🐞 GDB 常用命令速查表（含注释与示意）

> 本文档适用于 GDB 调试 C 程序、RISC-V 内核（如 uCore / xv6）以及 QEMU 虚拟机环境。

🧩 一、启动与连接

命令	说明
`gdb <program>`	启动 GDB 并加载可执行文件
`target remote :1234`	连接 QEMU 或远程调试目标
`symbol-file <file>`	加载符号信息（常用于内核调试）
`set architecture riscv:rv64`	设置架构为 RISC-V 64 位
`set endian little`	设置为小端模式
`quit`	退出 GDB

⚙️ 二、运行控制

命令	说明
`run` / `r`	从头运行程序
`continue` / `c`	继续执行直到下一个断点
`si`（`stepi`）	单步执行一条汇编指令（进入函数）
`ni`（`nexti`）	单步执行一条汇编指令（不进入函数）
`step` / `s`	单步执行一行 C 代码（进入函数）
`next` / `n`	单步执行一行 C 代码（不进入函数）
`finish`	执行到当前函数返回
`until <addr>`	执行到指定地址
`jump *<addr>`	直接跳转到某个地址（慎用）

🧨 三、断点管理

命令	说明
`break <func>`	在函数入口处设置断点
`break *0x80200000`	在指定地址设置断点（常用于内核入口）
`info breakpoints` / `info b`	查看所有断点

命令	说明
`delete <num>`	删除断点
`disable <num>`	禁用断点
`enable <num>`	启用断点
`clear <func>`	清除函数断点

🧭 四、寄存器与内存

命令	说明
`info registers` / `i r`	查看所有寄存器
`info registers sp ra pc`	查看部分寄存器
`set $sp = 0x80210000`	修改寄存器的值
`x/<n><f> <addr>`	查看内存内容（非常常用）
示例：	
`x/10i $pc`	查看从当前 PC 开始的 10 条指令
`x/8x 0x80200000`	查看 8 个 32 位十六进制数
`x/4gx $sp`	查看栈内容（64 位）
`x/s $a0`	将地址内容当作字符串打印

🔍 五、反汇编与代码查看

命令	说明
`list` / `l`	显示当前源代码
`list <func>`	显示指定函数
`disassemble` / `disas`	反汇编当前函数
`disas /r`	同时显示机器码与汇编
`x/20i <addr>`	从指定地址反汇编 20 条指令

🧱 六、查看函数调用栈

命令	说明
`backtrace` / `bt`	查看函数调用栈

命令	说明
`frame <n>` / `f <n>`	切换到第 n 层栈帧
`info frame`	显示当前栈帧信息
`info args`	显示当前函数参数
`info locals`	显示当前函数局部变量
`up` / `down`	在调用栈中上下切换

七、变量与表达式

命令	说明
`print <expr>` / `p <expr>`	打印表达式的值
`p/x $pc`	十六进制显示
`p/d $t0`	十进制显示
`set var <expr>`	修改变量值
`display <expr>`	程序每次停止时自动显示
`undisplay <num>`	取消自动显示

八、文件与符号信息

命令	说明
`info files`	查看加载文件信息
`info functions`	查看符号表中所有函数
`info variables`	查看全局变量
`info line <addr>`	查找某地址对应源码行

九、日志与输出控制

命令	说明
`set pagination off`	关闭分页（防止 “--More--” 卡顿）
`set disassemble-next-line on`	执行时自动显示当前指令
`set confirm off`	禁用确认提示
`set print asm-demangle on`	显示解码后的函数名（C++）

命令	说明
`set logging on`	将输出记录到日志文件

🧭 十、RISC-V 调试常用命令

命令	说明
`break *0x80200000`	在内核入口设置断点
`info registers`	查看通用寄存器
`p/x $mstatus`、`p/x $sstatus`	查看特权级状态寄存器
`p/x $satp`	查看页表根地址寄存器
`x/10i $pc`	查看当前执行的 10 条指令

💡 十一、组合示例：调试 uCore 启动

```
make qemu-gdb
```

在另一个终端启动 GDB：

```
target remote :1234
symbol-file bin/kernel
break *0x80200000
continue
x/10i $pc
```

单步跟踪内核入口：

```
si
info registers pc ra sp
```

✅ 十二、记忆口诀：三看两控一打断

类型	命令	说明
三看	`info registers`、`x/i`、`bt`	看寄存器、内存、栈
两控	`si`、`c`	控制执行
一打断	`break`	设置断点