show the best way to calculate ( $\frac{0}{2}$ CCC) using shifts and adds.  Show the best way to calculate ( $\frac{0}{2}$ CCC) using shifts and 3 adds.  Ox72 = 0111 0010 → $2^6 + 2^5 + 2^4 + 2^1 \Rightarrow 6 + 5 + 4 + 1 = 16 \Rightarrow 16$ shifts and 3 adds.  OxCC = 1100 1100 → $2^7 + 2^6 + 2^3 + 2^2 \Rightarrow 7 + 6 + 3 + 2 = 18 \Rightarrow 18$ shifts and 3 adds.  Best Way ⇒ ( $2^6 + 2^5 + 2^4 + 2^1$ ) ≈ OxCC ⇒ ( $2^6 * oxCC$ ) + ( $2^5 * oxCC$ ) + ( $2^4 * oxCC$ )  Operation A B Overflow when result is left Floating Point bias sign exponent fraction A+B ≥ 0 ≥ 0 < 0 Single Precision 32 bits 127 1 8 23  A+B < 0 < 0 > 0 Double Precision 64 bits 1023 1 111 52  A-B ≥ 0 < 0 Single min-max -126 127  A-B ≥ 0 < 0 Single min-max -126 127  A-B < 0 ≥ 0 ≥ 0 Double min-max -1022 1023  Exponent Field Represents Denormalized number All 0s All 0s Double min-max -1022 1023  Exponent Field Represents Denormalized number ±0. fraction <sub>2</sub> * 2(exp-bias)  All 0s Not All 0s ±0. fraction <sub>2</sub> * 2(exp-bias)  All 1s Any ±1. fraction <sub>2</sub> * 2(exp-bias)  All 1s Not All 0s NaN  Convert the number, 216.875 to IEEE double-precision floating point.	$ (Exec \ time) \ old \Rightarrow \ t = \frac{\sum (IC + CPI)}{(0.5p^+ CPI)} \Rightarrow \frac{(1.28e^9 + 3) + (2.56e^9 + 8) + (64e^6 + 5)}{4e^9} = \frac{24.64e^9}{4e^9} = 6.16s $ $ (Exec \ time) \ new \Rightarrow \ t = \frac{\sum (IC + CPI)}{(0.5p^+ CPI)} \Rightarrow \frac{(1.28e^9 + 3) + (2.56e^9 + 8) + (64e^6 + 5)}{(0.5 + 16)^+ 3} + \frac{24.64e^9}{(0.5 + 16)^+ 8} = \frac{3.36e^9}{4e^9} = 840e - 3s $ $ = \frac{3.36e^9}{4e^9} = 840e - 3s $ $ = \frac{3.36e^9}{4e^9} = \frac$					
Branch ⇒ Op = 4 (same as  w)       Jump ⇒ OP = 2 (b. 26 br4 s)         X=2 and Y=3. If X and Y are represented by signed 4-bit integers, calculate X-Y using saturating arithmetic.       0       0       1       0       0       0       1       0       1       0       1       0       1       0       1       0       1       0       1       0       1       0       1       0       1       0       1       0       1       0       1       0       1       0       1       0       1       0       1       0       1       0       0       1	$ (Exec \ time) \ old \Rightarrow \ t = \frac{\sum (IC + CPI)}{(0.5p^+ CPI)} \Rightarrow \frac{(1.28e^9 + 3) + (2.56e^9 + 8) + (64e^6 + 5)}{4e^9} = \frac{24.64e^9}{4e^9} = 6.16s $ $ (Exec \ time) \ new \Rightarrow \ t = \frac{\sum (IC + CPI)}{(0.5p^+ CPI)} \Rightarrow \frac{(1.28e^9 + 3) + (2.56e^9 + 8) + (64e^6 + 5)}{(0.5 + 16)^+ 3} + \frac{24.64e^9}{(0.5 + 16)^+ 8} = \frac{3.36e^9}{4e^9} = 840e - 3s $ $ = \frac{3.36e^9}{4e^9} = 840e - 3s $ The following table shows results for SPEC benchmark programs running on a processor. Find the geometric mean. Name $   \ C * 10e^9   \ Execution \ Time \ (s) \ Reference \ Time$					
Branch ⇒ Op = 4 (same as  w)         Jump ⇒ OP = 2 (b. 26 brts)           X=2 and Y=3. If X and Y are represented by signed 4-bit integers, calculate X-Y using saturating arithmetic.         0         0         1         0         0         1         0         0         1         0         0         1         0         1         0         1         0         1         0         1         0         1         0         1         0         1         0         1         0         1         0         1         0         1         0         1         0         1         0         0         1         0         0         1         0         0         1         0         0         1         0         0         1         0         0         1         1         0         0         1<	$ (Exec \ time) \ old \Rightarrow \ t = \frac{\sum (IC + CPI)}{(0.5p^+ CPI)} \Rightarrow \frac{(1.28e^9 + 3) + (2.56e^9 + 8) + (64e^6 + 5)}{4e^9} = \frac{24.64e^9}{4e^9} = 6.16s $ $ (Exec \ time) \ new \Rightarrow \ t = \frac{\sum (IC \\ 0.5p^+ CPI)}{f} \Rightarrow \frac{(1.28e^9 + 3) + (2.56e^9 + 8) + (64e^6 + 5)}{(0.5 + 16)^+ 3} = \frac{24.64e^9}{4e^9} = 6.16s $ $ (Exec \ time) \ new \Rightarrow \ t = \frac{\sum (IC \\ 0.5p^+ CPI)}{f} \Rightarrow \frac{(1.28e^9 + 3) + (2.56e^9 + 8) + (64e^6 + 5)}{(0.5 + 16)^+ 3} = \frac{24.64e^9}{4e^9} = 6.16s $ $ (Exec \ time) \ new \Rightarrow \ t = \frac{\sum (IC \\ 0.5p^+ CPI)}{f} \Rightarrow \frac{(1.28e^9 + 3) + (2.56e^9 + 8) + (64e^6 + 5)}{f} = \frac{24.64e^9}{4e^9} = 6.16s $ $ (Exec \ time) \ new \Rightarrow \ t = \frac{\sum (IC \\ 0.5p^+ CPI)}{f} \Rightarrow \frac{(1.28e^9 + 3) + (2.56e^9 + 8) + (64e^6 + 5)}{f} = \frac{24.64e^9}{4e^9} = 6.16s $ $ (Exec \ time) \ new \Rightarrow \ t = \frac{\sum (IC \\ 0.5p^+ CPI)}{f} \Rightarrow \frac{(1.28e^9 + 3) + (2.56e^9 + 8) + (64e^6 + 5)}{f} = \frac{24.64e^9}{4e^9} = 6.16s $ $ (Exec \ time) \ new \Rightarrow \ t = \frac{1.28e^9}{f} = \frac{2.56e^9}{(0.5 + 16)^4} + \frac{1}{8} + (64e^6 + 5) = \frac{24.64e^9}{4e^9} = 6.16s $ $ (Exec \ time) \ new \Rightarrow \ t = \frac{\sum (IC \\ 0.5p^+ CPI)}{f} \Rightarrow \frac{(1.28e^9 + 3) + (2.6e^9 + 8)}{6.16} + \frac{1}{8} + \frac{1}{8} + \frac{1}{8} = \frac{1.36e^9}{4e^9} = 840e - 3s $ $ (Exec \ time) \ new \Rightarrow \ t = (Exec \ time) \ new \Rightarrow \ t = \frac{(Exec \ time) \ new \Rightarrow \ t = $					
Branch ⇒ Op = 4 (same as  w)         Jump ⇒ OP = 2 (b. 26 brts)           X=2 and Y=3. If X and Y are represented by signed 4-bit integers, calculate X-Y using saturating arithmetic.         0         0         1         0         0         1         0         0         1         0         0         1         0         1         0         1         0         1         0         1         0         1         0         1         0         1         0         1         0         1         0         1         0         1         0         1         0         1         0         0         1         0         0         1         0         0         1         0         0         1         0         0         1         0         0         1         1         0         0         1<	$ (Exec \ time) \ old \Rightarrow \ t = \frac{\sum (IC + CPI)}{(0.5p^+ CPI)} \Rightarrow \frac{(1.28e^9 + 3) + (2.56e^9 + 8) + (64e^6 + 5)}{4e^9} = \frac{24.64e^9}{4e^9} = 6.16s $ $ (Exec \ time) \ new \Rightarrow \ t = \frac{\sum (IC + CPI)}{(0.5p^+ CPI)} \Rightarrow \frac{(1.28e^9 + 3) + (2.56e^9 + 8) + (64e^6 + 5)}{(0.5 + 16)^+ 3} + \frac{24.64e^9}{(0.5 + 16)^+ 8} = \frac{3.36e^9}{4e^9} = 840e - 3s $ $ = \frac{3.36e^9}{4e^9} = 840e - 3s $ The following table shows results for SPEC benchmark programs running on a processor. Find the geometric mean. Name $   C * 10e^9   Execution \ Time \ (s)   Reference   Referenc$					
## Branch → Op = 4 (same as   \( \omega\$)   \( \omega\$)   \omega\$   \( \omega\$)   \(	$ (Exec \ time) \ old \Rightarrow \ t = \frac{\sum (IC + CPI)}{(0.5p^+ CPI)} \Rightarrow \frac{(1.28e^9 + 3) + (2.56e^9 + 8) + (64e^6 + 5)}{4e^9} = \frac{24.64e^9}{4e^9} = 6.16s $ $ (Exec \ time) \ new \Rightarrow \ t = \frac{\sum (IC + CPI)}{(0.5p^+ CPI)} \Rightarrow \frac{(1.28e^9 + 3) + (2.56e^9 + 8) + (64e^6 + 5)}{(0.5 + 16)^+ 3} + \frac{24.64e^9}{(0.5 + 16)^+ 8} = \frac{3.36e^9}{4e^9} = 840e - 3s $ $ = \frac{3.36e^9}{4e^9} = 840e - 3s $ The following table shows results for SPEC benchmark programs running on a processor. Find the geometric mean. Name $   C * 10e^9   Execution \ Time \ (s)   Reference \ Time \ (s)   Ref \ time \ Execution \ Time \ (s)   Ref \ time \ Execution \ Time \ (s)   Ref \ time \ Execution \ Time \ (s)   Ref \ time \ Execution \ Time \ (s)   Ref \ time \ Execution \ Time \ (s)   Ref \ time \ Execution \ Time \ (s)   Ref \ time \ Execution \ Time \ (s)   Ref \ time \ Execution \ Time \ (s)   Ref \ time \ Execution \ Time \ (s)   Ref \ time \ Execution \ Time \ (s)   Ref \ time \ Execution \ Time \ (s)   Ref \ time \ Execution \ Time \ (s)   Ref \ time \ Execution \ Time \ (s)   Ref \ time \ Execution \ Time \ (s)   Ref \ time \ Execution \ Time \ (s)   Ref \ time \ Execution \ Time \ (s)   Ref \ time \ Execution \ Time \ (s)   Ref \ time \ Execution \ Time \ (s)   Ref \ time \ Execution \ Time \ (s)   Ref \ time \ Time$					
	$ (Exec \ time) \ old \Rightarrow \ t = \frac{\sum (IC \circ CPI)}{f} \Rightarrow \frac{(1.28e^9 \cdot 3) + (2.56e^9 \cdot 8) + (64e^6 \cdot 5)}{4e^9} = \frac{24.64e^9}{4e^9} = 6.16s $ $ (Exec \ time) \ new \Rightarrow \ t = \frac{\sum (IC \ Sp}{(0.5p} \circ CPI)} \Rightarrow \frac{(1.28e^9 \cdot 3) + (2.56e^9 \cdot 8) + (64e^6 \cdot 5)}{(0.5 \cdot 16)^4 \cdot 3} + \frac{24.64e^9}{(0.5 \cdot 16)^4 \cdot 8} = \frac{24.64e^9}{4e^9} = 6.16s $ $ (Exec \ time) \ new \Rightarrow \ t = \frac{\sum (IC \ Sp}{(0.5p \cdot 16)^4} \circ \frac{(1.28e^9)}{(0.5 \cdot 16)^4} \circ (1.28e^$					
Branch   ⇒ Op = 4 (same as   ω)   Jump   ⇒ Op = 2 (b. 26 bt s)   S	$ (Exec \ time) \ old \Rightarrow \ t = \frac{\sum (IC \circ \circ FI)}{f} \Rightarrow \frac{(1.28e^9 \cdot 3) + (2.56e^9 \cdot 8) + (64e^6 \cdot 5)}{4e^9} = \frac{24.64e^9}{4e^9} = 6.16s $ $ (Exec \ time) \ new \Rightarrow \ t = \frac{\sum (IC \circ FI)}{(0.5p^+ \circ FI)} \Rightarrow \frac{(1.28e^9 \cdot 3) + (2.56e^9 \cdot 8) + (64e^6 \cdot 5)}{(0.5 \cdot 16)^+ \cdot 3} + \frac{3.36e^9}{(0.5 \cdot 16)^+ \cdot 8} = \frac{3.36e^9}{4e^9} = 840e - 3s $ $ The following table shows results for SPEC benchmark programs running on a processor. Find the geometric mean. Name                                   $					
## Branch → Op = 4 (same as   \( \omega\$)	$ (Exec \ time) \ old \Rightarrow t = \frac{\sum (1.28e^{9} \cdot 3) + (2.56e^{9} \cdot 8) + (64e6^{8} \cdot 5)}{4e^{9}} = \frac{2.64e^{9}}{4e^{9}} = 6.16s  $ $ (Exec \ time) \ new \Rightarrow t = \frac{\sum \left(\frac{1.05}{0.5p} \cdot CPI\right)}{\int \frac{1.28e^{9}}{(0.5 + 16)} \cdot 3\right) + \left(\frac{2.55e^{9}}{(0.5 + 16)} \cdot 8\right) + (64e6^{8} \cdot 5)}{4e^{9}} = \frac{3.36e^{9}}{4e^{9}} = 840e^{-3}s  $ $ = \frac{5 \left(\frac{1.05}{0.5p} \cdot CPI\right)}{4e^{9}} = \frac{\left(\frac{1.28e^{9}}{(0.5 + 16)} \cdot 8\right) + \left(\frac{2.55e^{9}}{(0.5 + 16)} \cdot 8\right) + \left(\frac{64e6^{8} \cdot 5}{4e^{9}}\right)}{4e^{9}} = \frac{840e^{-3}s}{4e^{9}} = \frac{840e^{-3}s}{4$					
## Branch → Op = 4 (same as   \( \omega\$)	$ (Exec \ time) \ old \Rightarrow t = \frac{\sum (1.28e^{9} \cdot 3) + (2.56e^{9} \cdot 8) + (6.4e6^{8} \cdot 5)}{4e^{9}} = \frac{2.64e^{9}}{4e^{9}} = 6.16s $ $ (Exec \ time) \ old \Rightarrow t = \frac{\sum (\frac{1.5}{0.5p} \cdot CPI)}{\int (0.5p^{8} \cdot CPI)} \Rightarrow \frac{(\frac{1.28e^{9}}{(0.5+16)} \cdot 3) + (\frac{2.56e^{9}}{(0.5+16)} \cdot 8) + (64e6^{8} \cdot 5)}{4e^{9}} = \frac{3.36e^{9}}{4e^{9}} = 840e - 3s $ $ The following table shows results for SPEC benchmark programs running on a processor. Find the geometric mean.  Name                                  $					
	$(Exec \ time) \ old \Rightarrow \ t = \frac{\sum (IC * OPI)}{f} \Rightarrow \frac{(1.28e^9 \cdot 3) + (2.56e^9 \cdot 8) + (64e^6 \cdot 5)}{4\cdot 9} = \frac{24.64e^9}{4\cdot 9} = 6.16s$ $(Exec \ time) \ new \Rightarrow \ t = \frac{\sum (IC \cdot OPI)}{f} \Rightarrow \frac{(1.28e^9 \cdot 3) + (2.56e^9 \cdot 8) + (64e^6 \cdot 5)}{f} = \frac{3.36e^9}{4\cdot 9} = 840e - 3s$ $The following table shows results for SPEC benchmark programs running on a processor. Find the geometric mean. Name                                   $					
A B Overflow when result is  A+B ≥ 0 ≥ 0 < 0 Single Precision 32 bits 127 1 8 23  A+B ≥ 0 < 0 < 0 Single Precision 32 bits 127  A-B ≥ 0 < 0 < 0 Single Precision 32 bits 1023 1 11 52  Exponent Field Fraction Field Represented By Signed With 18 All 0s  All 0s  Not All 0	$(Exec \ time) \ old \Rightarrow t = \frac{\sum (IC \circ CPI)}{f} \Rightarrow \frac{(1.28e^9 \cdot 3) + (2.56e^9 \cdot 8) + (64e^6 \cdot 5)}{4\cdot 9} = \frac{24.64e^9}{4\cdot 9} = 6.16s$ $(Exec \ time) \ new \Rightarrow t = \frac{\sum (IC \cdot CPI)}{f} \Rightarrow \frac{(1.28e^9 \cdot 3) + (2.56e^9 \cdot 8) + (64e^6 \cdot 5)}{(Exec \ time) \ new \Rightarrow t} = \frac{3.36e^9}{4\cdot 69} = 840e - 3s$ $The following table shows results for SPEC benchmark programs running on a processor. Find the geometric mean. Name                                   $					
A B vanch → Op = 4 (same as   ω)	$ (Exec time) old \Rightarrow t = \frac{\sum (IC \circ CPI)}{S} \Rightarrow \frac{(1.28e9 \circ 3) + (2.58e9 \circ 8) + (64e6 \circ 5)}{4e9} = \frac{24.69e}{4e9} = 6.16s $ $ (Exec time) new \Rightarrow t = \frac{\sum (IC \circ CPI)}{I} \Rightarrow \frac{(1.28e9 \circ 3) + (2.58e9 \circ 8) + (64e6 \circ 5)}{I} \Rightarrow \frac{24.69e}{4e9} = \frac{24.69e}{4e9} = 840e - 3s $ $ (Exec time) new \Rightarrow t = \frac{\sum (IC \circ CPI)}{I} \Rightarrow \frac{(1.28e9 \circ 3) + (2.58e9 \circ 8) + (64e6 \circ 5)}{I} \Rightarrow \frac{24.69e}{4e9} = \frac{840e - 3s}{4e9} = \frac{840e - 3s}{$					
A	$ (Exec time) old \Rightarrow t = \frac{\sum (IC \circ CPI)}{S} \Rightarrow \frac{(1.28e9 - 3) + (2.56e9 + 8) + (64e6 + 5)}{4e9} = \frac{24.64e9}{4e9} = 6.16s $ $ (Exec time) new \Rightarrow t = \frac{\sum (IC \circ CPI)}{I} \Rightarrow \frac{(1.28e9 - 3) + (2.56e9 + 8) + (64e6 + 5)}{I} \Rightarrow \frac{24.64e9}{I} \Rightarrow \frac{24.64e9}{I} = 6.16s $ $ (Exec time) new \Rightarrow t = \frac{\sum (IC \circ CPI)}{I} \Rightarrow \frac{(1.28e9 - 3) + (2.56e9 + 8) + (64e6 + 5)}{I} \Rightarrow \frac{24.64e9}{I} \Rightarrow \frac{24.64e9}{I} = 8.40e - 3s $ $ (Exec time) new \Rightarrow t = \frac{\sum (IC \circ CPI)}{I} \Rightarrow \frac{(IC \circ TO \circ CPI)}{I} \Rightarrow \frac{1.28e9}{I} \Rightarrow 1.28$					
A B   Overflow when result is   IEEE Floating Point   Diason   Single Precision 64 bits   Diason   Fraction   Single Precision	$ (Exec time) old \Rightarrow t = \frac{\sum (IC \circ CPI)}{1} \Rightarrow \frac{(1.28e9 - 3) + (2.58e9 + 8) + (64e6 + 5)}{4e9} = \frac{24.64e9}{4e9} = 6.16s $ $ (Exec time) new \Rightarrow t = \frac{\sum (IC \circ CPI)}{I} \Rightarrow \frac{(1.28e9 - 3) + (2.55e69 + 8) + (64e6 + 5)}{I} \Rightarrow \frac{24.64e9}{I} = 840e - 3s $ $ (Exec time) new \Rightarrow t = \frac{\sum (IC \circ CPI)}{I} \Rightarrow \frac{(1.28e9 - 3) + (2.55e69 + 8) + (64e6 + 5)}{I} \Rightarrow \frac{24.64e9}{I} = 840e - 3s $ $ (Exec time) new \Rightarrow t = \frac{\sum (IC \circ CPI)}{I} \Rightarrow \frac{(1.28e9 - 3) + (2.55e69 + 8) + (64e6 + 5)}{I} \Rightarrow \frac{3.36e9}{4e9} = 840e - 3s $ $ (Exec time) new \Rightarrow t = \frac{\sum (IC \circ CPI)}{I} \Rightarrow \frac{(1.28e9 - 3) + (2.56e9 + 8) + (64e6 + 5)}{I} \Rightarrow \frac{3.36e9}{I} = 840e - 3s $ $ (Exec time) new \Rightarrow t = \frac{Execution time}{I} \Rightarrow \frac{669 + (696 - 3)}{I} \Rightarrow \frac{3.36e9}{I} = 840e - 3s $ $ (Exec time) new \Rightarrow t = \frac{Execution time}{I} \Rightarrow \frac{669 + (696 - 3)}{I} \Rightarrow \frac{3.36e9}{I} = 840e - 3s $ $ (Exec time) new \Rightarrow t = \frac{Execution time}{I} \Rightarrow \frac{669 + (696 - 3)}{I} \Rightarrow \frac{3.36e9}{I} = 840e - 3s $ $ (Exec time) new \Rightarrow t = \frac{Execution time}{I} \Rightarrow \frac{669 + (696 - 3)}{I} \Rightarrow \frac{3.36e9}{I} \Rightarrow \frac{8993}{I} \Rightarrow \frac{3.36e9}{I} \Rightarrow \frac{8993}{I} \Rightarrow \frac{8993}{I} \Rightarrow \frac{8993}{I} \Rightarrow \frac{18993}{I} \Rightarrow 189$					
A	$ (Exec time) old \Rightarrow t = \frac{\sum (IC \circ CPI)}{S} \Rightarrow \frac{(1.28e9 \circ 3) + (2.58e9 \circ 8) + (64e6 \circ 5)}{4e9} = \frac{24.64e9}{4e9} = 6.16s $ $ (Exec time) new \Rightarrow t = \frac{\sum (IC \circ CPI)}{I} \Rightarrow \frac{(1.28e9 \circ 3) + (2.56e9 \circ 8) + (64e6 \circ 5)}{I} \Rightarrow \frac{24.64e9}{I} \Rightarrow \frac{24.64e9}{I} = 8.10e $ $ (Exec time) new \Rightarrow t = \frac{\sum (IC \circ CPI)}{I} \Rightarrow \frac{(1.28e9 \circ 3) + (2.56e9 \circ 8) + (64e6 \circ 5)}{I} \Rightarrow \frac{24.64e9}{I} \Rightarrow \frac{24.64e9}{I} = 8.40e - 3s $ The following table shows results for SPEC benchmark programs running on a processor. Find the geometric mean. Name $ (IC \circ 10e9) = \frac{Execution Time (s)}{Execution} = \frac{Execution Time (s)}{Execution} = \frac{Execution Time (s)}{I} = \frac{Execution Execution}{I} = \frac{Execution Execution}{I} = \frac{Execution Execution Execution}{I} = \frac{Execution Execution Execution}{I} = \frac{Execution Execution}{I} = \frac{Execution Execution Execution}{I} = Execution Execution$					