Project ‘GAMERA’

(Semi-Powerful Console(Windows & Linux) Tools & gigabytes of English texts, downloadable from www.sanmayce.com)

WHERE THE WORD COUNTS

Caterpillar(LZSS-King of Brute-Force Heavy Sentence Dumpers, 32bit console application), revision 14+
Kazuya(LZ-Sovereign of Brute-Force Heavy Sentence Dumpers, 32bit console application), revision 14++
Salah-ed-din(GZ-Sultan of Brute-Force Heavy Sentence Dumpers, 32bit console application), revision 14++
Raccoondog(LZMA-Baron of Brute-Force Heavy Sentence Dumpers, 32bit console application), revision 14++
Yoshi(Filelist creator and more, 32bit console application), revision 06
Leprechaun(Fast and Greedy Word_Ripper, 32bit console application), revision 12

WinRAR archive in ten 621MB volumes • Required HDD space: 5.92 GB • 2009 MAR 04

Kazuya delivers english sentences at 85-255MB/s speed(Obtained with Toshiba Satellite L305 (Intel Pentium(Merom-1M) T3400 2.16GHz))
Salah-ed-din delivers english sentences at 114-117MB/s speed(Obtained with Toshiba Satellite L305 (Intel Pentium(Merom-1M) T3400 2.16GHz))
Raccoondog delivers english sentences at 39MB/s speed(Obtained with Toshiba Satellite L305 (Intel Pentium(Merom-1M) T3400 2.16GHz))
Leprechaun rips 2,636,331 words per second(Obtained with Toshiba Satellite L305 (Intel Pentium(Merom-1M) T3400 2.16GHz))
LBL stands for Line-By-Line(GRAMMATICAL ENGLISH LINES) i.e. sentences not merely CRLF or LF lines!
.LBL files are made from .TXT files which are made from respective .DOC, .RTF, .LIT, .PDF, .CHM, .HTM[L], .DJV[U] files;
Number and size of *.LBL files: 512,561 files(24.9GB or 26,800,604,148 bytes);
Lines and words in *.LBL files: 408,530,953 lines(with 4,386,856,249 words of them 8,817,135 distinct);

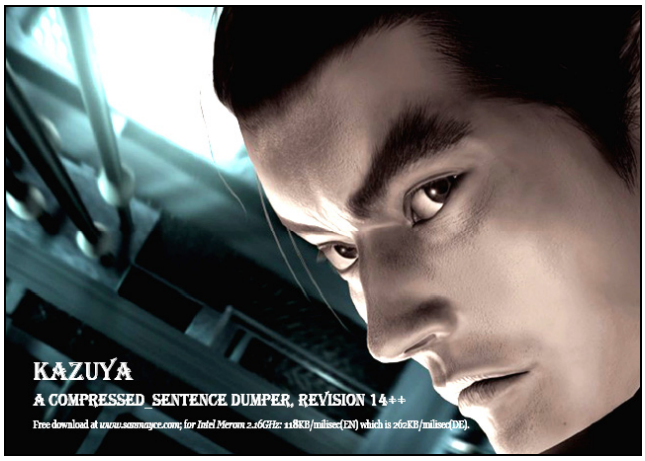
'**Monstrous Dumpers**' package, revision 12

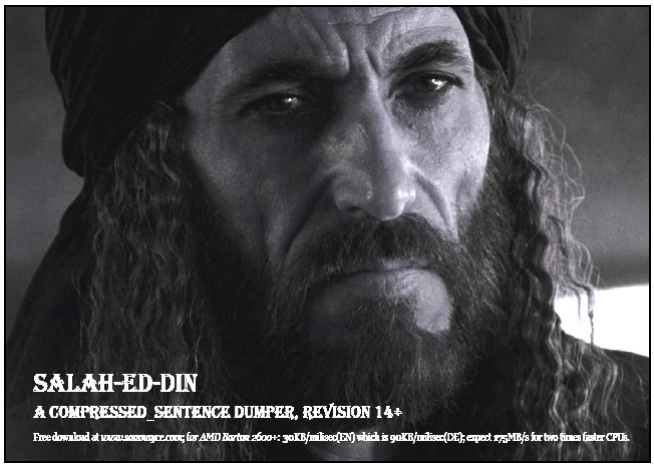
OR

How words can be mixed into sentences!?

With this package(the main part of project "**GAMERA**") you can make full-text(brute-force) requests into millions of lines(sentences). For example: make a search for **requests???????????into** to see whether that preposition has place near on right side of "requests".
This package(a Winrar archive) is intended as shareware and contains six very fast 32bit console text tools: **Caterpillar** (its rivals **Raccoondog**, **Salah-ed-din** and **Kazuya**), **Leprechaun**, **Yoshi** and of course 100++ million sentences(in English language) from various sources.

The package allows easily to create:
- a FILELIST(a text file with filenames);
- a WORDLIST(a text file with sorted distinct words);
- and as a main feature a text-pattern to be searched into LF(Unix)|CRLF(Windows) lines(or files) via filelist and to dump resultant hits(lines or filenames) into .HTML file.





Main features:

- 24.9GB english-ASCII-texts converted to .LBL(same as .TXT but each line is a sentence) format;
- File-by-file listings of all texts included:

```
    164,128 KAZE_G.S._Corpus_'.chm'_Caterpillar.html
     16,817 KAZE_G.S._Corpus_'.djv-'_Caterpillar.html
     61,142 KAZE_G.S._Corpus_'.doc'_Caterpillar.html
 22,742,832 KAZE_G.S._Corpus_'.htm-'_Caterpillar.html
     32,717 KAZE_G.S._Corpus_'.lit'_Caterpillar.html
  1,905,255 KAZE_G.S._Corpus_'.pdf'_Caterpillar.html
     30,535 KAZE_G.S._Corpus_'.rtf'_Caterpillar.html
 27,389,047 KAZE_G.S._Corpus_'.txt'_Caterpillar.html
```

- Wildcards(*,?) available for patterns: very slow(pattern ***underdog*** took 490 seconds to look for into 400+ million sentences) but powerful;
- Fast(for laptops with a non-SSD disk)(90++MB/s) full-text traversing due to **zlib** used;
- Extra Fast(for laptops with a SSD disk)(200++MB/s) full-text traversing due to **QuickLZ** used;
- Results are delivered as *screen output* immediately and as *pure HTML files* finally;
- 'Karp_Rabin_Kaze'(patterns ***, underdog** took 70 seconds to look for into 400+ million sentences) compared to 'strstr'(85s) & 'Boyer-Moore-Horspool'(86s) is ((85-70)/70)*100%≈21% faster when running on sentences.

Installation(i.e. extracting) notes:

- Unrar in D:\ if possible, "Caterpillar.lnk", "Go to PROMPT.lnk", "Raccoondog.lnk", "Salah-ed-din.lnk", "Kazuya.lnk" need manual adjustments if not D:\, 6GB must be free.
- To use "**Caterpillar.lnk**" and "**Salah-ed-din.lnk**" and "**Kazuya.lnk**" must run (R2O.BAT) and (R2G.BAT) and (R2L.BAT) respectively.

Current revisions of tools:

- ***EXEs(Windows):***

Caterpillar	r.14+
Leprechaun	r.12
Yoshi	r.06
Salah-ed-din	r.14++
Raccoondog	r.14++
Kazuya	r.14++ r.15 has an ability to search non-compressed files too!

- ***ELFs(Linux):***

Caterpillar	r.14+
Leprechaun	r.12
Yoshi	r.06
Salah-ed-din	r.14+

Note1: Revisions 14++ are Experimental(but operational, not beta) Karp-Rabin function with my hash, see last page.
Note2: Predecessor of *Caterpillar*, *Salah-ed-din* & *Raccoondog* was **Kazuya**(with more functionality and critical parts written in 16bit assembler), someday I will resurrect him in 64bit.

Convert at will:

Use G2R.BAT for .gz -> .lzma (1000+ minutes needed to convert, grmbl)
Use R2G.BAT for .lzma -> .gz (11:05 PM - 12:18 AM i.e 73 minutes needed to convert)
Use R2L.BAT for .lzma -> .Lasse (06:57 PM - 07:45 PM i.e 48 minutes needed to convert)
Use R2O.BAT for .lzma -> .Okumura (11:33 PM - 01:22 AM i.e 109 minutes needed to convert)

Some experience(Machine: Toshiba Satellite L305 - Intel Pentium Dual CPU T3400 @ 2.16GHz):

- **Caterpillar** uses **LZSS**(based on LZSS.C written by H.Okumura) compression;
  24.9GB -> 11.4 GB (12,341,932,922 bytes);
  delivering text at 82KB(*149KB when in system cache*)/clock i.e. 80MB/s;
  suitable for FAST HDDs 80+MB/s.
- **Raccoondog** uses **LZMA**(based on LZMA SDK 4.65 written by I.Pavlov) compression;
  24.9GB -> 5.5 GB (5,944,631,607 bytes);
  delivering text at 40KB(*bottleneck is CPU power alone*)/clock i.e. 39MB/s;
  suitable for flash cards like CFs, SDs.
- **Salah-ed-din** uses **GZ**(based on zlib 1.2.3 written by J.Gailly and M.Adler) compression;
  24.9GB -> 8.4 GB (9,051,049,655 bytes);
  delivering text at 117KB(*120KB when in system cache*)/clock i.e. 114MB/s;
  suitable for FAST CPUs 3+GHz.
- **Kazuya** uses **LZ**(based on QuickLZ 1.4.0 written by Lasse Reinhold) compression;
  24.9GB -> 10.6 GB (11,402,975,168 bytes);
  delivering text at 88KB(*262KB(118KB(EN)) when in system cache*)/clock i.e. 85MB/s;
  suitable for FAST SSDs 115+MB/s. Near future dreams: CPU(2x faster) and SSD(2x115Mb/s read) will give 2x255Mb/s.

Search|Seek|Find in order to Explore|Learn|Avoid Different Styles:

["Супруга съм на три деца. С чувекъ сбрахме пари и купихми триустаен партамент. Една вечер звъни вратата. Звънецъ чука. Отварям - НИНДЖА. И без да каже нищо, с карате в бъбреците. Дукат съ усета ми би два шамара с КРАК и един на детето в гръбначнийъ кош! От ударната вълна отльитам на 20-30 метра. Абстрахираха децата. А чувекъ го нема. Ако общината в града не вземе спешни мерки, ще се самуубеся ильи ще изчезна безкрайно."]

/Интервю с ромка излъчено по КАНАЛ 1 за акция на НСБОП по залавяне на опасни рецидивисти в Пазарджишко./

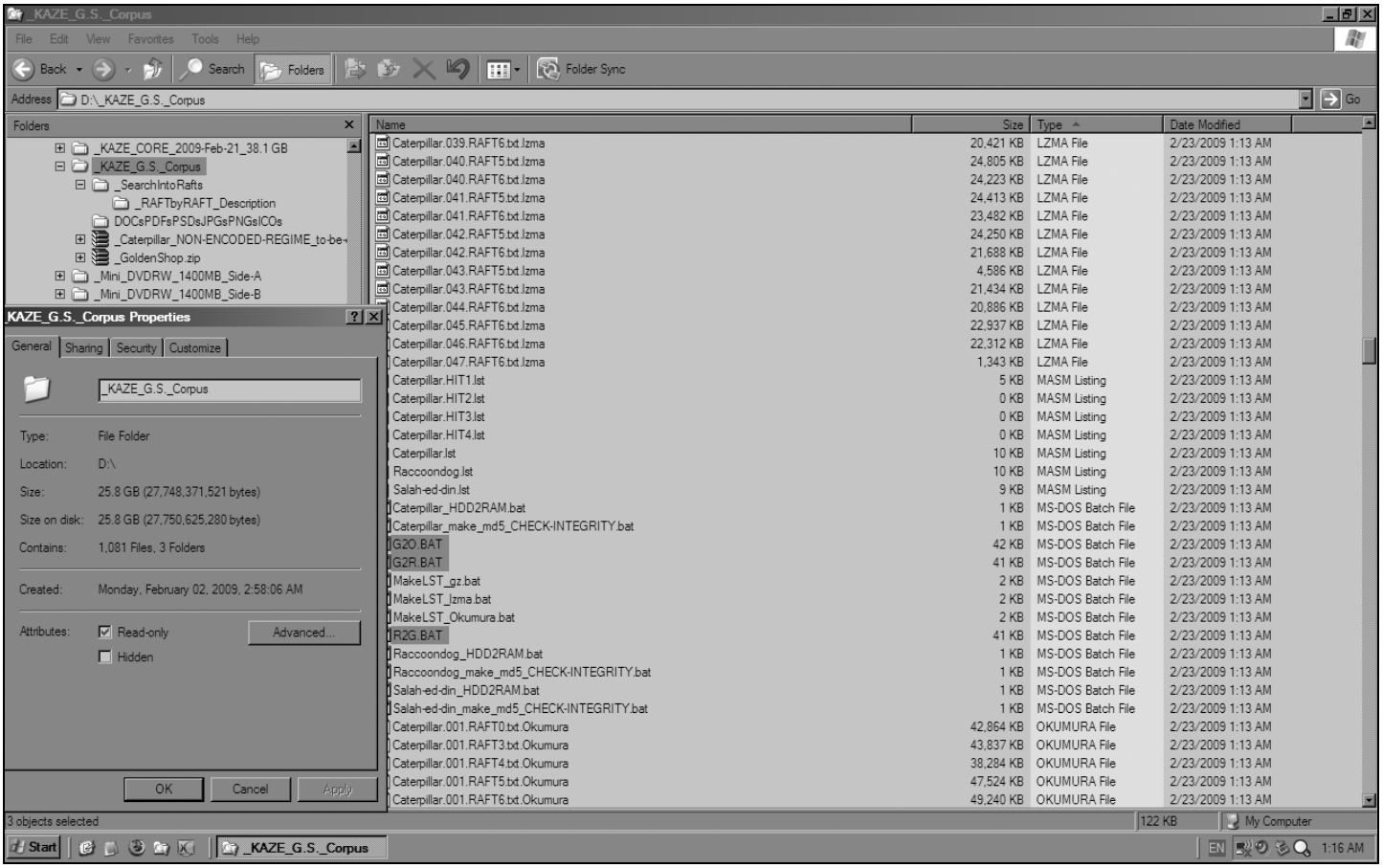
Enjoy!
Sanmayce 'Kaze', 2009 Mar 13.

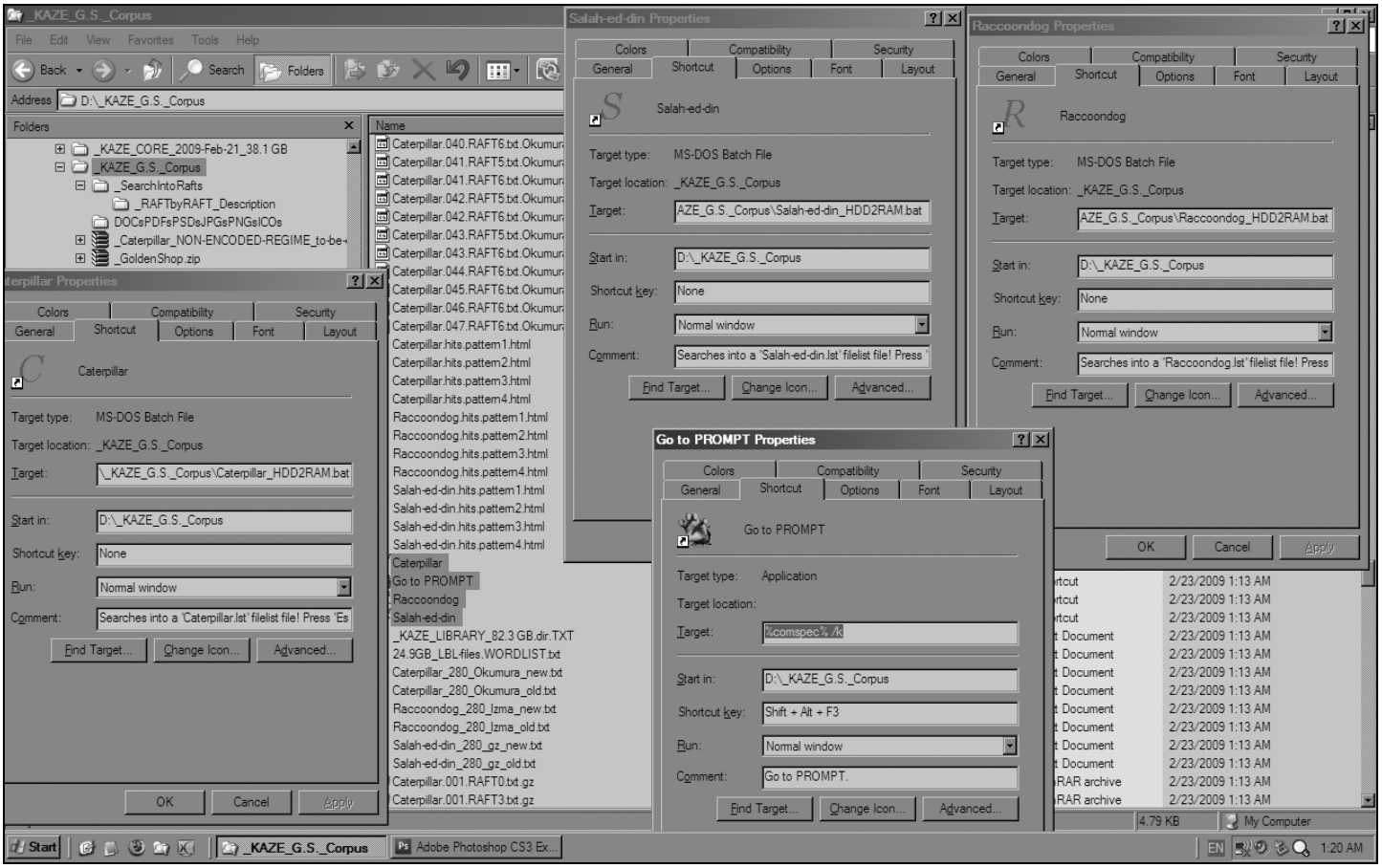
```
D:\_KAZE_G.S._Corpus>yoshi
Yoshi(Filelist Creator), revision 06, written by Svalqyatchx,
in fact based on SWEEP.C from 'Open Watcom Project', thanks-thanks.

Note1: So far, it works for current directory only.
Note2: Default method is depth-first traversal;
       may use pipe 'Yoshi|sort' for breadth-first_like traversal results.
Note3: Make notice that '*.*'(extensionfull only) is not equal to '*'(all);
       one disadvantage is an inability to list only extensionless filenames.
Note4: Search is case-insensitive as-must.
Note5: This revision allows multiple '*', and meaning of masks is:
       '?' - any character AND NOT EMPTY(default, for OR EMPTY see option -e);
       '*' - any character(s) or empty.
Note6: What is a .LBL(LineByLine) file?
       it is a bunch of GRAMMATICAL lines not mere LF or CRLF lines;
       it contains not symbols under 32(except CR and LF) and above 127;
       it contains not space symbol sequences.
Usage:
      Yoshi [option(s)] [filename(s)]
      option(s):
         -v           i.e. verbose mode; output goes to console;
         -f           i.e. fullpath mode for output;
         -e           i.e. treat '?' as any character OR EMPTY;
         -t           i.e. touch all encountered files;
         -2           i.e. convert all encountered .TXT files to .LBL files;
         -o<filename> i.e. output goes to file(in append mode).
      filename(s):
         Wildcards '*' and wildcards '?' are allowed i.e. "str*.c??";
         default filename is '*'; DO NOT FORGET TO PUT
         filename(s) WITH WILDCARD(S) INTO QUOTE MARKS!
Examples:
      Yoshi -v -f -oCaterpillar_NON.lst "*.lbl" "*.txt" "*.htm" "*.html"
      Yoshi -f -oMyEbooks.txt "*wiley*essential*.pdf" "*russian*.*htm"

Yoshi: Total size of files: 00,027,750,342,332 bytes.
Yoshi: Total files: 000,000,001,088.
Yoshi: Total folders: 0,000,000,003.

D:\_KAZE_G.S._Corpus>

D:\_KAZE_G.S._Corpus>Leprechaun
Leprechaun(Fast and Greedy Word_Ripper), revision 12, written by Svalqyatchx.
Leprechaun: 'Oh, well, didn't you hear? Bigger is good, but jumbo is dear.'

'The Little Monster' short notes:
Note1: I wish to thank to R.N. Horspool, Ranjan Sinha, Dmitry Shkarin, Michael
       Abrash, J. Bentley, R. Sedgewick for sharing their knowledge to public.
Note2: Run it without parameters to get usage and short notes:
Note3: This simple amateurish(more over I am not versed well neither in C nor
       in mathematics nor in english language, but I am persistent in INDEXING
       GBs of english TEXTS) tool is written in ANSI C(at least its source is
       compileable for CL(Windows) and GCC(Linux)), and its purpose is to
       create a WordList for a group of files(given via filelist).
       Its name comes(according to Heritage Dictionary) from 'low corpus' or
       'little body', in fact from amazing movie saga 'Leprechaun 1-2-3-4-5-?'
       starring by Warwick Davis.
Note4: Only words up to 31 chars are proceeded.
Note5: Cursor hiding in C - mission impossible for me.
Note6: By default(third parameter is 527) allocated memory is 221MB.
       Due to 'malloc()' limitation under WINDOWS, maximum value of third
       parameter is 4582 which is 1914MB allocated block.
Note7: File Leprechaun.LOG is a log, where new statistics are appended.
Note8: For 'D:\temp>dir E:\KAZEHOME\*.lbl/b/s>KAZE_library_LBL.lst'
           'D:\temp>leprechaun KAZE_library_LBL.lst KAZE_library.lep 900'
       on SIS741GX-M + Duron 1300MHz FSB(2x100MHz) + 256MB DDR 266MHz(2x133MHz)
       with Windows Server 2003 on IDE HDD Hitachi Deskstar 160GB 7200 2MB and
       Allocated memory in MB: 376
       Number of Files in KAZE_library_LBL.lst: 67,894(14,052,659,814 bytes)
       Number of Words(total): 2,414,464,937
       Number of Words(distinct in total) in KAZE_library.lep: 3,873,717
       result is 2,765 seconds.

Usage: Leprechaun InFile OutFile [BufferSize] [SortMethod]
      <InFile>: Input file with files for Leprechauning, in WINDOWS console
                you can create it by 'E:\KAZEHOME>dir *.txt/s/b>Leprechaun.lst'
      <OutFile>: Output WORDLIST(sorted since r.9, CRLF) file
      <BufferSize>: Optional Dynamic RAM buffer in KB, default(and minimum
                    in the same time) is 527, i.e. omit or specify greater one
      <SortMethod>: Optional Sort Method, default is 'D',
                    A - InsertionSort
                    B - InsertionX26Sort
                    C - MultiKeyQuickSortSort by J. Bentley, R. Sedgewick
                    D - MultiKeyQuickSortX26Sort' by J. Bentley, R. Sedgewick

Have a nice Leprechauning.
For contacts: sanmayce@hotmail.com
Sanmayce Svalqyatchx 'Kaze', 2005 Feb 07.

D:\_KAZE_G.S._Corpus>

D:\_KAZE_G.S._Corpus>Caterpillar
Caterpillar(Sentence_Dumper), revision 14+, written by Svalqyatchx,
in fact adapted from Haruhiko Okumura's excellent LZSS.C program.

       How near are these words_forms to me: Masakari, Massacre, Steel-Coloss,
Monster-Truck, Dump-Mining-Truck, Caterpillar 797, Liebherr, Komatsu. They
resemble one thing: strong-devoid-of-ambition-power(i.e. a pure work/time).

       'Caterpillar' is a simple pattern searcher(from 'Masakari' family tools)
into archived english-text files, designed to achieve up to 90% higher read
speed than the HDD READ BURST i.e. 'copy hugefile nul' gaining at same time
50% compression of searched data.

       Its main feature is somewhat hidden nowadays, because of
pseudo-transparent decompression used, which leads to doubling(unreachable in
fact) uploaded data for search function(written by N. Horspool, thanks a lot)
due to LZSS algorithm implemented by H. Okumura(greetings to him). Okumura's
variant(HDD2RAM) which is much faster(!!!) and needs less memory than tuned
memory-to-memory decompression(RAM2RAM) variant. I am still stunned.

       In few words: feeding search function is 100-% faster with very fast
CPU-Physical_RAM subsystems, in this way reducing the ugly penalty which comes
```

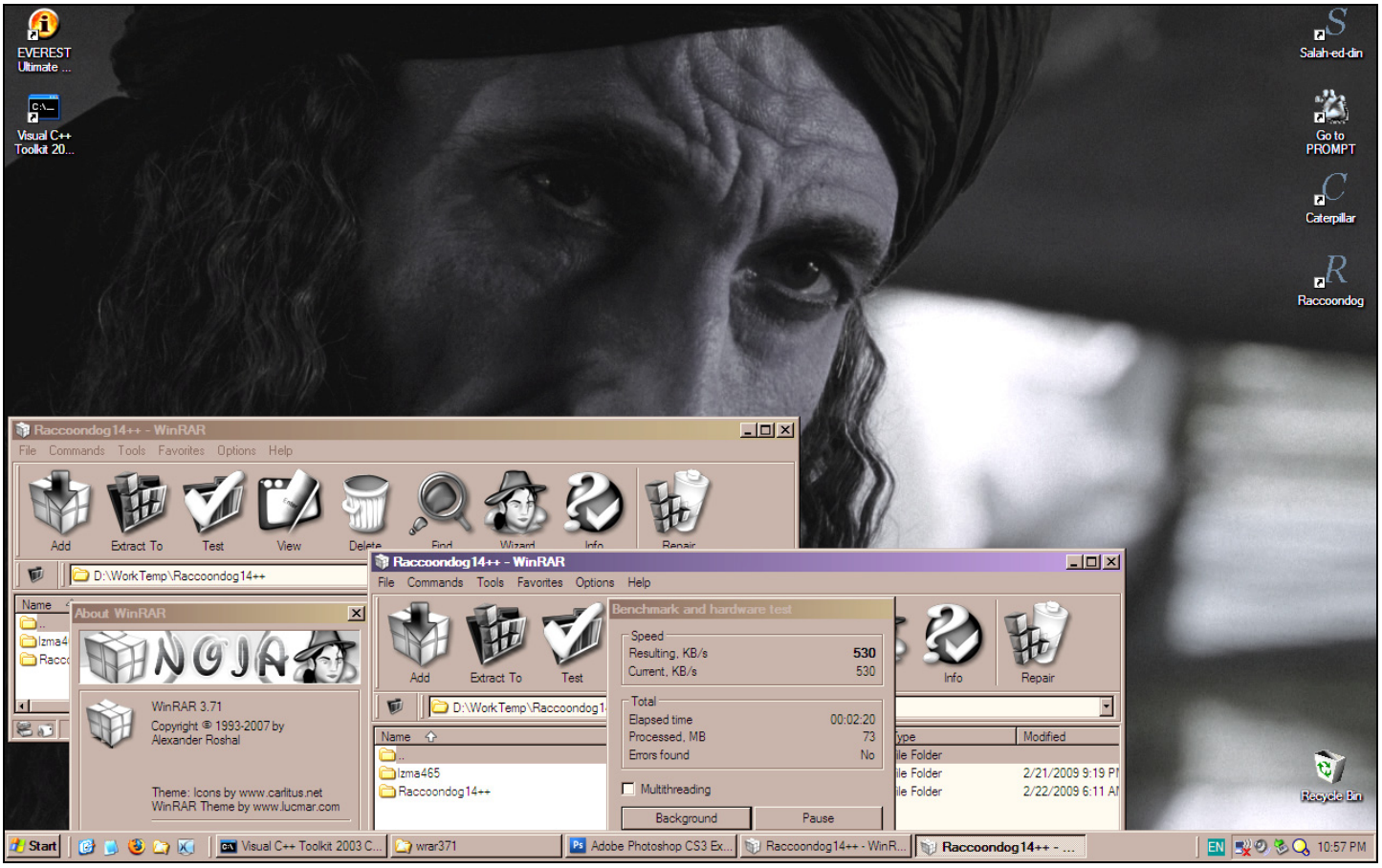
```
from reading a HDD. In numbers: me IDE HITACHI 7200rpm 2MB gives up to 60MB/s
READ BURST, 'Caterpillar' almost doubles(i.e. 120-MB/s) it in case of 3+++GHz
CPU and 533+++MHz RAM.

       For Windows 2003, VIA KT600, AMD XP 2500+(1836.12MHz=11x166.92MHz),
FSB 333.84MHz(2x166.92MHz), 512KB L2 cache, 1 DIMM DDR 512MB 333MHz(2x166MHz),
Caterpillar(in fact LZSS) decompresses 58,000KB per second i.e.
boost is negative: 60MB/s=61,440KB/s(READ BURST) is greater than 58,000KB/s.
But for two times faster CPU-RAM sub-system(SERVER) than described above OR
for two times slower HDD sub-system(LAPTOP) boost will be positive:
(1 - READ BURST SPEED / DECOMPRESSION SPEED) * READ BURST SPEED or
(1 - (61,440KB/s) / (2 * 58,000KB/s)) * 61,440KB/s = (0.471) * 61,440KB/s.

       Since revision 5 'fread()' was changed with 'read()', for speed.

'The Monster-Dump-Truck' short notes:
Note1: Thanks a lot to N. Horspool, Dmitry Shkarin, H. Okumura, Igor Pavlov.
Note2: Run it without parameters to get usage and short notes.
Note3: Current revision searches only for case-sensitive and unexact matches.
Note4: This simple amateurish(more over I am not versed well neither in C nor
       in mathematics nor in english language, but I am persistent in INDEXING
       GBs of english TEXTS) tool is written in ANSI C(at least its source is
       compileable for CL(Windows) and not yet for GCC(Linux) because of
       'O_BINARY in open(), gets(), getch(), kbhit()', and its purpose is to
       create a SentenceList for a group of compressed(with it) text files(LF
       and CRLF) given via filelist.
       Its name comes from a heavy-nopride-dumper-truck 'Caterpillar'.
Note5: By default allocated memory is 95MB i.e. decoding is HDD2RAM.
Note6: Disastrous performance in case 95MB|147MB not fully physical!
Note7: For me digital library:
       where files are 54, ENcoded 6,917,425,566, DEcoded 14,419,485,826
       with Windows XP, VIA KT600, AMD XP 2500+(1836.12MHz=11x166.92MHz),
       FSB 333.84MHz(2x166.92MHz), 512KB L2 cache, DDR 512MB 333MHz(2x166MHz),
       IDE HDD Maxtor 80GB 7200 8MB and
       'D:\temp>dir E:\KAZEHOME\KAZUYA.0??/b>Caterpillar.lst'
       'D:\temp>Caterpillar Caterpillar.lst CaterpillarRAM2RAM.ini'
       result is: 282 seconds or 41000KB/s upload, 11000KB/s boost,
       52000KB/s boosted upload, 56000KB/s decode.
       'D:\temp>Caterpillar Caterpillar.lst CaterpillarHDD2RAM.ini'
       result is: 142 seconds or 99000KB/s boosted upload!!!
Note8: Matches(hits) containing neither '<' nor '>' are written
       to 'Caterpillar.hits.pattern?.html' file.
Note9: Works both on UNIX(LF) and Windows(CRLF) text files.
NoteA: Never forget the importance of defragmented_AND_grouped files located at
       fastest area of disk - first partition is faster than second one, etc.
NoteB: In ANSI, clock is defined as '#define CLOCKS_PER_SEC 1000'.
NoteC: Since Caterpillar 13++:
       - limits(just skip longer ones) lines to 960 chars; OTHERWISE: HUGE TIME
         DELAYS due to recursive function;
       - shows hits to console too; MORE VIVID;
NoteD: During execution hitting a 'Esc' causes termination(i.e. skipping rest).
NoteE: At last NON-ENCODED regime has two modes: in addition to LINE(i.e.
       hits are lines) there is a FILE(i.e. hits are filenames) mode.
NoteF: For all regimes files Caterpillar.HIT?.lst are created for each
       pattern(1,2,3 and 4) - containing hits filelist i.e. filenames
       containing HITS(either LINEs or FILENAMEs).

Below is LINE(default for DECODING ????RAM regimes) mode pattern description:
Pattern(s) note: You may specify(four times) a main-pattern(case insensitive
                 with wildcards '*' i.e. any character(s) or empty and '?'
                 i.e. any character or empty) with three nested-patterns(case
                 sensitive and unexact), all four connected with AND.
                 Due to different line endings(CRLF in Windows; LF in UNIX)
                 you must add a '?' wildcard in place of CR: for example in
                 case of searching for '*.pdf' write '*.pdf?'.
Pattern(s) example: Pattern1: *take? *it*
                    Pattern1_NestedPattern1: you
                    Possible hit: ... your reason is so taken by It.

Usage: 'Caterpillar e file1 file2' encodes file1 into file2
       'Caterpillar d file2 file1' decodes file2 into file1
       'Caterpillar m ListOfFilesFile SolidSize'
         <ListOfFilesFile>: Files to be merged into Caterpillar.??? files
         <SolidSize>: Caterpillar.??? files size limit in MB.
       'Caterpillar ListOfFilesFile [OptionsFile]'
         <ListOfFilesFile>: Input file with files for Caterpillaring
         <OptionsFile>: Optional input file with options with following format:
         Optional line #1 contains method of decoding:
           'DECODING HDD2RAM' | 'DECODING RAM2RAM' | 'NON-ENCODED'
           'NON-ENCODED' allocates 95MB, size of biggest file must be lower;
           'DECODING HDD2RAM' needs less physical memory(95MB) but is faster!
           'DECODING RAM2RAM' needs more physical memory(147MB) but is slower!
         Optional line #2 contains terminal hits:
           '0' | 'long integer'
           '0' means all hits are needed
           'long integer' means reaching this value termination follows
         Optional line #3 contains Pattern1: 'string'
           if 'string' is specified then input from keyboard arise not
           if 'string' is not specified then input from keyboard arise
         Optional line #4 contains Pattern1_NestedPattern1: 'string'
         Optional line #5 contains Pattern1_NestedPattern2: 'string'
         Optional line #5 contains Pattern1_NestedPattern3: 'string'
Note1: One useful way to make 'ListOfFilesFile=Caterpillar_NON.lst' is next:
D:\Caterpillar>copy con MAKElst.bat
@echo off
dir Caterpillar_tree\*.lbl /s/b>Caterpillar_NON.lst
dir Caterpillar_tree\*.txt /s/b>>Caterpillar_NON.lst
echo.
F6

Have a nice Caterpillaring.
For contacts: sanmayce@hotmail.com
Sanmayce Svalqyatchx 'Kaze', 2009 Jan 29.

D:\_KAZE_G.S._Corpus>

D:\_KAZE_G.S._Corpus>Raccoondog.exe

LZMA Utility 4.65 : Igor Pavlov : Public domain : 2009-02-03

Usage:  lzma <e|d> inputFile outputFile
  e: encode file
```

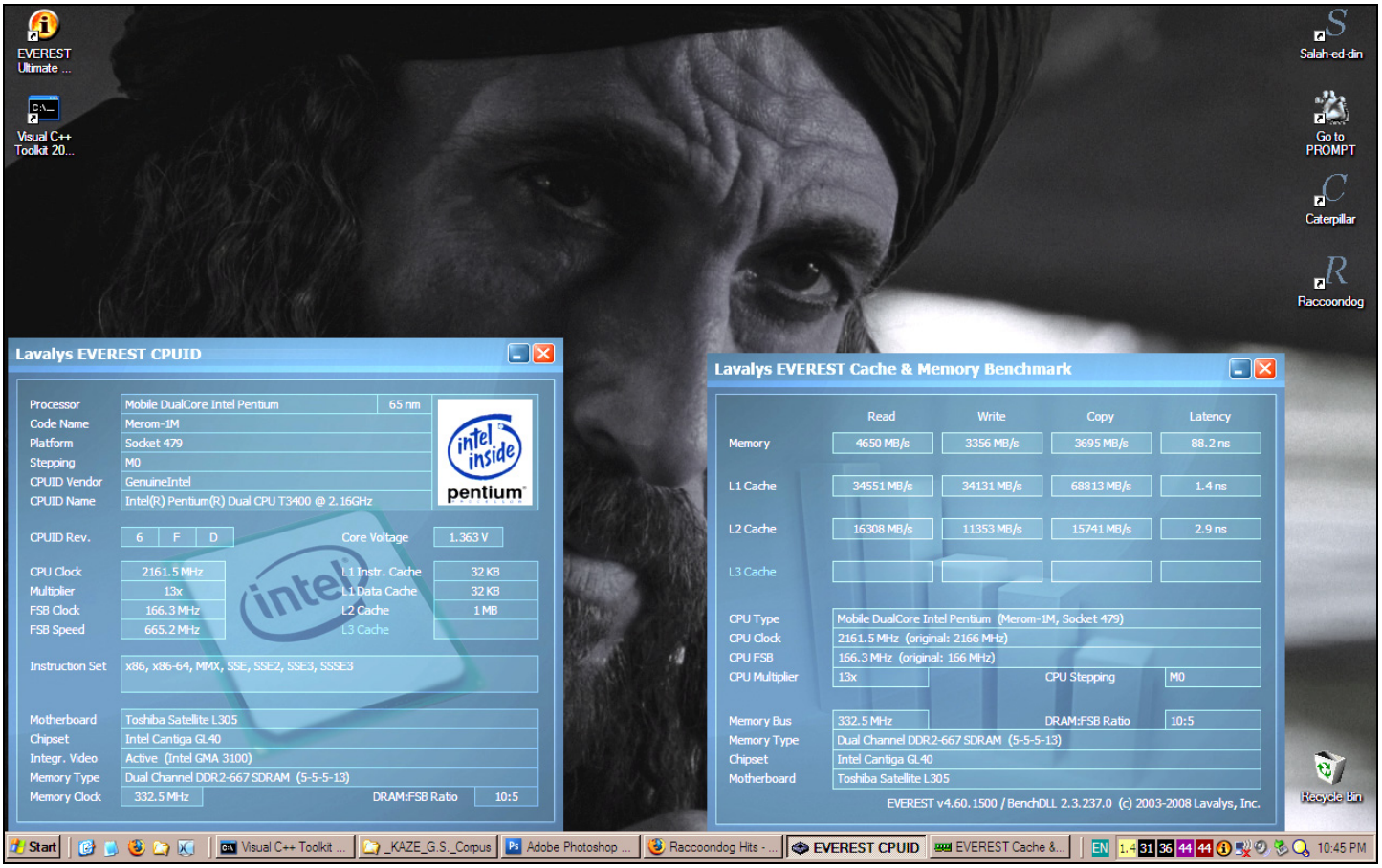
```
  d: decode file

D:\_KAZE_G.S._Corpus>

D:\_KAZE_G.S._Corpus>Raccoondog -SA4 Raccoondog.lst
Raccoondog(LZMA Sentence_Dumper), revision 14++, written by Svalqyatchx,
in fact adapted from Igor Pavlov's excellent LZMA 4.56 SDK.

Usage1: Raccoondog [-SA1|-SA2|-SA3|-SA4] filename
        Decodes all files from a list(filename)
        -SA1 : Brute_Force Search Algorithm
        -SA2 : Quick_Boyer_Moore Search Algorithm
        -SA3 : SMITH_Boyer_Moore Search Algorithm
        -SA4 : Karp_Rabin_Kaze Search Algorithm
        Default is HORSPOOL_Boyer_Moore Search Algorithm
Usage2: Raccoondog <e|d> inputFile outputFile
        e: encode file
        d: decode file
Example1: Raccoondog Raccoondog.lst
Example2: Raccoondog -SA2 Raccoondog.lst
Example3: Raccoondog e Caterpillar.001.txt Caterpillar.001.txt.lzma
Note1: Benchmark:
       Raccoondog(EN:8KB/clock, DE:39KB/clock) for 24.9GB(5.53GB LZMA) texts.
       Me machine is:
       Motherboard Name:                              Toshiba Satellite L305
       CPU Type:         Mobile DualCore Intel Pentium, 2166 MHz (13 x 167)
       CPU Alias:                                                   Merom-1M
       L1 Code Cache:                                          32 KB per core
       L1 Data Cache:                                          32 KB per core
       L2 Cache:                       1 MB (On-Die, ECC, ASC, Full-Speed)
       Bus Type:                                             Dual DDR2 SDRAM
       Bus Width:                                                  128-bit
       Real Clock:                                          333 MHz (DDR)
       Effective Clock:                                            666 MHz
Note2: Disastrous performance in case 128MB not fully physical!
Note3: Matches(hits) are overwritten to Raccoondog.hits.Pattern?.html files.
Note4: Works both on UNIX(LF) and Windows(CRLF) text files.
Note5: Never forget the importance of defragmented_AND_grouped files located at
       fastest area of disk - first partition is faster than second one, etc.
Note6: In ANSI, clock is defined as '#define CLOCKS_PER_SEC 1000'.
Note7: Since Raccoondog 13++:
       - limits(just skip longer ones) lines to 960 chars; OTHERWISE: HUGE TIME
         DELAYS due to recursive function;
       - shows hits to console too; MORE VIVID;
Note8: Since Raccoondog 14:
       - No deletion of input file after compressing/decompressing;
Note9: During execution hitting a 'Esc' causes termination(i.e. skipping rest).

Pattern(s) note: You may specify(four times) a main-pattern(case insensitive
                 with wildcards '*' i.e. any character(s) or empty and '?'
                 i.e. any character or empty) with three nested-patterns(case
                 sensitive and unexact), all four connected with AND.
                 Due to different line endings(CRLF in Windows; LF in UNIX)
                 you must add a '?' wildcard in place of CR: for example in
                 case of searching for '*.pdf' write '*.pdf?'.
Pattern(s) example: Pattern1: *take? *it*
                    Pattern1_NestedPattern1: you
                    Possible hit: ... your reason is so taken by It.

Have a nice Raccoondoging.
For contacts: sanmayce@hotmail.com
Sanmayce Svalqyatchx 'Kaze', 2009 Feb 22.

Allocated memory for DEcoded file in MB: 256
Size of input file with files for Raccoondoging: 9240
Pattern(s) note: You may specify(four times) a main-pattern(case insensitive
                 with wildcards '*' i.e. any character(s) or empty and '?'
                 i.e. any character or empty) with three nested-patterns(case
                 sensitive and unexact), all four connected with AND.
                 Due to different line endings(CRLF in Windows; LF in UNIX)
                 you must add a '?' wildcard in place of CR: for example in
                 case of searching for '*.pdf' write '*.pdf?'.
Pattern(s) example: Pattern1: *take? *it*
                    Pattern1_NestedPattern1: you
                    Possible hit: ... your reason is so taken by It.

Input Pattern1(hit only 'Enter' to skip): *not anymore*
- Input Pattern1_NestedPattern1(hit only 'Enter' to skip):
Input Pattern2(hit only 'Enter' to skip):
Processing .\Caterpillar.001.RAFT3.txt.lzma ...
Doing DECODE from HDD to RAM ...
Overall decode performance so far: 000,008KB/clock(EN) or 000,044KB/clock(DE)
Doing SEARCH for Pattern1 at once and flushing hit-sentences ...
000,000,001 M: Not anymore.
000,000,002 M: Not anymore.
000,000,003 "Not anymore," Lidia replied.
000,000,004 Not anymore.
000,000,005 Not anymore.
000,000,006 "Not anymore," Lidia replied.
000,000,007 Not anymore.
000,000,008 Not anymore.
000,000,009 "Not anymore," Lidia replied.
000,000,010 Not anymore.
000,000,011 Not anymore.
Found 11 case-insensitive and unexact matches(hits), so far.
'Esc' was pressed, so skip the rest files and quit!

Total Rough Upload and Decode time: 2,187 clocks
Total Rough Search time: 1,875 clocks
Total time: 4 seconds
Total Lines encountered: 1,835,098
Total Search(non-mask) function invocations: 0
Total Search(MASK i.e. wildcard) function invocations: 1,834,650
Total Boyer-Moore-Horspool(whole chunks, not lines) hits: 0
Total Boyer-Moore-Horspool(whole chunks, not lines) time: 0 clocks
Total Karp_Rabin_Kaze(whole chunks, not lines) hits: 0
Total Karp_Rabin_Kaze(whole chunks, not lines) time: 0 clocks
Raccoondog: Done successfully.

D:\_KAZE_G.S._Corpus>

D:\_KAZE_G.S._Corpus>Salah-ed-din -SA4 Salah-ed-din.lst
```

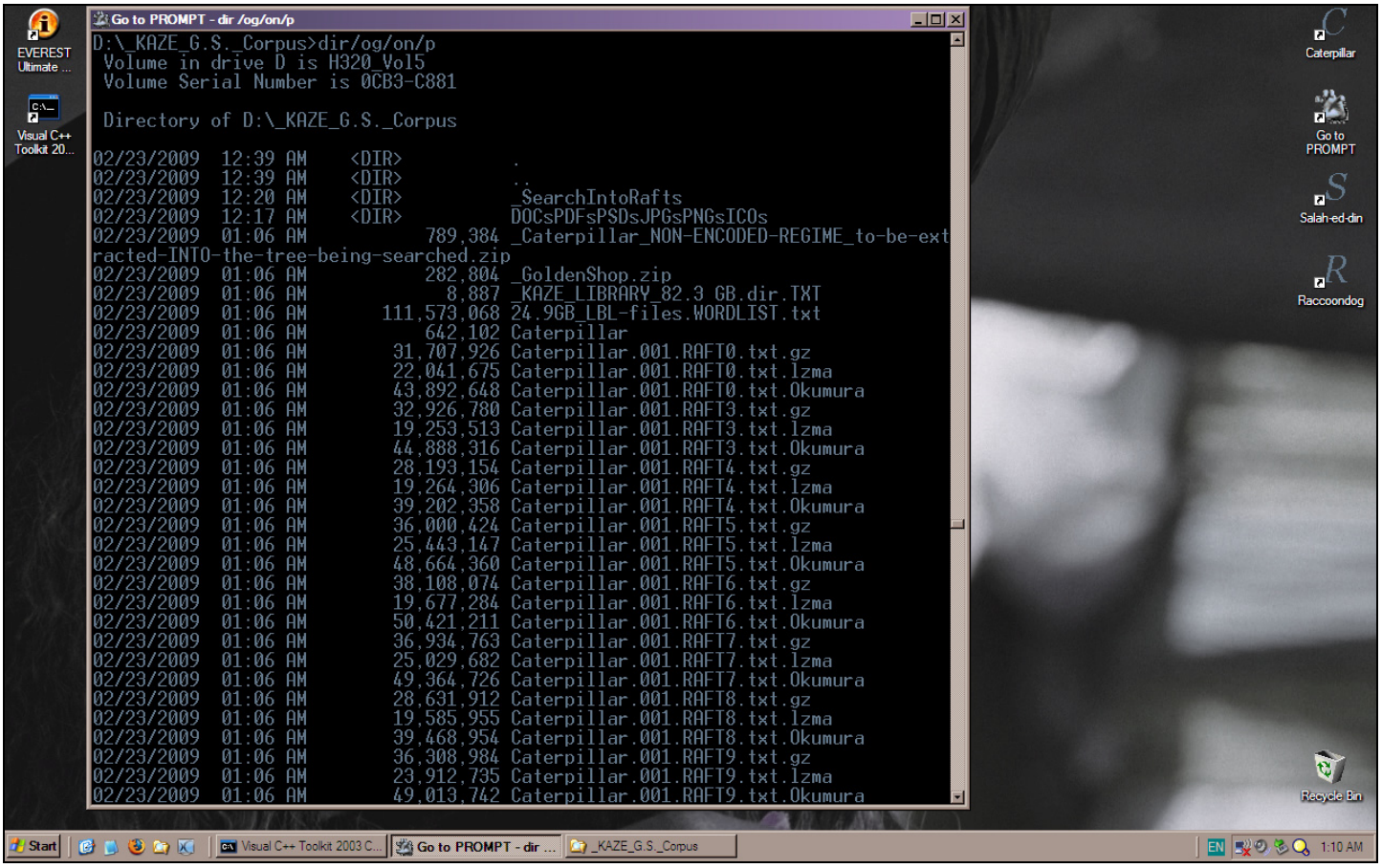
```
Salah-ed-din(Sentence_Dumper), revision 14++, written by Svalqyatchx,
in fact adapted from Mark Adler's and Jean-loup Gailly's ZLIB package.

Usage1: Salah-ed-din [-SA1|-SA2|-SA3|-SA4] filename
        Decodes all files from a list(filename)
        -SA1 : Brute_Force Search Algorithm
        -SA2 : Quick_Boyer_Moore Search Algorithm
        -SA3 : SMITH_Boyer_Moore Search Algorithm
        -SA4 : Karp_Rabin_Kaze Search Algorithm
        Default is HORSPOOL_Boyer_Moore Search Algorithm
Usage2: Salah-ed-din [-d] [-f] [-h] [-r] [-1 to -9] [files...]
        -d : decompress
        -f : compress with Z_FILTERED
        -h : compress with Z_HUFFMAN_ONLY
        -r : compress with Z_RLE
        -1 to -9 : compression level
Example1: Salah-ed-din Salah-ed-din.lst
Example2: Salah-ed-din -SA2 Salah-ed-din.lst
Example3: Salah-ed-din -f -6 Caterpillar.001.txt
Example4: Salah-ed-din -d Caterpillar.001.txt.gz
Note1: Benchmark:
       Raccoondog(EN:39KB/clock, DE:117KB/clock) for 24.9GB(8.42GB GZ) texts.
       Me machine is:
       Motherboard Name:                              Toshiba Satellite L305
       CPU Type:         Mobile DualCore Intel Pentium, 2166 MHz (13 x 167)
       CPU Alias:                                                  Merom-1M
       L1 Code Cache:                                          32 KB per core
       L1 Data Cache:                                          32 KB per core
       L2 Cache:                        1 MB (On-Die, ECC, ASC, Full-Speed)
       Bus Type:                                             Dual DDR2 SDRAM
       Bus Width:                                                    128-bit
       Real Clock:                                            333 MHz (DDR)
       Effective Clock:                                               666 MHz
Note2: Disastrous performance in case 128MB not fully physical!
Note3: Matches(hits) are overwritten to Salah-ed-din.hits.Pattern?.html files.
Note4: Works both on UNIX(LF) and Windows(CRLF) text files.
Note5: Never forget the importance of defragmented_AND_grouped files located at
       fastest area of disk - first partition is faster than second one, etc.
Note6: In ANSI, clock is defined as '#define CLOCKS_PER_SEC 1000'.
Note7: Since Salah-ed-din 13++:
       - limits(just skip longer ones) lines to 960 chars; OTHERWISE: HUGE TIME
         DELAYS due to recursive function;
       - shows hits to console too; MORE VIVID;
Note8: Since Salah-ed-din 14:
       - No deletion of input file after compressing/decompressing;
Note9: During execution hitting a 'Esc' causes termination(i.e. skipping rest).

Pattern(s) note: You may specify(four times) a main-pattern(case insensitive
                 with wildcards '*' i.e. any character(s) or empty and '?'
                 i.e. any character or empty) with three nested-patterns(case
                 sensitive and unexact), all four connected with AND.
                 Due to different line endings(CRLF in Windows; LF in UNIX)
                 you must add a '?' wildcard in place of CR: for example in
                 case of searching for '*.pdf' write '*.pdf?'.
Pattern(s) example: Pattern1: *take? *it*
                    Pattern1_NestedPattern1: you
                    Possible hit: ... your reason is so taken by It.

Have a nice Salah-ed-dining.
For contacts: sanmayce@hotmail.com
Sanmayce Svalqyatchx 'Kaze', 2009 Feb 22.

Allocated memory for DEcoded file in MB: 96
Size of input file with files for Salah-ed-dining: 8680
Pattern(s) note: You may specify(four times) a main-pattern(case insensitive
                 with wildcards '*' i.e. any character(s) or empty and '?'
                 i.e. any character or empty) with three nested-patterns(case
                 sensitive and unexact), all four connected with AND.
                 Due to different line endings(CRLF in Windows; LF in UNIX)
                 you must add a '?' wildcard in place of CR: for example in
                 case of searching for '*.pdf' write '*.pdf?'.
Pattern(s) example: Pattern1: *take? *it*
                    Pattern1_NestedPattern1: you
                    Possible hit: ... your reason is so taken by It.

Input Pattern1(hit only 'Enter' to skip): *not anymore*
- Input Pattern1_NestedPattern1(hit only 'Enter' to skip):
Input Pattern2(hit only 'Enter' to skip):
Processing .\Caterpillar.001.RAFT3.txt.gz ...
Doing DECODE from HDD to RAM ...

Salah-ed-din decoded buffer size: 99,614,459
Overall decode performance so far: 000,033KB/clock(EN) or 000,102KB/clock(DE)
Doing SEARCH for Pattern1 at once and flushing hit-sentences ...
000,000,001 M: Not anymore.
000,000,002 M: Not anymore.
000,000,003 "Not anymore," Lidia replied.
000,000,004 Not anymore.
000,000,005 Not anymore.
000,000,006 "Not anymore," Lidia replied.
000,000,007 Not anymore.
000,000,008 Not anymore.
000,000,009 "Not anymore," Lidia replied.
000,000,010 Not anymore.
000,000,011 Not anymore.
Found 11 case-insensitive and unexact matches(hits), so far.
'Esc' was pressed, so skip the rest files and quit!

Total Rough Upload and Decode time: 953 clocks
Total Rough Search time: 1,907 clocks
Total time: 3 seconds
Total Lines encountered: 1,835,098
Total Search(non-mask) function invocations: 0
Total Search(MASK i.e. wildcard) function invocations: 1,834,650
Total Boyer-Moore-Horspool(whole chunks, not lines) hits: 0
Total Boyer-Moore-Horspool(whole chunks, not lines) time: 0 clocks
Total Karp_Rabin_Kaze(whole chunks, not lines) hits: 0
Total Karp_Rabin_Kaze(whole chunks, not lines) time: 0 clocks
Salah-ed-din: Done successfully.

D:\_KAZE_G.S._Corpus>
```

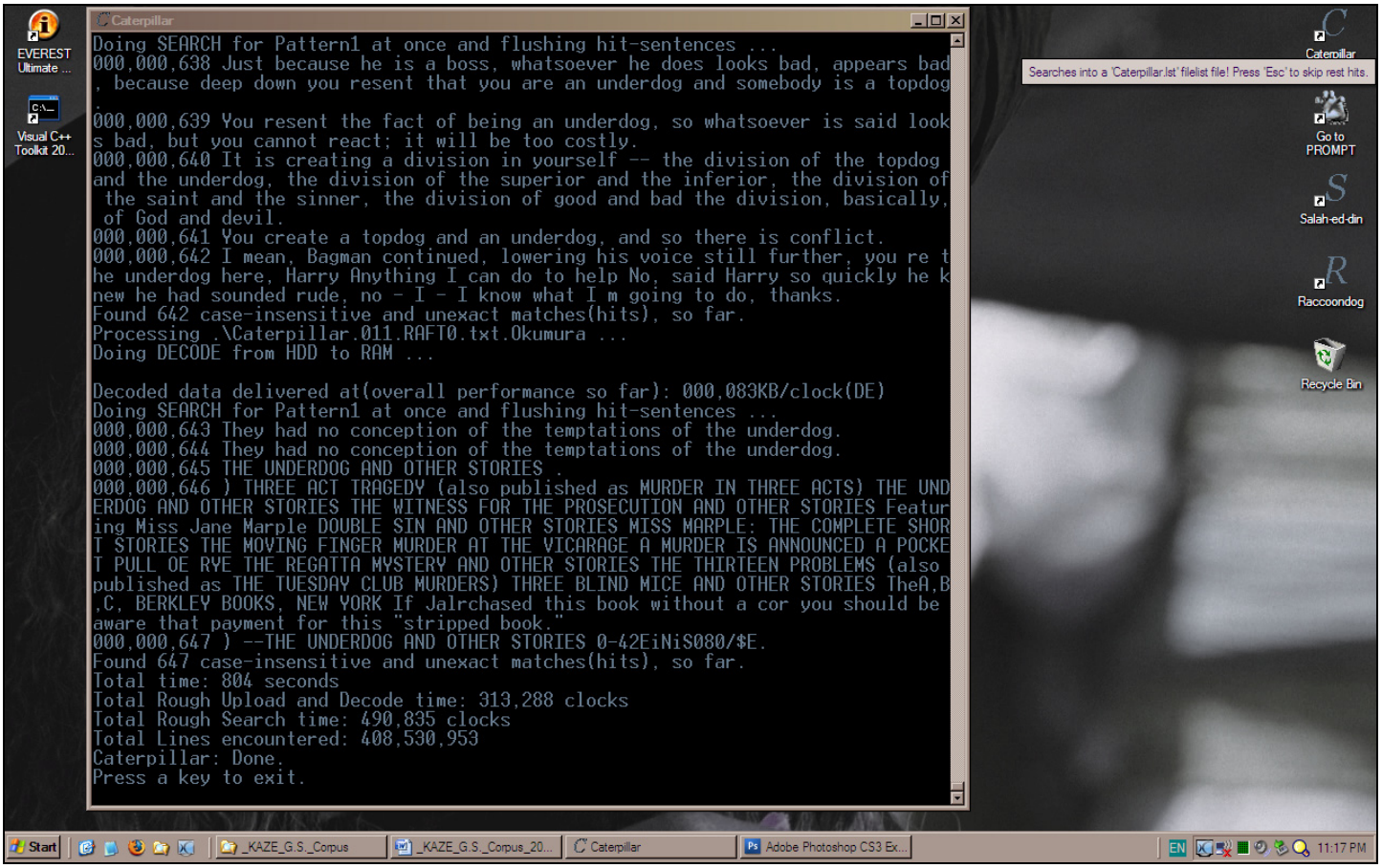

_KAZE_G.S._Corpus
File Edit View Favorites Tools Help
Back Search Folders Folder Sync
Address D:_KAZE_G.S._Corpus
Folders
_KAZE_CORE_2009-Feb-21_38.1 GB
_KAZE_G.S._Corpus
_SearchIntoRafts
_RAFTbyRAFT_Description
DOCsPDFsPSDsJPGsPNGsICOs
_Caterpillar_NON-ENCODED-REGIME_to-be-
_GoldenShop.zip
_Mini_DVDRW_1400MB_Side-A
_Mini_DVDRW_1400MB_Side-B
KAZE_G.S._Corpus Properties
General Sharing Security Customize
_KAZE_G.S._Corpus
Type: File Folder
Location: D:\
Size: 25.8 GB (27,748,371,521 bytes)
Size on disk: 25.8 GB (27,750,625,280 bytes)
Contains: 1,081 Files, 3 Folders
Created: Monday, February 02, 2009, 2:58:06 AM
Attributes: Read-only Hidden Advanced...
OK Cancel Apply
Name Size Type Date Modified
Caterpillar.039.RAFT6.txt.lzma 20,421 KB LZMA File 2/23/2009 1:13 AM
Caterpillar.040.RAFT5.txt.lzma 24,805 KB LZMA File 2/23/2009 1:13 AM
Caterpillar.040.RAFT6.txt.lzma 24,223 KB LZMA File 2/23/2009 1:13 AM
Caterpillar.041.RAFT5.txt.lzma 24,413 KB LZMA File 2/23/2009 1:13 AM
Caterpillar.041.RAFT6.txt.lzma 23,482 KB LZMA File 2/23/2009 1:13 AM
Caterpillar.042.RAFT5.txt.lzma 24,250 KB LZMA File 2/23/2009 1:13 AM
Caterpillar.042.RAFT6.txt.lzma 21,688 KB LZMA File 2/23/2009 1:13 AM
Caterpillar.043.RAFT5.txt.lzma 4,586 KB LZMA File 2/23/2009 1:13 AM
Caterpillar.043.RAFT6.txt.lzma 21,434 KB LZMA File 2/23/2009 1:13 AM
Caterpillar.044.RAFT6.txt.lzma 20,886 KB LZMA File 2/23/2009 1:13 AM
Caterpillar.045.RAFT6.txt.lzma 22,937 KB LZMA File 2/23/2009 1:13 AM
Caterpillar.046.RAFT6.txt.lzma 22,312 KB LZMA File 2/23/2009 1:13 AM
Caterpillar.047.RAFT6.txt.lzma 1,343 KB LZMA File 2/23/2009 1:13 AM
Caterpillar.HIT1.lst 5 KB MASM Listing 2/23/2009 1:13 AM
Caterpillar.HIT2.lst 0 KB MASM Listing 2/23/2009 1:13 AM
Caterpillar.HIT3.lst 0 KB MASM Listing 2/23/2009 1:13 AM
Caterpillar.HIT4.lst 0 KB MASM Listing 2/23/2009 1:13 AM
Caterpillar.lst 10 KB MASM Listing 2/23/2009 1:13 AM
Raccoondog.lst 10 KB MASM Listing 2/23/2009 1:13 AM
Salah-ed-din.lst 9 KB MASM Listing 2/23/2009 1:13 AM
Caterpillar_HDD2RAM.bat 1 KB MS-DOS Batch File 2/23/2009 1:13 AM
Caterpillar_make_md5_CHECK-INTEGRITY.bat 1 KB MS-DOS Batch File 2/23/2009 1:13 AM
G2O.BAT 42 KB MS-DOS Batch File 2/23/2009 1:13 AM
G2R.BAT 41 KB MS-DOS Batch File 2/23/2009 1:13 AM
MakeLST_gz.bat 2 KB MS-DOS Batch File 2/23/2009 1:13 AM
MakeLST_lzma.bat 2 KB MS-DOS Batch File 2/23/2009 1:13 AM
MakeLST_Okumura.bat 2 KB MS-DOS Batch File 2/23/2009 1:13 AM
R2G.BAT 41 KB MS-DOS Batch File 2/23/2009 1:13 AM
Raccoondog_HDD2RAM.bat 1 KB MS-DOS Batch File 2/23/2009 1:13 AM
Raccoondog_make_md5_CHECK-INTEGRITY.bat 1 KB MS-DOS Batch File 2/23/2009 1:13 AM
Salah-ed-din_HDD2RAM.bat 1 KB MS-DOS Batch File 2/23/2009 1:13 AM
Salah-ed-din_make_md5_CHECK-INTEGRITY.bat 1 KB MS-DOS Batch File 2/23/2009 1:13 AM
Caterpillar.001.RAFT0.txt.Okumura 42,864 KB OKUMURA File 2/23/2009 1:13 AM
Caterpillar.001.RAFT3.txt.Okumura 43,837 KB OKUMURA File 2/23/2009 1:13 AM
Caterpillar.001.RAFT4.txt.Okumura 38,284 KB OKUMURA File 2/23/2009 1:13 AM
Caterpillar.001.RAFT5.txt.Okumura 47,524 KB OKUMURA File 2/23/2009 1:13 AM
Caterpillar.001.RAFT6.txt.Okumura 49,240 KB OKUMURA File 2/23/2009 1:13 AM
3 objects selected
122 KB
My Computer
Start _KAZE_G.S._Corpus
1:16 AM

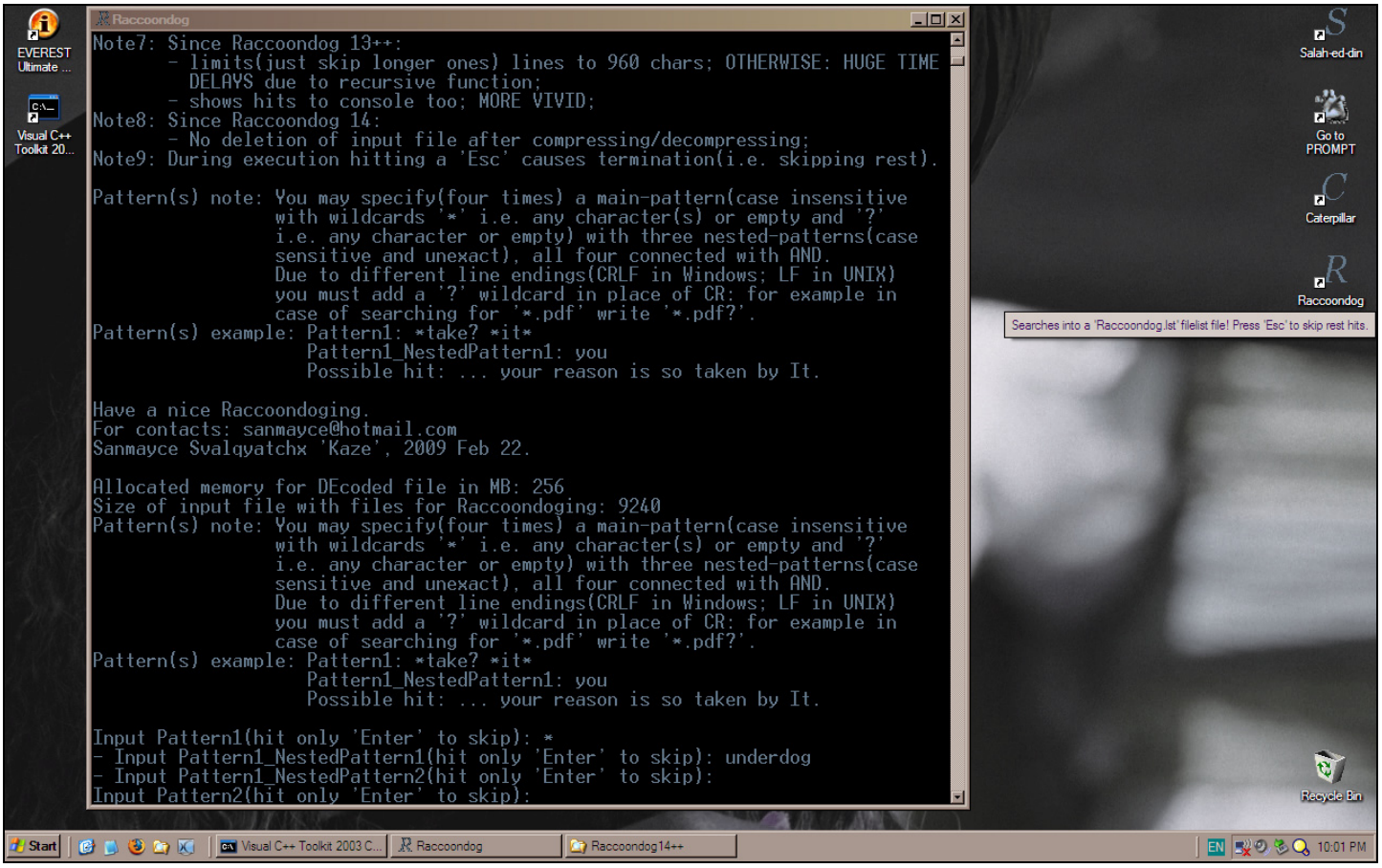

Caterpillar Properties
Colors Compatibility Security
General Shortcut Options Font Layout
Caterpillar
Target type: MS-DOS Batch File
Target location: _KAZE_G.S._Corpus
Target: _KAZE_G.S._Corpus\Caterpillar_HDD2RAM.bat
Start in: D:_KAZE_G.S._Corpus
Shortcut key: None
Run: Normal window
Comment: Searches into a 'Caterpillar.lst' filelist file! Press 'Es
Find Target... Change Icon... Advanced...
OK Cancel Apply
Salah-ed-din Properties
Salah-ed-din
Target type: MS-DOS Batch File
Target location: _KAZE_G.S._Corpus
Target: AZE_G.S._Corpus\Salah-ed-din_HDD2RAM.bat
Start in: D:_KAZE_G.S._Corpus
Shortcut key: None
Run: Normal window
Comment: Searches into a 'Salah-ed-din.lst' filelist file! Press '
Find Target... Change Icon... Advanced...
Raccoondog Properties
Raccoondog
Target type: MS-DOS Batch File
Target location: _KAZE_G.S._Corpus
Target: AZE_G.S._Corpus\Raccoondog_HDD2RAM.bat
Start in: D:_KAZE_G.S._Corpus
Shortcut key: None
Run: Normal window
Comment: Searches into a 'Raccoondog.lst' filelist file! Press
Find Target... Change Icon... Advanced...
Go to PROMPT Properties
Go to PROMPT
Target type: Application
Target location:
Target: %comspec% /k
Start in: D:_KAZE_G.S._Corpus
Shortcut key: Shift + Alt + F3
Run: Normal window
Comment: Go to PROMPT.
Find Target... Change Icon... Advanced...
Caterpillar.hits.pattern1.html
Caterpillar.hits.pattern2.html
Caterpillar.hits.pattern3.html
Caterpillar.hits.pattern4.html
Raccoondog.hits.pattern1.html
Raccoondog.hits.pattern2.html
Raccoondog.hits.pattern3.html
Raccoondog.hits.pattern4.html
Salah-ed-din.hits.pattern1.html
Salah-ed-din.hits.pattern2.html
Salah-ed-din.hits.pattern3.html
Salah-ed-din.hits.pattern4.html
Caterpillar
Go to PROMPT
Raccoondog
Salah-ed-din
_KAZE_LIBRARY_82.3 GB.dir.TXT
24.9GB_LBL-files.WORDLIST.txt
Caterpillar_280_Okumura_new.txt
Caterpillar_280_Okumura_old.txt
Raccoondog_280_lzma_new.txt
Raccoondog_280_lzma_old.txt
Salah-ed-din_280_gz_new.txt
Salah-ed-din_280_gz_old.txt
Caterpillar.001.RAFT0.txt.gz
Caterpillar.001.RAFT3.txt.gz
4.79 KB
My Computer
Start _KAZE_G.S._Corpus Adobe Photoshop CS3 Ex...
1:20 AM

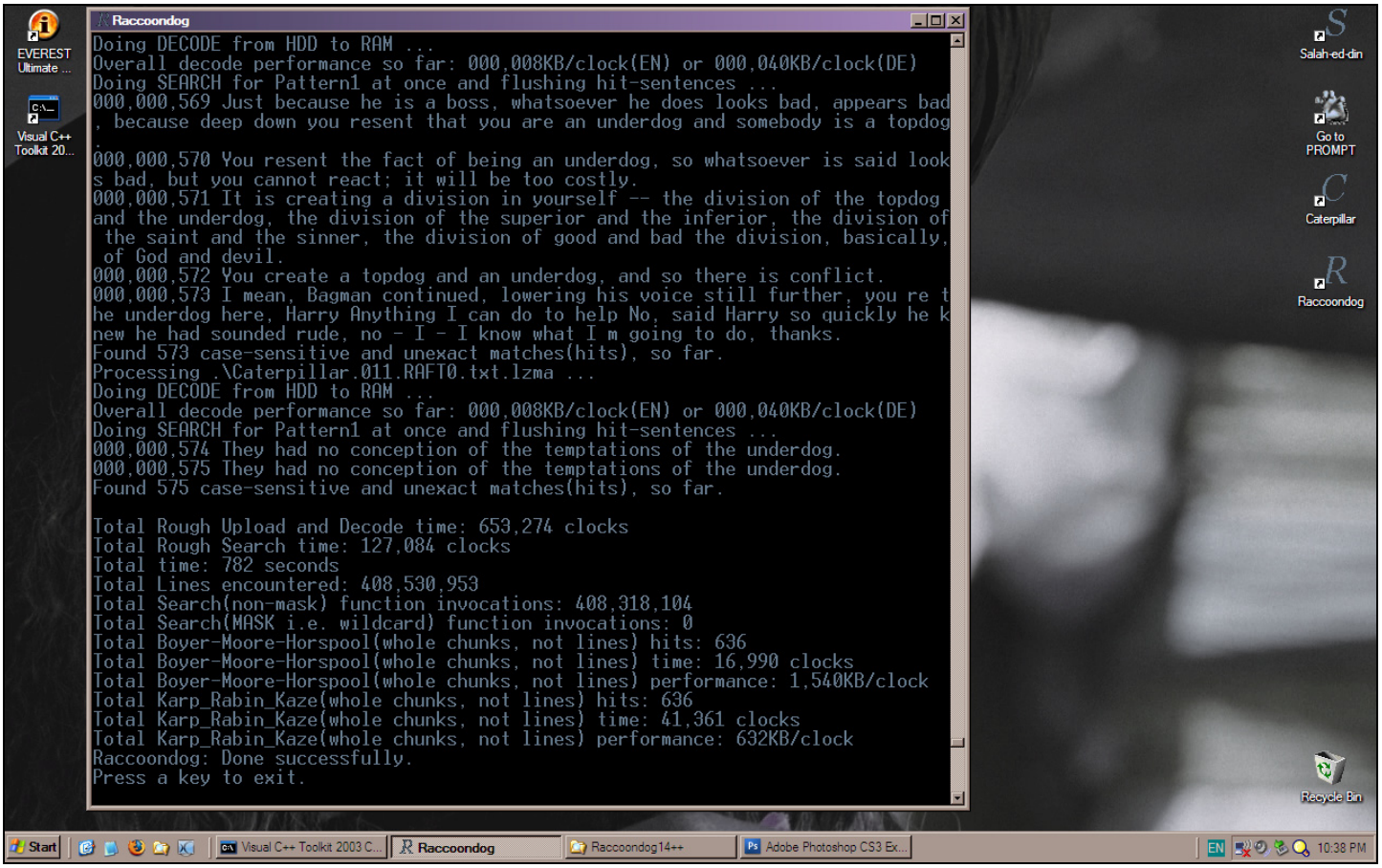

EVEREST Ultimate ...
Visual C++ Toolkit 20...
Salah-ed-din
Go to PROMPT
Caterpillar
Raccoondog
Recycle Bin
Raccoondog14++ - WinRAR
File Commands Tools Favorites Options Help
Add
Extract To
Test
View
Delete
Find
Wizard
Info
Repair
D:\WorkTemp\Raccoondog14++
About WinRAR
WinRAR 3.71
Copyright © 1993-2007 by Alexander Roshal
Theme: Icons by www.carlitus.net
WinRAR Theme by www.lucmar.com
Benchmark and hardware test
Speed
Resulting, KB/s 530
Current, KB/s 530
Total
Elapsed time 00:02:20
Processed, MB 73
Errors found No
Multithreading
Background
Pause
lzma465
Raccoondog14++
Start
Visual C++ Toolkit 2003 C...
wrar371
Adobe Photoshop CS3 Ex...
Raccoondog14++ - WinR...
10:57 PM

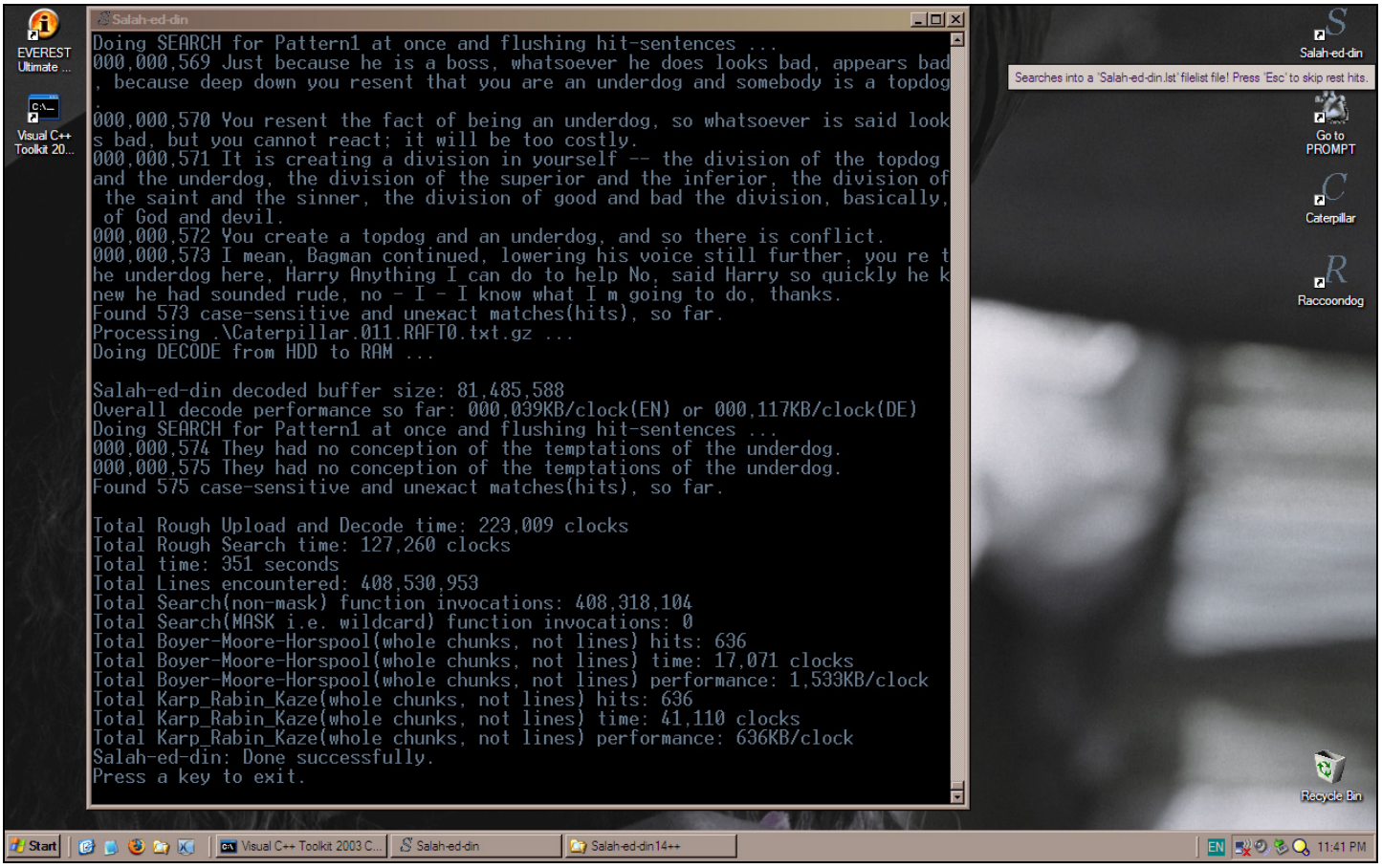

EVEREST Ultimate ...
Visual C++ Toolkit 20...
Salah-ed-din
Go to PROMPT
Caterpillar
Raccoondog
Recycle Bin
Lavalys EVEREST CPUID
Processor Mobile DualCore Intel Pentium 65 nm
Code Name Merom-1M
Platform Socket 479
Stepping M0
CPUID Vendor GenuineIntel
CPUID Name Intel(R) Pentium(R) Dual CPU T3400 @ 2.16GHz
CPUID Rev. 6 F D
Core Voltage 1.363 V
CPU Clock 2161.5 MHz
Multiplier 13x
FSB Clock 166.3 MHz
FSB Speed 665.2 MHz
L1 Instr. Cache 32 KB
L1 Data Cache 32 KB
L2 Cache 1 MB
L3 Cache
Instruction Set x86, x86-64, MMX, SSE, SSE2, SSE3, SSSE3
Motherboard Toshiba Satellite L305
Chipset Intel Cantiga GL40
Integr. Video Active (Intel GMA 3100)
Memory Type Dual Channel DDR2-667 SDRAM (5-5-5-13)
Memory Clock 332.5 MHz
DRAM:FSB Ratio 10:5
Lavalys EVEREST Cache & Memory Benchmark
Read Write Copy Latency
Memory 4650 MB/s 3356 MB/s 3695 MB/s 88.2 ns
L1 Cache 34551 MB/s 34131 MB/s 68813 MB/s 1.4 ns
L2 Cache 16308 MB/s 11353 MB/s 15741 MB/s 2.9 ns
L3 Cache
CPU Type Mobile DualCore Intel Pentium (Merom-1M, Socket 479)
CPU Clock 2161.5 MHz (original: 2166 MHz)
CPU FSB 166.3 MHz (original: 166 MHz)
CPU Multiplier 13x
CPU Stepping M0
Memory Bus 332.5 MHz
DRAM:FSB Ratio 10:5
Memory Type Dual Channel DDR2-667 SDRAM (5-5-5-13)
Chipset Intel Cantiga GL40
Motherboard Toshiba Satellite L305
EVEREST v4.60.1500 / BenchDLL 2.3.237.0 (c) 2003-2008 Lavalys, Inc.
Start
Visual C++ Toolkit ...
_KAZE_G.S._Corpus
Adobe Photoshop ...
Raccoondog Hits - ...
EVEREST CPUID
EVEREST Cache &...
10:45 PM

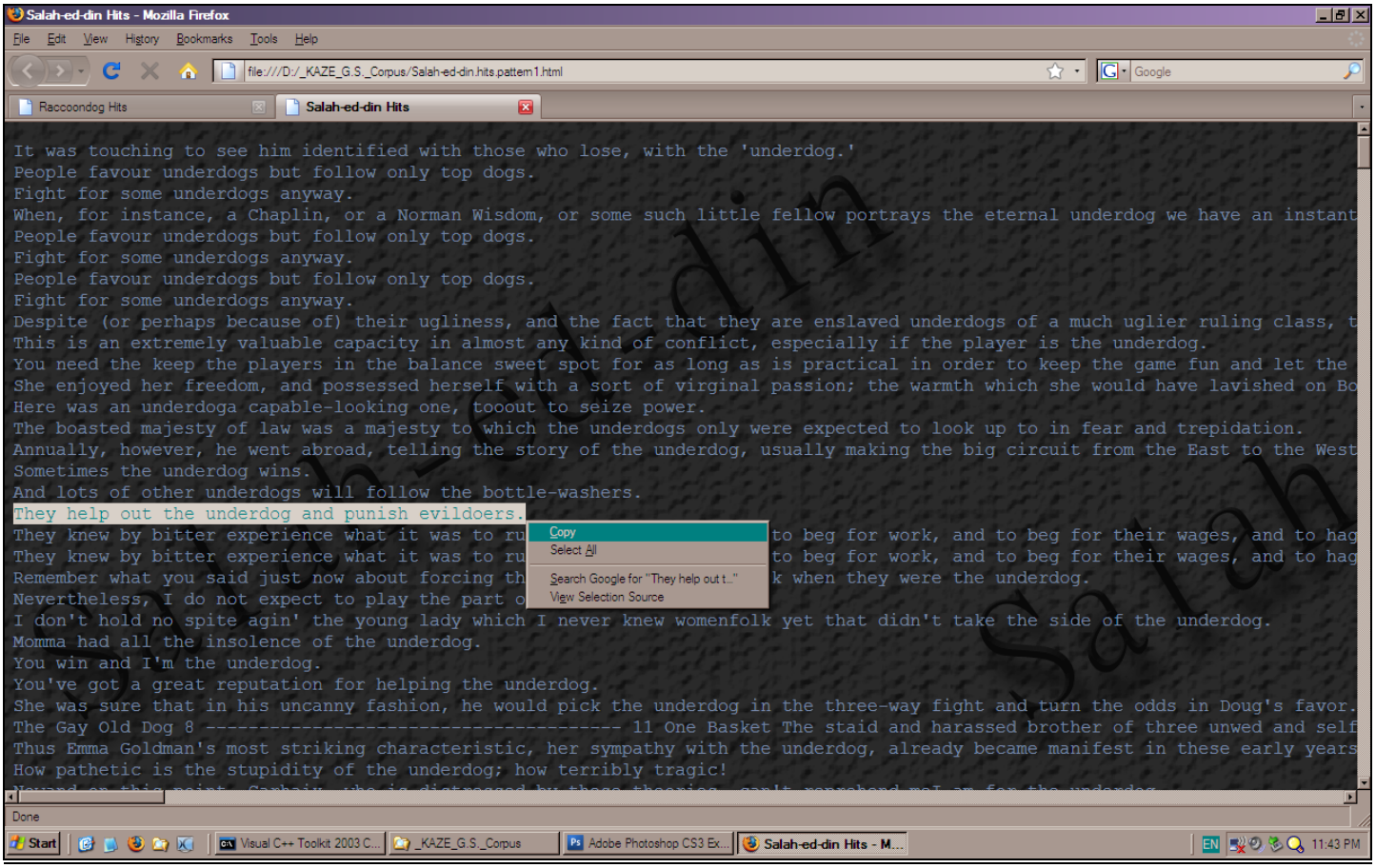

Go to PROMPT - dir /og/on/p
D:_KAZE_G.S._Corpus>dir/og/on/p
Volume in drive D is H320_Vol5
Volume Serial Number is 0CB3-C881
Directory of D:_KAZE_G.S._Corpus
02/23/2009 12:39 AM <DIR> .
02/23/2009 12:39 AM <DIR> ..
02/23/2009 12:20 AM <DIR> _SearchIntoRafts
02/23/2009 12:17 AM <DIR> DOCsPDFsPSDsJPGsPNGsICOs
02/23/2009 01:06 AM 789,384 _Caterpillar_NON-ENCODED-REGIME_to-be-extracted-INTO-the-tree-being-searched.zip
02/23/2009 01:06 AM 282,804 _GoldenShop.zip
02/23/2009 01:06 AM 8,887 _KAZE_LIBRARY_82.3 GB.dir.TXT
02/23/2009 01:06 AM 111,573,068 24.9GB_LBL-files.WORDLIST.txt
02/23/2009 01:06 AM 642,102 Caterpillar
02/23/2009 01:06 AM 31,707,926 Caterpillar.001.RAFT0.txt.gz
02/23/2009 01:06 AM 22,041,675 Caterpillar.001.RAFT0.txt.lzma
02/23/2009 01:06 AM 43,892,648 Caterpillar.001.RAFT0.txt.Okumura
02/23/2009 01:06 AM 32,926,780 Caterpillar.001.RAFT3.txt.gz
02/23/2009 01:06 AM 19,253,513 Caterpillar.001.RAFT3.txt.lzma
02/23/2009 01:06 AM 44,888,316 Caterpillar.001.RAFT3.txt.Okumura
02/23/2009 01:06 AM 28,193,154 Caterpillar.001.RAFT4.txt.gz
02/23/2009 01:06 AM 19,264,306 Caterpillar.001.RAFT4.txt.lzma
02/23/2009 01:06 AM 39,202,358 Caterpillar.001.RAFT4.txt.Okumura
02/23/2009 01:06 AM 36,000,424 Caterpillar.001.RAFT5.txt.gz
02/23/2009 01:06 AM 25,443,147 Caterpillar.001.RAFT5.txt.lzma
02/23/2009 01:06 AM 48,664,360 Caterpillar.001.RAFT5.txt.Okumura
02/23/2009 01:06 AM 38,108,074 Caterpillar.001.RAFT6.txt.gz
02/23/2009 01:06 AM 19,677,284 Caterpillar.001.RAFT6.txt.lzma
02/23/2009 01:06 AM 50,421,211 Caterpillar.001.RAFT6.txt.Okumura
02/23/2009 01:06 AM 36,934,763 Caterpillar.001.RAFT7.txt.gz
02/23/2009 01:06 AM 25,029,682 Caterpillar.001.RAFT7.txt.lzma
02/23/2009 01:06 AM 49,364,726 Caterpillar.001.RAFT7.txt.Okumura
02/23/2009 01:06 AM 28,631,912 Caterpillar.001.RAFT8.txt.gz
02/23/2009 01:06 AM 19,585,955 Caterpillar.001.RAFT8.txt.lzma
02/23/2009 01:06 AM 39,468,954 Caterpillar.001.RAFT8.txt.Okumura
02/23/2009 01:06 AM 36,308,984 Caterpillar.001.RAFT9.txt.gz
02/23/2009 01:06 AM 23,912,735 Caterpillar.001.RAFT9.txt.lzma
02/23/2009 01:06 AM 49,013,742 Caterpillar.001.RAFT9.txt.Okumura
EVEREST Ultimate ...
Visual C++ Toolkit 20...
Caterpillar
Go to PROMPT
Salah-ed-din
Raccoondog
Recycle Bin
Start
Visual C++ Toolkit 2003 C...
Go to PROMPT - dir ...
_KAZE_G.S._Corpus
1:10 AM

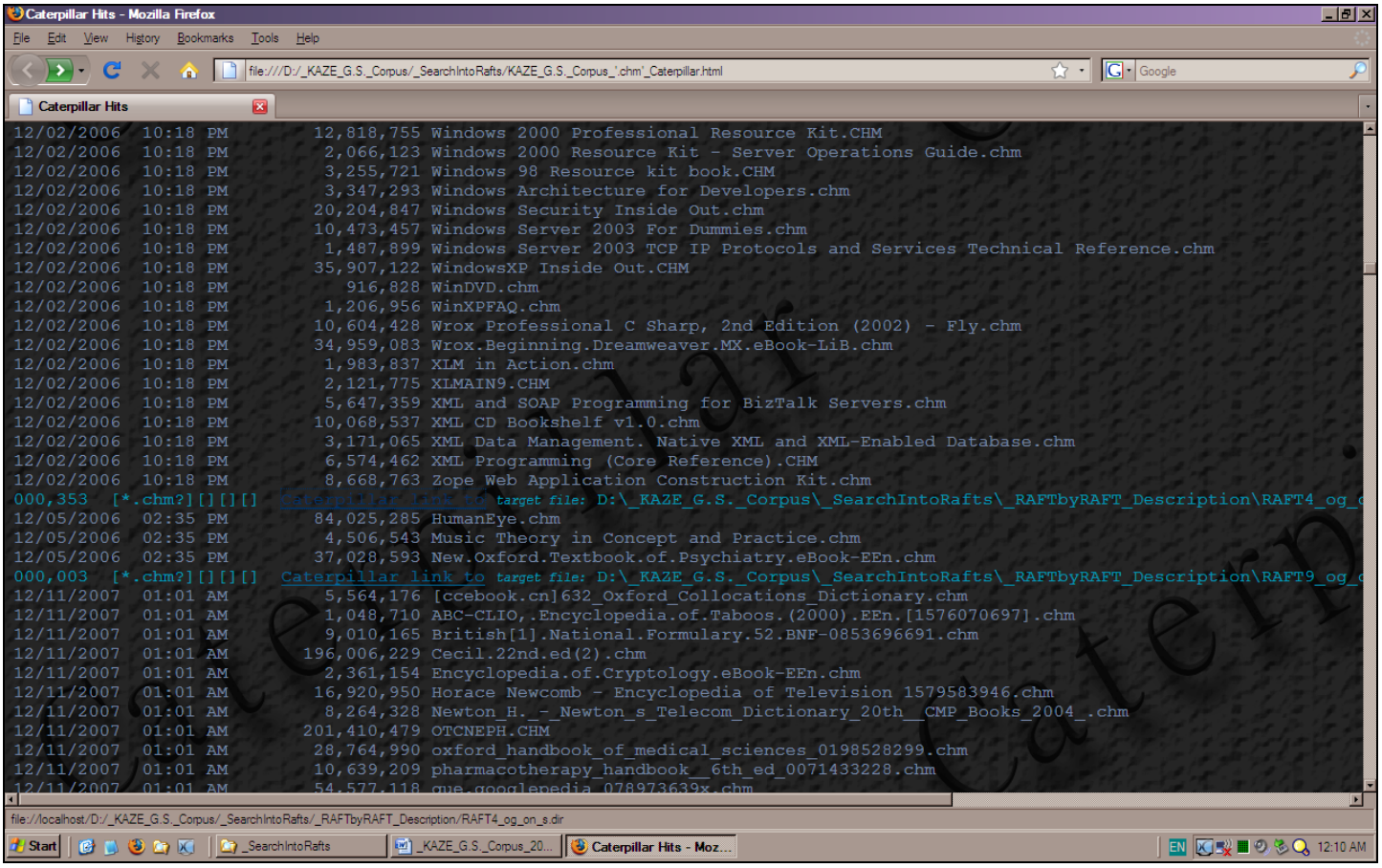

Caterpillar
Doing SEARCH for Pattern1 at once and flushing hit-sentences ...
000,000,638 Just because he is a boss, whatsoever he does looks bad, appears bad, because deep down you resent that you are an underdog and somebody is a topdog.
000,000,639 You resent the fact of being an underdog, so whatsoever is said looks bad, but you cannot react; it will be too costly.
000,000,640 It is creating a division in yourself -- the division of the topdog and the underdog, the division of the superior and the inferior, the division of the saint and the sinner, the division of good and bad the division, basically, of God and devil.
000,000,641 You create a topdog and an underdog, and so there is conflict.
000,000,642 I mean, Bagman continued, lowering his voice still further, you re the underdog here, Harry Anything I can do to help No, said Harry so quickly he knew he had sounded rude, no - I - I know what I m going to do, thanks.
Found 642 case-insensitive and unexact matches(hits), so far.
Processing .\Caterpillar.011.RAFT0.txt.Okumura ...
Doing DECODE from HDD to RAM ...
Decoded data delivered at(overall performance so far): 000,083KB/clock(DE)
Doing SEARCH for Pattern1 at once and flushing hit-sentences ...
000,000,643 They had no conception of the temptations of the underdog.
000,000,644 They had no conception of the temptations of the underdog.
000,000,645 THE UNDERDOG AND OTHER STORIES .
000,000,646) THREE ACT TRAGEDY (also published as MURDER IN THREE ACTS) THE UNDERDOG AND OTHER STORIES THE WITNESS FOR THE PROSECUTION AND OTHER STORIES Featuring Miss Jane Marple DOUBLE SIN AND OTHER STORIES MISS MARPLE: THE COMPLETE SHORT STORIES THE MOVING FINGER MURDER AT THE VICARAGE A MURDER IS ANNOUNCED A POCKET PULL OE RYE THE REGATTA MYSTERY AND OTHER STORIES THE THIRTEEN PROBLEMS (also published as THE TUESDAY CLUB MURDERS) THREE BLIND MICE AND OTHER STORIES TheA,B,C, BERKLEY BOOKS, NEW YORK If Jalrchased this book without a cor you should be aware that payment for this "stripped book."
000,000,647) --THE UNDERDOG AND OTHER STORIES 0-42EiNiS080/$E.
Found 647 case-insensitive and unexact matches(hits), so far.
Total time: 804 seconds
Total Rough Upload and Decode time: 313,288 clocks
Total Rough Search time: 490,835 clocks
Total Lines encountered: 408,530,953
Caterpillar: Done.
Press a key to exit.
Searches into a 'Caterpillar.lst' filelist file! Press 'Esc' to skip rest hits.
EVEREST Ultimate ...
Visual C++ Toolkit 20...
Caterpillar
Go to PROMPT
Salah-ed-din
Raccoondog
Recycle Bin
Start
_KAZE_G.S._Corpus
_KAZE_G.S._Corpus_20...
Caterpillar
Adobe Photoshop CS3 Ex...
11:17 PM

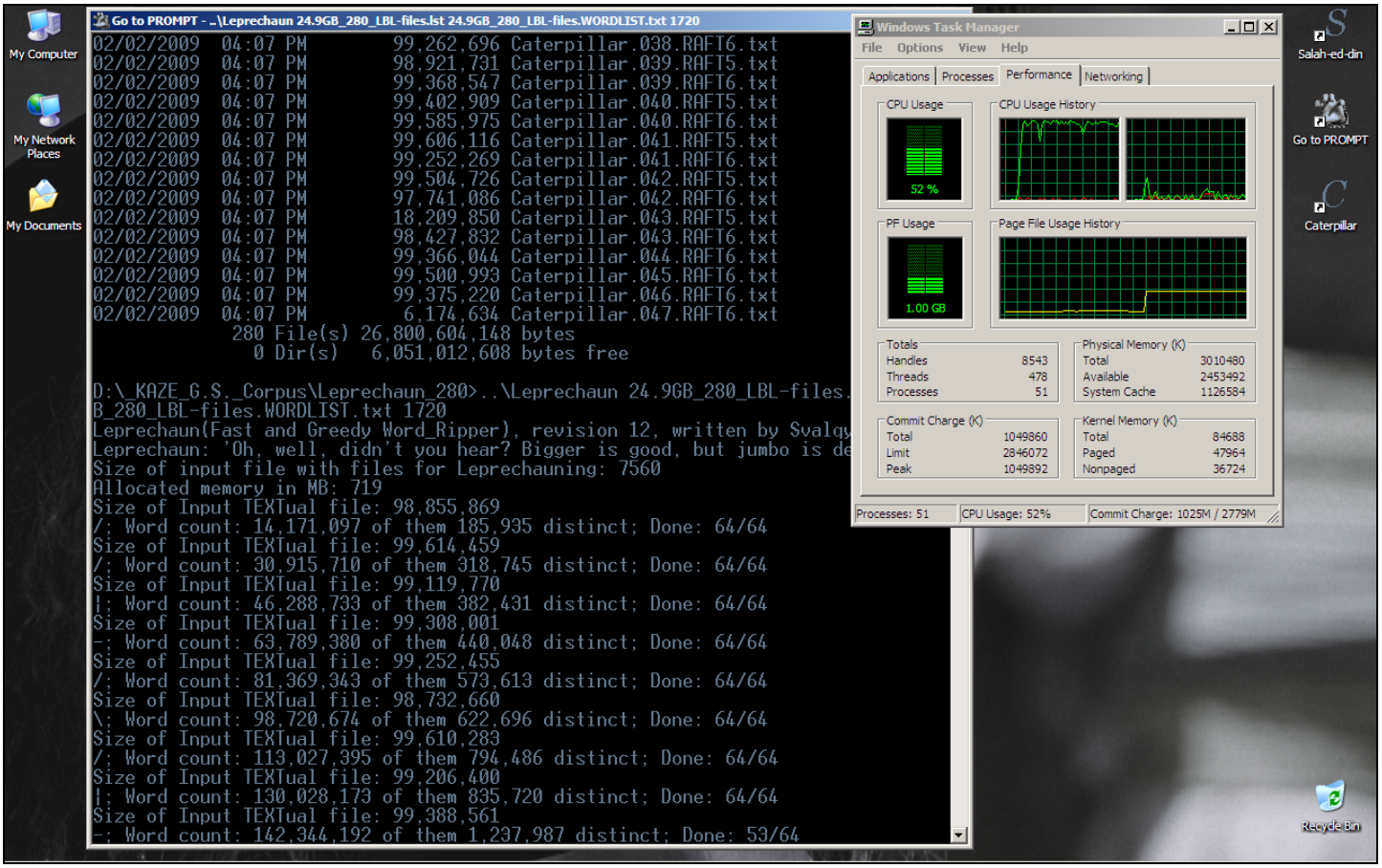

Raccoondog
Note7: Since Raccoondog 13++:
- limits(just skip longer ones) lines to 960 chars; OTHERWISE: HUGE TIME
DELAYS due to recursive function;
- shows hits to console too; MORE VIVID;
Note8: Since Raccoondog 14:
- No deletion of input file after compressing/decompressing;
Note9: During execution hitting a 'Esc' causes termination(i.e. skipping rest).
Pattern(s) note: You may specify(four times) a main-pattern(case insensitive
with wildcards '*' i.e. any character(s) or empty and '?'
i.e. any character or empty) with three nested-patterns(case
sensitive and unexact), all four connected with AND.
Due to different line endings(CRLF in Windows; LF in UNIX)
you must add a '?' wildcard in place of CR: for example in
case of searching for '*.pdf' write '*.pdf?'.
Pattern(s) example: Pattern1: *take? *it*
Pattern1_NestedPattern1: you
Possible hit: ... your reason is so taken by It.
Have a nice Raccoondoging.
For contacts: sanmayce@hotmail.com
Sanmayce Svalqyatchx 'Kaze', 2009 Feb 22.
Allocated memory for DEcoded file in MB: 256
Size of input file with files for Raccoondoging: 9240
Pattern(s) note: You may specify(four times) a main-pattern(case insensitive
with wildcards '*' i.e. any character(s) or empty and '?'
i.e. any character or empty) with three nested-patterns(case
sensitive and unexact), all four connected with AND.
Due to different line endings(CRLF in Windows; LF in UNIX)
you must add a '?' wildcard in place of CR: for example in
case of searching for '*.pdf' write '*.pdf?'.
Pattern(s) example: Pattern1: *take? *it*
Pattern1_NestedPattern1: you
Possible hit: ... your reason is so taken by It.
Input Pattern1(hit only 'Enter' to skip): *
- Input Pattern1_NestedPattern1(hit only 'Enter' to skip): underdog
- Input Pattern1_NestedPattern2(hit only 'Enter' to skip):
Input Pattern2(hit only 'Enter' to skip):
Searches into a 'Raccoondog.lst' filelist file! Press 'Esc' to skip rest hits.
EVEREST Ultimate ...
Visual C++ Toolkit 20...
Salah-ed-din
Go to PROMPT
Caterpillar
Raccoondog
Recycle Bin
Start
Visual C++ Toolkit 2003 C...
Raccoondog
Raccoondog14++
10:01 PM


Raccoondog
Doing DECODE from HDD to RAM ...
Overall decode performance so far: 000,008KB/clock(EN) or 000,040KB/clock(DE)
Doing SEARCH for Pattern1 at once and flushing hit-sentences ...
000,000,569 Just because he is a boss, whatsoever he does looks bad, appears bad
, because deep down you resent that you are an underdog and somebody is a topdog
.
000,000,570 You resent the fact of being an underdog, so whatsoever is said look
s bad, but you cannot react; it will be too costly.
000,000,571 It is creating a division in yourself -- the division of the topdog
and the underdog, the division of the superior and the inferior, the division of
the saint and the sinner, the division of good and bad the division, basically,
of God and devil.
000,000,572 You create a topdog and an underdog, and so there is conflict.
000,000,573 I mean, Bagman continued, lowering his voice still further, you re t
he underdog here, Harry Anything I can do to help No, said Harry so quickly he k
new he had sounded rude, no - I - I know what I m going to do, thanks.
Found 573 case-sensitive and unexact matches(hits), so far.
Processing .\Caterpillar.011.RAFT0.txt.lzma ...
Doing DECODE from HDD to RAM ...
Overall decode performance so far: 000,008KB/clock(EN) or 000,040KB/clock(DE)
Doing SEARCH for Pattern1 at once and flushing hit-sentences ...
000,000,574 They had no conception of the temptations of the underdog.
000,000,575 They had no conception of the temptations of the underdog.
Found 575 case-sensitive and unexact matches(hits), so far.
Total Rough Upload and Decode time: 653,274 clocks
Total Rough Search time: 127,084 clocks
Total time: 782 seconds
Total Lines encountered: 408,530,953
Total Search(non-mask) function invocations: 408,318,104
Total Search(MASK i.e. wildcard) function invocations: 0
Total Boyer-Moore-Horspool(whole chunks, not lines) hits: 636
Total Boyer-Moore-Horspool(whole chunks, not lines) time: 16,990 clocks
Total Boyer-Moore-Horspool(whole chunks, not lines) performance: 1,540KB/clock
Total Karp_Rabin_Kaze(whole chunks, not lines) hits: 636
Total Karp_Rabin_Kaze(whole chunks, not lines) time: 41,361 clocks
Total Karp_Rabin_Kaze(whole chunks, not lines) performance: 632KB/clock
Raccoondog: Done successfully.
Press a key to exit.
EVEREST Ultimate ...
Visual C++ Toolkit 20...
Salah-ed-din
Go to PROMPT
Caterpillar
Raccoondog
Recycle Bin
Start
Visual C++ Toolkit 2003 C...
Raccoondog
Raccoondog14++
Adobe Photoshop CS3 Ex...
10:38 PM


Salah-ed-din
Doing SEARCH for Pattern1 at once and flushing hit-sentences ...
000,000,569 Just because he is a boss, whatsoever he does looks bad, appears bad
, because deep down you resent that you are an underdog and somebody is a topdog
.
000,000,570 You resent the fact of being an underdog, so whatsoever is said look
s bad, but you cannot react; it will be too costly.
000,000,571 It is creating a division in yourself -- the division of the topdog
and the underdog, the division of the superior and the inferior, the division of
the saint and the sinner, the division of good and bad the division, basically,
of God and devil.
000,000,572 You create a topdog and an underdog, and so there is conflict.
000,000,573 I mean, Bagman continued, lowering his voice still further, you re t
he underdog here, Harry Anything I can do to help No, said Harry so quickly he k
new he had sounded rude, no - I - I know what I m going to do, thanks.
Found 573 case-sensitive and unexact matches(hits), so far.
Processing .\Caterpillar.011.RAFT0.txt.gz ...
Doing DECODE from HDD to RAM ...
Salah-ed-din decoded buffer size: 81,485,588
Overall decode performance so far: 000,039KB/clock(EN) or 000,117KB/clock(DE)
Doing SEARCH for Pattern1 at once and flushing hit-sentences ...
000,000,574 They had no conception of the temptations of the underdog.
000,000,575 They had no conception of the temptations of the underdog.
Found 575 case-sensitive and unexact matches(hits), so far.
Total Rough Upload and Decode time: 223,009 clocks
Total Rough Search time: 127,260 clocks
Total time: 351 seconds
Total Lines encountered: 408,530,953
Total Search(non-mask) function invocations: 408,318,104
Total Search(MASK i.e. wildcard) function invocations: 0
Total Boyer-Moore-Horspool(whole chunks, not lines) hits: 636
Total Boyer-Moore-Horspool(whole chunks, not lines) time: 17,071 clocks
Total Boyer-Moore-Horspool(whole chunks, not lines) performance: 1,533KB/clock
Total Karp_Rabin_Kaze(whole chunks, not lines) hits: 636
Total Karp_Rabin_Kaze(whole chunks, not lines) time: 41,110 clocks
Total Karp_Rabin_Kaze(whole chunks, not lines) performance: 636KB/clock
Salah-ed-din: Done successfully.
Press a key to exit.
Searches into a 'Salah-ed-din.lst' filelist file! Press 'Esc' to skip rest hits.
Recycle Bin
11:41 PM

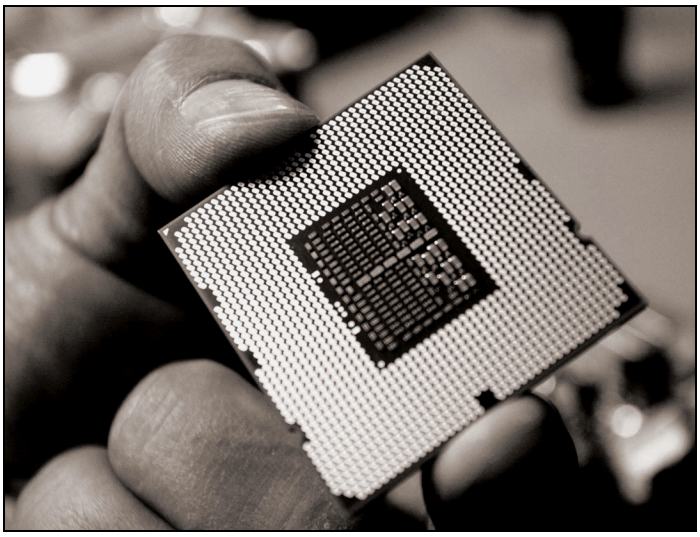

Salah-ed-din Hits - Mozilla Firefox
file:///D:/_KAZE_G.S._Corpus/Salah-ed-din.hits.pattern1.html
Raccoondog Hits
Salah-ed-din Hits
It was touching to see him identified with those who lose, with the 'underdog.'
People favour underdogs but follow only top dogs.
Fight for some underdogs anyway.
When, for instance, a Chaplin, or a Norman Wisdom, or some such little fellow portrays the eternal underdog we have an instant
People favour underdogs but follow only top dogs.
Fight for some underdogs anyway.
People favour underdogs but follow only top dogs.
Fight for some underdogs anyway.
Despite (or perhaps because of) their ugliness, and the fact that they are enslaved underdogs of a much uglier ruling class, t
This is an extremely valuable capacity in almost any kind of conflict, especially if the player is the underdog.
You need the keep the players in the balance sweet spot for as long as is practical in order to keep the game fun and let the
She enjoyed her freedom, and possessed herself with a sort of virginal passion; the warmth which she would have lavished on Bo
Here was an underdoga capable-looking one, tooout to seize power.
The boasted majesty of law was a majesty to which the underdogs only were expected to look up to in fear and trepidation.
Annually, however, he went abroad, telling the story of the underdog, usually making the big circuit from the East to the West
Sometimes the underdog wins.
And lots of other underdogs will follow the bottle-washers.
They help out the underdog and punish evildoers.
Copy
Select All
Search Google for "They help out t..."
View Selection Source
Momma had all the insolence of the underdog.
You win and I'm the underdog.
You've got a great reputation for helping the underdog.
She was sure that in his uncanny fashion, he would pick the underdog in the three-way fight and turn the odds in Doug's favor.
The Gay Old Dog 8 ---------- 11 One Basket The staid and harassed brother of three unwed and self
Thus Emma Goldman's most striking characteristic, her sympathy with the underdog, already became manifest in these early years
How pathetic is the stupidity of the underdog; how terribly tragic!
Done
11:43 PM


Caterpillar Hits - Mozilla Firefox
File Edit View History Bookmarks Tools Help
file:///D:/_KAZE_G.S._Corpus/_SearchIntoRafts/KAZE_G.S._Corpus_'.chm'_Caterpillar.html
Caterpillar Hits
12/02/2006 10:18 PM 12,818,755 Windows 2000 Professional Resource Kit.CHM
12/02/2006 10:18 PM 2,066,123 Windows 2000 Resource Kit - Server Operations Guide.chm
12/02/2006 10:18 PM 3,255,721 Windows 98 Resource kit book.CHM
12/02/2006 10:18 PM 3,347,293 Windows Architecture for Developers.chm
12/02/2006 10:18 PM 20,204,847 Windows Security Inside Out.chm
12/02/2006 10:18 PM 10,473,457 Windows Server 2003 For Dummies.chm
12/02/2006 10:18 PM 1,487,899 Windows Server 2003 TCP IP Protocols and Services Technical Reference.chm
12/02/2006 10:18 PM 35,907,122 WindowsXP Inside Out.CHM
12/02/2006 10:18 PM 916,828 WinDVD.chm
12/02/2006 10:18 PM 1,206,956 WinXPFAQ.chm
12/02/2006 10:18 PM 10,604,428 Wrox Professional C Sharp, 2nd Edition (2002) - Fly.chm
12/02/2006 10:18 PM 34,959,083 Wrox.Beginning.Dreamweaver.MX.eBook-LiB.chm
12/02/2006 10:18 PM 1,983,837 XLM in Action.chm
12/02/2006 10:18 PM 2,121,775 XLMAIN9.CHM
12/02/2006 10:18 PM 5,647,359 XML and SOAP Programming for BizTalk Servers.chm
12/02/2006 10:18 PM 10,068,537 XML CD Bookshelf v1.0.chm
12/02/2006 10:18 PM 3,171,065 XML Data Management. Native XML and XML-Enabled Database.chm
12/02/2006 10:18 PM 6,574,462 XML Programming (Core Reference).CHM
12/02/2006 10:18 PM 8,668,763 Zope Web Application Construction Kit.chm
000,353 [*.chm?][][][] Caterpillar link to target file: D:_KAZE_G.S._Corpus_SearchIntoRafts_RAFTbyRAFT_Description\RAFT4_og_
12/05/2006 02:35 PM 84,025,285 HumanEye.chm
12/05/2006 02:35 PM 4,506,543 Music Theory in Concept and Practice.chm
12/05/2006 02:35 PM 37,028,593 New.Oxford.Textbook.of.Psychiatry.eBook-EEn.chm
000,003 [*.chm?][][][] Caterpillar link to target file: D:_KAZE_G.S._Corpus_SearchIntoRafts_RAFTbyRAFT_Description\RAFT9_og_
12/11/2007 01:01 AM 5,564,176 [ccebook.cn]632_Oxford_Collocations_Dictionary.chm
12/11/2007 01:01 AM 1,048,710 ABC-CLIO,.Encyclopedia.of.Taboos.(2000).EEn.[1576070697].chm
12/11/2007 01:01 AM 9,010,165 British[1].National.Formulary.52.BNF-0853696691.chm
12/11/2007 01:01 AM 196,006,229 Cecil.22nd.ed(2).chm
12/11/2007 01:01 AM 2,361,154 Encyclopedia.of.Cryptology.eBook-EEn.chm
12/11/2007 01:01 AM 16,920,950 Horace Newcomb - Encyclopedia of Television 1579583946.chm
12/11/2007 01:01 AM 8,264,328 Newton_H._-_Newton_s_Telecom_Dictionary_20th__CMP_Books_2004_.chm
12/11/2007 01:01 AM 201,410,479 OTCNEPH.CHM
12/11/2007 01:01 AM 28,764,990 oxford_handbook_of_medical_sciences_0198528299.chm
12/11/2007 01:01 AM 10,639,209 pharmacotherapy_handbook__6th_ed_0071433228.chm
12/11/2007 01:01 AM 54,577,118 que.googlepedia.078973639x.chm
file://localhost/D:/_KAZE_G.S._Corpus/_SearchIntoRafts/_RAFTbyRAFT_Description/RAFT4_og_on_s.dir
Start
_SearchIntoRafts
_KAZE_G.S._Corpus_20...
Caterpillar Hits - Moz...
12:10 AM

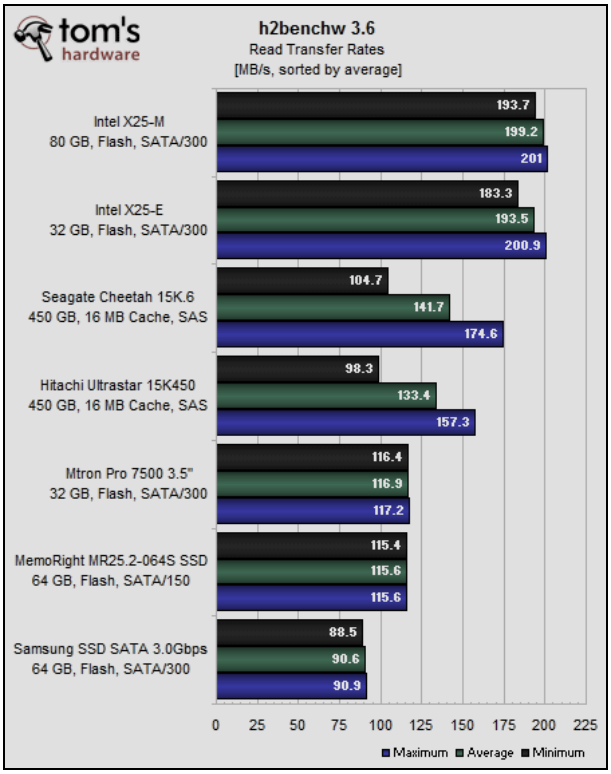

My Computer
My Network Places
My Documents
Go to PROMPT - ..\Leprechaun 24.9GB_280_LBL-files.lst 24.9GB_280_LBL-files.WORDLIST.txt 1720
02/02/2009 04:07 PM 99,262,696 Caterpillar.038.RAFT6.txt
02/02/2009 04:07 PM 98,921,731 Caterpillar.039.RAFT5.txt
02/02/2009 04:07 PM 99,368,547 Caterpillar.039.RAFT6.txt
02/02/2009 04:07 PM 99,402,909 Caterpillar.040.RAFT5.txt
02/02/2009 04:07 PM 99,585,975 Caterpillar.040.RAFT6.txt
02/02/2009 04:07 PM 99,606,116 Caterpillar.041.RAFT5.txt
02/02/2009 04:07 PM 99,252,269 Caterpillar.041.RAFT6.txt
02/02/2009 04:07 PM 99,504,726 Caterpillar.042.RAFT5.txt
02/02/2009 04:07 PM 97,741,086 Caterpillar.042.RAFT6.txt
02/02/2009 04:07 PM 18,209,850 Caterpillar.043.RAFT5.txt
02/02/2009 04:07 PM 98,427,832 Caterpillar.043.RAFT6.txt
02/02/2009 04:07 PM 99,366,044 Caterpillar.044.RAFT6.txt
02/02/2009 04:07 PM 99,500,993 Caterpillar.045.RAFT6.txt
02/02/2009 04:07 PM 99,375,220 Caterpillar.046.RAFT6.txt
02/02/2009 04:07 PM 6,174,634 Caterpillar.047.RAFT6.txt
280 File(s) 26,800,604,148 bytes
0 Dir(s) 6,051,012,608 bytes free
D:_KAZE_G.S._Corpus\Leprechaun_280>..\Leprechaun 24.9GB_280_LBL-files.
B_280_LBL-files.WORDLIST.txt 1720
Leprechaun(Fast and Greedy Word_Ripper), revision 12, written by Svalqy
Leprechaun: 'Oh, well, didn't you hear? Bigger is good, but jumbo is de
Size of input file with files for Leprechauning: 7560
Allocated memory in MB: 719
Size of Input TEXTual file: 98,855,869
/; Word count: 14,171,097 of them 185,935 distinct; Done: 64/64
Size of Input TEXTual file: 99,614,459
/; Word count: 30,915,710 of them 318,745 distinct; Done: 64/64
Size of Input TEXTual file: 99,119,770
|; Word count: 46,288,733 of them 382,431 distinct; Done: 64/64
Size of Input TEXTual file: 99,308,001
-; Word count: 63,789,380 of them 440,048 distinct; Done: 64/64
Size of Input TEXTual file: 99,252,455
/; Word count: 81,369,343 of them 573,613 distinct; Done: 64/64
Size of Input TEXTual file: 98,732,660
\; Word count: 98,720,674 of them 622,696 distinct; Done: 64/64
Size of Input TEXTual file: 99,610,283
/; Word count: 113,027,395 of them 794,486 distinct; Done: 64/64
Size of Input TEXTual file: 99,206,400
|; Word count: 130,028,173 of them 835,720 distinct; Done: 64/64
Size of Input TEXTual file: 99,388,561
-; Word count: 142,344,192 of them 1,237,987 distinct; Done: 53/64
Windows Task Manager
File Options View Help
Applications Processes Performance Networking
CPU Usage
52 %
CPU Usage History
PF Usage
1.00 GB
Page File Usage History
Totals
Handles 8543
Threads 478
Processes 51
Physical Memory (K)
Total 3010480
Available 2453492
System Cache 1126584
Commit Charge (K)
Total 1049860
Limit 2846072
Peak 1049892
Kernel Memory (K)
Total 84688
Paged 47964
Nonpaged 36724
Processes: 51
CPU Usage: 52%
Commit Charge: 1025M / 2779M
Salah-ed-din
Go to PROMPT
Caterpillar
Recycle Bin

```
#define ulPrime ((unsigned long) 0x00FF00F1)
#define ulBase  ((unsigned long) 127)
        // 9,223,372,036,854,775,807
// 257^7 =     74,051,159,531,521,793
// 257^8 = 19,031,147,999,601,100,801
// 127^9 =  8,594,754,748,609,397,887
// 57^10 =    362,033,331,456,891,249
// 13^16 =    665,416,609,183,179,841
//  5^12 =                244,140,625
// 13^8  =                815,730,721

long KarpRabinKazeHits (char * pbTarget,
     char * pbPattern,
     unsigned long cbTarget,
     unsigned long cbPattern)
{
    unsigned int    i;
    char *  pbTargetMax = pbTarget + cbTarget;
    char *  pbPatternMax = pbPattern + cbPattern;
    unsigned long  ulBaseToPowerMod = 1;
    register unsigned long  ulHashPattern = 0;
    unsigned long  ulHashTarget = 0;
long hits = 0;
//unsigned long count;
    //char *  buf1;
    //char *  buf2;

    if (cbPattern > cbTarget)
        return(0);

    // Compute the power of the left most character in base ulBase
    //for (i = 1; i < cbPattern; i++) ulBaseToPowerMod = (ulBase * ulBaseToPowerMod);

    // Calculate the hash function for the src (and the first dst)
    while (pbPattern < pbPatternMax)
    {
        // Below lines give 366KB/clock for 'underdog':
        //ulHashPattern = (ulHashPattern*ulBase + *pbPattern);
        //ulHashTarget = (ulHashTarget*ulBase + *pbTarget);
        pbPattern++;
        pbTarget++;
    }
        // Below lines give 436KB/clock for 'underdog' + requirement pattern to be 4 chars min.:
        //ulHashPattern = ( (*(long *)(pbPattern-cbPattern)) & 0xffffff00 ) + *(pbPattern-1);
        //ulHashTarget = ( (*(long *)(pbTarget-cbPattern)) & 0xffffff00 ) + *(pbTarget-1);
        // Below lines give 482KB/clock for 'underdog' + requirement pattern to be 2 chars min.:
        //ulHashPattern = ( (*(unsigned short *)(pbPattern-cbPattern)) | *(pbPattern-1) );
        //ulHashTarget = ( (*(unsigned short *)(pbTarget-cbPattern)) | *(pbTarget-1) );
        // Below lines give 482KB/clock for 'underdog' + requirement pattern to be 2 chars min.:
        //ulHashPattern = ( (*(unsigned short *)(pbPattern-cbPattern)) & 0xff00 ) + *(pbPattern-1);
        //ulHashTarget = ( (*(unsigned short *)(pbTarget-cbPattern)) & 0xff00 ) + *(pbTarget-1);
        // Below lines give 605KB/clock for 'underdog' + requirement pattern to be 2 chars min.:
        //ulHashPattern = ( (*(unsigned short *)(pbPattern-cbPattern))<<8 ) + *(pbPattern-1);
        //ulHashTarget = ( (*(unsigned short *)(pbTarget-cbPattern))<<8 ) + *(pbTarget-1);
        // Below lines give 668KB/clock for 'underdog':
        ulHashPattern = ( (*(char *)(pbPattern-cbPattern))<<8 ) + *(pbPattern-1);
        ulHashTarget = ( (*(char *)(pbTarget-cbPattern))<<8 ) + *(pbTarget-1);

    // Dynamically produce hash values for the string as we go
    for ( ;; )
    {
        if ( (ulHashPattern == ulHashTarget) && !memcmpKAZE(pbPattern-cbPattern, pbTarget-cbPattern, (unsigned int)cbPattern) )
       // if ( ulHashPattern == ulHashTarget ) {
       //
       //  count = cbPattern;
       //  buf1 = pbPattern-cbPattern;
       //  buf2 = pbTarget-cbPattern;
       //  while ( --count && *(char *)buf1 == *(char *)buf2 ) {
       //          buf1 = (char *)buf1 + 1;
       //          buf2 = (char *)buf2 + 1;
       //  }
       //
       //  if ( *((unsigned char *)buf1) - *((unsigned char *)buf2) == 0) hits++;
       //  }
            hits++;
            //return((long)(pbTarget-cbPattern));

        if (pbTarget == pbTargetMax)
            return(hits);

        // Below line gives 482KB/clock for 'underdog' + requirement pattern to be 2 chars min.:
        //ulHashTarget = ( (*(unsigned short *)(pbTarget+1-cbPattern)) | *pbTarget );
        // Below line gives 436KB/clock for 'underdog' + requirement pattern to be 4 chars min.:
        //ulHashTarget = ( (*(long *)(pbTarget+1-cbPattern)) & 0xffffff00 ) + *pbTarget;
//; Line 696
//        movsx   esi, BYTE PTR [ebx]
//        mov     ecx, DWORD PTR [edx+1]
//        and     ecx, -256                               ; ffffff00H
//        add     ecx, esi
        // Below line gives 482KB/clock for 'underdog' + requirement pattern to be 2 chars min.:
        //ulHashTarget = ( (*(unsigned short *)(pbTarget+1-cbPattern)) & 0xff00 ) + *pbTarget;
//; Line 691
//        movsx   esi, BYTE PTR [ebx]
//        xor     ecx, ecx
//        mov     cx, WORD PTR [edx+1]
//        and     ecx, 65280                              ; 0000ff00H
//        add     ecx, esi
        // Below line gives 605KB/clock for 'underdog' + requirement pattern to be 2 chars min.:
        //ulHashTarget = ( (*(unsigned short *)(pbTarget+1-cbPattern))<<8 ) + *pbTarget;
        // Below line gives 668KB/clock for 'underdog':
        ulHashTarget = ( (*(char *)(pbTarget+1-cbPattern))<<8 ) + *pbTarget;
//; Line 718
//        movsx   ecx, BYTE PTR [eax+1]
//        movsx   edx, BYTE PTR [ebp]
//        shl     ecx, 8
//        add     ecx, edx
        // Below line gives 366KB/clock for 'underdog':
        //ulHashTarget = (ulHashTarget - *(pbTarget-cbPattern)*ulBaseToPowerMod)*ulBase + *pbTarget;
        pbTarget++;
    }
}
```

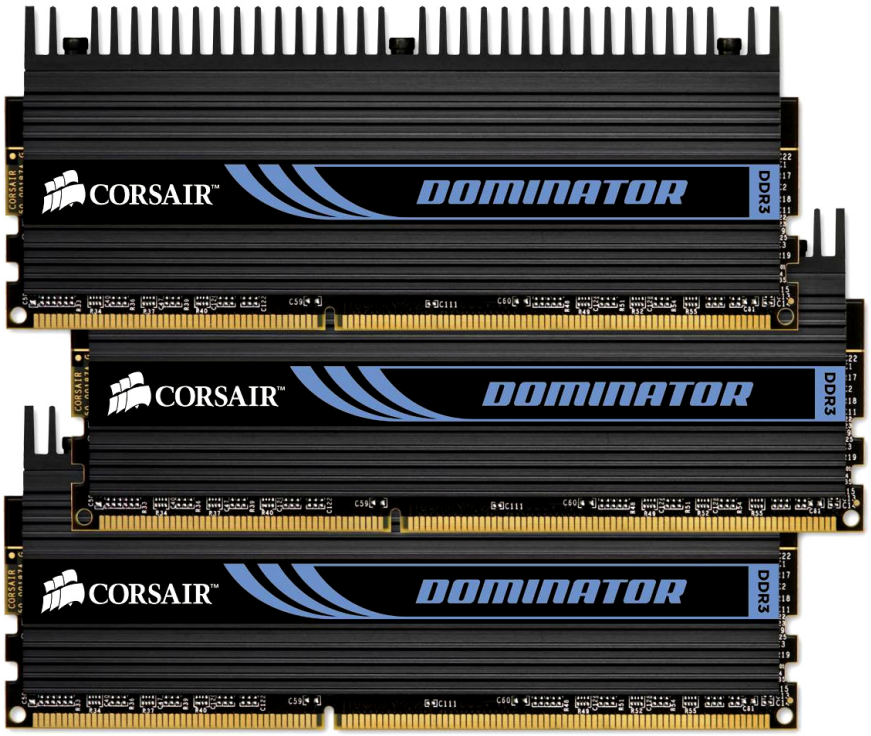
Core i7 *chip(731 million transistors) - the current dominator; plus* ***heart-touching-data-storages:***



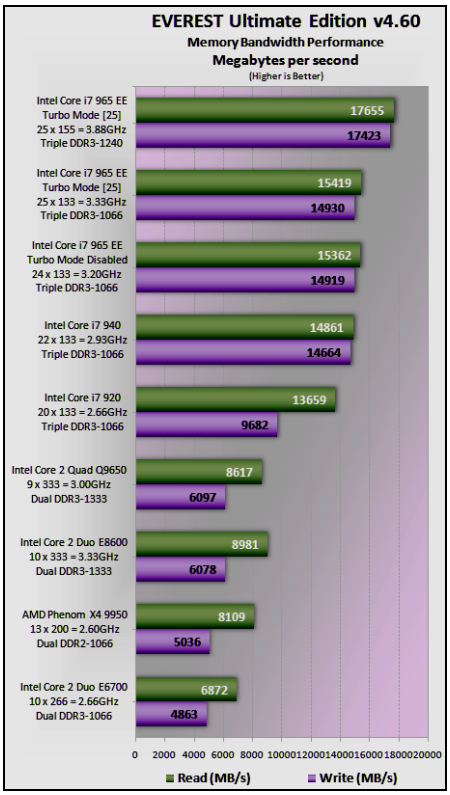
SSD drives *- simply the future:*

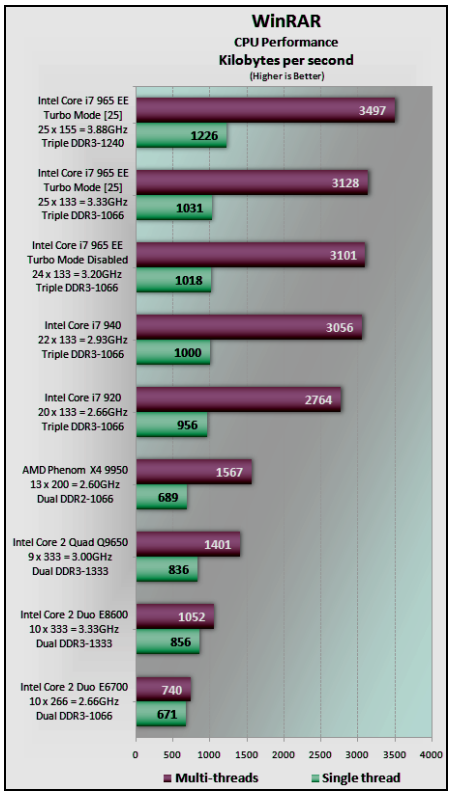


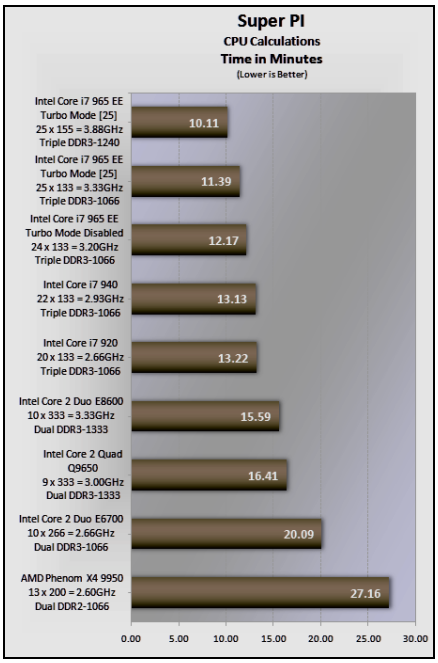
Triple channel DDR3 RAM - three pieces of beauty:

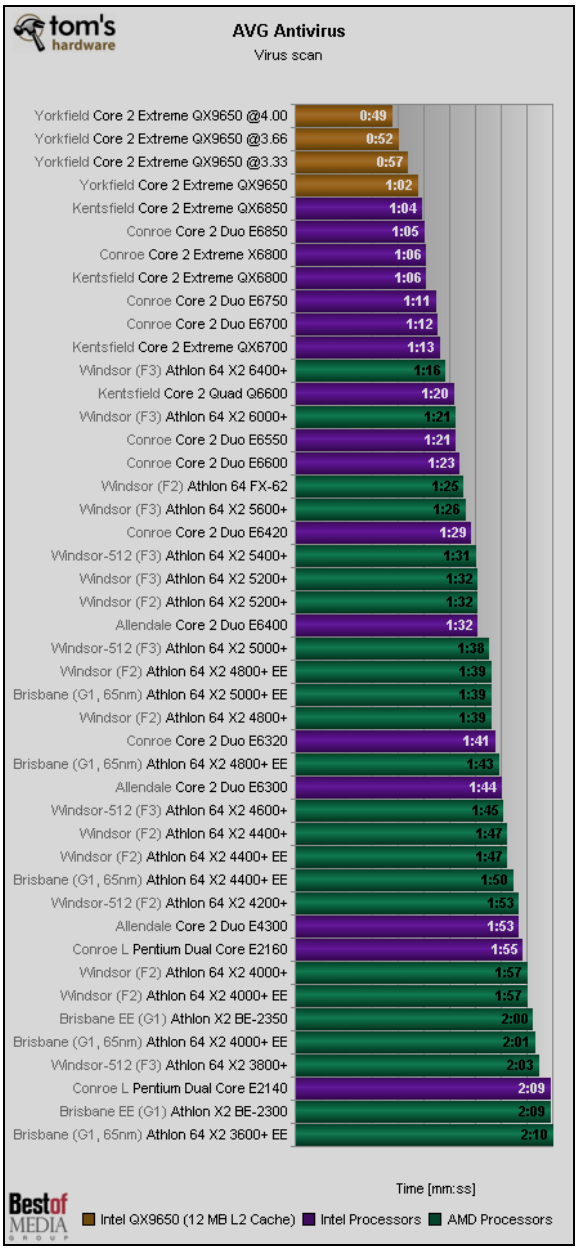


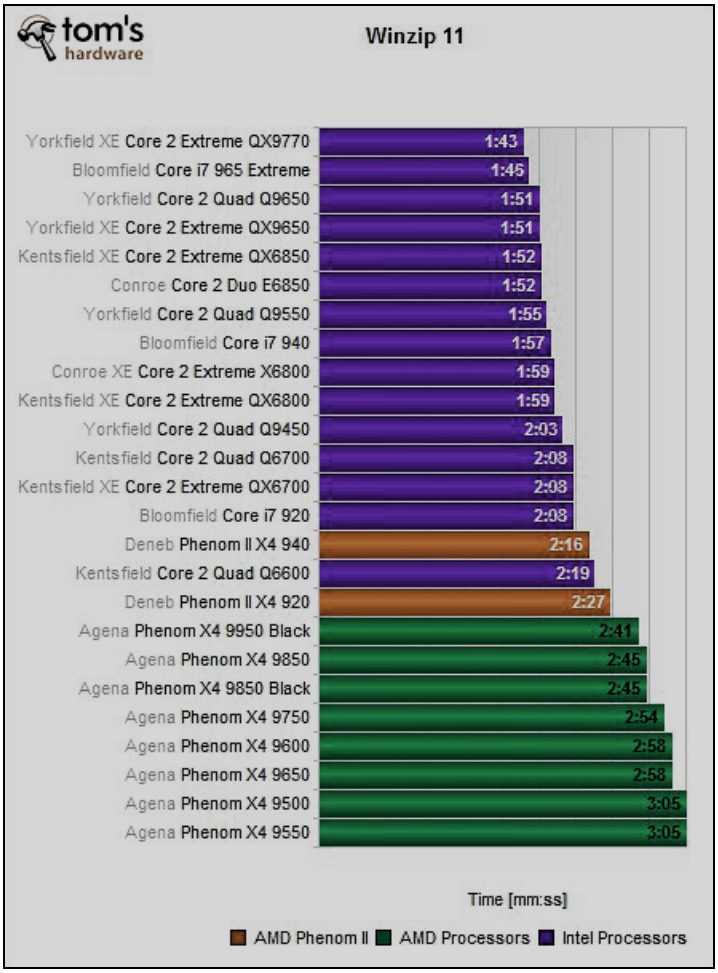
***Benchmarks** – single thread: what a shame:*











Dual Socket F (1207 FX) memory interface : 2x dual channel DDR2-800

CPU-Name	Clock	Multi	FSB / Hyper Tras.	Cache L1/L2	Voltage	Instruction sets	Energy Features	Power	Temp.	Transistors	Core	Step.	Process
Athlon 64 FX-74 (4x4)	4x 3000 MHz	15x free	200 MHz / HT1000	4x 64+64KB / 1MB	1.35 - 1.40V	MMX 3DNow! NX SSE SSE2 SSE3 x86-64 PA	Cool'n'Quiet	250 W	56°C	454 Mio.	Windsor FX	F3	90 nm
Athlon 64 FX-72 (4x4)	4x 2800 MHz	14x free	200 MHz / HT1000	4x 64+64KB / 1MB	1.35 - 1.40V	MMX 3DNow! NX SSE SSE2 SSE3 x86-64 PA	Cool'n'Quiet	250 W	63°C	454 Mio.	Windsor FX	F3	90 nm
Athlon 64 FX-70 (4x4)	4x 2600 MHz	13x free	200 MHz / HT1000	4x 64+64KB / 1MB	1.35 - 1.40V	MMX 3DNow! NX SSE SSE2 SSE3 x86-64 PA	Cool'n'Quiet	250 W	63°C	454 Mio.	Windsor FX	F3	90 nm

Socket AM2 / AM2+ (940 Pins) memory interface : dual channel DDR2-800 / 1066

CPU-Name	Clock	Multi	FSB / Hyper Tras.	Cache L1/L2	Voltage	Instruction sets	Energy Features	Power	Temp.	Transistors	Core	Step.	Process
Phenom X4 9950 Black	4x 2600 MHz	13x open	200 MHz / HT2000	4x 64+64KB / 4x 512 KB / 2 MB	1.05 - 1.30V	MMX 3DNow! NX SSE SSE2 SSE3 SSE4A x86-64 PA	Cool'n'Quiet 2	140 W	64°C	463 Mio.	Agena	B3	65 nm
Phenom X4 9850 Black	4x 2500 MHz	12.5x open	200 MHz / HT2000	4x 64+64KB / 4x 512 KB / 2 MB	1.05 - 1.3V	MMX 3DNow! NX SSE SSE2 SSE3 SSE4A x86-64 PA	Cool'n'Quiet 2	125 W	61°C	463 Mio.	Agena	B3	65 nm
Phenom X4 9850	4x 2500 MHz	12.5x	200 MHz / HT2000	4x 64+64KB / 4x 512 KB / 2 MB	1.05 - 1.3V	MMX 3DNow! NX SSE SSE2 SSE3 SSE4A x86-64 PA	Cool'n'Quiet 2	125 W	61°C	463 Mio.	Agena	B3	65 nm
Phenom X4 9750	4x 2400 MHz	12x	200 MHz / HT2000	4x 64+64KB / 4x 512 KB / 2 MB	1.20/1.25/1.30V	MMX 3DNow! NX SSE SSE2 SSE3 SSE4A x86-64 PA	Cool'n'Quiet 2	125 W	61°C	463 Mio.	Agena	B3	65 nm
Phenom X4 9750	4x 2400 MHz	12x	200 MHz / HT2000	4x 64+64KB / 4x 512 KB / 2 MB	1.10/1.15/1.20/1.25V	MMX 3DNow! NX SSE SSE2 SSE3 SSE4A x86-64 PA	Cool'n'Quiet 2	95 W	70 °C	463 Mio.	Agena	B3	65 nm
Phenom X4 9650	4x 2300 MHz	11.5x	200 MHz / HT1800	4x 64+64KB / 4x 512 KB / 2 MB	1.10/1.15/1.20/1.25V	MMX 3DNow! NX SSE SSE2 SSE3 SSE4A x86-64 PA	Cool'n'Quiet 2	95 W	70 °C	463 Mio.	Agena	B3	65 nm
Phenom X4 9550	4x 2200 MHz	11x	200 MHz / HT1800	4x 64+64KB / 4x 512 KB / 2 MB	1.10/1.15/1.20/1.25V	MMX 3DNow! NX SSE SSE2 SSE3 SSE4A x86-64 PA	Cool'n'Quiet 2	95 W	70 °C	463 Mio.	Agena	B3	65 nm
Phenom X4 9350e	4x 2000 MHz	10x	200 MHz / HT1800	4x 64+64KB / 4x 512 KB / 2 MB	1.0 - 1.125V	MMX 3DNow! NX SSE SSE2 SSE3 SSE4A x86-64 PA	Cool'n'Quiet 2	65 W	70 °C	463 Mio.	Agena	B3	65 nm
Phenom X4 9150e	4x 1800 MHz	9x	200 MHz / HT1800	4x 64+64KB / 4x 512 KB / 2 MB	1.0 - 1.125V	MMX 3DNow! NX SSE SSE2 SSE3 SSE4A x86-64 PA	Cool'n'Quiet 2	65 W	70 °C	463 Mio.	Agena	B3	65 nm
Phenom X4 9100e	4x 1800 MHz	9x	200 MHz / HT1800	4x 64+64KB / 4x 512 KB / 2 MB	1.10/1.125/1.15V	MMX 3DNow! NX SSE SSE2 SSE3 SSE4A x86-64 PA	Cool'n'Quiet 2	65 W	61°C	463 Mio.	Agena	B2	65 nm
Phenom X4 9700 Sample	4x 2400 MHz	12x	200 MHz / HT2000	4x 64+64KB / 4x 512 KB / 2 MB	unknown	MMX 3DNow! NX SSE SSE2 SSE3 SSE4A x86-64 PA	Cool'n'Quiet 2	95 W	unknown	463 Mio.	Agena	B2	65 nm
Phenom X4 9600 Black	4x 2300 MHz	11.5x open	200 MHz / HT1800	4x 64+64KB / 4x 512 KB / 2 MB	1.10/1.15/1.20/1.25V	MMX 3DNow! NX SSE SSE2 SSE3 SSE4A x86-64 PA	Cool'n'Quiet 2	95 W	70 °C	463 Mio.	Agena	B2	65 nm
Phenom X4 9600	4x 2300 MHz	11.5x	200 MHz / HT1800	4x 64+64KB / 4x 512 KB / 2 MB	1.10/1.15/1.20/1.25V	MMX 3DNow! NX SSE SSE2 SSE3 SSE4A x86-64 PA	Cool'n'Quiet 2	95 W	70 °C	463 Mio.	Agena	B2	65 nm
Phenom X4 9500	4x 2200 MHz	11x	200 MHz / HT1800	4x 64+64KB / 4x 512 KB / 2 MB	1.10/1.15/1.20/1.25V	MMX 3DNow! NX SSE SSE2 SSE3 SSE4A x86-64 PA	Cool'n'Quiet 2	95 W	70 °C	463 Mio.	Agena	B2	65 nm
Phenom X3 8750	3x 2400 MHz	12x	200 MHz / HT1800	3x 64+64KB / 3x 512 KB / 2 MB	1.05V - 1.25V	MMX 3DNow! NX SSE SSE2 SSE3 SSE4A x86-64 PA	Cool'n'Quiet 2	95 W	70 °C		Toliman	B3	65 nm
Phenom X3 8650	3x 2300 MHz	11.5x	200 MHz / HT1800	3x 64+64KB / 3x 512 KB / 2 MB	1.05V - 1.25V	MMX 3DNow! NX SSE SSE2 SSE3 SSE4A x86-64 PA	Cool'n'Quiet 2	95 W	70 °C		Toliman	B3	65 nm
Phenom X3 8600	3x 2300 MHz	11.5x	200 MHz / HT1800	3x 64+64KB / 3x 512 KB / 2 MB	1.05V - 1.25V	MMX 3DNow! NX SSE SSE2 SSE3 SSE4A x86-64 PA	Cool'n'Quiet 2	95 W	70 °C		Toliman	B2	65 nm
Phenom X3 8450	3x 2100 MHz	10.5x	200 MHz / HT1800	3x 64+64KB / 3x 512 KB / 2 MB	1.05V - 1.25V	MMX 3DNow! NX SSE SSE2 SSE3 SSE4A x86-64 PA	Cool'n'Quiet 2	95 W	70 °C		Toliman	B3	65 nm
Phenom X3 8400	3x 2100 MHz	10.5x	200 MHz / HT1800	3x 64+64KB / 3x 512 KB / 2 MB	1.05V - 1.25V	MMX 3DNow! NX SSE SSE2 SSE3 SSE4A x86-64 PA	Cool'n'Quiet 2	95 W	70 °C		Toliman	B2	65 nm
Athlon 64 X2 6500+	2x 2300 MHz	11,5x		2x 64+64KB / 2x 512 KB / 2 MB		MMX 3DNow! NX SSE SSE2 SSE3 SSE4A x86-64 PA	Cool'n'Quiet 2	95 W			Kuma	B3	65 nm
Athlon 64 FX FX-62	2x 2800 MHz	14x free	200 MHz / HT1000	2x 64+64KB / 1MB	1.35 - 1.40V	MMX 3DNow! NX SSE SSE2 SSE3 x86-64 PA	Cool'n'Quiet	125 W	55-63°C	227 Mio.	Windsor	F2	90 nm
Athlon 64 X2 6400+ Black	2x 3200 MHz	16x	200 MHz / HT1000	2x 64+64KB / 1MB	1.35-1.40V	MMX 3DNow! NX SSE SSE2 SSE3 x86-64 PA	Cool'n'Quiet	125 W	55-63°C	227 Mio.	Windsor	F3	90 nm
Athlon 64 X2 6000+	2x 3000 MHz	15x	200 MHz / HT1000	2x 64+64KB / 1MB	1.35 - 1.40V	MMX 3DNow! NX SSE SSE2 SSE3 x86-64 PA	Cool'n'Quiet	125 W	55-63°C	227 Mio.	Windsor	F3	90 nm
Athlon 64 X2 5600+	2x 2800 MHz	14x	200 MHz / HT1000	2x 64+64KB / 1MB	1.30 - 1.35V	MMX 3DNow! NX SSE SSE2 SSE3 x86-64 PA	Cool'n'Quiet	89 W	55-70°C	227 Mio.	Windsor	F3	90 nm
Athlon 64 X2 5400+ Black	2x 2800 MHz	14x	200 MHz / HT1000	2x 64+64KB / 512 KB	1.30 - 1.35V	MMX 3DNow! NX SSE SSE2 SSE3 x86-64 PA	Cool'n'Quiet	89 W	55-70°C	154 Mio.	Windsor-512	F3	90 nm
Athlon 64 X2 5400+	2x 2800 MHz	14x	200 MHz / HT1000	2x 64+64KB / 512 KB	1.30 - 1.35V	MMX 3DNow! NX SSE SSE2 SSE3 x86-64 PA	Cool'n'Quiet	89 W	55-70°C	154 Mio.	Windsor-512	F3	90 nm
Athlon 64 X2 5200+	2x 2600 MHz	13x	200 MHz / HT1000	2x 64+64KB / 1MB	1.30V / 1.35V	MMX 3DNow! NX SSE SSE2 SSE3 x86-64 PA	Cool'n'Quiet	89 W	55-70°C	227 Mio.	Windsor	F2	90 nm
Athlon 64 X2 5000+ Black	2x 2600 MHz	13x	200 MHz / HT1000	2x 64+64KB / 512 KB	1.30V / 1.35V	MMX 3DNow! NX SSE SSE2 SSE3 x86-64 PA	Cool'n'Quiet	89 W	55-70°C	154 Mio.	Windsor-512	F2	90 nm
Athlon 64 X2 5000+	2x 2600 MHz	13x	200 MHz / HT1000	2x 64+64KB / 512 KB	1.30V / 1.35V	MMX 3DNow! NX SSE SSE2 SSE3 x86-64 PA	Cool'n'Quiet	89 W	55-70°C	154 Mio.	Windsor-512	F2	90 nm
Athlon 64 X2 5000+	2x 2600 MHz	13x	200 MHz / HT1000	2x 64+64KB / 512 KB	1.30V / 1.35V	MMX 3DNow! NX SSE SSE2 SSE3 x86-64 PA	Cool'n'Quiet	89 W	55-70°C	154 Mio.	Windsor-512	F2	90 nm
Athlon 64 X2 4800+	2x 2400 MHz	12x	200 MHz / HT1000	2x 64+64KB / 1MB	1.30V / 1.35V	MMX 3DNow! NX SSE SSE2 SSE3 x86-64 PA	Cool'n'Quiet	89 W	55-70°C	227 Mio.	Windsor	F2	90 nm
Athlon 64 X2 4600+	2x 2400 MHz	12x	200 MHz / HT1000	2x 64+64KB / 512 KB	1.30V / 1.35V	MMX 3DNow! NX SSE SSE2 SSE3 x86-64 PA	Cool'n'Quiet	89 W	55-70°C	154 Mio.	Windsor-512	F2	90 nm
Athlon 64 X2 4400+	2x 2200 MHz	11x	200 MHz / HT1000	2x 64+64KB / 1MB	1.30V / 1.35V	MMX 3DNow! NX SSE SSE2 SSE3 x86-64 PA	Cool'n'Quiet	89 W	55-70°C	227 Mio.	Windsor	F2	90 nm
Athlon 64 X2 4200+	2x 2200 MHz	11x	200 MHz / HT1000	2x 64+64KB / 512 KB	1.30V / 1.35V	MMX 3DNow! NX SSE SSE2 SSE3 x86-64 PA	Cool'n'Quiet	89 W	55-72°C	154 Mio.	Windsor-512	F2	90 nm
Athlon 64 X2 4000+	2x 2000 MHz	10x	200 MHz / HT1000	2x 64+64KB / 1MB	1.30V / 1.35V	MMX 3DNow! NX SSE SSE2 SSE3 x86-64 PA	Cool'n'Quiet	89 W	55-70°C	227 Mio.	Windsor	F2	90 nm
Athlon 64 X2 3800+	2x 2000 MHz	10x	200 MHz / HT1000	2x 64+64KB / 512 KB	1.30V / 1.35V	MMX 3DNow! NX SSE SSE2 SSE3 x86-64 PA	Cool'n'Quiet	89 W	55-70°C	154 Mio.	Windsor-512	F2	90 nm

Socket 771 memory interface : dual / quad channel FB-DIMM DDR3-667/800

CPU-Name	Clock	Multi	FSB	Cache L1/L2	Voltage	Instruction sets	Energy Features	Power	Temp.	Trans.	Core	Stepping	Process
Core 2 Extreme QX9775	4x 3200 MHz	8x (open)	400 MHz QDR	4x 32+32 / 2x 6144 KB	1.212 V	MMX SSE SSE2 SSE3 SSSE3 SSE4.1 NX EM64T VT	TM2 C1E EIST	150 W	63°C	820 Mio.	Yorkfield XE	C0	45 nm

Socket 775 memory interface : dual channel DDR2-533/667/800 DDR3-1066/1333

CPU-Name	Clock	Multi	FSB	Cache L1/L2	Voltage	Instruction sets	Energy Features	Power	Temp.	Trans.	Core	Stepping	Process
Core 2 Extreme QX9770	4x 3200 MHz	free (8x)	400 MHz QDR	4x 32+32 / 2x 6144 KB	1.3625 V	MMX SSE SSE2 SSE3 SSSE3 SSE4.1 NX EM64T VT	TM2 C1E EIST	136 W	55.5°C	820 Mio.	Yorkfield XE	C1	45 nm
Core 2 Extreme QX9650	4x 3000 MHz	9x (open)	333 MHz QDR	4x 32+32 / 2x 6144 KB	1.3625 V	MMX SSE SSE2 SSE3 SSSE3 SSE4.1 NX EM64T VT	TM2 C1E EIST	130 W	64.5°C	820 Mio.	Yorkfield XE	C0 C1	45 nm
Core 2 Quad Q9650	4x 3000 MHz	9x	333 MHz QDR	4x 32+32 / 2x 6144 KB		MMX SSE SSE2 SSE3 SSSE3 SSE4.1 NX EM64T VT	TM2 C1E EIST	95 W		820 Mio.	Yorkfield	C1	45 nm
Core 2 Quad Q9550	4x 2833 MHz	8.5x	333 MHz QDR	4x 32+32 / 2x 6144 KB	1.3625 V	MMX SSE SSE2 SSE3 SSSE3 SSE4.1 NX EM64T VT	TM2 C1E EIST	95 W	71.4°C	820 Mio.	Yorkfield	C1	45 nm
Core 2 Quad Q9450	4x 2666 MHz	8x	333 MHz QDR	4x 32+32 / 2x 6144 KB	1.3625 V	MMX SSE SSE2 SSE3 SSSE3 SSE4.1 NX EM64T VT	TM2 C1E EIST	95 W	71.4°C	820 Mio.	Yorkfield	C1	45 nm
Core 2 Quad Q9400	4x 2666 MHz	8x	333 MHz QDR	4x 32+32 / 2x 3072 KB		MMX SSE SSE2 SSE3 SSSE3 SSE4.1 NX EM64T VT	TM2 C1E EIST	95 W			Yorkfield	C1	45 nm
Core 2 Quad Q9300	4x 2500 MHz	7.5x	333 MHz QDR	4x 32+32 / 2x 3072 KB	1.3625 V	MMX SSE SSE2 SSE3 SSSE3 SSE4.1 NX EM64T VT	TM2 C1E EIST	95 W	71.4°C		Yorkfield	M1	45 nm
Core 2 Quad Q8200	4x 2333 MHz	7x	333 MHz QDR	4x 32+32 / 2x 2048 KB		MMX SSE SSE2 SSE3 SSSE3 SSE4.1 NX EM64T VT	TM2 C1E EIST	95 W			Yorkfield	M1	45 nm
Core 2 Duo E8600	2x 3333 MHz	10x	333 MHz QDR	2x 32+32 / 1x 6144 KB		MMX SSE SSE2 SSE3 SSSE3 SSE4.1 NX EM64T VT	TM2 C1E EIST	65 W		410 Mio.	Wolfdale	E0	45 nm
Core 2 Duo E8500	2x 3166 MHz	9.5x	333 MHz QDR	2x 32+32 / 1x 6144 KB	1.3625 V	MMX SSE SSE2 SSE3 SSSE3 SSE4.1 NX EM64T VT	TM2 C1E EIST	65 W	72.4°C	410 Mio.	Wolfdale	C0	45 nm
Core 2 Duo E8400	2x 3000 MHz	9x	333 MHz QDR	2x 32+32 / 1x 6144 KB	1.3625 V	MMX SSE SSE2 SSE3 SSSE3 SSE4.1 NX EM64T VT	TM2 C1E EIST	65 W	72.4°C	410 Mio.	Wolfdale	C0	45 nm
Core 2 Duo E8300	2x 2833 MHz	8.5x	333 MHz QDR	2x 32+32 / 1x 6144 KB	1.3625 V	MMX SSE SSE2 SSE3 SSSE3 SSE4.1 NX EM64T VT	TM2 C1E EIST	65 W	72.4°C	410 Mio.	Wolfdale	C0	45 nm
Core 2 Duo E8200	2x 2666 MHz	8x	333 MHz QDR	2x 32+32 / 1x 6144 KB	1.3625 V	MMX SSE SSE2 SSE3 SSSE3 SSE4.1 NX EM64T VT	TM2 C1E EIST	65 W	72.4°C	410 Mio.	Wolfdale	C0	45 nm
Core 2 Duo E8190	2x 2666 MHz	8x	333 MHz QDR	2x 32+32 / 1x 6144 KB	1.3625 V	MMX SSE SSE2 SSE3 SSSE3 SSE4.1 NX EM64T VT	TM2 C1E EIST	65 W	72.4°C	410 Mio.	Wolfdale	C0	45 nm
Core 2 Duo E7300	2x 2666 MHz	10x	266 MHz QDR	2x 32+32 / 1x 3072 KB		MMX SSE SSE2 SSE3 SSSE3 SSE4.1 NX EM64T VT	TM2 C1E EIST	65 W			Wolfdale	M0	45 nm
Core 2 Duo E7200	2x 2533 MHz	9.5x	266 MHz QDR	2x 32+32 / 1x 3072 KB	1.3625 V	MMX SSE SSE2 SSE3 SSSE3 SSE4.1 NX EM64T VT	TM2 C1E EIST	65 W	74.1°C		Wolfdale	M0	45 nm
Core 2 Extreme QX6850	4x 3000 MHz	9x (open)	333 MHz QDR	4x 32+32 / 2x 4096 KB		MMX SSE SSE2 SSE3 SSSE3 NX EM64T VT	TM2 C1E EIST	130 W	64.5°C	582 Mio.	Kentsfield XE	G0	65 nm
Core 2 Extreme QX6800	4x 2933 MHz	11x (open)	266 MHz QDR	4x 32+32 / 2x 4096 KB	1.37 V	MMX SSE SSE2 SSE3 SSSE3 NX EM64T VT	TM2 C1E EIST	130 W	54.8°C	582 Mio.	Kentsfield XE	B3 G0	65 nm
Core 2 Extreme QX6700	4x 2666 MHz	10x (open)	266 MHz QDR	4x 32+32 / 2x 4096 KB	1.37 V	MMX SSE SSE2 SSE3 SSSE3 NX EM64T VT	TM2 C1E EIST	130 W	65°C	582 Mio.	Kentsfield XE	B3	65 nm
Core 2 Quad Q6700	4x 2666 MHz	10x	266 MHz QDR	4x 32+32 / 2x 4096 KB	1.37 V	MMX SSE SSE2 SSE3 SSSE3 NX EM64T VT	TM2 C1E EIST	95 W	71°C	582 Mio.	Kentsfield	G0	65 nm
Core 2 Quad Q6600	4x 2400 MHz	9x	266 MHz QDR	4x 32+32 / 2x 4096 KB	1.37 V	MMX SSE SSE2 SSE3 SSSE3 NX EM64T VT	TM2 C1E EIST	105 W	62.2°C	582 Mio.	Kentsfield	B3	65 nm
Core 2 Quad Q6600	4x 2400 MHz	9x	266 MHz QDR	4x 32+32 / 2x 4096 KB	1.37 V	MMX SSE SSE2 SSE3 SSSE3 NX EM64T VT	TM2 C1E EIST	95 W	71°C	582 Mio.	Kentsfield	G0	65 nm
Core 2 Extreme X6800	2x 2933 MHz	11x (open)	266 MHz QDR	2x 32+32 / 1x 4096 KB	1.35 V	MMX SSE SSE2 SSE3 SSSE3 NX EM64T VT	TM2 C1E EIST	75 W	60.4°C	291 Mio.	Conroe XE	B0 B1 B2	65 nm
Core 2 Duo E6850	2x 3000 MHz	9x	333 MHz QDR	4x 32+32 / 2x 4096 KB	1.5 V	MMX SSE SSE2 SSE3 SSSE3 NX EM64T VT	TM2 C1E EIST	65 W	72°C	291 Mio.	Conroe	G0	65 nm
Core 2 Duo E6750	2x 2666 MHz	8x	333 MHz QDR	4x 32+32 / 2x 4096 KB	1.5 V	MMX SSE SSE2 SSE3 SSSE3 NX EM64T VT	TM2 C1E EIST	65 W	72°C	291 Mio.	Conroe	G0	65 nm
Core 2 Duo E6550	2x 2333 MHz	7x	333 MHz QDR	4x 32+32 / 2x 4096 KB	1.5 V	MMX SSE SSE2 SSE3 SSSE3 NX EM64T VT	TM2 C1E EIST	65 W	72°C	291 Mio.	Conroe	G0	65 nm
Core 2 Duo E6540	2x 2333 MHz	7x	333 MHz QDR	4x 32+32 / 2x 4096 KB	1.5 V	MMX SSE SSE2 SSE3 SSSE3 NX EM64T VT	TM2 C1E EIST	65 W	72°C	291 Mio.	Conroe	G0	65 nm
Core 2 Duo E6700	2x 2667 MHz	10x	266 MHz QDR	2x 32+32 / 1x4096 KB	1.35 V	MMX SSE SSE2 SSE3 SSSE3 NX EM64T VT	TM2 C1E EIST	65 W	60.1°C	291 Mio.	Conroe	B0 B1 B2	65 nm
Core 2 Duo E6600	2x 2400 MHz	9x	266 MHz QDR	2x 32+32 / 1x4096 KB	1.32 V	MMX SSE SSE2 SSE3 SSSE3 NX EM64T VT	TM2 C1E EIST	65 W	60.1°C	291 Mio.	Conroe	B0 B1 B2	65 nm
Core 2 Duo E6400	2x 2133 MHz	8x	266 MHz QDR	2x 32+32 / 1x2048 KB	1.35 V	MMX SSE SSE2 SSE3 SSSE3 NX EM64T VT	TM2 C1E EIST	65 W	61.4°C	291 Mio.	Conroe-2048	B2	65 nm
Core 2 Duo E6300	2x 1866 MHz	7x	266 MHz QDR	2x 32+32 / 1x2048 KB	1.35 V	MMX SSE SSE2 SSE3 SSSE3 NX EM64T VT	TM2 C1E EIST	65 W	61.4°C	294 Mio.	Conroe-2048	B2	65 nm
Core 2 Duo E6420	2x 2133 MHz	8x	266 MHz QDR	2x 32+32 / 1x4096 KB		MMX SSE SSE2 SSE3 SSSE3 NX EM64T VT	TM2 C1E EIST	65 W	60.1°C	291 Mio.	Conroe	B2	65 nm
Core 2 Duo E6320	2x 1866 MHz	7x	266 MHz QDR	2x 32+32 / 1x4096 KB		MMX SSE SSE2 SSE3 SSSE3 NX EM64T VT	TM2 C1E EIST	65 W	60.1°C	291 Mio.	Conroe	B2	65 nm
Core 2 Duo E6400	2x 2133 MHz	8x	266 MHz QDR	2x 32+32 / 1x2048 KB	1.32 V	MMX SSE SSE2 SSE3 SSSE3 NX EM64T VT	TM2 C1E EIST	65 W	61.4°C	167 Mio.	Allendale	B0 B1 B2 L2	65 nm
Core 2 Duo E6300	2x 1866 MHz	7x	266 MHz QDR	2x 32+32 / 1x2048 KB	1.32 V	MMX SSE SSE2 SSE3 SSSE3 NX EM64T VT	TM2 C1E EIST	65 W	61.4°C	167 Mio.	Allendale	B1 B2 L2	65 nm
Core 2 Duo E4700	2x 2600 MHz	13x	200 MHz QDR	2x 32+32 / 1x2048 KB	1.312 V	MMX SSE SSE2 SSE3 SSSE3 NX EM64T	TM2 C1E EIST	65 W	73.3°C	167 Mio.	Allendale	G0	65 nm
Core 2 Duo E4600	2x 2400 MHz	12x	200 MHz QDR	2x 32+32 / 1x2048 KB	1.312 V	MMX SSE SSE2 SSE3 SSSE3 NX EM64T	TM2 C1E EIST	65 W	73.3°C	167 Mio.	Allendale	M0	65 nm
Core 2 Duo E4500	2x 2200 MHz	11x	200 MHz QDR	2x 32+32 / 1x2048 KB		MMX SSE SSE2 SSE3 SSSE3 NX EM64T	TM2 C1E EIST	65 W	73.3°C	167 Mio.	Allendale	L2 M0	65 nm
Core 2 Duo E4400	2x 2000 MHz	10x	200 MHz QDR	2x 32+32 / 1x2048 KB	1.31 V	MMX SSE SSE2 SSE3 SSSE3 NX EM64T	TM2 C1E EIST	65 W	61.4°C	167 Mio.	Allendale	L2	65 nm
Core 2 Duo E4300	2x 1800 MHz	9x	200 MHz QDR	2x 32+32 / 1x2048 KB	1.32 V	MMX SSE SSE2 SSE3 SSSE3 NX EM64T	TM2 C1E EIST	65 W	61.4°C	167 Mio.	Allendale	L2	65 nm
Pentium Dual Core E2220	2x 2400 MHz	12x	200 MHz QDR	2x 32+32 / 1x1024 KB	1.31 V	MMX SSE SSE2 SSE3 SSSE3 NX EM64T	TM2 C1E EIST	65 W			Conroe-L	M0	65 nm
Pentium Dual Core E2200	2x 2200 MHz	11x	200 MHz QDR	2x 32+32 / 1x1024 KB	1.31 V	MMX SSE SSE2 SSE3 SSSE3 NX EM64T	TM2 C1E EIST	65 W			Conroe-L	M0	65 nm
Pentium Dual Core E2180	2x 2000 MHz	10x	200 MHz QDR	2x 32+32 / 1x1024 KB	1.31 V	MMX SSE SSE2 SSE3 SSSE3 NX EM64T	TM2 C1E EIST	65 W			Conroe-L	M0	65 nm
Pentium Dual Core E2160	2x 1800 MHz	9x	200 MHz QDR	2x 32+32 / 1x1024 KB	1.31 V	MMX SSE SSE2 SSE3 SSSE3 NX EM64T	TM2 C1E EIST	65 W	61.4°C		Conroe-L	L2 G0 M0	65 nm
Pentium Dual Core E2140	2x 1600 MHz	8x	200 MHz QDR	2x 32+32 / 1x1024 KB	1.31 V	MMX SSE SSE2 SSE3 SSSE3 NX EM64T	TM2 C1E EIST	65 W	61.4°C		Conroe-L	L2 G0 M0	65 nm
Celeron E1400	2x 2000 MHz	10x	200 MHz QDR	2x 32+32 / 512 KB	1.312 V	MMX SSE SSE2 SSE3 SSSE3 NX EM64T	TM2	65 W	73.3°C		Conroe-L	M0	65 nm
Celeron E1200	2x 1600 MHz	8x	200 MHz QDR	2x 32+32 / 512 KB	1.312 V	MMX SSE SSE2 SSE3 SSSE3 NX EM64T	TM2	65 W	73.3°C		Conroe-L	M0	65 nm
Celeron 440	2000 MHz	10x	200 MHz QDR	32+32 / 512 KB	1.3375 V	MMX SSE SSE2 SSE3 SSSE3 NX EM64T	TM2	35 W	60.4°C		Conroe-L	A1	65 nm
Celeron 430	1800 MHz	9x	200 MHz QDR	32+32 / 512 KB	1.3375 V	MMX SSE SSE2 SSE3 SSSE3 NX EM64T	TM2	35 W	60.4°C		Conroe-L	A1	65 nm
Celeron 420	1600 MHz	8x	200 MHz QDR	32+32 / 512 KB	1.3375 V	MMX SSE SSE2 SSE3 SSSE3 NX EM64T	TM2	35 W	60.4°C		Conroe-L	A1	65 nm
Pentium EE 965	2x 3724 MHz	14x	266 MHz QDR	12KµOps+16/2048 KB	1.30 V	MMX SSE SSE2 SSE3 NX EM64T VT HT	TM1 C1E	130 W	68.6°C	376 Mio.	Presler	C1	65 nm
Pentium EE 955	2x 3466 MHz	13x	266 MHz QDR	12KµOps+16/2048 KB	1.30 V	MMX SSE SSE2 SSE3 NX EM64T VT HT	TM1	130 W	68.6°C	376 Mio.	Presler	B1	65 nm
Pentium D 960	2x 3600 MHz	18x	200 MHz QDR	12KµOps+16/2048 KB	1.30 V	MMX SSE SSE2 SSE3 NX EM64T VT	TM1 C1E	130 W	86.6°C	376 Mio.	Presler	C1	65 nm
Pentium D 960	2x 3600 MHz	18x	200 MHz QDR	12KµOps+16/2048 KB	1.31 V	MMX SSE SSE2 SSE3 NX EM64T VT	TM1 C1E	95 W	63.4°C	376 Mio.	Presler	D0	65 nm
Pentium D 950	2x 3400 MHz	17x	200 MHz QDR	12KµOps+16/2048 KB	1.40 V	MMX SSE SSE2 SSE3 NX EM64T VT	TM1 C1E	130 V	86.6°C	376 Mio.	Presler	B1	65 nm
Pentium D 950	2x 3400 MHz	17x	200 MHz QDR	12KµOps+16/2048 KB	1.31 V	MMX SSE SSE2 SSE3 NX EM64T VT	TM1 C1E	95 W	63.4°C	376 Mio.	Presler	C1 D0	65 nm
Pentium D 945	2x 3200 MHz	16x	200 MHz QDR	12KµOps+16/2048 KB	1.32 V	MMX SSE SSE2 SSE3 NX EM64T	TM1 C1E	95 W	63.4°C	376 Mio.	Presler	C1 D0	65 nm
Pentium D 940	2x 3200 MHz	16x	200 MHz QDR	12KµOps+16/2048 KB	1.40 V	MMX SSE SSE2 SSE3 NX EM64T VT	TM1 C1E	130 V	86.6°C	376 Mio.	Presler	B1	65 nm
Pentium D 940	2x 3200 MHz	16x	200 MHz QDR	12KµOps+16/2048 KB	1.31 V	MMX SSE SSE2 SSE3 NX EM64T VT	TM1 C1E	95 W	63.4°C	376 Mio.	Presler	C1	65 nm
Pentium D 935	2x 3000 MHz	15x	200 MHz QDR	12KµOps+16/2048 KB	1.32 V	MMX SSE SSE2 SSE3 NX EM64T	TM1 C1E	95 W	63.4°C	376 Mio.	Presler	D0	65 nm
Pentium D 930	2x 3000 MHz	15x	200 MHz QDR	12KµOps+16/2048 KB	1.40 V	MMX SSE SSE2 SSE3 NX EM64T VT	TM1 C1E	95 W	63.4°C	376 Mio.	Presler	B1 C1	65 nm
Pentium D 925	2x 2800 MHz	14x	200 MHz QDR	12KµOps+16/2048 KB	1.31 V	MMX SSE SSE2 SSE3 NX EM64T	TM1 C1E	95 W	63.4°C	376 Mio.	Presler	C1 D0	65 nm
Pentium D 920	2x 2800 MHz	14x	200 MHz QDR	12KµOps+16/2048 KB	1.40 V	MMX SSE SSE2 SSE3 NX EM64T VT	TM1 C1E	95 W	63.4°C	376 Mio.	Presler	B1	65 nm
Pentium D 915	2x 2800 MHz	14x	200 MHz QDR	12KµOps+16/2048 KB	1.31 V	MMX SSE SSE2 SSE3 NX EM64T	TM1 C1E	95 W	63.4°C	376 Mio.	Presler	C1 D0	65 nm
Pentium 4 "E" 661	3600 MHz	18x	200 MHz QDR	12KµOps+16/2048 KB	1.32 V	MMX SSE SSE2 SSE3 NX EM64T HT	TM1 C1E	86 W	69.2°C	188 Mio.	Cedar Mill	B1 C1	65 nm
Pentium 4 "E" 651	3400 MHz	17x	200 MHz QDR	12KµOps+16/2048 KB	1.32 V	MMX SSE SSE2 SSE3 NX EM64T HT	TM1 C1E	86 W	69.2°C	188 Mio.	Cedar Mill	B1 C1	65 nm
Pentium 4 "E" 651	3400 MHz	17x	200 MHz QDR	12KµOps+16/2048 KB	1.32 V	MMX SSE SSE2 SSE3 NX EM64T HT	TM1 C1E	65 W	64.4°C	188 Mio.	Cedar Mill	D0	65 nm
Pentium 4 "E" 641	3200 MHz	16x	200 MHz QDR	12KµOps+16/2048 KB	1.32 V	MMX SSE SSE2 SSE3 NX EM64T HT	TM1 C1E	86 W	69.2°C	188 Mio.	Cedar Mill	B1 C1	65 nm
Pentium 4 "E" 641	3200 MHz	16x	200 MHz QDR	12KµOps+16/2048 KB	1.32 V	MMX SSE SSE2 SSE3 NX EM64T HT	TM1 C1E	65 W	64.4°C	188 Mio.	Cedar Mill	D0	65 nm
Pentium 4 "E" 631	3000 MHz	15x	200 MHz QDR	12KµOps+16/2048 KB	1.32 V	MMX SSE SSE2 SSE3 NX EM64T HT	TM1 C1E	86 W	69.2°C	188 Mio.	Cedar Mill	B1 C1	65 nm
Pentium 4 "E" 631	3000 MHz	15x	200 MHz QDR	12KµOps+16/2048 KB	1.32 V	MMX SSE SSE2 SSE3 NX EM64T HT	TM1 C1E	65 W	69.2°C	188 Mio.	Cedar Mill	D0	65 nm

CPU Queen

CPU	CPU Clock	Motherboard	Chipset	Memory	CL-RCD-RP-RAS	Score
4x Core i7 Extreme 965 HT	3333 MHz	Asus P6T Deluxe	X58	Triple DDR3-1333	9-9-9-24 CR1	30786
8x Xeon E5462	2800 MHz	Intel S5400SF	i5400	Quad DDR2-640FB	5-5-5-15	30472
4x Core 2 Extreme QX9650	3000 MHz	Gigabyte GA-EP35C-DS3R	P35	Dual DDR3-1066	8-8-8-20 CR2	21421
8x Xeon L5320	1866 MHz	Intel S5000VCL	i5000V	Dual DDR2-533FB	4-4-4-12	20452
4x Core 2 Extreme QX6700	2666 MHz	Intel D975XBX2	i975X	Dual DDR2-667	5-5-5-15	19166
4x Phenom II X4 Black 940	3000 MHz	Asus M3N78-EM	GeForce8300 Int.	Ganged Dual DDR2-800	5-5-5-18 CR2	18636
4x Xeon 5140	2333 MHz	Intel S5000VSA	i5000V	Dual DDR2-667FB	5-5-5-15	16729
8x Opteron HE 2344	1700 MHz	Tyan Thunder n3600R	nForcePro-3600	Unganged Dual DDR2-667R	5-5-5-15 CR1	16146
4x Phenom X4 9500	2200 MHz	Asus M3A	AMD770	Ganged Dual DDR2-800	5-5-5-18 CR2	13693
2x Core 2 Duo E6700	2666 MHz	Abit AB9	P965	Dual DDR2-800	5-5-5-18 CR2	11406
2x Athlon64 X2 Black 6400+	3200 MHz	MSI K9N SLI Platinum	nForce570SLI	Dual DDR2-800	4-4-4-11 CR1	11169
2x Core 2 Duo P8400	2266 MHz	MSI MegaBook PR201	GM45 Int.	Dual DDR2-667	5-5-5-15	9578
2x Pentium T3400	**2166 MHz**	**Toshiba Satellite L305**	**GL40 Int.**	**Dual DDR2-667**	**5-5-5-13**	**9145**
2x Core Duo T2500	2000 MHz	Asus N4L-VM DH	i945GT Int.	Dual DDR2-667	5-5-5-15	7793
2x Core 2 Duo T5600	1833 MHz	Asus F3000Jc Notebook	i945PM	Dual DDR2-667	5-5-5-15	7717
2x Athlon64 X2 4000+	2100 MHz	ASRock ALiveNF7G-HDready	nForce7050-630a Int.	Dual DDR2-700	5-5-5-18 CR2	7280
2x Pentium EE 955 HT	3466 MHz	Intel D955XBK	i955X	Dual DDR2-667	4-4-4-11	7098
2x Xeon	3066 MHz	Asus PCH-DL	i875P + PAT	Dual DDR333	2-2-2-5	6188
2x Opteron 240	1400 MHz	MSI K8D Master3-133 FS	AMD8100	Dual DDR400R	3-4-4-8 CR1	4863
2x PIII-S	1266 MHz	MSI Pro266TD Master-LR	ApolloPro266TD	DDR266 SDRAM	2-3-3-6 CR2	4857
P4EE HT	3733 MHz	Intel SE7230NH1LX	iE7230	Dual DDR2-667	5-5-5-15	4210
Opteron 248	2200 MHz	MSI K8T Master1-FAR	K8T800	Dual DDR266R	2-3-3-6 CR1	3851
Atom 230	1600 MHz	Intel D945GCLF	i945GC	DDR2-533 SDRAM	4-4-4-12	3779

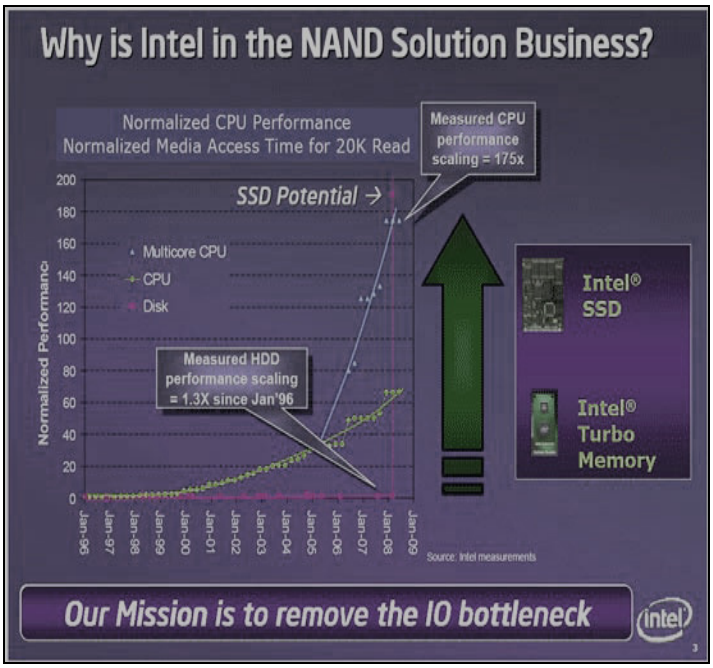
CPU AES

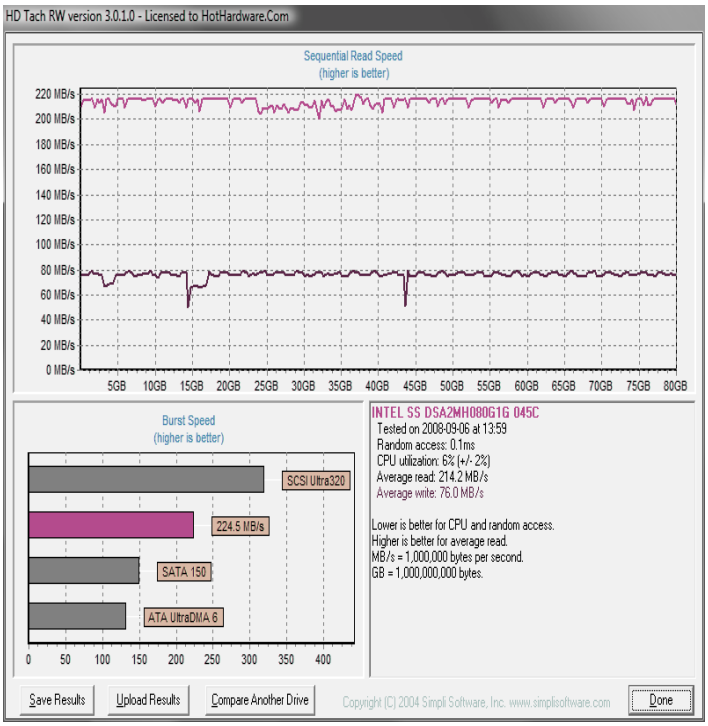
CPU	CPU Clock	Motherboard	Chipset	Memory	CL-RCD-RP-RAS	Score
8x Xeon E5462	2800 MHz	Intel S5400SF	i5400	Quad DDR2-640FB	5-5-5-15	41625
8x Xeon L5320	1866 MHz	Intel S5000VCL	i5000V	Dual DDR2-533FB	4-4-4-12	27698
4x Core i7 Extreme 965 HT	3333 MHz	Asus P6T Deluxe	X58	Triple DDR3-1333	9-9-9-24 CR1	26703
8x Opteron HE 2344	1700 MHz	Tyan Thunder n3600R	nForcePro-3600	Unganged Dual DDR2-667R	5-5-5-15 CR1	22599
4x Core 2 Extreme QX9650	3000 MHz	Gigabyte GA-EP35C-DS3R	P35	Dual DDR3-1066	8-8-8-20 CR2	22435
4x Phenom II X4 Black 940	3000 MHz	Asus M3N78-EM	GeForce8300 Int.	Ganged Dual DDR2-800	5-5-5-18 CR2	21658
4x Core 2 Extreme QX6700	2666 MHz	Intel D975XBX2	i975X	Dual DDR2-667	5-5-5-15	19896
C7	1500 MHz	VIA EPIA EN	CN700 Int.	DDR2-533 SDRAM	4-4-4-12 CR2	17358
4x Xeon 5140	2333 MHz	Intel S5000VSA	i5000V	Dual DDR2-667FB	5-5-5-15	17320
4x Phenom X4 9500	2200 MHz	Asus M3A	AMD770	Ganged Dual DDR2-800	5-5-5-18 CR2	14802
2x Core 2 Duo E6700	2666 MHz	Abit AB9	P965	Dual DDR2-800	5-5-5-18 CR2	9969
2x Pentium EE 955 HT	3466 MHz	Intel D955XBK	i955X	Dual DDR2-667	4-4-4-11	8970
2x Core 2 Duo P8400	2266 MHz	MSI MegaBook PR201	GM45 Int.	Dual DDR2-667	5-5-5-15	8443
2x Athlon64 X2 Black 6400+	3200 MHz	MSI K9N SLI Platinum	nForce570SLI	Dual DDR2-800	4-4-4-11 CR1	8339
2x Pentium T3400	**2166 MHz**	**Toshiba Satellite L305**	**GL40 Int.**	**Dual DDR2-667**	**5-5-5-13**	**7966**
2x Core Duo T2500	2000 MHz	Asus N4L-VM DH	i945GT Int.	Dual DDR2-667	5-5-5-15	7109

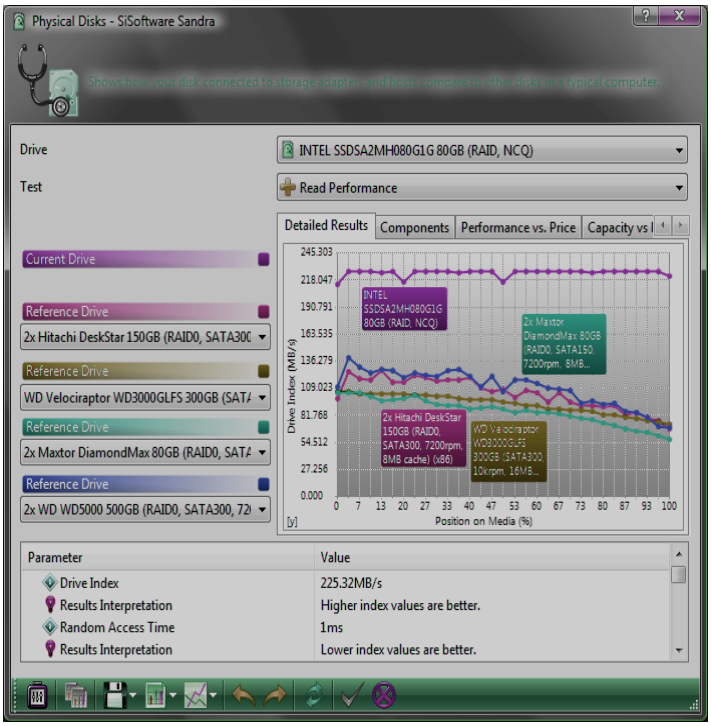
CPU ZLib

CPU	CPU Clock	Motherboard	Chipset	Memory	CL-RCD-RP-RAS	Score
8x Xeon E5462	2800 MHz	Intel S5400SF	i5400	Quad DDR2-640FB	5-5-5-15	139481 KB/s
4x Core i7 Extreme 965 HT	3333 MHz	Asus P6T Deluxe	X58	Triple DDR3-1333	9-9-9-24 CR1	112330 KB/s
8x Xeon L5320	1866 MHz	Intel S5000VCL	i5000V	Dual DDR2-533FB	4-4-4-12	95887 KB/s
8x Opteron HE 2344	1700 MHz	Tyan Thunder n3600R	nForcePro-3600	Unganged Dual DDR2-667R	5-5-5-15 CR1	88845 KB/s
4x Phenom II X4 Black 940	3000 MHz	Asus M3N78-EM	GeForce8300 Int.	Ganged Dual DDR2-800	5-5-5-18 CR2	80081 KB/s
4x Core 2 Extreme QX9650	3000 MHz	Gigabyte GA-EP35C-DS3R	P35	Dual DDR3-1066	8-8-8-20 CR2	77167 KB/s
4x Core 2 Extreme QX6700	2666 MHz	Intel D975XBX2	i975X	Dual DDR2-667	5-5-5-15	69756 KB/s
4x Xeon 5140	2333 MHz	Intel S5000VSA	i5000V	Dual DDR2-667FB	5-5-5-15	60995 KB/s
4x Phenom X4 9500	2200 MHz	Asus M3A	AMD770	Ganged Dual DDR2-800	5-5-5-18 CR2	58396 KB/s
2x Athlon64 X2 Black 6400+	3200 MHz	MSI K9N SLI Platinum	nForce570SLI	Dual DDR2-800	4-4-4-11 CR1	38059 KB/s
2x Core 2 Duo E6700	2666 MHz	Abit AB9	P965	Dual DDR2-800	5-5-5-18 CR2	35355 KB/s
2x Pentium EE 955 HT	3466 MHz	Intel D955XBK	i955X	Dual DDR2-667	4-4-4-11	29844 KB/s
2x Core 2 Duo P8400	2266 MHz	MSI MegaBook PR201	GM45 Int.	Dual DDR2-667	5-5-5-15	29223 KB/s
2x Pentium T3400	**2166 MHz**	**Toshiba Satellite L305**	**GL40 Int.**	**Dual DDR2-667**	**5-5-5-13**	**28519 KB/s**
2x Xeon	3066 MHz	Asus PCH-DL	i875P + PAT	Dual DDR333	2-2-2-5	25040 KB/s
2x Core 2 Duo T5600	1833 MHz	Asus F3000Jc Notebook	i945PM	Dual DDR2-667	5-5-5-15	23969 KB/s
2x Athlon64 X2 4000+	2100 MHz	ASRock ALiveNF7G-HDready	nForce7050-630a Int.	Dual DDR2-700	5-5-5-18 CR2	23852 KB/s

Intel claims 230MB/s read, (benchmark tools say 214MB/s read) - already on market:

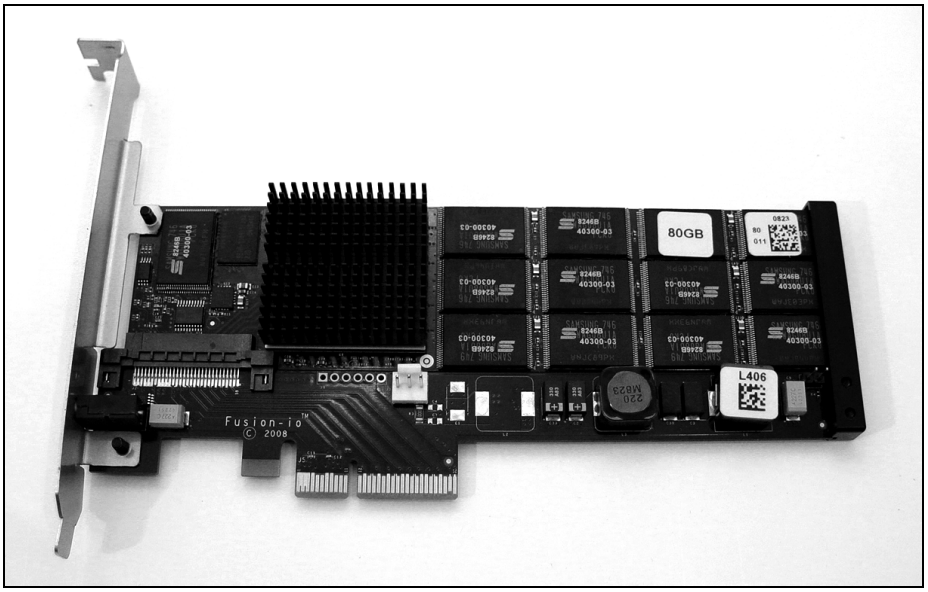


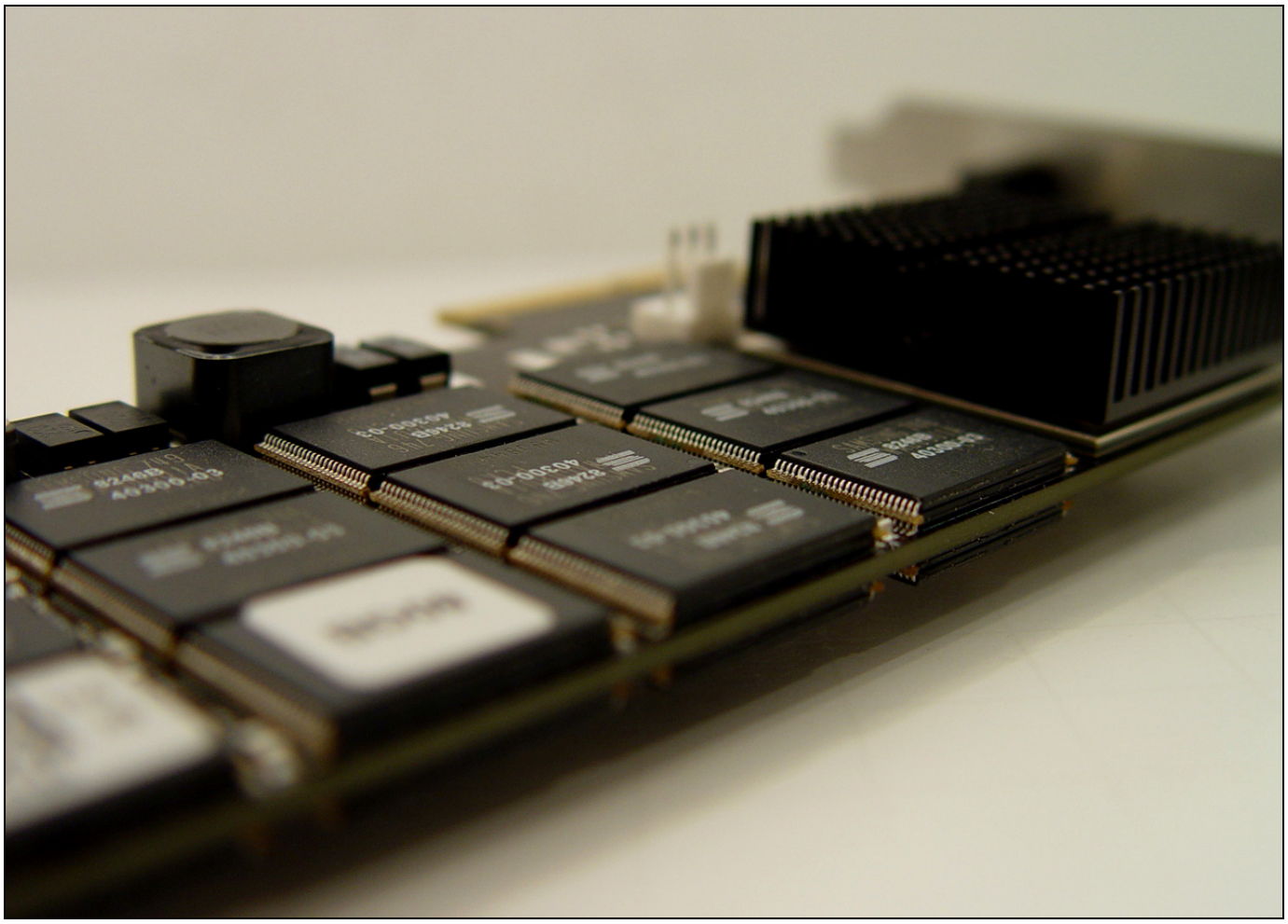




***Fusion ioDrive** SPECIFICATIONS:*

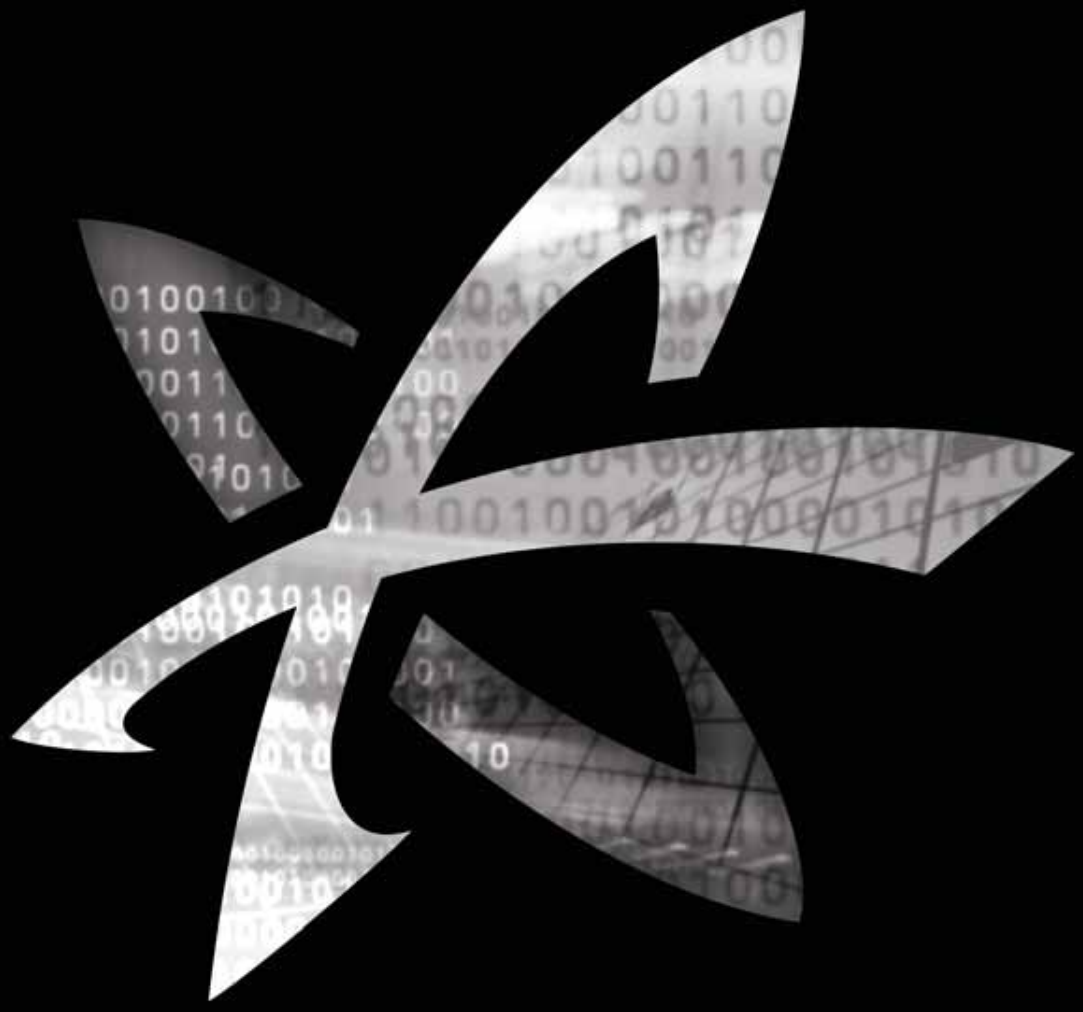
NAND Type: Single Level Cell (SLC)
Read Bandwidth: 700 MB/s (random 16K)
Access Latency: 50µs
Bus Interface: PCI-Express x4
Operating Systems: Microsoft 64-Bit Windows(64-Bit Windows XP, Vista, Server 2003 & 2008)





* * *

With the **ioDrive Duo**, it is now possible for application, database and system administrators to get previously unheard-of levels of performance, protection and capacity utilization from a single server. Performance for multiple **ioDrive Duos** scales linearly, allowing any enterprise to scale performance to **six gigabytes per-second** (Gbytes/sec) of read bandwidth and over 500,000 read IOPS by using just four **ioDrive Duos**.



Fusion-io's Solid State Storage – A New Standard for Enterprise-Class Reliability

FUSION-IO

Fusion-io's Solid State Storage – A New Standard for Enterprise-Class Reliability

Fusion-io offers solid state storage solutions based on NAND flash that provide a level of integrity and availability for mission-critical data that exceeds today's solid state storage solutions and significantly surpasses that of enterprise-class rotating magnetic storage devices.

With throughput and seek times many times faster than the fastest disk arrays, it is little wonder that enterprise data centers have been keen to include NAND flash as part of their server infrastructure. The primary reason NAND flash has not been widely adopted in the computer industry is its reputation for unreliability. There is a long-standing view that NAND flash storage works well for non-mission-critical applications, such as media storage devices (where the occasional bit error generally translates into a slight audio hiss or a stray errant pixel in a video), but cannot be relied upon for applications where a bit error could crash an operating system or compromise the integrity of critical data.

System architects face a number of storage-related challenges and NAND flash technology presents its own set of unique problems. But Fusion-io has developed patent-pending techniques to create NAND flash-based storage with reliability equal to or exceeding that of disk-based storage. This paper describes several inventions and advancements Fusion-io has introduced to ensure data is not corrupted or lost. Additionally, this paper discusses the probability of catastrophic storage device failure and how Fusion-io's architecture ensures predicable, controlled management of early device failure, long-term device attrition and data changes due to external and data transport interference.

NAND Flash

Flash memory chips are a non-volatile storage medium (i.e., they can retain their information even in the absence of power). The most common types of flash chips are silicon-based NOR and NAND, named after the types of logic gates used in their design. NAND flash, introduced in 1989, has become the most commonly used type of flash chip, due to its quicker write speed. Flash memory continues to grow in popularity as its price steadily declines, its storage capacity increases, and its physical size continues to decrease.

In Fusion-io's storage devices, NAND flash chips are stacked several at a time (to increase density), operated in parallel (to increase throughput) and mounted on a printed circuit board (PCB) that plugs into a PCI-Express (PCIe) slot on the server or in the CPU. The flash media is integrated with the controller onto a single PCI-Express card.

NAND flash, as a storage medium, offers a number of benefits in comparison to rotating magnetic storage devices (aka HDD, Hard Disk Drives). NAND flash has no moving parts and is therefore significantly less prone to shock or movement disturbance. It is a high speed solution in both latency and throughput. Temperature and humidity resistance mean that it can operate in a number of different environments. Finally, NAND flash consumes significantly less power than rotating magnetic storage devices, particularly when you take into account secondary power requirements for device cooling.

However, NAND flash does introduce a number of potential failure points including:

- Media – Media failures can occur on the NAND flash chips themselves.
- Transport – Transport errors can occur anywhere along the path carrying data from the CPU through to the NAND flash chips.
- Management – There is a small chance that management problems can occur within the logic of the device itself. The code that controls the operation can contain technical problems that can result in data failures.
- External – External problems can affect any part of the process.
- Device Failure – Catastrophic hardware failure can also occur. This includes the possibility of internal short circuits and open circuits within the memory array itself.

Protecting the Data

Implementing a variety of design and architectural strategies for protecting data integrity, Fusion-io's NAND flash devices greatly exceed the reliability of rotating magnetic media storage devices, while providing performance that is orders of magnitude better. Fusion-io protects your data at every step, ensuring that nothing is lost or corrupted in transit or on the media.

Data Integrity

Data integrity means having a high degree of confidence that what you put into a storage system is exactly what you get out when you request that data and it is the most important function of a storage system. While being moved from a computer's RAM or CPU to the Fusion-io device, several proven industry-standard approaches are used to ensure data integrity. The CPU, chipset, and RAM use SECDED (Single Error Correct Double Error Detect) or chipkill (method for on-the-fly replacement of a failed chip) to ensure accuracy. Once data is written to the storage medium, it is again checked for accuracy.

When data is read from the storage medium, error correction techniques are again employed to ensure that the data being retrieved is correct. The device can correct a substantial portion of the data being read. NAND's reputation for unreliability is based on studies that show potential data loss without utilizing error correction – or less correction than that employed by the Fusion-io device. Using the methods described here, Fusion-io devices can produce results that exceed target error probability by about four times. Fusion-io's devices also use a patent-pending approach when writing data, which allows the data's path to be reconstructed from information generated during the write process.

Data Availability

Data availability means having a high degree of confidence that data stored will not be lost, either while in transition to the storage device or after it has been written to the media.

Fusion-io employs a wide variety of techniques to overcome some of the common problems associated with data availability in general, and also addresses some that are particular to NAND flash as a storage medium. Generally speaking, NAND flash is substantially more reliable than rotating magnetic media. It eliminates the chance of mechanical failure

(the failure associated with moving parts). There is, however, a chance of bad chips and chip wear-out. Fusion-io mitigates this risk using a variety of approaches.

Fusion-io's redundant, patent-pending approach to writing data allows data to be rebuilt at a very high rate of speed, ensuring rapid data availability. Data is also regularly moved and checked for accuracy to ensure that it does not deteriorate on the flash chip. This also consolidates good data and reallocates space on the drive to ensure greater data availability. This system also spreads data evenly across the device, ensuring uniform wear across all chips.

Additionally, Fusion-io uses multiple error correction code (ECC) techniques to identify and correct faulty data. Using ECCs, the device controller can correct up to 11 missing or incorrect bits out of every 240 bytes. One of the biggest benefits of ECC routines is that it they allow the device to predict the likelihood of failure on individual chips. When a particular area of a chip has passed a set unreliability threshold, its data can be moved and that are will be taken out of service. The controller continues to identify and remove bad blocks, regions of chips or even entire chips so that ordinary wear-out does not cause catastrophic failure rather a very predictable wear-out.

Device Longevity

The majority of this paper has concentrated on NAND flash in an enterprise-class storage device, and how to leverage its strengths while overcoming its weaknesses. NAND flash, however, is only part of a Fusion-io's storage device. The flash chips reside on a PCIe adapter card that has a number of other parts as well, all of which are susceptible to failure. The life of a NAND flash storage device can be estimated by examining the failure rate of its component parts. Wear-out is generally a function of having lost enough storage cells that both capacity and reliability drop below acceptable thresholds. This can be assessed by evaluating and keeping a record of the amount of errors detected at each physical location.

NAND flash wears out at a predictable rate as described by the formulas below. Effective use of wear-leveling strategies employed by Fusion-io can significantly improve the life expectancy of its drives. Please note that the formulas are applied to both MLC and SLC NAND-based non-volatile memory technologies. Single-Level Cell (SLC) NAND and Multi-Level Cell (MLC) NAND offer capabilities that serve two very different types of applications – respectively, those requiring high performance at an attractive cost-per-bit and those seeking even higher performance over time, that are less cost-sensitive:

Average-lifetime = lifetime / read-write- ratio

TYPE / WRITE DUTY	AVERAGE ESTIMATED LIFETIME FORMULA
SLC flash @ 40% write duty	25 calendar years
MLC flash @ 20% write duty	10 calendar years
MLC flash @ 40% write duty	5 calendar years

Average estimated lifetime based on Fusion-io lab testing

The read/write ratio is difficult to predict, and will vary considerably from environment to environment. As a point of reference, the International Disk-drive Equipment and Materials Association (IDEMA), an industry trade group that publishes storage device standards, recommends a read/write ratio of 60%/40% for its server-class device reliability testing (IDEMA Standards, Document R3-98).

Flashback Protection

Enterprises have long sought to take advantage of the speed, size, low-power and high-performance of NAND Flash because of its potential to change the way they manage large amounts of active data. The primary objection to NAND flash has been the reliability of the medium. Fusion-io has eliminated this barrier by inventing a revolutionary self-healing technology, known as Flashback Protection, in our controllers that instantaneously restores, corrects and resurrects lost data in the flash-based storage sub-system. Flashback Protection is accomplished by collectively using advanced bit error correction, proactive data integrity monitoring of stored data and the recent addition of a dedicated chip to repair failed devices.

Fusion-io is the first and only company to bring RAID-class redundancy and reliability using Flashback Protection down to the card level. The Flashback Protection system allows users to diagnose and correct system errors. Fusion-io integrates dedicated NAND flash chips, which offer information that enables the detection of single bit errors. This technique eliminates data loss due to chip failures and extends the usable lifetime of the NAND flash-based storage device. The NAND flash chips on Fusion-io's products contain an innovative storage architecture that enable it to deliver the performance, and now the reliability, of a storage area network (SAN) at a fraction of the power, size and cost of traditional disk arrays.

Controlled Predictable Usage Versus Catastrophic Failure

Among the greatest reliability benefits of the Fusion-io storage device is its ability to:

- Restore and Protect data
- Monitor and predict media wear-out
- Correct bad data as necessary
- Take blocks out of service when their failure rate becomes unacceptable
- Replace bad chips on-the-fly
- Move the data to a known good location (and update corresponding mapping information)

Data stored on the Fusion-io medium is double protected using both ECCs and parity data on the redundant chip. The net effect is that wear-out of the device, instead of being catastrophic, is predictable and incremental.

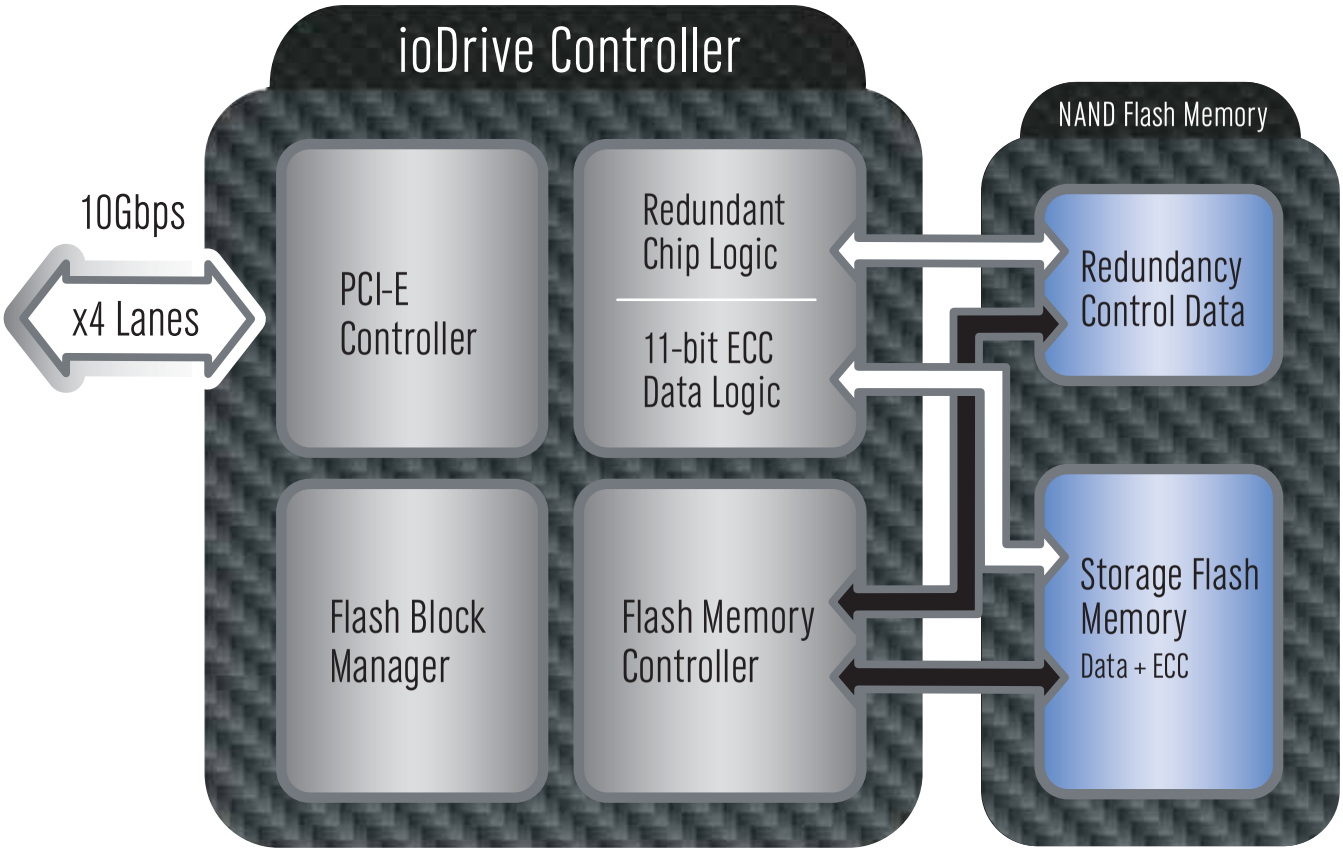
A Fusion-io device provides advanced warning prior to wear-out. Fusion-io supports today's monitoring management functions to measure and report on the device's status and usable life. In almost all cases, device upgrade is a smooth and predictable process, rather than an emergency situation.

Fusion-io protects your data at every stage of its path from your applications to the NAND flash storage medium, ensuring that nothing is lost or corrupted along the way or while the data is being stored. Data is checked multiple times, using several error detection methods. Once it reaches the storage medium, it is stored with robust error correction encoding that lets the flash device not only identify but correct bit errors. Fusion-io's data integrity design target is a 1 in 10^{30} probability of undetected bad data and a 1 in 10^{20} probability of uncorrectable data, as compared to a 1 in 10^{16} probability of undetected or uncorrectable errors for rotating magnetic storage devices.

Conclusion

Now with Fusion-io's comprehensive approach to data integrity, it is safe to exploit the exponential performance gains and many other benefits offered by NAND flash storage. The storage architecture pioneered by Fusion-io ensures predictable, controlled mitigation of early device failure, long-term device attrition and data changes due to external and data transport interference—issues that have up to now limited the adoption of NAND flash-based storage at the enterprise level. Fusion-io's NAND flash devices exceed the reliability of rotating magnetic media storage devices while providing an order of magnitude performance improvement.

Flash Back Block Diagram



Robert Brumfield
Fusion Public Relations
212.651.4215
robert.brumfield@fusionpr.com

Fusion-io Announces the ioDrive Duo—The World's Fastest and Most Innovative SSD

PCI Express, server-based solid-state storage offering sets a new standard for enterprise application-centric storage, with up to 640 gigabytes of capacity and 1.5 gigabytes per-second of sustained throughput

SALT LAKE CITY - March 11, 2009 - Fusion-io, the leader in solid-state architecture and high-performance I/O solutions, today announced the ioDrive Duo, which doubles the slot capacity of Fusion-io's successful PCI Express-based ioDrive storage solution. The new ioDrive Duo is the market's fastest and most innovative server-based solid-state storage solution.

With the ioDrive Duo, it is now possible for application, database and system administrators to get previously unheard-of levels of performance, protection and capacity utilization from a single server. Performance for multiple ioDrive Duos scales linearly, allowing any enterprise to scale performance to six gigabytes per-second (Gbytes/sec) of read bandwidth and over 500,000 read IOPS by using just four ioDrive Duos.

"Many database and system administrators are finding that SANs are too expensive and don't meet performance, protection and capacity utilization expectations," said David Flynn, CTO of Fusion-io. "This is why more and more application vendors are moving toward application-centric solid-state storage. The ioDrive Duo offers the enterprise the advantages of application-centric storage without application-specific programming."

ioDrive Duo Product Details

The following specifications describe the physical and performance characteristics of the ioDrive Duo.

Performance

Based on PCI Express x8 or PCI Express 2.0 x4 standards, which can sustain up to 20 gigabits per-second of raw throughput, the ioDrive Duo has more than enough bandwidth to obtain industry-leading performance from a single card. The ioDrive Duo can easily sustain 1.5 Gbytes/sec of read bandwidth and nearly 200,000 read IOPS. Its performance metrics are as follows:

- Sustained read bandwidth: 1500 MB/sec (32k packet size)
- Sustained write bandwidth: 1400 MB/sec (32k packet size)
- Read IOPS: 186,000 (4k packet size)
- Write IOPS: 167,000 (4k packet size)
- Latency < 50 µsec

Reliability

The ioDrive Duo offers unmatched solid-state protection for data integrity and reliability with triple redundancy for a single storage component.

- Multi-bit error detection and correction
- Patent-pending Flashback protection, offering chip-level N+1 redundancy and on-board self-healing so that no servicing is required
- Optional RAID-1 mirroring between two ioMemory modules on the same ioDrive Duo, offering complete redundancy on a single PCIe card

Capacity

The ioDrive Duo comes in the following capacities:

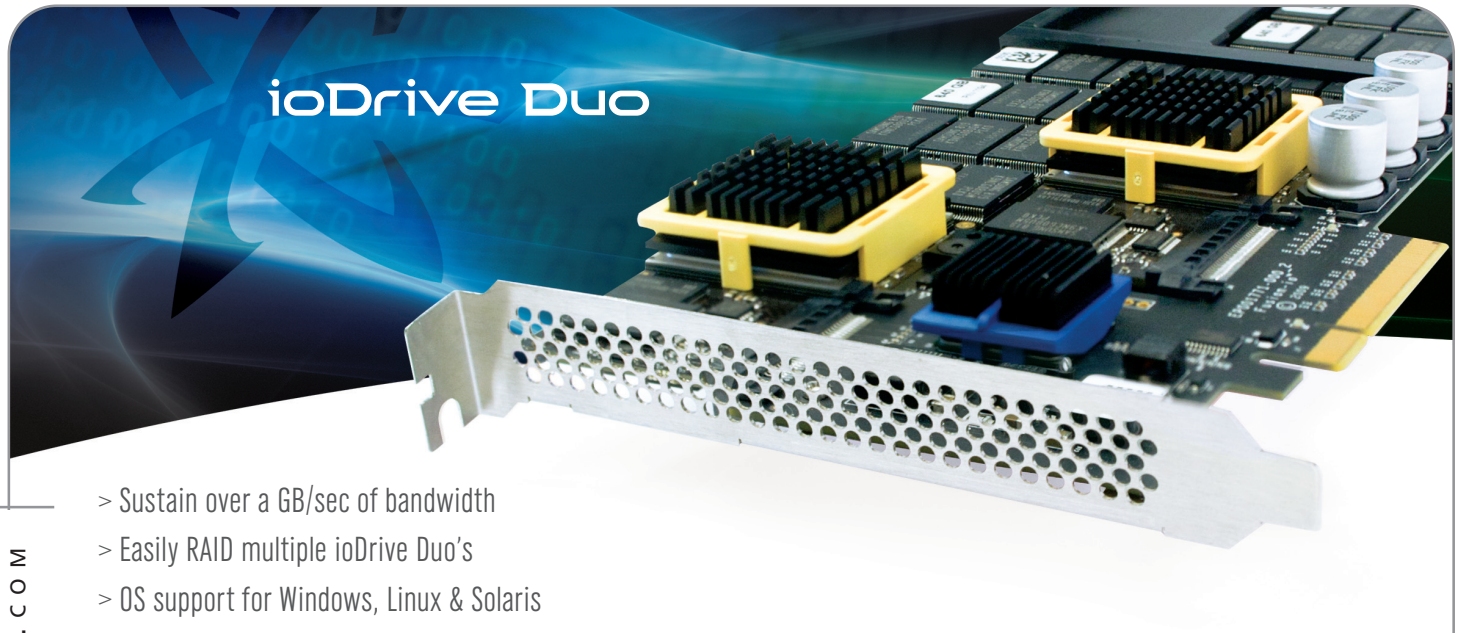
- 160 Gbytes
- 320 Gbytes
- 640 Gbytes
- 1.28 TB (second half of 2009)

The ioDrive Duo will be available in April 2009. To find out more about how this and Fusion-io's other enterprise solid-state storage products can benefit your organization, please visit www.fusionio.com.

About Fusion-io

Fusion-io is a leading provider of enterprise solid-state technology and high-performance I/O solutions. The company's solid-state storage technology closes the gap between processing power and storage needs delivering breakthrough performance at a fraction of the cost of traditional disk-based storage systems. The result is a world of possibilities for performance-starved applications.

ioDrive Duo



> Sustain over a GB/sec of bandwidth
> Easily RAID multiple ioDrive Duo's
> OS support for Windows, Linux & Solaris

ioDrive Duo Capacity	160GB	320GB	640GB
NAND Type	Single Level Cell (SLC)	Single Level Cell (SLC)	Multi Level Cell (MLC)
Write Bandwidth	1.1 GB/s (32k packet size)	1.4 GB/s (32k packet size)	1.0 GB/s (32k packet size)
Read Bandwidth	1.5 GB/s (32k packet size)	1.5 GB/s (32k packet size)	1.4 GB/s (32k packet size)
IOPS*	200,832 reads (4k packet size) 132,118 writes 4k packet size)	185,022 reads (4k packet size) 167,784 writes (4k packet size)	126,601 reads (4k packet size) 180,530 writes (4k packet size)
Access Latency	50µs Read	50µs Read	80µs Read
Bus Interface	PCI-Express x8 and PCI Express 2.0 x4	PCI-Express x8 and PCI Express 2.0 x4	PCI-Express x8 and PCI Express 2.0 x4
Weight	Less than 10 ounces	Less than 10 ounces	Less than 10 ounces
Operating Systems	Microsoft Windows**, Open Solaris 10 Solaris 10, RHEL 4 & 5; SLES 9 & 10	Microsoft Windows**, Open Solaris 10 Solaris 10, RHEL 4 & 5; SLES 9 & 10	Microsoft Windows**, Open Solaris 10 Solaris 10, RHEL 4 & 5; SLES 9 & 10
Wear Leveling and Sophisticated ECC (@ 5-TB write-erase / day)	24yrs	48yrs	16yrs

** Performance achieved using multiprocessor enterprise server ** 64-Bit Windows XP, Vista, Server 2003 & 2008*

STANDARDS

Form Factor	Full height, 3/4 length PCI Express 2.0
Connectivity	PCI Express electromechanical spec 2.0
Power	PCI Express power spec 2.0

AGENCY

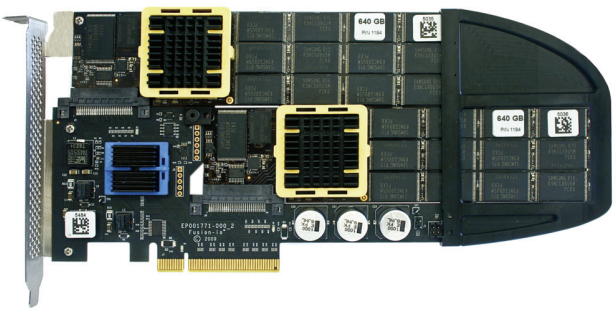
US / Canada	FCC Part 15, ICES-003, Class A
Europe	2004/108/EC EMC Directive CE Mark;
Japan	VCCI, Class A
Taiwan	BSMI, Class A
New Zealand /Australia	AS/NZS 3548 Class A
RoHS	R5 (Directive 2002/95/EC)

ENVIRONMENTAL SPECIFICATIONS

		Min	Max
Temperature (°C)*	Operational	0	55
	Non-operational	- 40	70
Air Flow (LFM)		300	
Humidity (%)	Non-condensing	5	95
Altitude (ft)	Operational		10,000
	Non-operational		30,000

** Temperature derated 1 C per 1000 ft elevation above sea level*

100% Assembled in the U.S.A.



FUSION-IO

> Less than 50 µs latency

> Easily RAID multiple ioDrives together

> Managed like simple block storage

WWW.FUSIONIO.COM

ioDrive Capacity	80GB	160GB	320GB
NAND Type	Single Level Cell (SLC)	Single Level Cell (SLC)	Multi Level Cell (MLC)
Write Bandwidth	550 MB/s (random 16K)	600 MB/s (random 16K)	500 MB/s (random 8K)
Read Bandwidth	700 MB/s (random 16K)	700 MB/s (random 16K)	700 MB/s (random 32K)
IOPS*	102,000 (random 4k reads) 91,000 (random 4k writes) 88,000 (70/30 random 4k mix)	104,400 (random 4k reads) 103,925 (random 4k writes) 95,000 (70/30 random 4k mix)	60,000 (random 4k reads) 79,000 (random 4k writes) 65,000 (70/30 random 4k mix)
Access Latency	50µs Read	50µs Read	80µs Read
Bus Interface	PCI-Express x4	PCI-Express x4	PCI-Express x4
Weight	Less than 2 ounces	Less than 2 ounces	Less than 2 ounces
Operating Systems	RHEL 4 & 5; SLES 9 & 10 Microsoft 64-Bit Windows**	RHEL 4 & 5; SLES 9 & 10 Microsoft 64-Bit Windows**	RHEL 4 & 5; SLES 9 & 10 Microsoft 64-Bit Windows**
Wear Leveling and Sophisticated ECC (@ 5-TB write-erase / day)	24yrs	48yrs	16yrs

** Performance data provided by Medusa Labs. ** 64-Bit Windows XP, Vista, Server 2003 & 2008*

STANDARDS

Form Factor	Low profile PCI Express x4 slot (spec 1.1)
Connectivity	PCI Express x4 (electromechanical spec 1.1)
Power	PCI Express x4 (power spec 1.1)

ENVIRONMENTAL SPECIFICATIONS

		Min	Max
Temperature (°C)*	Operational	0	55
	Non-operational	- 40	70
Air Flow (LFM)		300	
Humidity (%)	Non-condensing	5	95
Altitude (ft)	Operational		10,000
	Non-operational		30,000

** Temperature derated 1 C per 1000 ft elevation above sea level*

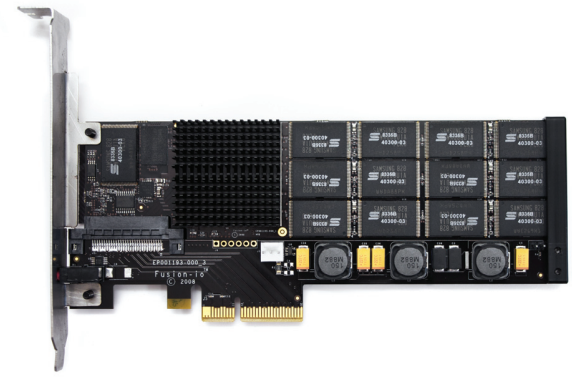
SAFETY

US / Canada	UL60950, CSA C22.2 No.60950-1-03
Europe	TUV EN60950-1:2001; 3N50825-1:

100% Assembled in the U.S.A.

AGENCY

US / Canada	FCC Part 15, ICES-003, Class A
Europe	2004/108/EC EMC Directive CE Mark;
Japan	VCCI, Class A
Taiwan	BSMI, Class A
New Zealand /Australia	AS/NZS 3548 Class A
RoHS	R5 (Directive 2002/95/EC)



FUSION-IO