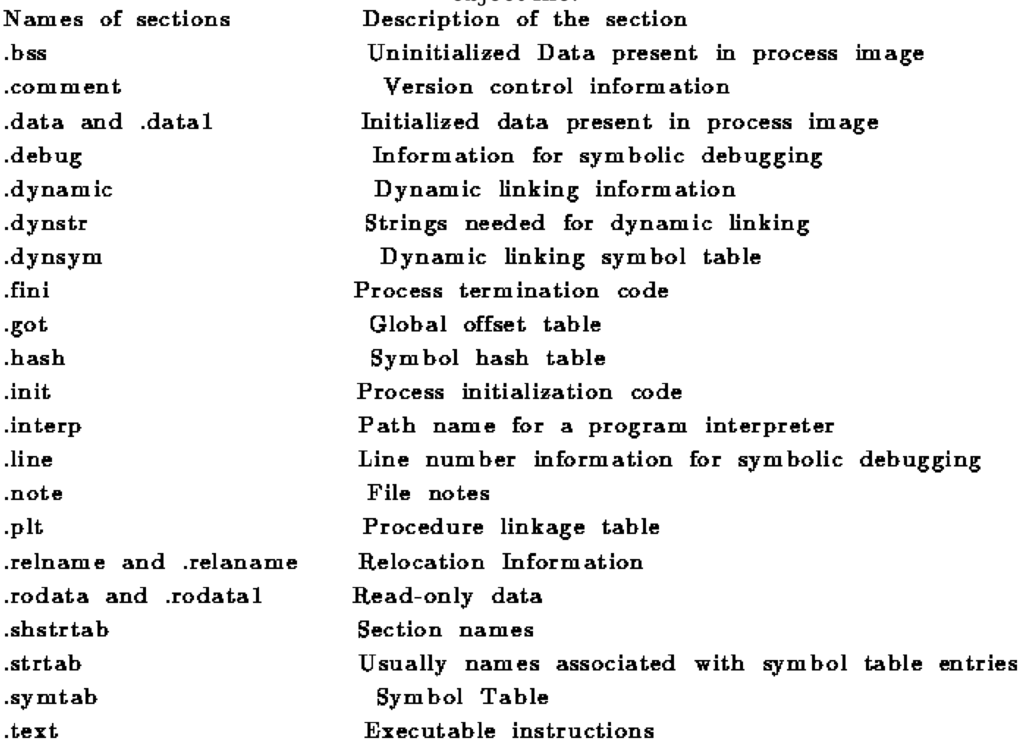
## The Section Header Table

All sections in object files can be found using the Section header table. The section header, similar to the program header, is an array of structures. Each entry correlates to a section in the file. The entry provides the name, type, memory image starting address (if loadable), file offset, the section’s size in bytes, alignment, and how the information in the section should be interpreted. Figure 2.8 details the specific fields of the structure. The name provided in the structure is actually an index into the string table (a section in the object file) where the actual string representation of the name of the section exists. Sections will be discussed further below.

**Figure:** Special Sections. A brief description of sections that can appear in an ELF object file.



| Names of sections | Description of the section |
|---|---|
| .bss | Uninitialized Data present in process image |
| .comment | Version control information |
| .data and .data1 | Initialized data present in process image |
| .debug | Information for symbolic debugging |
| .dynamic | Dynamic linking information |
| .dynstr | Strings needed for dynamic linking |
| .dynsym | Dynamic linking symbol table |
| .fini | Process termination code |
| .got | Global offset table |
| .hash | Symbol hash table |
| .init | Process initialization code |
| .interp | Path name for a program interpreter |
| .line | Line number information for symbolic debugging |
| .note | File notes |
| .plt | Procedure linkage table |
| .relname and .relaname | Relocation Information |
| .rodata and .rodata1 | Read-only data |
| .shstrtab | Section names |
| .strtab | Usually names associated with symbol table entries |
| .symtab | Symbol Table |
| .text | Executable instructions |