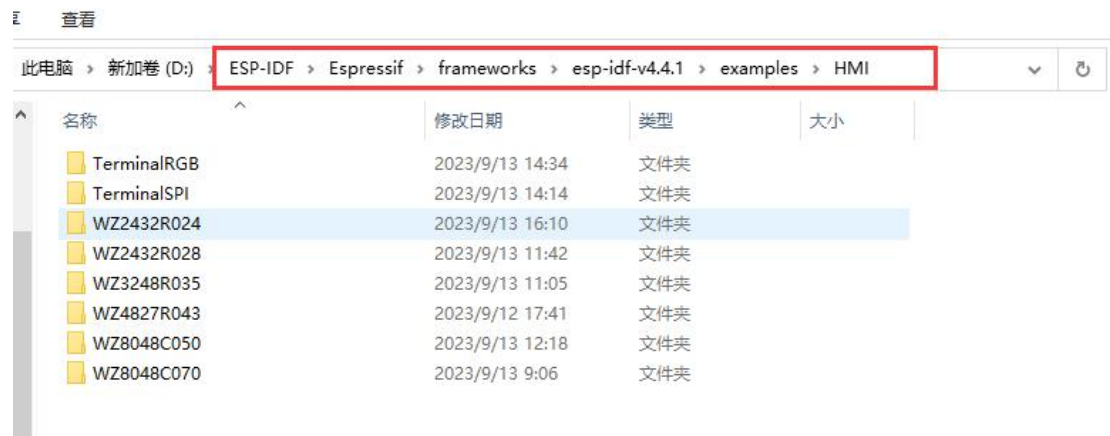


WZ8048C070 Use a tutorial

Place the downloaded project under the IDF directory (as shown below):



Let's first learn at the use of commands :

cd xxx---Moving to the xxx directory, xxx represents the name of the directory, for example: cd example

idf.py set-target esp32s3---Set the target chip for example: esp32s3

idf.py fullclean---Delete the entire build directory, including all the CMake configuration output files.

idf.py clean---It removes the building output files from the building directory and cleans up the entire project..

idf.py menuconfig---Configure the target chip

idf.py build---Compile a private code base

idf.py -p com3 flash---Download the program to the target chip

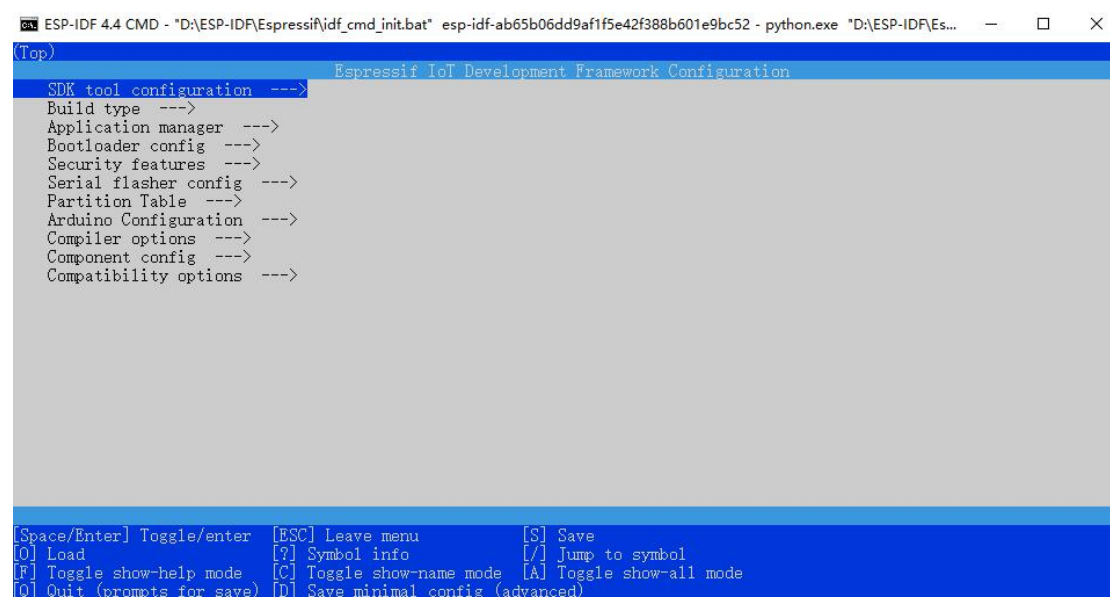
idf.py -p com3 flash monitor---Once compile burn and open monitoring

Now we open the terminal and go to the WZ8048C070 project catalog

```
D:\ESP-IDF\Esspressif\frameworks\esp-idf-v4.4.1\examples\HMI>cd WZ8048C070
D:\ESP-IDF\Esspressif\frameworks\esp-idf-v4.4.1\examples\HMI\WZ8048C070>
```

Now we have to empty the project `idf.py fullclean` once first, and then go into the configuration

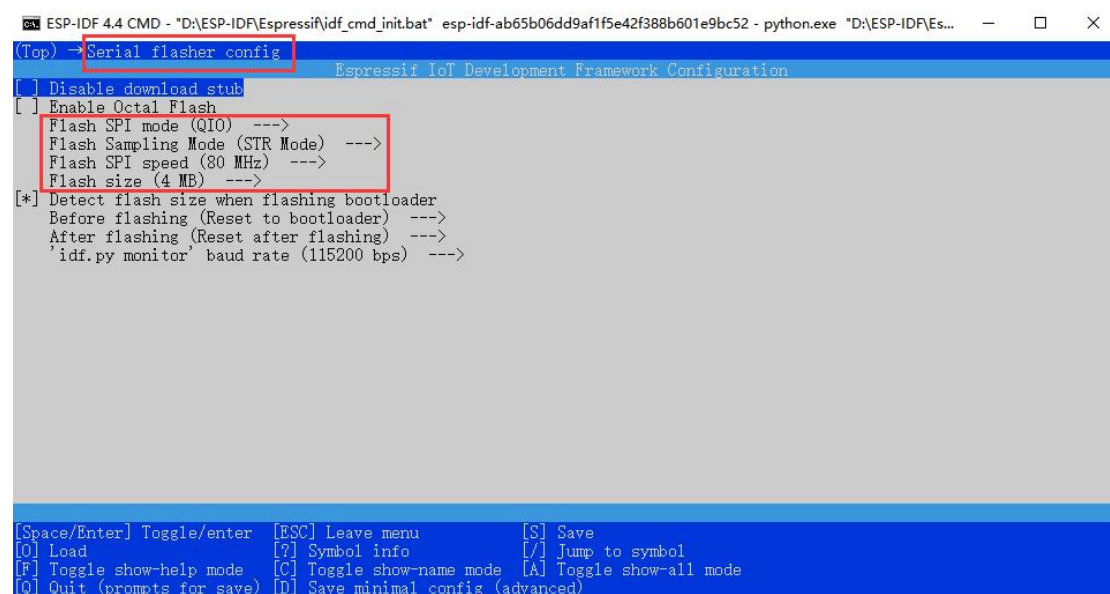
```
D:\ESP-IDF\Esspressif\frameworks\esp-idf-v4.4.1\examples\HMI\WZ8048C070>idf.py fullclean
Executing action: fullclean
Done
```



```
ESP-IDF 4.4 CMD - "D:\ESP-IDF\Esspressif\idf_cmd_init.bat" esp-idf-ab65b06dd9af1f5e42f388b601e9bc52 - python.exe "D:\ESP-IDF\Es...
(Top)
Espressif IoT Development Framework Configuration
SDK tool configuration --->
Build type --->
Application manager --->
Bootloader config --->
Security features --->
Serial flasher config --->
Partition Table --->
Arduino Configuration --->
Compiler options --->
Component config --->
Compatibility options --->

[Space/Enter] Toggle/enter  [ESC] Leave menu          [S] Save
[O] Load                   [?] Symbol info          [/] Jump to symbol
[F] Toggle show-help mode  [C] Toggle show-name mode [A] Toggle show-all mode
[Q] Quit (prompts for save) [D] Save minimal config (advanced)
```

Now modify the options by following the following steps:



```
ESP-IDF 4.4 CMD - "D:\ESP-IDF\Esspressif\idf_cmd_init.bat" esp-idf-ab65b06dd9af1f5e42f388b601e9bc52 - python.exe "D:\ESP-IDF\Es...
(Top) -> Serial flasher config
Espressif IoT Development Framework Configuration
[ ] Disable download stub
[ ] Enable Octal Flash
Flash SPI mode (QIO) --->
Flash Sampling Mode (STR Mode) --->
Flash SPI speed (80 MHz) --->
Flash size (4 MB) --->
[*] Detect flash size when flashing bootloader
Before flashing (Reset to bootloader) --->
After flashing (Reset after flashing) --->
'idf.py monitor' baud rate (115200 bps) --->

[Space/Enter] Toggle/enter  [ESC] Leave menu          [S] Save
[O] Load                   [?] Symbol info          [/] Jump to symbol
[F] Toggle show-help mode  [C] Toggle show-name mode [A] Toggle show-all mode
[Q] Quit (prompts for save) [D] Save minimal config (advanced)
```

```
ESP-IDF 4.4 CMD - "D:\ESP-IDF\Espressif\idf_cmd_init.bat" esp-idf-ab65b06dd9af1f5e42f388b601e9bc52 - python.exe "D:\ESP-IDF\Es...
(Top) -> Partition Table
Espressif IoT Development Framework Configuration
Partition Table (Custom partition table CSV) --->
(huge app.csv) Custom partition CSV file
(0x8000) Offset of partition table
[*] Generate an MD5 checksum for the partition table

[Space/Enter] Toggle/enter [ESC] Leave menu [S] Save
[O] Load [?] Symbol info [J] Jump to symbol
[F] Toggle show-help mode [C] Toggle show-name mode [A] Toggle show-all mode
[Q] Quit (prompts for save) [D] Save minimal config (advanced)
```

```
ESP-IDF 4.4 CMD - "D:\ESP-IDF\Espressif\idf_cmd_init.bat" esp-idf-ab65b06dd9af1f5e42f388b601e9bc52 - python.exe "D:\ESP-IDF\Es...
(Top) -> Component config -> ESP32S3-Specific
Espressif IoT Development Framework Configuration
CPU frequency (160 MHz) --->
Cache config --->
[*] Support for external, SPI-connected RAM
SPI RAM config --->
[ ] Use IRAX tracing feature
[ ] Enable Ultra Low Power (ULP) Coprocessor
[*] Make exception and panic handlers JTAG/OCD aware
[*] Hardware brownout detect & reset
Brownout voltage level (2.44V) --->
Timers used for gettimeofday function (RTC and high-resolution timer) --->
RTC clock source (Internal 150kHz RC oscillator) --->
(1024) Number of cycles for RTC_SLOW_CLK calibration
(2000) Extra delay in deep sleep wake stub (in us)
[ ] No Binary Blobs
[ ] Place RTC_DATA_ATTR and RTC_RODATA_ATTR variables into RTC fast memory segment
[ ] Use fixed static RAM size

[Space/Enter] Toggle/enter [ESC] Leave menu [S] Save
[O] Load [?] Symbol info [J] Jump to symbol
[F] Toggle show-help mode [C] Toggle show-name mode [A] Toggle show-all mode
[Q] Quit (prompts for save) [D] Save minimal config (advanced)
```

```
ESP-IDF 4.4 CMD - "D:\ESP-IDF\Espressif\idf_cmd_init.bat" esp-idf-ab65b06dd9af1f5e42f388b601e9bc52 - python.exe "D:\ESP-IDF\Es...
(Top) -> Component config -> ESP32S3-Specific -> Support for external, SPI-connected RAM -> SPI RAM config
Espressif IoT Development Framework Configuration
Mode (QUAD/OCT) of SPI RAM chip in use (Octal Mode PSRAM) --->
Type of SPIRAM chip in use (Auto-detect) --->
PSRAM Clock and CS IO for ESP32S3 --->
[ ] Cache fetch instructions from SPI RAM
[ ] Cache load read only data from SPI RAM
Set RAM clock speed (80MHz clock speed) --->
[*] Initialize SPI RAM during startup
[ ] Ignore PSRAM when not found
SPI RAM access method (Make RAM allocatable using malloc() as well) --->
[*] Run memory test on SPI RAM initialization
(16384) Maximum malloc() size, in bytes, to always put in internal memory
[ ] Try to allocate memories of WiFi and LWIP in SPIRAM firstly. If failed, allocate internal memory
(32768) Reserve this amount of bytes for data that specifically needs to be in DMA or internal memory

[Space/Enter] Toggle/enter [ESC] Leave menu [S] Save
[O] Load [?] Symbol info [J] Jump to symbol
[F] Toggle show-help mode [C] Toggle show-name mode [A] Toggle show-all mode
[Q] Quit (prompts for save) [D] Save minimal config (advanced)
```

```
ESP-IDF 4.4 CMD - "D:\ESP-IDF\Espressif\idf_cmd_init.bat" esp-idf-ab65b06dd9af1f5e42f388b601e9bc52 - python.exe "D:\ESP-IDF\Esp...
(Top) -> Component config -> LVGL configuration
[ ] Uncheck this to use custom lv conf.h
[ ] LVGL minimal configuration.
  Color settings --->
  Memory settings --->
  HAL Settings --->
  Feature configuration --->
  Font usage --->
  Text Settings --->
  Widget usage --->
  Extra Widgets --->
  Themes --->
  Layouts --->
  3rd Party Libraries --->
  Others --->
  Examples --->
  Demos --->

[Space/Enter] Toggle/enter [ESC] Leave menu [S] Save
[O] Load [?] Symbol info [J] Jump to symbol
[F] Toggle show-help mode [C] Toggle show-name mode [A] Toggle show-all mode
[Q] Quit (prompts for save) [M] Save minimal config (advanced)
```

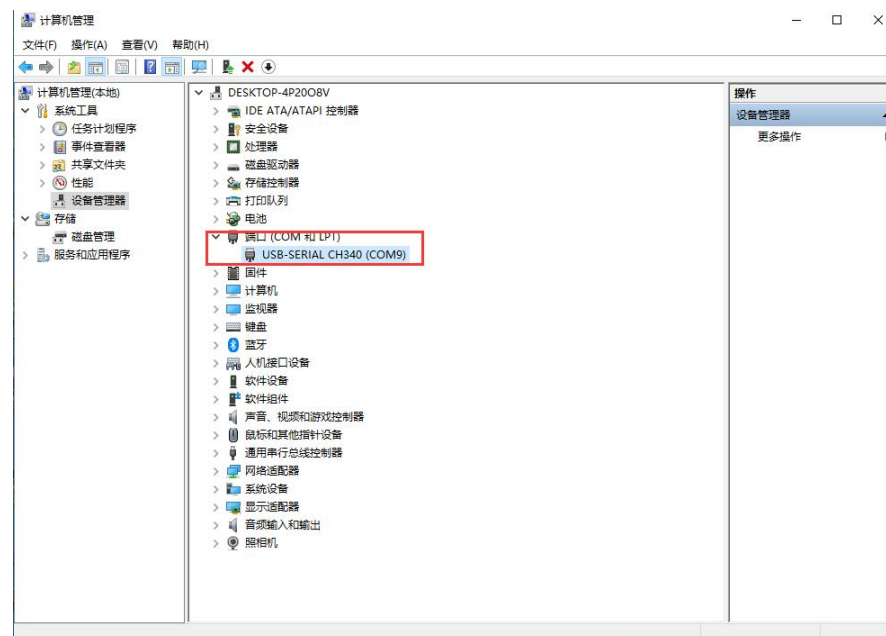
Save the exit after setup, and then execute the **idf.py build**

```
ESP-IDF 4.4 CMD - "D:\ESP-IDF\Espressif\idf_cmd_init.bat" esp-idf-ab65b06dd9af1f5e42f388b601e9bc52 - python.exe "D:\ESP-IDF\Espre...
.../lvgl-3 D:/ESP-IDF/Espressif/frameworks/esp-idf-v4.4.1/components/lwip D:/ESP-IDF/Espressif/frameworks/esp-idf-v4.4.1/examples/HMI/WZ2432R024/main D:/ESP-IDF/Espressif/frameworks/esp-idf-v4.4.1/components/mbdrtls D:/ESP-IDF/Espressif/fr
ameworks/esp-idf-v4.4.1/components/mdns D:/ESP-IDF/Espressif/frameworks/esp-idf-v4.4.1/components/mqtt D:/ESP-IDF/Espres
sif/frameworks/esp-idf-v4.4.1/components/newlib D:/ESP-IDF/Espressif/frameworks/esp-idf-v4.4.1/components/nghttp D:/ESP-
IDF/Espressif/frameworks/esp-idf-v4.4.1/components/nvs_flash D:/ESP-IDF/Espressif/frameworks/esp-idf-v4.4.1/components/o
penssl D:/ESP-IDF/Espressif/frameworks/esp-idf-v4.4.1/components/openthread D:/ESP-IDF/Espressif/frameworks/esp-idf-v4.4
.1/components/partition_table D:/ESP-IDF/Espressif/frameworks/esp-idf-v4.4.1/components/perfmon D:/ESP-IDF/Espressif/fra
ameworks/esp-idf-v4.4.1/components/protobuf-c D:/ESP-IDF/Espressif/frameworks/esp-idf-v4.4.1/components/protocomm D:/ESP-
IDF/Espressif/frameworks/esp-idf-v4.4.1/components/pthread D:/ESP-IDF/Espressif/frameworks/esp-idf-v4.4.1/components/sdm
mc D:/ESP-IDF/Espressif/frameworks/esp-idf-v4.4.1/components/soc D:/ESP-IDF/Espressif/frameworks/esp-idf-v4.4.1/componen
ts/spi_flash D:/ESP-IDF/Espressif/frameworks/esp-idf-v4.4.1/components/spiffs D:/ESP-IDF/Espressif/frameworks/esp-idf-v4
.4.1/components/tcp_transport D:/ESP-IDF/Espressif/frameworks/esp-idf-v4.4.1/components/tcpip_adapter D:/ESP-IDF/Espres
sif/frameworks/esp-idf-v4.4.1/components/tinyusb D:/ESP-IDF/Espressif/frameworks/esp-idf-v4.4.1/components/ulp D:/ESP-IDF
/Espressif/frameworks/esp-idf-v4.4.1/components/unity D:/ESP-IDF/Espressif/frameworks/esp-idf-v4.4.1/components/usb D:/E
SP-IDF/Espressif/frameworks/esp-idf-v4.4.1/components/vfs D:/ESP-IDF/Espressif/frameworks/esp-idf-v4.4.1/components/wear
levelling D:/ESP-IDF/Espressif/frameworks/esp-idf-v4.4.1/components/wifi_provisioning D:/ESP-IDF/Espressif/frameworks/e
sp-idf-v4.4.1/components/wpa_supplicant D:/ESP-IDF/Espressif/frameworks/esp-idf-v4.4.1/components/xtensa
-- Configuring done
-- Generating done
-- Build files have been written to: D:/ESP-IDF/Espressif/frameworks/esp-idf-v4.4.1/examples/HMI/WZ2432R024/build
[7/1614] Generating ../partition_table/partition-table.bin
Partition table binary generated. Contents:
*****
# ESP-IDF Partition Table
# Name, Type, SubType, Offset, Size, Flags
nvs,data,nvs,0x9000,24K,
phy_init,data,phy,0xf000,4K,
factory,app,factory,0x10000,1500K,
*****
[107/1614] Building C object esp-idf/spi_flash/CMakeFiles/ idf_spi_flash.dir/spi_flash_chip_th.c.obj
```


Waiting for the compilation to complete, the following figure interface appears:

```
ESP-IDF 4.4 CMD - "D:\ESP-IDF\Espressif\idf_cmd_init.bat" esp-idf-ab65b06dd9af1f5e42f388b601e9bc52
v_img_desc_t * (aka 'const struct <anonymous> *')
void ui_image_set_property(lv_obj_t * target, int id, uint8_t * val);
./components/UI/ui.c:124:16: warning: unused variable 'target' [-Wunused-variable]
    lv_obj_t * target = lv_event_get_target(e);
At top level:
./components/UI/ui.c:44:13: warning: 'anim_x_cb' defined but not used [-Wunused-function]
static void anim_x_cb(void * var, int32_t v)
[1610/1614] Generating ld/sections.ld
warning: the default selection TFT_SPI_2_HOST (undefined) of <choice TFT_SPI_PORT> (defined at D:\ESP-IDF\Espressif\frameworks/esp-idf-v4.4.1/examples/HMI/WZ2432R024/components/TFT_eSPI/Kconfig:216) is not contained in the choice
[1613/1614] Generating binary image from built executable
esptool.py v3.3-dev
Creating esp32 image...
Merged 2 ELF sections
Successfully created esp32 image.
Generated D:\ESP-IDF\Espressif\frameworks/esp-idf-v4.4.1/examples/HMI/WZ2432R024/build/WZ2432R024.bin
[1614/1614] cmd.exe /C cd /D D:\ESP-IDF\Espressif\frameworks/esp-idf-v4.4.1/examples/HMI/WZ2432R024/build/WZ2432R024\bin
WZ2432R024.bin binary size 0xf04a0 bytes. Smallest app partition is 0x177000 bytes. 0x86b60 bytes (36%) free.
Project build complete. To flash, run this command:
D:\ESP-IDF\Espressif\python_env\idf4.4_py3.8_env\Scripts\python.exe .\..\components\esptool_py\esptool\esptool.py -
(PORT) -b 460800 --before default_reset --after hard_reset --chip esp32 write_flash --flash_mode dio --flash_size det
et --flash_freq 40m 0x1000 build\bootloader\bootloader.bin 0x8000 build\partition_table\partition-table.bin 0x10000 bui
ld\WZ2432R024.bin
run 'idf.py -p (PORT) flash'
```

Perform the **idf.py -p com9 flash**



success!

```
ESP-IDF 4.4 CMD - "D:\ESP-IDF\Espressif\idf_cmd_init.bat" esp-idf-ab65b06dd9af1f5e42f388b601e9bc52
Writing at 0x00072d8e... (34 %)
Writing at 0x00098b7d... (38 %)
Writing at 0x000a28c1... (42 %)
Writing at 0x000a8111... (46 %)
Writing at 0x000adf3a... (50 %)
Writing at 0x000b3d13... (53 %)
Writing at 0x000b9990... (57 %)
Writing at 0x000c0d9c... (61 %)
Writing at 0x000c6345... (65 %)
Writing at 0x000cbcf9... (69 %)
Writing at 0x000d13da... (73 %)
Writing at 0x000d7285... (76 %)
Writing at 0x000dd8a3... (80 %)
Writing at 0x000e605f... (84 %)
Writing at 0x000ee710... (88 %)
Writing at 0x000ff46b... (92 %)
Writing at 0x000fa236... (96 %)
Writing at 0x00100051... (100 %)
Wrote 984224 bytes (410199 compressed) at 0x00010000 in 11.7 seconds (effective 673.4 kbit/s)...
Hash of data verified.
Compressed 3072 bytes to 105...
Writing at 0x00008000... (100 %)
Wrote 3072 bytes (105 compressed) at 0x00008000 in 0.1 seconds (effective 314.3 kbit/s)...
Hash of data verified.
Leaving...
Hard resetting via RTS pin...
Done
```