Offset	Size	Name	Description
00h	WORD	ne_magic	Signature word EMAGIC

On-disk segment entry struct new_seg { 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 */ WORD ns_cbseg; /* Length of the segment in the file, in bytes. Zero means 64K */ WORD ns_flags; /* Flag word */ WORD ns_minalloc; /* Minimum allocation size of the segment, in bytes. Total size of the segment. Zero means 64K */ }; In-memory segment entry struct new_seg1 {*

```
  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 */
  WORD  ns1_cbseg;                 /* Length of the segment in the file, in
bytes. Zero means 64K */
  WORD  ns1_flags;                 /* Flag word */
  WORD  ns1_minalloc;               /* Minimum allocation size of the
segment, in bytes. Total size

                                      of the segment. Zero means 64K */
  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;
```

};

struct new_rlcinfo {

```
  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;
```

};

#define NR_STYPE(x) (x).nr_stype #define NR_FLAGS(x) (x).nr_flags #define NR_SOFF(x) (x).nr_soff
#define NR_SEGNO(x) (x).nr_union.nr_intref.nr_segno #define NR_RES(x) (x).nr_union.nr_intref.nr_res
#define NR_ENTRY(x) (x).nr_union.nr_intref.nr_entry #define NR_MOD(x)
(x).nr_union.nr_import.nr_mod #define NR_PROC(x) (x).nr_union.nr_import.nr_proc #define
NR_OSTYPE(x) (x).nr_union.nr_osfix.nr_ostype #define NR_OSRES(x) (x).nr_union.nr_osfix.nr_osres

#define NRSTYP 0x0f #define NRSBYT 0x00 #define NRSSEG 0x02 #define NRSPTR 0x03 #define
NRSOFF 0x05 #define NRPTR48 0x06 #define NROFF32 0x07 #define NRSOFF32 0x08

#define NRADD 0x04 #define NRRTYP 0x03 #define NRRINT 0x00 #define NRRORD 0x01 #define
NRRNAM 0x02 #define NRROSF 0x03 #define NRICHAIN 0x08

#if (EXE386 == 0)

#define RS_LEN(x) (x).rs_len #define RS_STRING(x) (x).rs_string #define RS_ALIGN(x) (x).rs_align

#define RT_ID(x) (x).rt_id #define RT_NRES(x) (x).rt_nres #define RT_PROC(x) (x).rt_proc

#define RN_OFFSET(x) (x).rn_offset #define RN_LENGTH(x) (x).rn_length #define RN_FLAGS(x)
(x).rn_flags #define RN_ID(x) (x).rn_id #define RN_HANDLE(x) (x).rn_handle #define RN_USAGE(x)
(x).rn_usage

#define RSORDID 0x8000

#define RNMOVE 0x0010 #define RNPURE 0x0020 #define RNPRELOAD 0x0040 #define RNDISCARD
0xF000

#define NE_FFLAGS_LIBMODULE 0x8000

struct rsrc_string {

```
    char    rs_len;
    char    rs_string[1];
```

};

struct rsrc_typeinfo {

```
   WORD   rt_id;
   WORD   rt_nres;
   DWORD               rt_proc;
```

};

struct rsrc_nameinfo {

```
   WORD   rn_offset;
   WORD   rn_length;
   WORD   rn_flags;
   WORD   rn_id;
   WORD   rn_handle;
   WORD   rn_usage;
```

};

struct new_rsrc {

```
   WORD               rs_align;
   struct rsrc_typeinfo     rs_typeinfo;
```

};

#endif

#pragma pack(pop)

#ifdef __cplusplus } /* extern "C" */ #endif

#endif