DS50 Programming Guide

Version: V1.00

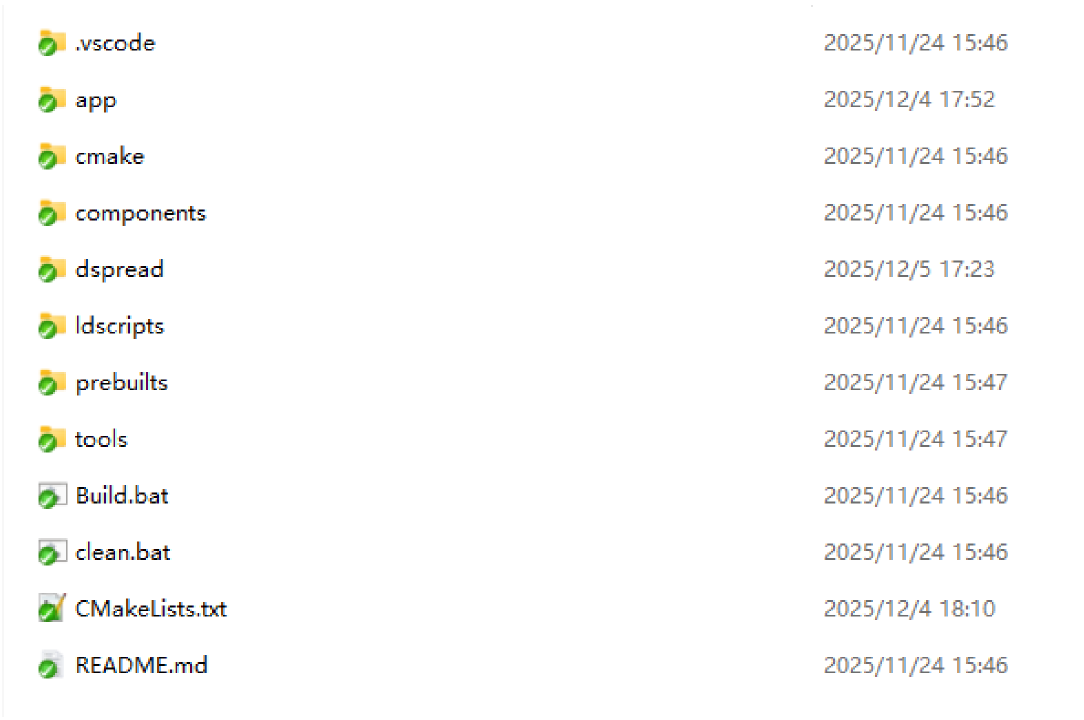
Version update record

Serial number	Version	Date	Modification record	Author	Reviewer
1	V1.00	2025-12-1	Initial version	Echo	LiYongliang

Contents

1. Engineering structure

We provide customers with a comprehensive development and compilation environment, upon which they can directly develop and compile their applications. An introduction to the development environment directory is as follows:



.vscode	2025/11/24 15:46
app	2025/12/4 17:52
cmake	2025/11/24 15:46
components	2025/11/24 15:46
dspread	2025/12/5 17:23
ldscripts	2025/11/24 15:46
prebuilts	2025/11/24 15:47
tools	2025/11/24 15:47
Build.bat	2025/11/24 15:46
clean.bat	2025/11/24 15:46
CMakeLists.txt	2025/12/4 18:10
README.md	2025/11/24 15:46

- **App**: Directory for storing application source code
- **Cmake：** cmake configuration file
- **Components：** C library standard components
- **Ldscripts：** Related link configuration files
- **Out：** Compiled output directory
- **Prebuilts：** Precompile tools

- **Dspread：** Related libraries and header files provided by Dspread
- **Tools：** gcc compiler
- **Build.bat：** Compile script
- **CMakeLists.txt：** Compile configuration file

Users need only to care about the app, the dspread directory, and the cmakelists.txt file.

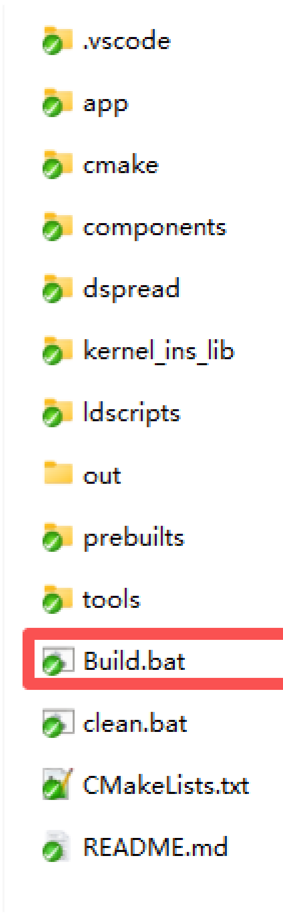
2. Development environment setup

We have integrated offline development environment, and users can develop directly on Windows(Win7/Win8/Win10/Win11).

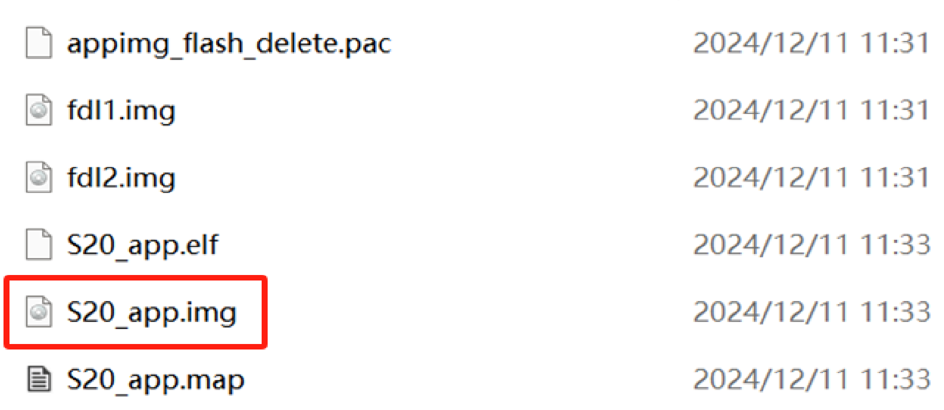
3. Program compilation

Find Build.bat in the project directory and double-click on it

The relevant files after a successful compilation are in the directory “out/appimage_debug/hex”.



.vscode
app
cmake
components
dspread
kernel_ins_lib
ldscripts
out
prebuilts
tools
Build.bat
clean.bat
CMakeLists.txt
README.md



```
appimg_flash_delete.pac        2024/12/11 11:31
fdl1.img                       2024/12/11 11:31
fdl2.img                       2024/12/11 11:31
S20_app.elf                    2024/12/11 11:33
S20_app.img                    2024/12/11 11:33
S20_app.map                    2024/12/11 11:33
```

Before downloading to the terminal, the S20_app.img file needs to be signed. Please refer to *DS50 Smart Tool Operation Guide.docx* for the signing steps.

4. How to debug

Use the coolwatcher tool to capture logs. The advantage of this is that both system-level and application-level logs can be captured simultaneously.
In case of any tricky issues in the future, user can debug and capture logs, then generate a log file, send the log file to the vendor's FAE. This method provides a convenient way for solving internal and external program issues in the future.
Printing interface can refer to *DS50 API Reference Manual.docx*:

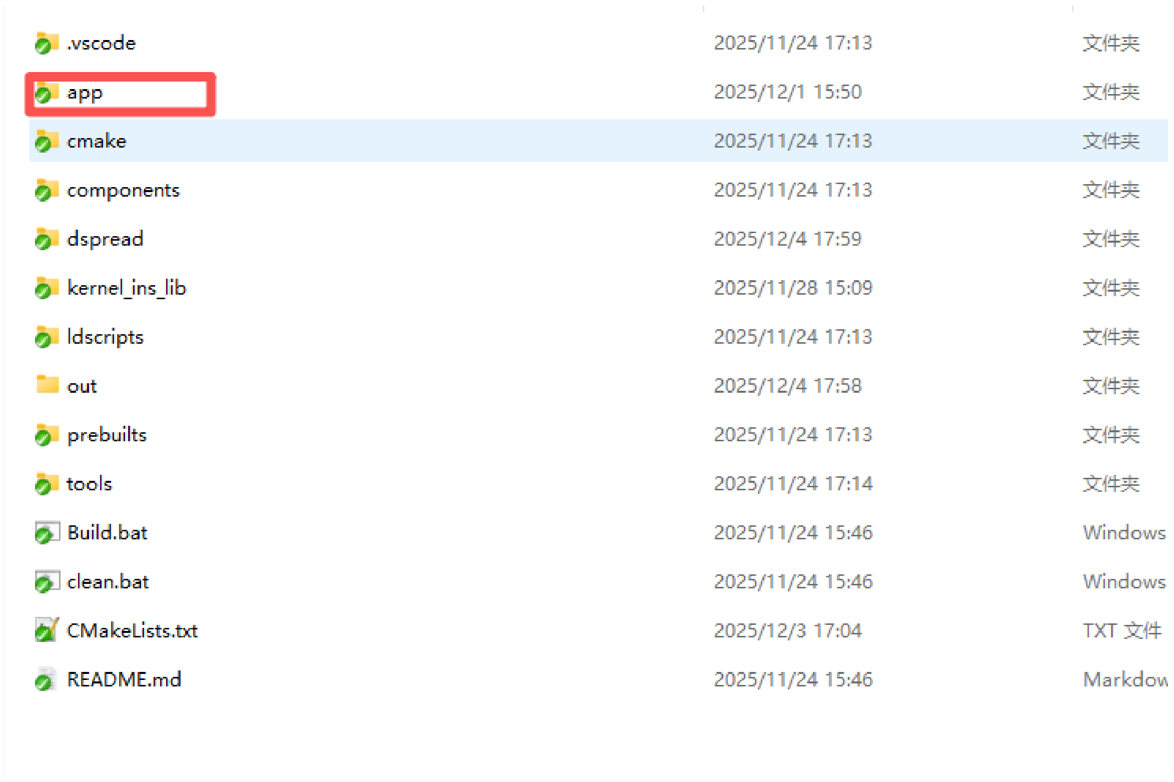
```
void api_log_output(unsigned char level, const char* format, ...);
```

The printed data will be displayed on the coolwatcher tool.

Please refer to the DS5*0 Capture Log.docx* for tool usage.

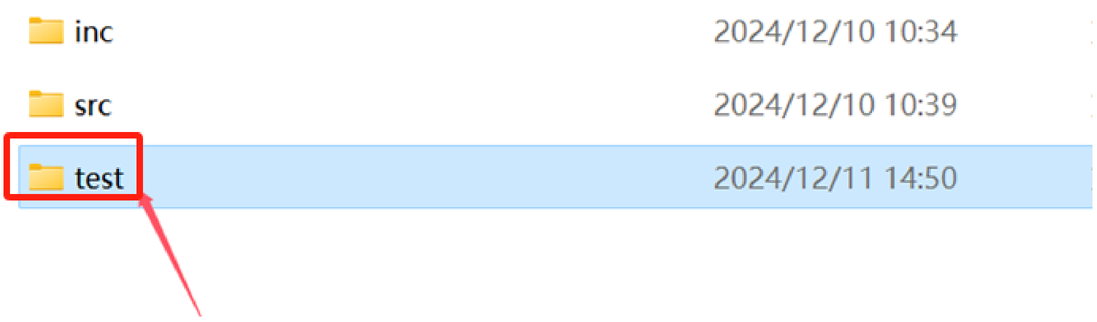
5. How to add code

Add the user code in the app directory



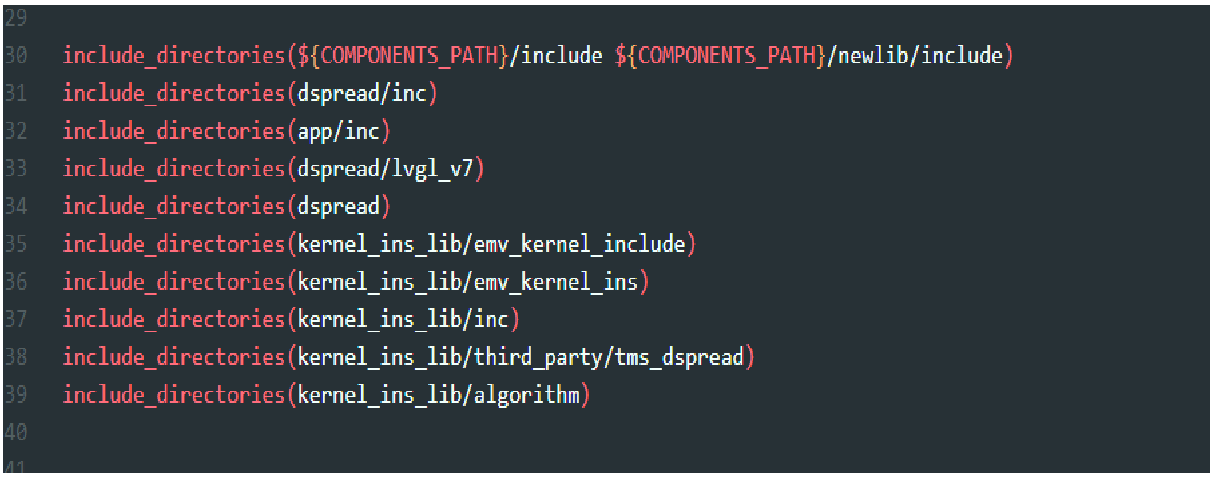
.vscode	2025/11/24 17:13	文件夹
app	2025/12/1 15:50	文件夹
cmake	2025/11/24 17:13	文件夹
components	2025/11/24 17:13	文件夹
dspread	2025/12/4 17:59	文件夹
kernel_ins_lib	2025/11/28 15:09	文件夹
ldscripts	2025/11/24 17:13	文件夹
out	2025/12/4 17:58	文件夹
prebuilts	2025/11/24 17:13	文件夹
tools	2025/11/24 17:14	文件夹
Build.bat	2025/11/24 15:46	Windows
clean.bat	2025/11/24 15:46	Windows
CMakeLists.txt	2025/12/3 17:04	TXT 文件
README.md	2025/11/24 15:46	Markdow

If the user creates another subdirectory in the app directory, you need to modify the cmakelists.txt and reference this subdirectory. For example, you add a test directory that contains some relevant codes.

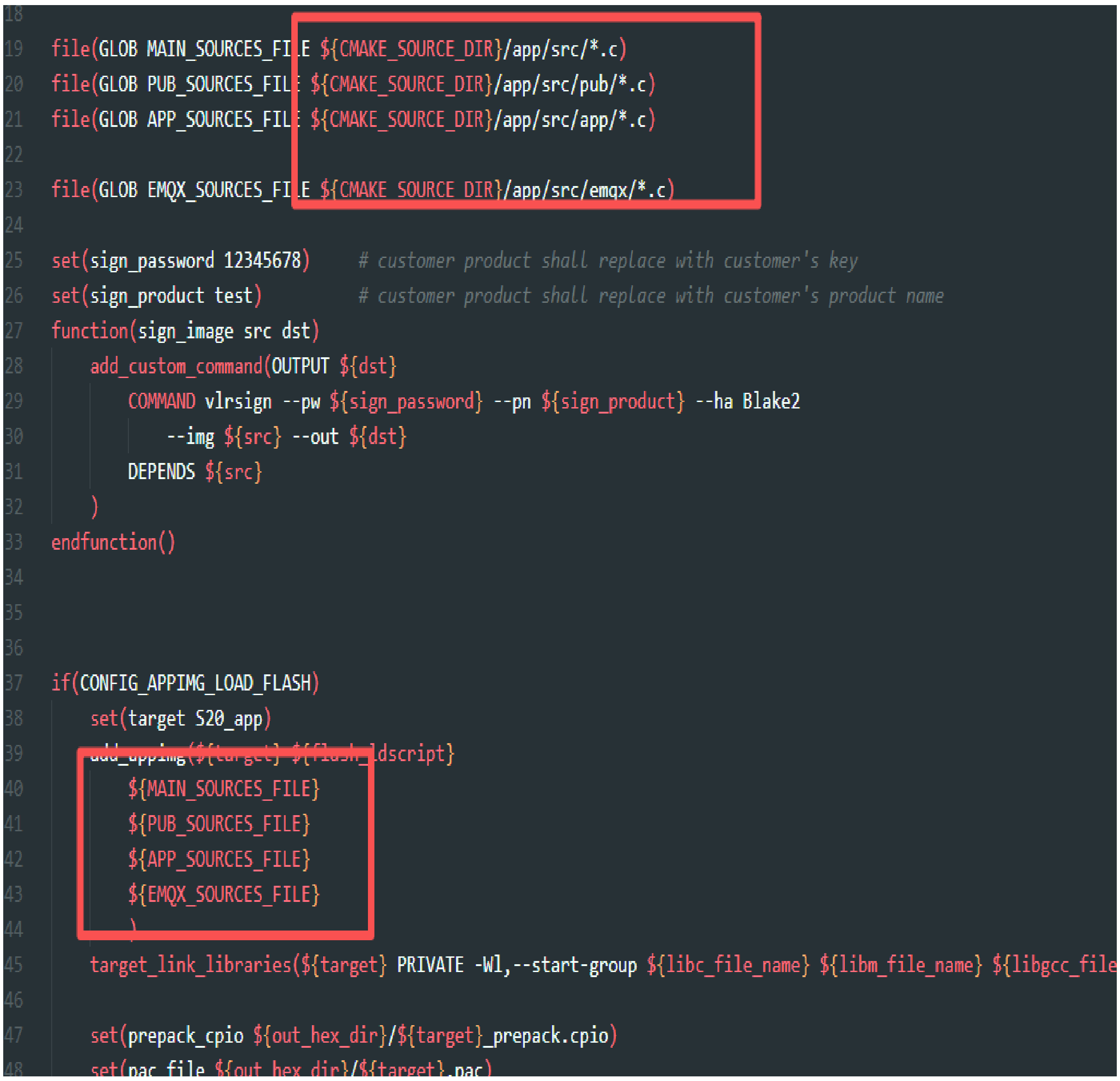


inc	2024/12/10 10:34
src	2024/12/10 10:39
test	2024/12/11 14:50

Modify cmakelists.txt as shown in the following figure:



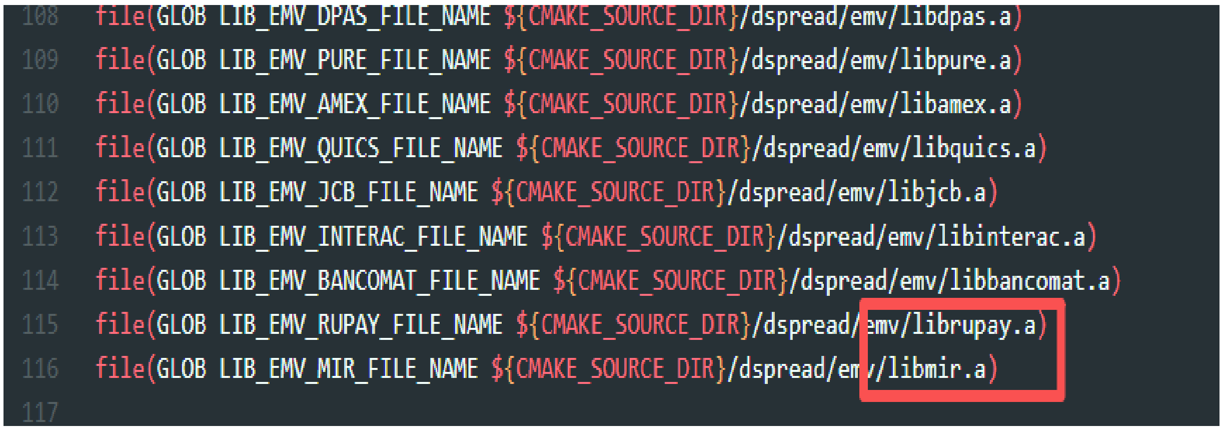
```
include_directories(${COMPONENTS_PATH}/include ${COMPONENTS_PATH}/newlib/include)
include_directories(dspread/inc)
include_directories(app/inc)
include_directories(dspread/lvgl_v7)
include_directories(dspread)
include_directories(kernel_ins_lib/emv_kernel_include)
include_directories(kernel_ins_lib/emv_kernel_ins)
include_directories(kernel_ins_lib/inc)
include_directories(kernel_ins_lib/third_party/tms_dspread)
include_directories(kernel_ins_lib/algorithm)
```



```
file(GLOB MAIN_SOURCES_FILE ${CMAKE_SOURCE_DIR}/app/src/*.c)
file(GLOB PUB_SOURCES_FILE ${CMAKE_SOURCE_DIR}/app/src/pub/*.c)
file(GLOB APP_SOURCES_FILE ${CMAKE_SOURCE_DIR}/app/src/app/*.c)

file(GLOB EMQX_SOURCES_FILE ${CMAKE_SOURCE_DIR}/app/src/emqx/*.c)

set(sign_password 12345678)     # customer product shall replace with customer's key
set(sign_product test)          # customer product shall replace with customer's product name
function(sign_image src dst)
    add_custom_command(OUTPUT ${dst}
        COMMAND vlrsign --pw ${sign_password} --pn ${sign_product} --ha Blake2
            --img ${src} --out ${dst}
        DEPENDS ${src}
    )
endfunction()



if(CONFIG_APPIMG_LOAD_FLASH)
    set(target S20_app)
    add_appimg(${target} ${flash_ldscript}
        ${MAIN_SOURCES_FILE}
        ${PUB_SOURCES_FILE}
        ${APP_SOURCES_FILE}
        ${EMQX_SOURCES_FILE}
        )
    target_link_libraries(${target} PRIVATE -Wl,--start-group ${libc_file_name} ${libm_file_name} ${libgcc_file

    set(prepack_cpio ${out_hex_dir}/${target}_prepack.cpio)
    set(pac_file ${out_hex_dir}/${target}.pac)
```

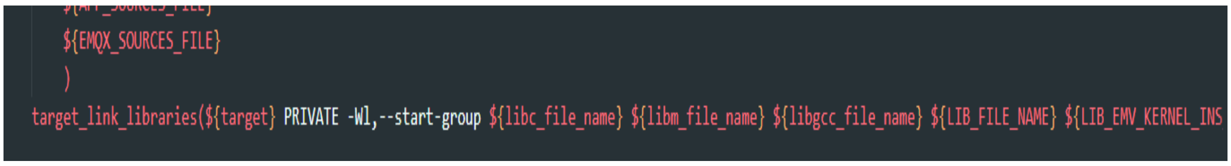
6. How to add a library file?

Define a variable in cmakelists.txt to specify the location of the library.



```
file(GLOB LIB_EMV_DPAS_FILE_NAME ${CMAKE_SOURCE_DIR}/dspread/emv/libdpas.a)
file(GLOB LIB_EMV_PURE_FILE_NAME ${CMAKE_SOURCE_DIR}/dspread/emv/libpure.a)
file(GLOB LIB_EMV_AMEX_FILE_NAME ${CMAKE_SOURCE_DIR}/dspread/emv/libamex.a)
file(GLOB LIB_EMV_QUICS_FILE_NAME ${CMAKE_SOURCE_DIR}/dspread/emv/libquics.a)
file(GLOB LIB_EMV_JCB_FILE_NAME ${CMAKE_SOURCE_DIR}/dspread/emv/libjcb.a)
file(GLOB LIB_EMV_INTERAC_FILE_NAME ${CMAKE_SOURCE_DIR}/dspread/emv/libinterac.a)
file(GLOB LIB_EMV_BANCOMAT_FILE_NAME ${CMAKE_SOURCE_DIR}/dspread/emv/libbancomat.a)
file(GLOB LIB_EMV_RUPAY_FILE_NAME ${CMAKE_SOURCE_DIR}/dspread/emv/librupay.a)
file(GLOB LIB_EMV_MIR_FILE_NAME ${CMAKE_SOURCE_DIR}/dspread/emv/libmir.a)
```

Then, use it in the macro definition CONFIG_APPIMG_LOAD_FLASH



```
    ${EMQX_SOURCES_FILE}
    )
target_link_libraries(${target} PRIVATE -Wl,--start-group ${libc_file_name} ${libm_file_name} ${libgcc_file_name} ${LIB_FILE_NAME} ${LIB_EMV_KERNEL_INS
```

7. How to customize boot logo?

Please provide image files for reference and contact vendor’s FAE.

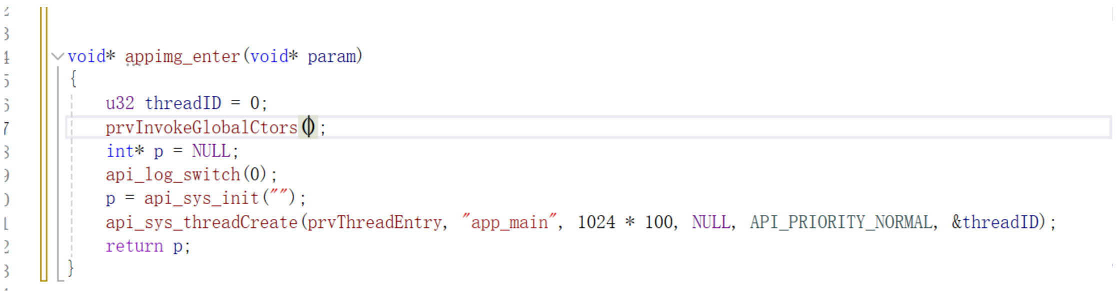
8. How to download DS50_app.img?

Refer to *DS50 Smart Tool Operation Guide.docx*.

9. Other precautions

9.1 Main Function

The main entry of the project is *void* appimg_enter(void* param):*



```
void* appimg_enter(void* param)
{
    u32 threadID = 0;
    prvInvokeGlobalCtors();
    int* p = NULL;
    api_log_switch(0);
    p = api_sys_init("");
    api_sys_threadCreate(prvThreadEntry, "app_main", 1024 * 100, NULL, API_PRIORITY_NORMAL, &threadID);
    return p;
}
```

When developing an application, it is necessary to call api_sys_init first; This interface will initialize many things, including network, flash, audio, etc. The application only needs to develop its own functional modules. Opening a thread with a large stack and developing application functions within it is recommended. You can refer to our application demo. Note: In the event of a program crash, priority should be given to checking the size of the thread stack and whether there are any dead loops. If necessary, the system delay function should be called. In addition, the system and application logs are turned off by default. If you want to view the logs, be sure to call api_log_switch (0); before the api_sys_init function.

9.2 Exit Function

Application exit entry function is void appimg_exit(void)

9.3 PrvInvokeGlobalCtorsFunction

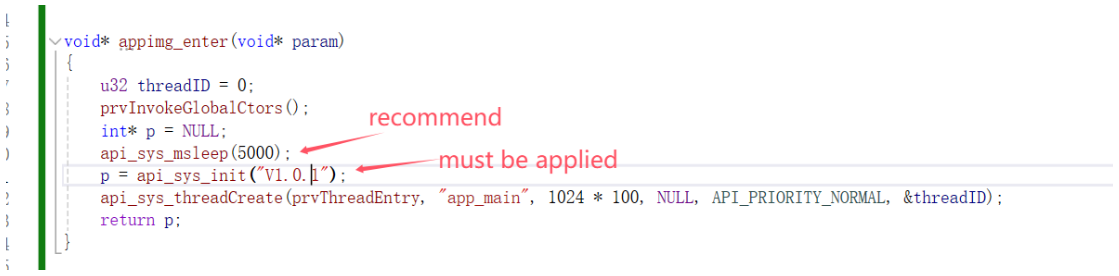
Global constructor

9.4 Important

The above three functions must be implemented, otherwise compilation errors will occur.

9.5 Development Precautions

If the application repeatedly restarts, using the Dspread tool to update the application is impossible. The only solution at this time is to download through BOOTLOADER, which requires disassembling the device and flying out of the boot download cable. Therefore, it is best to add a system delay function (approximately five seconds) to the main entry function during application development, and then remove it when development is complete and ready for release.

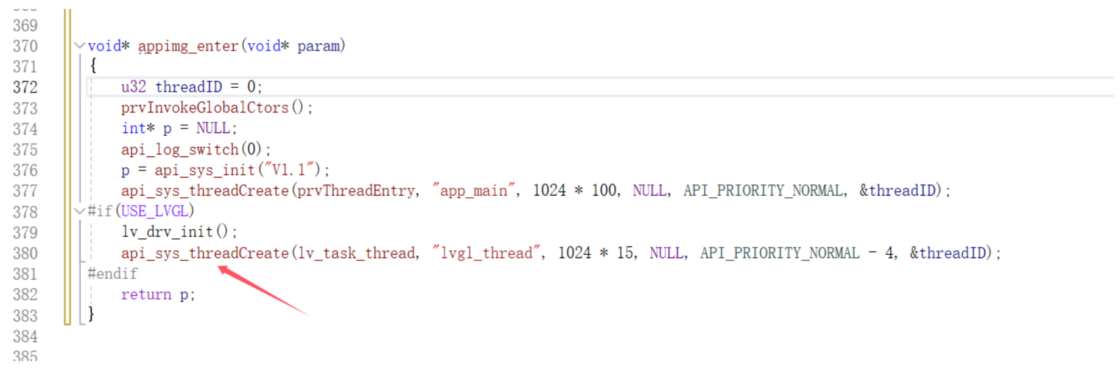


```
void* appimg_enter(void* param)
{
    u32 threadID = 0;
    prvInvokeGlobalCtors();
    int* p = NULL;
    api_sys_msleep(5000);
    p = api_sys_init("V1.0.1");
    api_sys_threadCreate(prvThreadEntry, "app_main", 1024 * 100, NULL, API_PRIORITY_NORMAL, &threadID);
    return p;
}
```

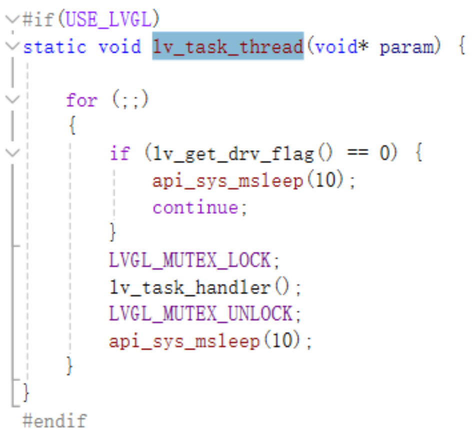
9.6 Using lvgl

LVGL (Light and Versatile Graphics Library) is an open source graphical user interface library designed to provide lightweight, portable, flexible and easy to use graphical user interface solutions.

We have integrated lvgl(using version v7) for our clients. If you want to use lvlg functionality, you need to do the following in the application:



```
void* appimg_enter(void* param)
{
    u32 threadID = 0;
    prvInvokeGlobalCtors();
    int* p = NULL;
    api_log_switch(0);
    p = api_sys_init("V1.1");
    api_sys_threadCreate(prvThreadEntry, "app_main", 1024 * 100, NULL, API_PRIORITY_NORMAL, &threadID);
#if(USE_LVGL)
    lv_drv_init();
    api_sys_threadCreate(lv_task_thread, "lvgl_thread", 1024 * 15, NULL, API_PRIORITY_NORMAL - 4, &threadID);
#endif
    return p;
}
```


```
#if(USE_LVGL)
static void lv_task_thread(void* param) {

    for (;;)
    {
        if (lv_get_drv_flag() == 0) {
            api_sys_msleep(10);
            continue;
        }
        LVGL_MUTEX_LOCK;
        lv_task_handler();
        LVGL_MUTEX_UNLOCK;
        api_sys_msleep(10);
    }
}
#endif
```

9.7 Using Mbedtls

MbedTLS is an open source, portable, easy to use, highly readable SSL library. Common encryption/decryption algorithms, X.509 certificate operations, and TLS/DTLS protocols can be implemented. We provide users with some common encryption and decryption algorithms (not all features of Mbedtls though), use it if you find fit.