Offset	Size	Name	Description
00h	WORD	ne_magic	Signature word NEMAGIC
On-disk			
02h	BYTE	ne_ver	Version number of the linker
03h	BYTE	ne_rev	Revision number of the linker
In-memory			
02h	WORD	count	Usage count (ne_ver/ne_rev on disk)
04h	WORD	ne_enttab	Entry Table file offset, relative to the beginning of the segmented EXE header
On-disk			
06h	WORD	ne_cbenttab	Number of bytes in the entry table
In-memory			
06h	WORD	next	Selector to next module
On-disk			
08h	DWORD	ne_crc	32-bit CRC of entire contents of file. These words are taken as 00 during the calculation
In-memory			
08h	WORD	dgroup_entry	Near ptr to segment entry for DGROUP
0Ah	WORD	fileinfo	Near ptr to file info (OFSTRUCT)
0Ch	WORD	ne_flags	Flag word
0Eh	WORD	ne_autodata	Segment number of automatic data segment. This value is set to zero if SINGLEDATA and MULTIPLEDATA flag bits are clear, NOAUTODATA is indicated in the flags word. A Segment number is an index into the module's segment table. The first entry in the segment table is segment number 1
10h	WORD	ne_heap	Initial size, in bytes, of dynamic heap added to the data segment. This value is zero if no initial local heap is allocated
12h	WORD	ne_stack	Initial size, in bytes, of stack added to the data segment. This value is zero to indicate no initial stack allocation, or when SS is not equal to DS
14h	DWORD	ne_csip	Segment number:offset of CS:IP
18h	DWORD	ne_sssp	Segment number:offset of SS:SP If SS equals the automatic data segment and SP equals zero, the stack pointer is set to the top of the automatic data segment just below the additional heap area. +——————————+ ! additional dynamic heap ! +——————————+ ← SP ! additional stack ! +——————————+ ! loaded auto data segment !% +——————————+ ← DS, SS
1Ch	WORD	ne_cseg	Number of entries in the Segment Table
1Eh	WORD	ne_cmod	Number of entries in the Module Reference Table
20h	WORD	ne_cbnrestab	Number of bytes in the Non-Resident Name Table
22h	WORD	ne_segtab	Segment Table file offset, relative to the beginning of the segmented EXE header
24h	WORD	ne_rsrctab	Resource Table file offset, relative to the beginning of the segmented EXE header

Offset	Size	Name	Description
26h	WORD	ne_restab	Resident Name Table file offset, relative to the beginning of the segmented EXE header
28h	WORD	ne_modtab	Module Reference Table file offset, relative to the beginning of the segmented EXE header
2Ah	WORD	ne_imptab	Imported Names Table file offset, relative to the beginning of the segmented EXE header
2Ch	DWORD	ne_nrestab	Non-Resident Name Table offset, relative to the beginning of the file
30h	WORD	ne_cmovent	Number of movable entries in the Entry Table
32h	WORD	ne_align	Logical sector alignment shift count, log(base 2) of the segment sector size (default 9)
34h	WORD	ne_cres	Number of resource entries
36h	BYTE	ne_exetyp	Executable type, used by loader. 02h = WINDOWS
37h	BYTE	ne_flagsothers	Operating system flags
38h	WORD	???	offset to return thunks or start of gangload area
3Ah	WORD	???	offset to segment reference thunks or length of gangload area
3Ch	WORD	???	minimum code swap area size
3Eh	2 BYTEs	???	expected Windows version (minor version first)

On-disk segment entry

Offset	Size	Name	Description
00h	WORD	ns_sector	Logical-sector offset (n byte) to the contents of the segment data, relative to the beginning of the file. Zero means no file data
02h	WORD	ns_cbseg	Length of the segment in the file, in bytes. Zero means 64K
04h	WORD	ns_flags	Flag word
06h	WORD	ns_minalloc	Minimum allocation size of the segment, in bytes. Total size of the segment. Zero means 64K

In-memory segment entry

Offset	Size	Name	Description
00h	WORD	ns1_sector	Logical-sector offset (n byte) to the contents of the segment data, relative to the beginning of the file. Zero means no file data
02h	WORD	ns1_cbseg	Length of the segment in the file, in bytes. Zero means 64K
04h	WORD	ns1_flags	Flag word
06h	WORD	ns1_minalloc	Minimum allocation size of the segment, in bytes. Total size of the segment. Zero means 64K
08h	WORD	ns1_handle	Selector or handle (selector - 1) of segment in memory

struct new_segdata {

```
  union {
      struct {
          WORD      ns_niter;
          WORD      ns_nbytes;
          char      ns_iterdata;
      } ns_iter;
      struct {
```

```
          char      ns_data;
      } ns_noniter;
  } ns_union;
```

};

Offset	Size	Name	Description
00h	WORD	nr_nreloc	???

struct new_rlc {

```
  char            nr_stype;
  char            nr_flags;
  WORD  nr_soff;
  union {
      struct {
          char            nr_segno;
          char            nr_res;
          WORD  nr_entry;
        } nr_intref;
      struct {
          WORD  nr_mod;
          WORD  nr_proc;
        } nr_import;
      struct {
          WORD  nr_ostype;
          WORD  nr_osres;
        } nr_osfix;
    } nr_union;
```

};

Offset	Size	Name	Description
00h	char	rs_len	???
01h	char	rs_string[1]	???

Offset	Size	Name	Description
00h	WORD	rt_id	???
02h	WORD	rt_nres	???
04h	DWORD	rt_proc	???

Offset	Size	Name	Description
00h	WORD	rn_offset	???
02h	WORD	rn_length	???
04h	WORD	rn_flags	???
06h	WORD	rn_id	???
08h	WORD	rn_handle	???
0Ah	WORD	rn_usage	???

Offset	Size	Name	Description
00h	WORD	rs_align	???
02h	struct rsrc_typeinfo	rs_typeinfo	???