GigaDevice Semiconductor Inc.

Arm® Cortex®- M3/M4/M23/M33 32-bit MCU

应用笔记

AN023

目录

图索引

表索引

1. 简介

Letter shell 是一个 C 语言编写的，可以嵌入在程序中的嵌入式 shell，主要面向嵌入式设备，以 C 语言函数为运行单位，可以通过命令行调用，运行程序中的函数。

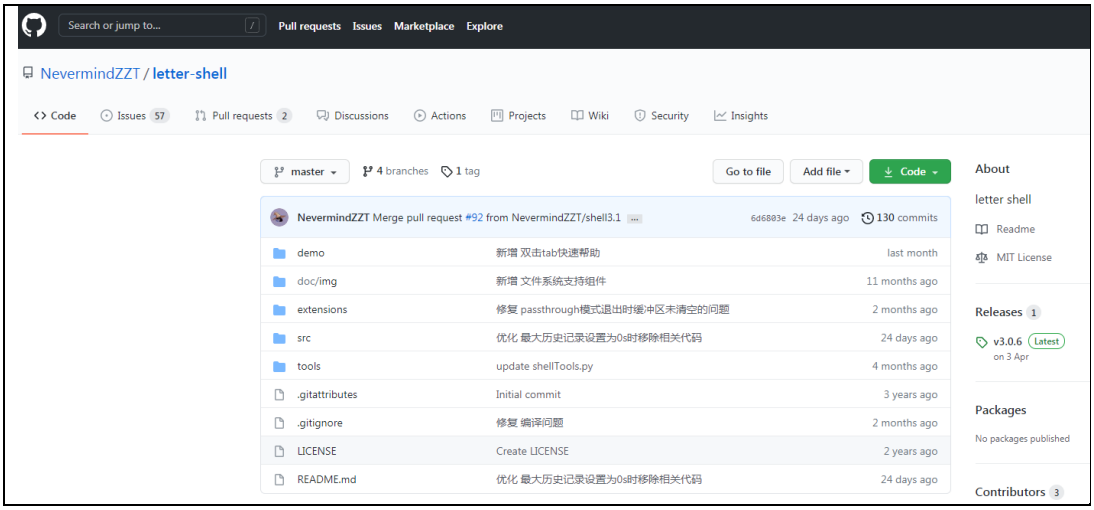
1.1. 主要特性

- 命令自动补全
- 快捷键功能定义
- 命令权限管理
- 用户管理
- 变量支持
- 代理函数和参数代理解析

2. 获取 Letter shell 源码

Letter shell 的 github 仓库地址为：https://github.com/NevermindZZT/letter-shell。具体如*图 2-1. Letter shell 的 github 仓库*所示。

图 2-1. Letter shell 的 github 仓库



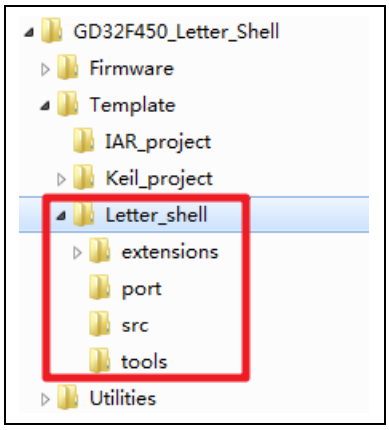
本 AN 使用的 Letter shell 版本为 3.0.6。

3. GD32F450 Letter shell 移植

3.1. 工程文件结构

本 AN 是以 GD32F4xx_Firmware_Library 默认的 Template 工程为基础搭建的，新建 Letter_shell 文件夹，导入 Letter_shell 核心文件，将 extensions、src 和 tools 文件夹直接拷贝到 Letter_shell 文件夹里，新建 port 文件夹，用于存放移植接口文件 shell_cfg.h、shell_port.h 和 shell_port.c 文件。具体如*图 3-1. 工程文件结构*所示。

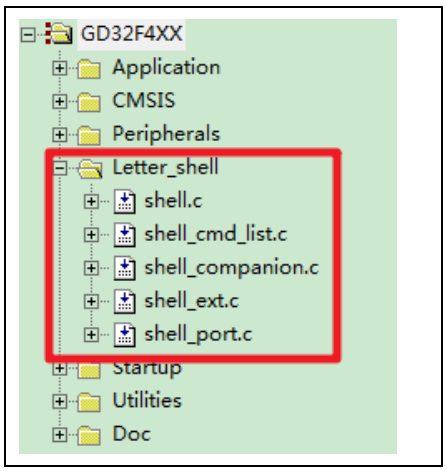
图 3-1. 工程文件结构



3.2. 搭建 Keil 工程框架

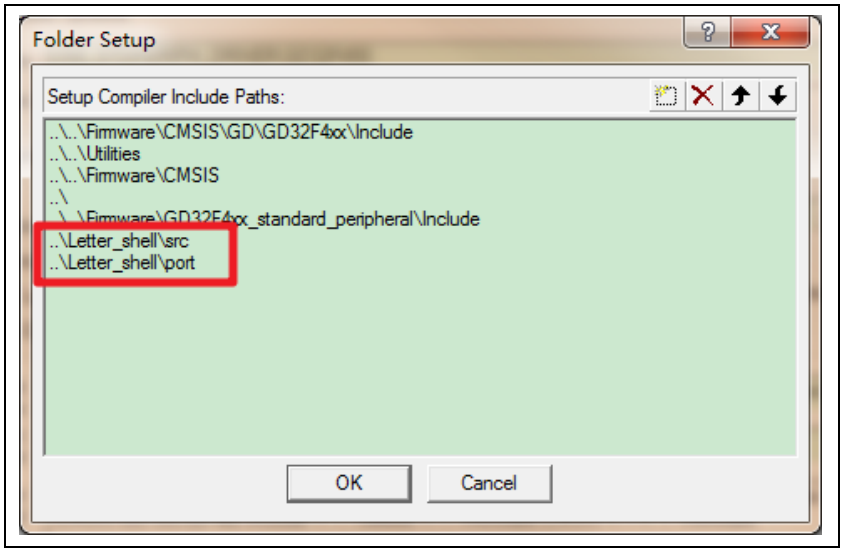
以 Keil 这个 IDE 为例（其他 IDE 工程搭建情况类似，这里不做赘述），在工程里添加 Letter_shell，并将 src 文件夹和 port 文件夹下的 c 文件添加进来，具体如*图 3-2. Keil 工程结构*所示。

图 3-2. Keil 工程结构



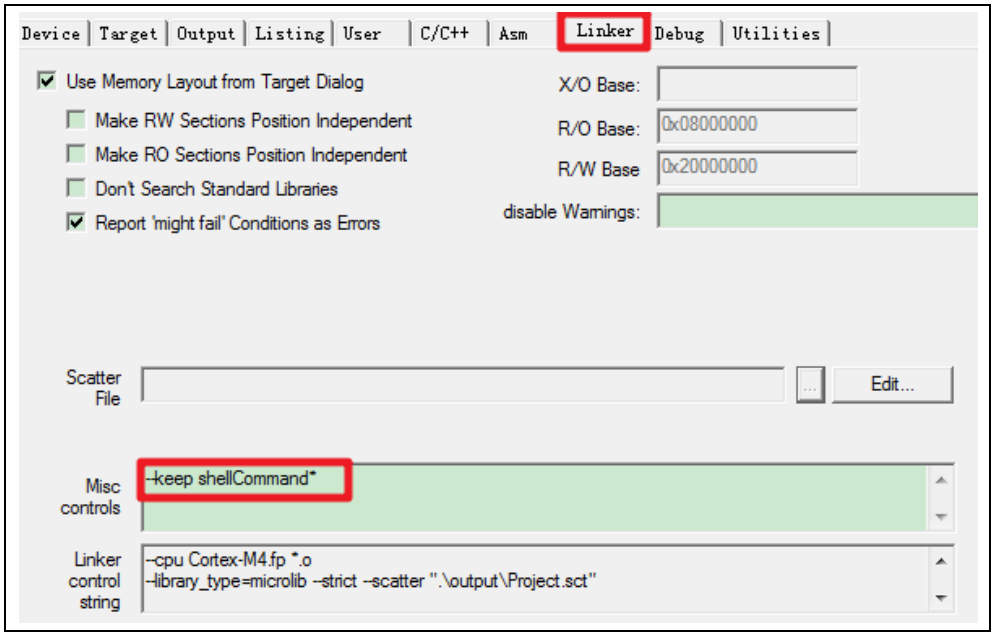
在 Setup Compiler Include Paths 中将 src 和 port 文件夹里的头文件包含进来，具体如*图 3-3. Keil Folder Setup 设置*所示。

图 3-3. Keil Folder Setup 设置



在 Keil 工程的 Linker 选项卡里，添加--keep shellCommand*，以防被编译优化，具体如*图 3-4. Keil Linker 设置*所示。

图 3-4. Keil Linker 设置



3.3. 编写移植接口文件

移植接口文件编写主要是 shell_port.c，具体如*表 3-1. Shell 结构体*所示。

表 3-1. Shell 结构体

```
typedef struct shell_def
{
    struct
    {
```

```
        const struct shell_command *user;
        int activeTime;
        char *path;
    #if SHELL_USING_COMPANION == 1
        struct shell_companion_object *companions;
    #endif
    } info;
    struct
    {
        unsigned short length;
        unsigned short cursor;
        char *buffer;
        char *param[SHELL_PARAMETER_MAX_NUMBER];
        unsigned short bufferSize;
        unsigned short paramCount;
        int keyValue;
    } parser;
    struct
    {
        char *item[SHELL_HISTORY_MAX_NUMBER];
        unsigned short number;
        unsigned short record;
        signed short offset;
    } history;
    struct
    {
        void *base;
        unsigned short count;
    } commandList;
    struct
    {
        unsigned char isChecked : 1;
        unsigned char isActive : 1;
        unsigned char tabFlag : 1;
    } status;
    signed char (*read)(char *);
    void (*write)(const char);
} Shell;
```

从对 Shell 结构的定义来看，需要实现 shell read 和 write 函数，具体如*表 3-2. userShellRead 函数实现*和*表 3-3. userShellWrite 函数实现*所示。

表 3-2. userShellRead 函数实现

```
signed char userShellRead(char *data)
```

```
{
    *data = 0;

    if (usart_flag_get(EVAL_COM0, USART_FLAG_RBNE) != RESET) {
        *data = usart_data_receive(EVAL_COM0);
    }

    if (*data == 0) {
        return -1;
    }
    return 0;
}
```

表 3-3. userShellWrite 函数实现

```
void userShellWrite(char data)
{
    while (RESET == usart_flag_get(EVAL_COM0,USART_FLAG_TC));
    usart_data_transmit(EVAL_COM0, (uint8_t) data);
}
```

完成 userShellRead 和 userShellWrite 函数实现以后，将其注册到 shell 结构体中，具体如*表 3-4. 注册 userShellRead 和 userShellWrite 函数*所示。

表 3-4. 注册 userShellRead 和 userShellWrite 函数

```
……
shell.write = userShellWrite;
shell.read = userShellRead;
shellInit(&shell, shellBuffer, sizeof(shellBuffer)/sizeof(shellBuffer[0]));
……
```

3.4. Letter shell 函数调用

Letter shell 的接口不是很多，主要有以下两点需要注意：

1、 在主程序里调用 userShellInit 完成 Letter shell 初始化；
2、 周期性调用 shellTask 任务。

具体如*表 3-5. Letter shell 的函数调用*所示。

表 3-5. Letter shell 的函数调用

```
……
userShellInit();
……
while (1){
    shellTask(&shell);
    delay_1ms(50);
}
……
```

4. Lettle shell 简单应用

4.1. 串口终端软件

对于基于串口移植，letter shell 建议使用 secureCRT 软件，letter shell 中的相关按键映射都是按照 secureCRT 进行设计的，使用其他串口软件时，可能需要修改键值。

打开 secureCRT，进行正确配置之后，将程序烧录到 GD32F450i-EVAL 开发板上，执行结果如*图 4-1. Letter shell 移植成功打印结果*所示。

图 4-1. Letter shell 移植成功打印结果

4.2. 添加自定义功能

Letter shell 支持添加自定义的功能，这里自定义了 3 个功能，具体如*表 4-1. reboot 功能*、*表 4-2. led 控制功能*和*表 4-3. random data 获取功能*所示。

表 4-1. reboot 功能

```
int reboot(int argc, char *agrv[])
{
    printf(" %dparameter(s)\r\n", argc);
    for (char i = 1; i < argc; i++)
    {
        printf("%s\r\n", agrv[i]);
    }
    NVIC_SystemReset();
    return 0;
}
SHELL_EXPORT_CMD(SHELL_CMD_PERMISSION(0)|SHELL_CMD_TYPE(SHELL_TYPE_CMD_
MAIN), reboot, reboot, reboot);
```

表 4-2. led 控制功能

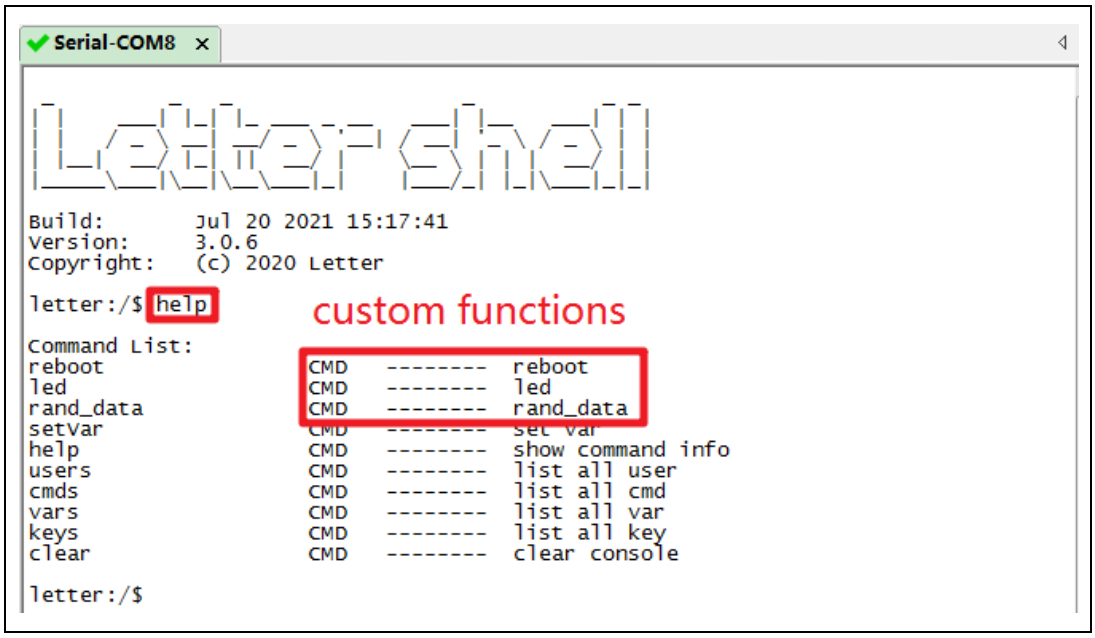
```
int led(int argc, char *agrv[])
{
    uint32_t temp, rtn, i;
    if(argc == 2){
        rtn = sscanf(agrv[1],"%d", &temp);
        if(rtn == 1){
            if(temp == 0){
                gd_eval_led_off(LED2);
                printf("LED2 is off!\r\n");
            }else if(temp == 1){
                gd_eval_led_on(LED2);
                printf("LED2 is on!\r\n");
            }else if(temp == 2){
                gd_eval_led_toggle(LED2);
                printf("LED2 is toggled!\r\n");
            }else if(temp == 3){
                for(i = 0; i < 6; i++){
                    gd_eval_led_toggle(LED2);
                    delay_1ms(400); // cannot be used in interrupt.
                }
                printf("LED is blinked!\r\n");
            }
        }
    }
    return 0;
}
SHELL_EXPORT_CMD(SHELL_CMD_PERMISSION(0)|SHELL_CMD_TYPE(SHELL_TYPE_CMD_
MAIN)|SHELL_CMD_DISABLE_RETURN, led, led, led);
```

表 4-3. random data 获取功能

```
int rand_data(int argc, char *agrv[])
{
    uint32_t temp;
    if(SUCCESS == trng_ready_check()){
        temp = trng_get_true_random_data();
        printf("Randon Data = 0x%08x\r\n", temp);
    }
    return 0;
}
SHELL_EXPORT_CMD(SHELL_CMD_PERMISSION(0)|SHELL_CMD_TYPE(SHELL_TYPE_CMD_
MAIN)|SHELL_CMD_DISABLE_RETURN, rand_data, rand_data, rand_data);
```

用户可以键入 help 查看支持哪些命令，具体如*图 4-2. 键入 help 的打印结果*所示。

图 4-2. 键入 help 的打印结果



4.3. reboot 功能执行结果

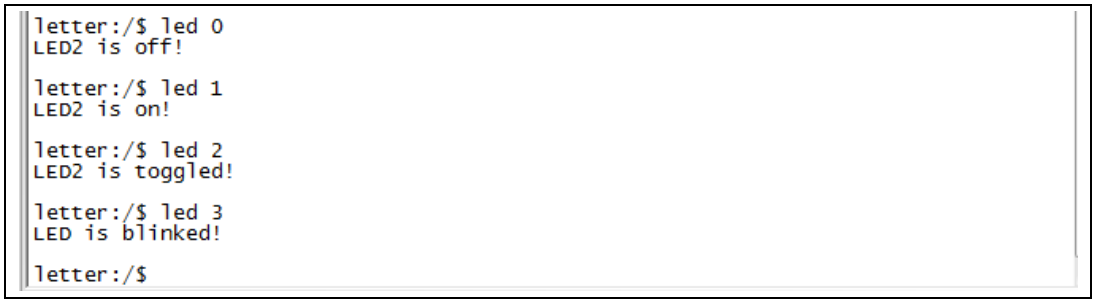
在 secureCRT 软件键入 reboot，可以实现打印和软复位芯片的效果，具体如*图 4-3. 键入 reboot 的打印结果*所示。

图 4-3. 键入 reboot 的打印结果

4.4. led 控制功能执行结果

在 secureCRT 软件键入 led+数字（0~3），可以实现对 LED 的控制效果，具体如*图 4-4. 键入 led 的打印结果*所示。

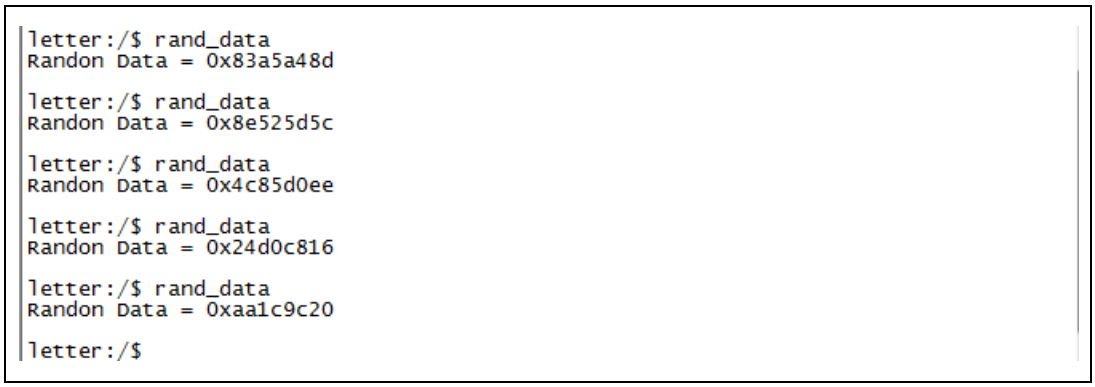
图 4-4. 键入 led 的打印结果



4.5. random data 获取功能执行结果

在 secureCRT 软件键入 rand_data，可以实现获取随机数的效果，具体如*图 4-5. 键入 rand_data 的打印结果*所示。

图 4-5. 键入 rand_data 的打印结果



注意：所有命令均可以按 Tab 实现命令自动补全。

5. 版本历史

表 5-1. 版本历史

版本号.	说明	日期
1.0	首次发布	2021 年 11 月 30 日

Important Notice

This document is the property of GigaDevice Semiconductor Inc. and its subsidiaries (the "Company"). This document, including any product of the Company described in this document (the “Product”), is owned by the Company under the intellectual property laws and treaties of the People’s Republic of China and other jurisdictions worldwide. The Company reserves all rights under such laws and treaties and does not grant any license under its patents, copyrights, trademarks, or other intellectual property rights. The names and brands of third party referred thereto (if any) are the property of their respective owner and referred to for identification purposes only.

The Company makes no warranty of any kind, express or implied, with regard to this document or any Product, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The Company does not assume any liability arising out of the application or use of any Product described in this document. Any information provided in this document is provided only for reference purposes. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. Except for customized products which has been expressly identified in the applicable agreement, the Products are designed, developed, and/or manufactured for ordinary business, industrial, personal, and/or household applications only. The Products are not designed, intended, or authorized for use as components in systems designed or intended for the operation of weapons, weapons systems, nuclear installations, atomic energy control instruments, combustion control instruments, airplane or spaceship instruments, transportation instruments, traffic signal instruments, life-support devices or systems, other medical devices or systems (including resuscitation equipment and surgical implants), pollution control or hazardous substances management, or other uses where the failure of the device or Product could cause personal injury, death, property or environmental damage ("Unintended Uses"). Customers shall take any and all actions to ensure using and selling the Products in accordance with the applicable laws and regulations. The Company is not liable, in whole or in part, and customers shall and hereby do release the Company as well as it’s suppliers and/or distributors from any claim, damage, or other liability arising from or related to all Unintended Uses of the Products. Customers shall indemnify and hold the Company as well as it’s suppliers and/or distributors harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of the Products.

Information in this document is provided solely in connection with the Products. The Company reserves the right to make changes, corrections, modifications or improvements to this document and Products and services described herein at any time, without notice.