```
#define BLK_SIZE 512

#define ENTRIES (BLK_SIZE/4)

#define DIRECT_BLOCKS 12
#define INDIRECT1_BLOCKS ENTRIES
#define INDIRECT2_BLOCKS (ENTRIES*ENTRIES)
#define INDIRECT3_BLOCKS (ENTRIES*ENTRIES*ENTRIES)

#define INDIRECT1_INDEX DIRECT_BLOCKS
#define INDIRECT2_INDEX (INDIRECT1_INDEX+1)
#define INDIRECT3_INDEX (INDIRECT2_INDEX+1)


struct inode {
   .
   .
   .
   int disk_num;
   .
   .
   .
   int map[INDIRECT3_INDEX+1];
   .
   .
   .
};

static unsigned int Level1(struct  inode *ip, unsigned int b,
                                              unsigned int fb) {
      if (b == 0)
         return (0);

      char buf[BLK_SIZE];
      unsigned *buf_i = (int *)buf;
      dread(ip->disk_num, b, buf, BLK_SIZE);
      return(buf_i[fb]);

}
unsigned int fb2db(struct inode *ip, unsigned int fb) {
   if (fb < DIRECT_BLOCKS)
      return (ip->map[fb]);

   fb -= DIRECT_BLOCKS;
   if (fb < INDIRECT1_BLOCKS)
      return(Level1(ip, ip->map[INDIRECT1_INDEX]), fb);

   fb -= INDIRECT1_BLOCKS;
   if (fb < INDIRECT2_BLOCKS) {
      unsigned int b;
      if ((b == Level1(ip, ip->map[INDIRECT2_INDEX],
                                  fb / INDIRECT1_BLOCKS)) == 0)
         return (0);

         return(Level1(ip, b, fb %  INDIRECT1_BLOCKS));
   }

   fb -= INDIRECT2_BLOCKS;
   if (fb < INDIRECT3_BLOCKS) {
      unsigned int b;
      if ((b == Level1(ip, ip->map[INDIRECT3_INDEX],
                               fb / INDIRECT2_BLOCKS)) == 0)
         return (0);

      if ((b == Level1(ip, b,
                  (fb % INDIRECT2_BLOCKS) / INDIRECT1_BLOCKS) == 0)
         return (0);

      return(Level1(ip, b,
                   (fb % INDIRECT2_BLOCKS) % INDIRECT1_BLOCKS);

   }
   return (0);
```