NAME

a.out − assembler and link editor output (a16.out when cross-support)

SYNOPSIS

#include <a.out.h>

DESCRIPTION

A.out is the output file of the assembler *as*(1) and the link editor *ld*(1). Both programs make *a.out* executable if there were no errors and no unresolved external references.

The terms ‘‘address’’ and ‘‘location’’ refer to addresses within the address space. The term ‘‘position’’ refers to a point in the *a.out* file. Layout information for the NS16032 is:

```
/*
 * a.out file layout information:
 *
 * "+" indicates an addition to the Berkeley 4.1 format.
 *
 * Note:
 *      This file contains only an incomplete description of
 *      information present in an a.out file; it omits symbol
 *      table definitions.
 *
 *      Complete symbol table definitions reside in <stab.h> and
 *      simplified definitions in <nlist.h>.  These sets of
 *      definitions are not fully compatible with each other; symbol
 *      table information as defined in <nlist.h> does not correspond
 *      directly to a.out file contents, but instead is obtained by
 *      translating such information with nlist(3).
 */

/*
 * Header prepended to each a.out file.
 */
struct exec {
         long    a_magic;         /* magic number */
unsigned long    a_text;          /* size of text segment */
unsigned long    a_data;          /* size of initialized data */
unsigned long    a_bss;           /* size of uninitialized data */
unsigned long    a_syms;          /* size of symbol table */
unsigned long    a_entry;         /* entry point address */
unsigned long    a_entry_mod;     /*+entry point - mod number */
unsigned long    a_trsize;        /* size of text relocation -
                                   * specifies relocation of link
                                   * tables at present */
unsigned long    a_drsize;        /* size of data relocation -
                                   * current assembler sets to zero */
unsigned long    a_mod;           /*+size of mod table within text */
unsigned long    a_link;          /*+size of link table within text */
unsigned long    a_strings;       /*+size of string table */
unsigned long    a_text_addr;     /*+address of text section in
                                   *+address space */
unsigned long    a_mod_addr;      /*+address of mod table in
                                   *+address space */
unsigned long    a_dat_addr;      /*+address of data section in
                                   *+address space */
};

#define OMAGIC   0407             /* old impure format */
#define NMAGIC   0410             /* read-only text */
#define ZMAGIC   0413             /* demand load format */
#define XMAGIC   0414             /*+demand load format
                                   *+locations 0-1023 unmapped */
/*
 * Macros that take exec structures as arguments and tell whether
 * the file has a reasonable magic number or give offsets to
 * text|symbols|strings.
 */
#define N_BADMAG(x) \
    (((x).a_magic)!=OMAGIC && ((x).a_magic)!=NMAGIC && \
     ((x).a_magic)!=XMAGIC && ((x).a_magic)!=ZMAGIC)

#define N_TXTOFF(x) \
        ((x).a_magic>NMAGIC ? 1024 : sizeof (struct exec))
#define N_SYMOFF(x) \
        (N_TXTOFF(x) + (x).a_text + (x).a_data \
                     + (x).a_trsize + (x).a_drsize)
#define N_STROFF(x) \
        (N_SYMOFF(x) + (x).a_syms)
```

This file has five sections: a header, the program text and data, relocation information, a symbol table, and a string table. At present, *ld*(1) ignores all relocation information. The assembler does not provide any relocation information. The last three sections may be omitted if the symbols and relocation have been removed by *strip*(1).

In the header the sizes of each section are given in bytes. The size of the header is not included in any of the other sizes.

When an *a.out* file is executed, three logical segments are set up: the text segment, the data segment (with uninitialized data, which starts off as all 0, following initialized), and a stack. The text segment begins at location 0 in the core image; the header is not loaded. The text segment begins with the module table, followed by the code and link tables for the modules. Magic numbers control where the text segment begins in the address space.

If the magic number is **OMAGIC** (0407), the text segment starts at location 0 in the address space and is not to be write-protected or shared, so the data segment is immediately contiguous with the text segment. This is rarely used.

If the magic number is **NMAGIC** (0410), the data segment begins at the first 0 mod 1024 byte boundary following the text segment, and the program cannot write on the text segment. Just as in the OMAGIC file, however, the data follows immediately after the text segment in the *a.out* file. This is the format used by the assembler.

If the magic number is **ZMAGIC** (0413), the data segment begins at the first 0 mod 1024 byte boundary following the text segment, the size of the text segment is a whole number of 1024 byte blocks, and the program cannot write on text segment. If other processes are executing the same file, they will share the text segment.

If the magic number is **XMAGIC** (0414), the first two pages of the address space are unmapped to trap references through zero pointers, and the text segment starts at address 0x400, but otherwise the format is as for ZMAGIC.

For both XMAGIC and ZMAGIC format, the text segment begins at offset 1024 in the *a.out* file and the remaining bytes after the header in the first block are reserved and should be zero. In this case the text and data sizes must both be multiples of 1024 bytes, and the pages of the file will be brought into the running image as needed. This is especially suitable for very large programs and is the default format produced by *ld*(1).

The stack will occupy the highest possible locations in the core image: growing downwards from 0xfffc00. The stack is automatically extended as required. The data segment is only extended as requested by *break*(2).

After the header in the file follow the text, data, text relocation, data relocation, symbol table, and string table in that order. The text begins at byte 1024 in the file for XMAGIC or ZMAGIC format or just after the header for the other formats. The N_TXTOFF macro returns this absolute file position when given the name of an exec structure as argument. The data segment is contiguous with the text and is immediately followed by the text relocation and then the data relocation information. The symbol table follows all this; its position is computed by the N_SYMOFF macro. Finally, the string table immediately follows the symbol table at a position that can be gotten easily using N_STROFF.

Address Mapping

Although the magic numbers specify how a file is to be mapped into the address space, there are also three numbers that specify this information redundantly for files that may be executed under GENIX. These numbers also allow more complex mappings if the file is not executed under GENIX. These numbers specify the starting addresses for three different parts of the address space: the mod table, the code and link tables, and the data (both initialized and uninitialized). Within the text section of the *a.out* file are two subsections: the mod table and everything else. The starting addresses of these two sections may be specified independently to the linker.

The addresses in the *a.out* file do not match quite as well to these sections of the file. One address is for the starting address of the mod table. Another address is for the starting address of the data. The last address is the starting address of the text section assuming that the mod table is at the beginning of the text section. If the mod table is really elsewhere, then nothing will actually exist until after the mod table. These three numbers indicate the assumptions that were made by the linker when it created the mod table, the link tables, and the entry point in the image.

Symbol Table

The symbol table is an array of 12-byte entries, each of which identifies its purpose. The symbol table contains all of the information that is not stored within the address space, but is needed by *ld*(1), by symbolic debuggers, or by any other programs. The symbol table format is described in *stab*(5).

FILES

/usr/include/a.out.h
/usr/include/stab.h

SEE ALSO

as(1), ddt(1), ld(1), nm(1), strip(1), stab(5)

CROSS-SUPPORT

In a cross support environment, the name of NS16000 *a.out* file is *a16.out*, while the include files are in /usr/NSC/include/a.out.h and /usr/NSC/include/stab.h.