MAKEFILE: OBJS= blocksort.o \ huffman.o \ crctable.o \ randtable.o \ compress.o \ decompress.o \ bzlib.o\ libbz2.a \ bzip2.o \ bzip2recover.o\ test  
DISTNAME=bzip2-1.0.6

MANUAL\_SRCS  
  
///////////////////////////////////////////////////////////////////////////////////////////////////////////  
Mk251 : Spew out a long sequence of the byte 251

Randtable: - #include "bzlib\_private.h"

Contiene array di 512 interi già riempito

Crctable: - bzlib\_private.h

Contiene array di 256 unsignedInt riempito in valori esadecimali

Huffman: : - #include "bzlib\_private.h"

Definisce 3 macro : peso = dalla terza all’ottava cifra (zz0)

Profondità= prime 2 cifre (zz1)

Massimo = (zz2, zz3)

Definisce ADDWEIGHTS, UPHEAP, DOWNHEAP

BZ2\_hbMakeCodeLengths: loop che si assicura la lunghezza dell’ Huffman-code-word sia nei limiti

BZ2\_hbAssignCodes: assegna codice corrispondenti a certe lunghezze?

BZ2\_hbCreateDecodeTables: crea tabelle interi32

Dlltest: -#include <stdio.h>

-#include <stdlib.h>

-#include "bzlib.h"

-#include <io.h>

-#define BZ2\_LIBNAME "libbz2-1.0.2.DLL"

-#include <windows.h>

Definisce 2 variabili globali: BZ2DLLLoaded(int), BZ2DLLhLib(HINSTANCE[finestra esecuzione])

BZ2DLLLoadLibrary: ottiene informazioni sul processo in esecuzione

BZ2DLLFreeLibrary: svuota libreria??

Main: /////////////

spewG: -#include <stdio.h>

-#include <stdlib.h>

#define 3 const: \_FILE\_OFFSET\_BITS 64 ;MEGABYTES 5000; N\_BUF 1000000(dimensione buffer private)

MAIN : inizializza random di seed 1 e un buffer.

Unzcrash:- #include <stdio.h> //# a << b: shift a di b passi

-#include <assert.h>

-#include "bzlib.h"

Bzerrorstrings = array di stringhe di errori

flip\_bit(bit) = aggiorna zbuf[bit/8] con il valore di oX1 \* bit%8

MAIN =

//////////////////////

Blocksort: -#include "bzlib\_private.h"

-fallbackSimpleSort( UInt32\* fmap, UInt32\* eclass, int32 lo, Int32 hi ):

Controlla lo != hi, sorting di fmap&eclass decrescente?

-define: fswap, fvswap, fmin, fpush, fpop, FALLBACK\_QSORT\_SMALL\_THRESH, FALLBACK\_QSORT\_STACK\_SIZE

-fallbackQSort3 ( UInt32\* fmap , eclass, Int32 loSt, hiSt ): fallbackSimpleSort, /////

-undef : fmin, fpush, fpop, FALLBACK\_QSORT\_SMALL\_THRESH, FALLBACK\_QSORT\_STACK\_SIZE

-#define funct\_\_BH(zz): n functions in binary

-fallbackSort(UInt32\* fmap, eclass, bhtab, Int32 nblock, verb ): dichiara 2 array, 8 int e un puntatore a array eclass di tipo Uchar.

“exponential radix sort” delle variabili in ingresso, ricompone UInt32\* eclass usando eclass8 e #undefine tutte funct\_\_BH(zz).

- mainGtU ( UInt32 i1, i2, UChar\* block, UInt16\* quadrant, nblock,

Int32\* budget ): controlla che i1 != i2, return (block[i1] > block[i2]) if block[i1]!= block[i2] per (i1++,i2++) 20 volte; if (i1 >= nblock) i1 -= nblock if (i2 >= nblock) i2 -= nblock; return false;

- Incerpi-Sedgewick : static Int32 incs[14], static void mainSimpleSort{};

- #define mswap, mvswap (swap array rispettivamente: tutto, da 0 a zzn passato in mvswap)

- mmed3(char a,b,c): return il maggiore

- #define mmin, mpush, mpop, mnextsize, mnextswap, MAIN\_QSORT\_SMALL\_THRESH, MAIN\_QSORT\_DEPTH\_THRESH, MAIN\_QSORT\_STACK\_SIZE

- mainQSort3 ( UInt32\* ptr, UChar\* block,UInt16\* quadrant,Int32 nblock, loSt, hiSt, dSt,\* budget):

///////////

-#undefine everything from here

- mainSort ( UInt32\* ptr, UChar\* block,UInt16\* quadrant, UInt32\* ftab,Int32 nblock, verb, \*budget ): dichiara variabili di support e set up the 2-byte frequency table (operazioni di shift in binario);

bucket sorting, Calculate the running order, from smallest to larges big bucket;

mainQSort3 per ordinare I contenitori(3 step: IRRILEVANTI PER QUELLO CHE DEVO FARE).

- BZ2\_blockSort ( EState\* s):

////////////////////////////

Bzip2recover: -#include <stdio.h>

-#include <errno.h>

-#include <stdlib.h>

-#include <string.h>

-definisce alcune costanti e rinomina alcuni tipi

-definisce 4 tipi di errori

-definisce struct(file+attributi) BitStream e funzioni di I/O

-bsPutBit: aggiunge intero in bit

-bsGetBit: Returns 0 or 1, or 2 to indicate EOF.

-bsPutUChar, bsPutUInt32: aggiunge intero/char in bit

-define constant windows/unix mode

-MAIN:

/////////////////////////////////////////

Bzlib.h: -#define errors code

Define struct bz\_stream

#import system libs

Ridefinisce funzioni di BZ\_EXTERN: (callable everywehere), return un bz\_stream

Molte funzioni BZ\_API (extern: inizializzate da qualche altra parte)[Bziplib.c]

-BZFILE ridefinito per puntatori a funzioni che tornano void

ALTRE FUNZIONI BZ\_API ridefinite in “Bziplib.c”

Bzlib\_private.h : -#include <stdlib.h> //failures, exceptions and sizevar

-#include <stdio.h> //input-output

-#include <ctype.h> //test and map characters

-#include <string.h>

-#include "bzlib.h"

Define some errors, some constants, rename some chartype using typedef

Struct Estate contiene variabili e funzioni per compressione file

Struct DState contiene variabili:” for undoing the Burrows-Wheeler transform” and “for doing the

final run- length decoding ” |[origPtr, tPos, k0, vari array] [state\_out\_ch, state\_out\_len,

blockRandomised, BZ\_RAND\_DECLS] . + dati decompressione

-define Macros for decompression/[BZ\_GET\_FAST, BZ\_GET\_FAST\_C, SET\_LL4, GET\_LL4, SET\_LL, GET\_LL, BZ\_GET\_SMALL]

-define externs for decompression/[BZ2\_indexIntoF, BZ2\_decompress, BZ2\_hbCreateDecodeTables]

-define “error NULL BZ\_NO\_STDIO” ???

Bzlib: - #include <stdio.h> <stdlib.h> <string.h> <signal.h><math.h><errno.h><ctype.h><fcntl.h> <sys/types.h><utime.h><unistd.h><sys/stat.h> <sys/times.h><io.h><fcntl.h>

-#define some costants

-

Compress:- #include "bzlib\_private.h"

-BZ2\_bsInitWrite ( EState\* s ) : setta live and buff di s a 0

-sFinishWrite ( EState\* s ): while stream active riempi buffer(come #define bsNEEDW(nz))

-bsW(estate \*s, int n,v): inline, bsNEEDW, s.buff in or con(v shift(32-s.live-n)), s.live += n;

-bsPutUInt32 ( EState\* s, UInt32 u ): for( i = 0,i<=24; i+8) {bsw(s,8,u+i)}

-bsPutUChar( EState\* s, UChar c ): bsw(s,8,c);/\*sembra che bsw sembra per unserire efficientemente dati nel buffer\*/

-makeMaps\_e(s):attiva bytes blocks(copia inUse in unseqToSeq e aggiorna contatori di quanti elementi ha copiato).

-generateMTFValues(s): makeMaps\_e, mette a 0 I valori di un MTF array, regola i valori delle altre MTF variables di s; setta array uint16 mtfv a 0 o 1 a seconda di?

-sendMTFValues(EState\* s ): gestisce coding MTF values, Decide how many coding tables to use, Generate an initial set of coding tables(MTF arays), 4 cilcli per migliorare tabelle[322-452] + assert(tutto corretto); assegna selettore(MTF arrays) al rispettivo valore su array di supporto(pos),assegna lunghezza minima e massima tabelle, Transmit the mapping table, stessa cosa con selectorMtf, len e selector.

-BZ2\_compressBlock(EState\* s, Bool is\_last\_block ): se nblock>0 combina e ordina CRC,

Cerca un blocco specific, bsw, estrai MTF value, create stream

decompress: -#include "bzlib\_private.h"

-makeMaps\_d(DState\* s): riempie variabile “inUse” di s con valori da 0 a 256 per ogni spazio di essa

-#define RETURN(rrr), rrr=goto save\_state\_and\_return

-#define GET\_BITS:

-#define GET\_UCHAR -#define GET\_BIT: GET\_BITS ma con la dimensione della variabile

-#define GET\_MTF\_VAL:

- BZ2\_decompress:

/////////////////////////////////////////////////////////////////////////////////////////////////////////////////

**burrows-wheeler transform:**

DState usata in: bzlib, bzlib\_private.h, decompress

Bzlib: pointer a DState in funct BZ\_API //line 492 : controlli, ritorna costante BZ\_OK

Function unRLE\_obuf\_to\_output\_FAST //538:

Function unRLE\_obuf\_to\_output\_SMALL// 706:

??? //808

Bzlib\_private: declared struct DState // 438

globalFunction BZ2\_decompress ( DState\* ); // return type: Int32

decompress: funct makeMaps\_d ( DState\* s ) //27: usa nbitused come array (prende binario, se 0 ->

non fa nulla ,se 1->aggiorna contatore e inserisce valore contatore in array

“seqToUnseq” della DState

Parametro passato per funzione BZ2\_decompress(DState s)//106: tutta la decompressione

Run-length encoding input: Estate {state\_in\_ch; state\_in\_len; BZ\_RAND\_DECLS}//bzlib\_private.h

Final run-length decoding: DState{state\_out\_ch; state\_out\_len; blockRandomised; BZ\_RAND\_DECLS;}//bzlib\_private.h

bsWr: bzip-recovery

bsw: compress[73/96 – 505/650]

/\* for undoing the Burrows-Wheeler transform \*/ :bzip-private.h

Int32 origPtr; k0; unzftab[256]; nblock\_used; cftab[257]; cftabCopy[257];UInt32 tPos;

**Run-length:** bzlib\_private.h[118-224-355], bzlib.c[700/900], compress.c[178/220]

Used in encoding: add\_pair\_to\_block[216 bzlib], ADD\_CHAR\_TO\_BLOCK[260 bzlib], bzlib [113,141], bz\_private[141-153]

Used in decoding(DState): unRLE\_obuf\_to\_output\_FAST[536 bzlib], unRLE\_obuf\_to\_output\_SMALL [706 bzlib](used in decompress)