A detailed list of a few OBJDUMP command options.

```
0000000000001140 <register_tm_clones>:
    1140:           48 8d 3d c9 2e 00 00        lea    0x2ec9(%rip),%rdi        # 4010 <__TMC_END__>
    1147:           48 8d 35 c2 2e 00 00        lea    0x2ec2(%rip),%rsi        # 4010 <__TMC_END__>
    114e:           48 29 fe                    sub    %rdi,%rsi
    1151:           48 89 f0                    mov    %rsi,%rax
    1154:           48 c1 ee 3f                 shr    $0x3f,%rsi
    1158:           48 c1 f8 03                 sar    $0x3,%rax
    115c:           48 01 c6                    add    %rax,%rsi
    115f:           48 d1 fe                    sar    %rsi
    1162:       /-- 74 14                       je     1178 <register_tm_clones+0x38>
    1164:       |   48 8b 05 85 2e 00 00       mov    0x2e85(%rip),%rax        # 3ff0 <_ITM_registerTMCloneTable@Base>
    116b:       |   48 85 c0                    test   %rax,%rax
    116e:       +-- 74 08                       je     1178 <register_tm_clones+0x38>
    1170:       |   ff e0                       jmp    *%rax
    1172:       |   66 0f 1f 44 00 00           nopw   0x0(%rax,%rax,1)
    1178:       \-> c3                          ret
    1179:           0f 1f 80 00 00 00 00        nopl   0x0(%rax)

0000000000001180 <__do_global_dtors_aux>:
    1180:               f3 0f 1e fa             endbr64
    1184:               80 3d 85 2e 00 00 00    cmpb   $0x0,0x2e85(%rip)        # 4010 <__TMC_END__>
    118b:       /------ 75 2b                   jne    11b8 <__do_global_dtors_aux+0x38>
    118d:       |       55                      push   %rbp
    118e:       |       48 83 3d 62 2e 00 00    cmpq   $0x0,0x2e62(%rip)        # 3ff8 <__cxa_finalize@GLIBC_2.2.5>
    1195:       |       00
    1196:       |       48 89 e5                mov    %rsp,%rbp
    1199:       |   /-- 74 0c                   je     11a7 <__do_global_dtors_aux+0x27>
    119b:       |   |   48 8b 3d 66 2e 00 00    mov    0x2e66(%rip),%rdi        # 4008 <__dso_handle>
    11a2:       |   |   e8 d9 fe ff ff          call   1080 <__cxa_finalize@plt>
    11a7:       |   \-> e8 64 ff ff ff          call   1110 <deregister_tm_clones>
    11ac:       |       c6 05 5d 2e 00 00 01    movb   $0x1,0x2e5d(%rip)        # 4010 <__TMC_END__>
    11b3:       |       5d                      pop    %rbp
    11b4:       |       c3                      ret
    11b5:       |       0f 1f 00                nopl   (%rax)
    11b8:       \-----> c3                      ret
    11b9:               0f 1f 80 00 00 00 00    nopl   0x0(%rax)
```

Command for above: objdump -d --visualize-jumps test_objdump

1. **Disassembly Options:**
    - **-d, --disassemble**: This option disassembles the executable sections of the object file. It's often used to see the assembly code corresponding to the compiled machine code.
    - **-D, --disassemble-all**: Similar to **-d**, but it disassembles all sections, not just the ones flagged as executable.
    - **-l, --line-numbers**: When used with disassembly options, it intermixes the assembly with source code line numbers, assuming debugging information is present. Useful for correlating assembly to source code.
    - **-S, --source**: Intermixes the source code with the disassembly. This gives a side-by-side view of the C/C++ code and its corresponding assembly, which is invaluable for understanding how code translates to machine instructions.
2. **Header Options:**
    - **-a, --archive-header**: If the object file is an archive (like a static library), this displays its header.
    - **-f, --file-headers**: Shows the overall file header, providing metadata about the object file or executable.
    - **-h, --section-headers, --headers**: Displays section headers. These headers give metadata about each section (like **.text** for code, **.data** for initialized data, etc.) including size, location, and flags.

3. **Symbol & Relocation Options:**
    - **-t, --syms**: Displays the symbol table. This table includes names and addresses of functions, variables, and other named entities in the object file or executable.
    - **-T, --dynamic-syms**: For shared libraries and dynamically linked executables, this displays the dynamic symbol table, which is used at runtime for linking.
    - **-r, --reloc**: Shows relocation entries. These indicate references that need adjustment when multiple object files are linked together.
4. **Debugging & Dynamic Linking Options:**
    - **-g, --debugging**: Displays debugging information, if present in the object file.
    - **-R, --dynamic-reloc**: Shows dynamic relocation entries, which are used when dynamically linked executables or shared libraries are loaded into memory.
5. **Content Display Options:**
    - **-s, --full-contents**: Instead of just headers or summaries, this option dumps the full contents of the specified sections, typically in a hexadecimal format.
    - **--dwarf**: Displays the contents of the file's DWARF debug sections, if present. DWARF is a standardized debugging data format used by many compilers and debuggers.
6. **Architecture & Target Specification:**
    - **-b, --target=BFDNAME**: Allows you to specify the format of the object file. Useful when working with cross-compiled binaries or non-standard object formats.
    - **-m, MACHINE**: Specifies the machine architecture for disassembly. Useful in cross-compilation scenarios.
7. **Address Manipulation Options:**
    - **--adjust-vma=OFFSET**: Adjusts the displayed virtual memory addresses by the specified offset. Useful when you want to see disassembly or other data as if it was loaded at a specific memory address.
    - **--start-address=ADDR** and **--stop-address=ADDR**: Limit the display to only include data within the specified address range.
8. **Output Control Options:**
    - **-j, --section=NAME**: Only process the section named **NAME**.
    - **--prefix=PREFIX**: Add **PREFIX** to absolute paths in debug information.
    - **--prefix-strip=LEVEL**: Remove **LEVEL** number of leading directories from the absolute paths.
    - **--show-raw-insn**: Display the instruction bytes alongside the disassembled output.
    - **--no-show-raw-insn**: Hide instruction bytes, showing only the disassembled output.
9. **Disassembly Display Options:**
    - **-M, --disassembler-options=OPTS**: Pass options to the disassembler. The available options depend on the target architecture.
    - **--insn-width=WIDTH**: Display **WIDTH** bytes on a single line when showing raw instruction bytes.

10. **Filtering Options:**
    - **--private-headers**: Display format-specific file headers.
    - **--start-address=ADDR**: Begin displaying data starting at address **ADDR**.
    - **--stop-address=ADDR**: Stop displaying data at address **ADDR**.
11. **File Format Selection:**
    - **-b, --target=BFDNAME**: Choose the object format of the input file (e.g., **elf32-i386**, **a.out-sunos-big**).
12. **Miscellaneous Options:**
    - **-i, --info**: Print a summary of the available formats and architectures.
    - **-v, --version**: Display the version of **objdump**.
    - **-W[lLiaprmfFsoRt] or --dwarf[=rawline,=decodedline,=info,=abbrev,=pubnames,=aranges,=macro,=frames,=frames-interp,=str,=loc,=Ranges,=pubtypes]**: Display the DWARF debug information in the file, if present. The optional arguments allow for specific DWARF sections to be displayed.
13. **Advanced Options:**
    - **--special-syms**: Include special symbols in the symbol table listing. These are symbols that the target uses internally and are normally not of interest to the user.
    - **--dynamic**: When displaying symbols, display dynamic symbols instead of normal symbols. This is only meaningful for dynamic objects, like certain types of shared libraries.
    - **--no-adjust-vma**: Don't adjust section addresses based on their VMA.
    - **--[no-]section-headers**: Display the section headers. The **--no-section-headers** option can be used to turn off this display.
    - **--section-groups**: Display the section groups in the object file, if present.
14. **Debugging Options:**
    - **-EB**: Assume big-endian format when interpreting the object file.
    - **-EL**: Assume little-endian format.
    - **--gdb**: Emit output in a format that can be used as input to some versions of **gdb**.

These are segregated list of few options, but there are many that are present to explore.