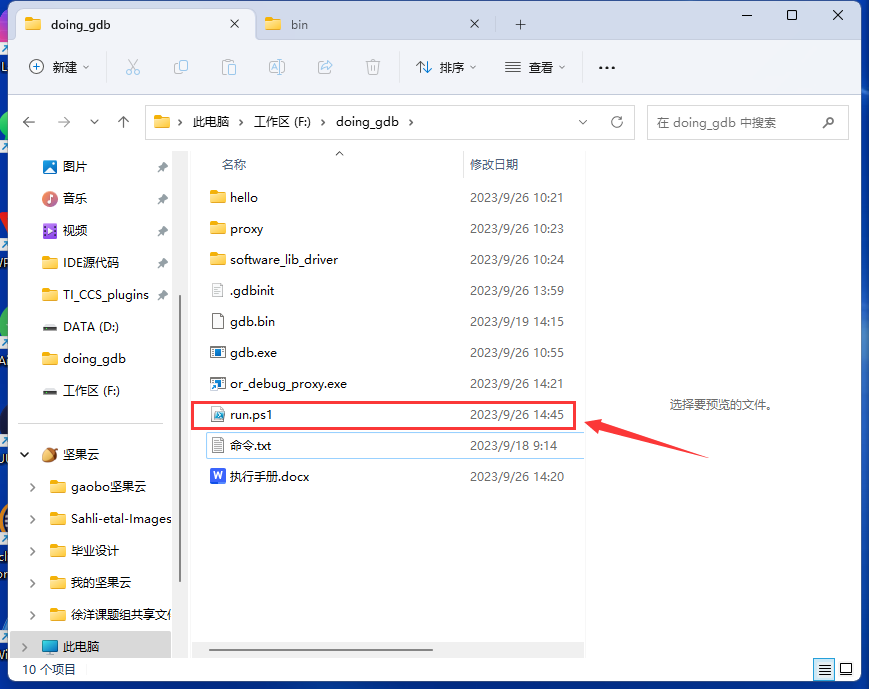
### GDB - proxy实时调试

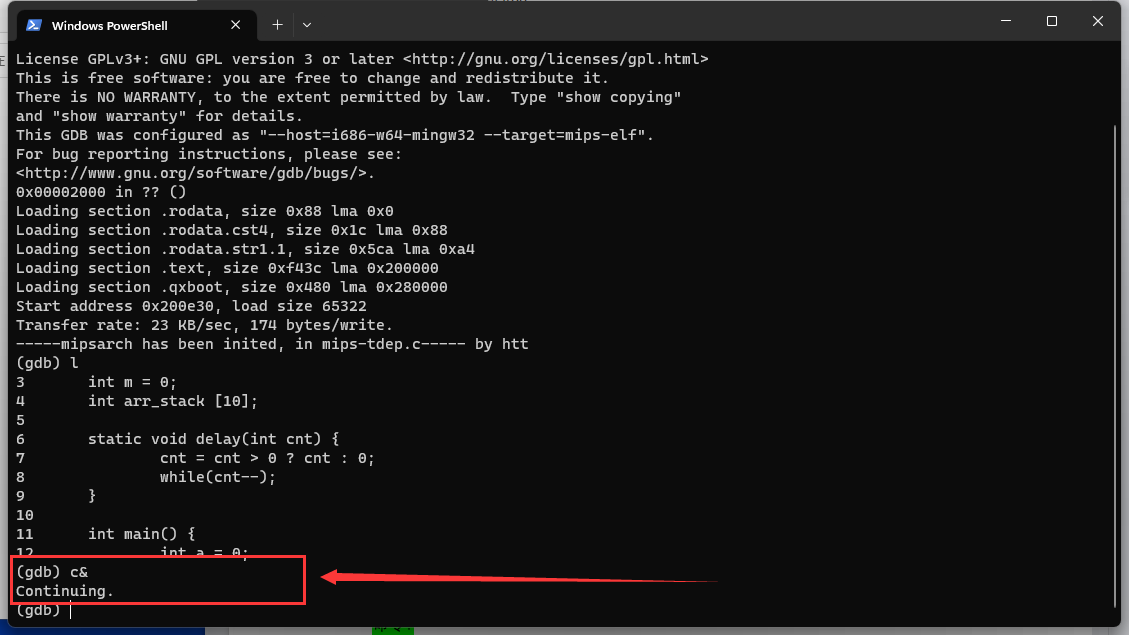
1 使用powershell脚本运行gdb和proxy



运行该脚本，该脚本会自动启动gdb和proxy,并会自动加载程序并下载到板子上

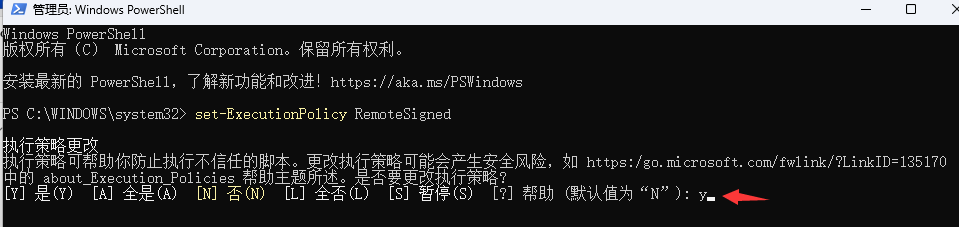
.\software\_lib\_driver\software\_lib\_driver\_core0\main.c

注意：原有的 “c” 命令 全部需要换成“c&”



后续的调试都可以按照正常的模式进行调试

注意：如果运行脚本的时候，出现无法访问的情况, 可以以管理员权限启动powershell，然后输入set-ExecutionPolicy RemoteSigned

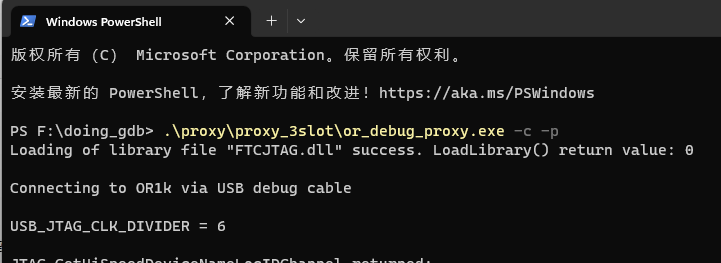


2 使用命令行启动

1. 启动proxy

命令：

.\proxy\proxy\_3slot\or\_debug\_proxy.exe -c -p



1. 启动默认启动gdb

如果想用gdb上写好的.gdbinit启动脚本，该脚本会自动连接proxy，并开启异步执行逻辑

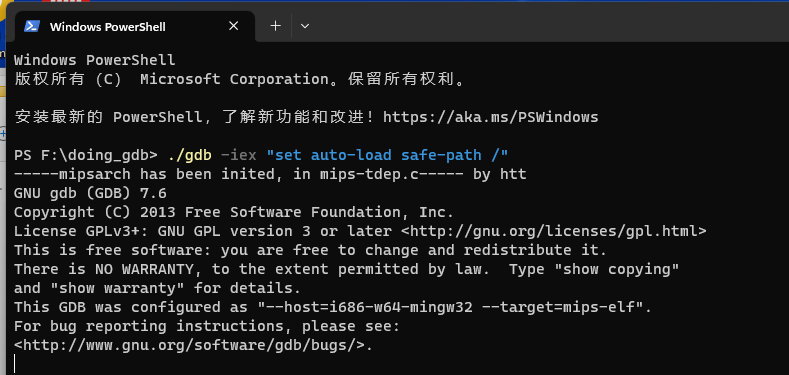
然后会加载

.\software\_lib\_driver\software\_lib\_driver\_core0\Release\software\_lib\_driver\_core0.out文件

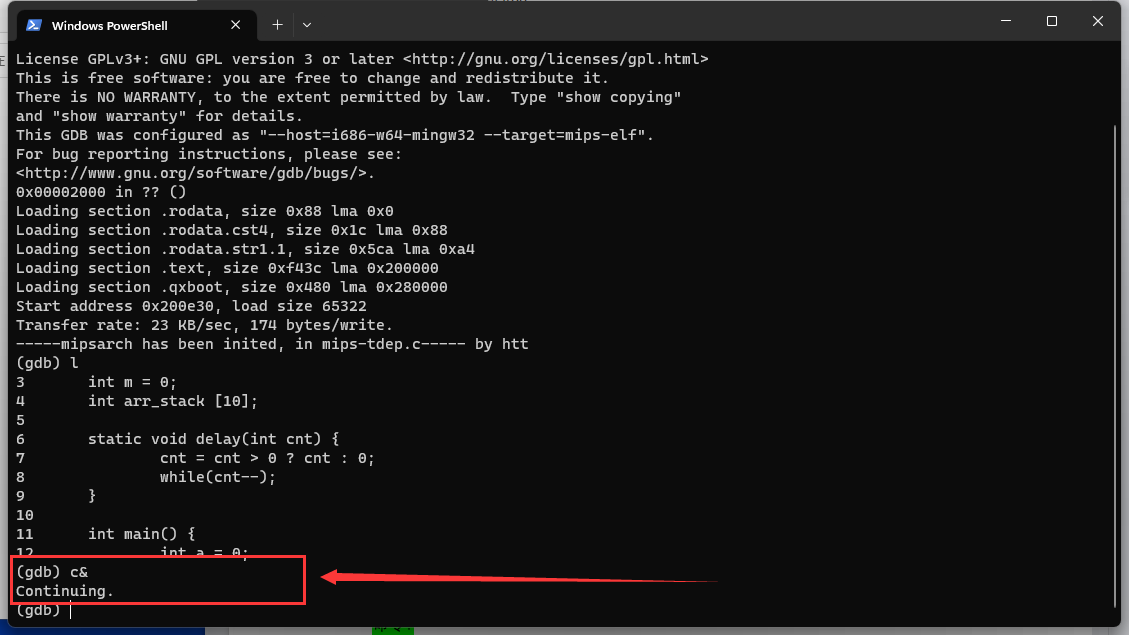
不需要再手动输入target remote:3333、load、file等

命令:

./gdb -iex "set auto-load safe-path /"



注意：原有的 “c” 命令 全部需要换成“c&”



后续的调试都可以按照正常的模式进行调试

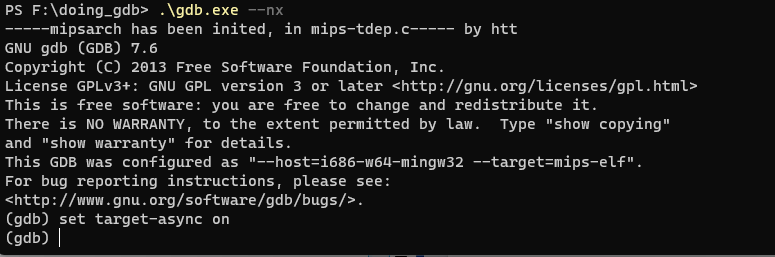
1. 如果不想使用gdb中的默认配置，可以手动启动gdb

命令:

.\gdb.exe --nx

然后输入gdb命令开启异步模式：

set target-async on



后续按照顺序进行 target remote :3333 load、file、c&等

附注

默认的调试程序：

software\_lib\_driver\_core0.out

#include "common.h"

*int* m = 0;

*int* arr\_stack [10];

static *void* delay(*int* *cnt*) {

*cnt* = *cnt* > 0 ? *cnt* : 0;

    while(*cnt*--);

}

*int* main() {

*int* a = 0;

*int* b = 0;

    while (1) {

        a += 1;

        a %= 10;

        m += 1;

        m %= 10;

        arr\_stack[a] = m;

        //delay(10000);

    }

    return 0;

}