Gowin FPGA

Quick Start Guide of RISCV Programming

IPUG546-1.0E,4/29/2019

Disclaimer

Revision History

Date	Version	Description
4/29/2019	1.0E	Initial version published.

Contents

List of Figures

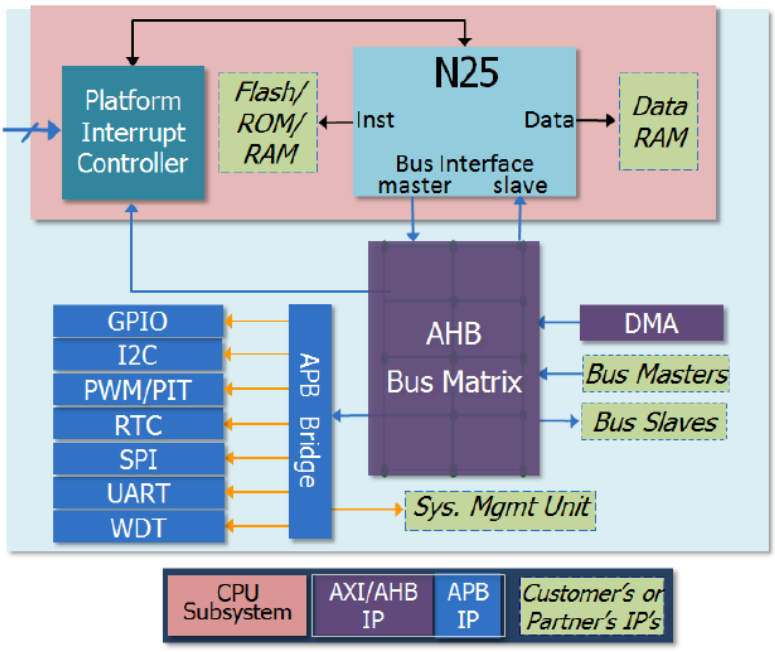
List of Tables

1 Introduction

1.1 AE250 Introduction

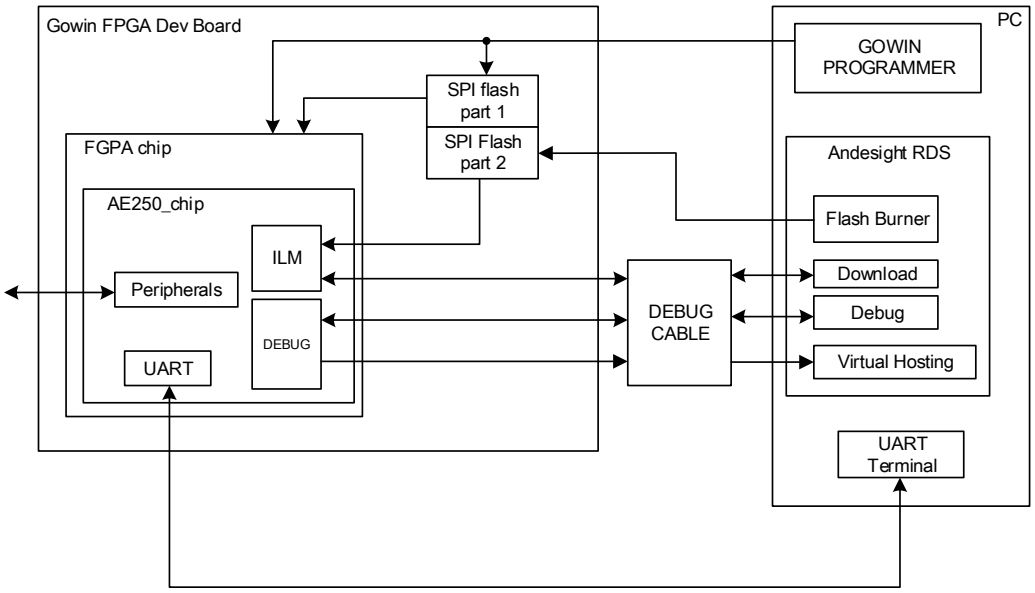
AE250 is a 32 bit RISC-V MCU system; its main structure is shown in Figure 1-1.

Figure 1-1 AE250 Structure Diagram



Based on Gowin FPGA development board, the RISC-V MCU development and debugging system is shown in Figure 1-2.

Figure 1-2 Development and Debugging System Structure Diagram



The FPGA chip on the development board is configured as a AE250 MCU using Gowin Programmer, and then the Debug Cable is connected to perform the embedded program development and debugging with Andesight.

1.2 Preparations

Before using Gowin FPGA and AE250 for development and debugging, the following tools need to be prepared:

1. Gowin GW2A series of FPGA development board;
2. Gowin IDE YunYuan software installation package for configuring and downloading FPGA chip;
3. Andesight installation package for developing and debugging embedded program;
4. Debug Cable for downloading and debugging embedded program. The selectable Debug Cable are:
   a). AICE MINI V2.0, which is supported by default;
   b). Olimex arm tiny h is a third-party Debug Cable. It needs to install libusbK before installing Andesight. Please refer to 2.2 for the installation method;

Note!

- If it needs to output information through UART, a UART to USB cable is needed to prepare;

- Other peripheral equipments required;

1.3 Development and Debugging Steps

The basic steps for developing and debugging RISC-V MCU based on the GW2A-18 development board are as follows:

1. Install softwares. GOWIN IDE YunYuan software is used to configure and generate AE250 RTL design and generate the bitstream file of the design; Andesight software is used to develop and debug embedded programs. Other softwares and drivers for debugging are required;
2. Configure the power supply and download cable of the development board. The bit-stream file of AE250_chip is downloaded to the FPGA chip on the development board using GOWIN PROGRAMMER, and AE250 is running on the development board;
3. Open Andesight, create a new embedded project or open an existing project for encoding, compiling and other operations. Connect the DEBUG CABLE used for AE250 debugging, download the project compilation result to the ILM (instruction memory) in AE250, and start debugging on the chip;
4. During debugging, you can use UART to USB cable to connect the UART interface of AE250_chip to PC, use the built-in serial terminal in Andesight to operate the input and output, or use the Virtual Hosting function to print the output information in the console (see 3.7 in this article). You can use GPIO to connect to LED indicators, buttons, or external pins for input-output operations, as well as I2C, SPI, Ethernet, and other peripherals to choose from;
5. AE250 MCU can connect Flash by SPI , download the compilation result of embedded program to Flash with andesight and Debug Cable, and automatically read and start the embedded program in SPI Flash when the chip is powered on. You can reuse the Flash that saves the FPGA Bitstream, one part to save the FPGA Bitstream, and the other to save the compilation results of embedded programs. This is a practical and economical method;

Please refer to chapter 2RISC-V Debug Cable Connection Guide , chapter 3Andesight RDS User Guide, and chapter 4Project Template and Example for detailed steps.

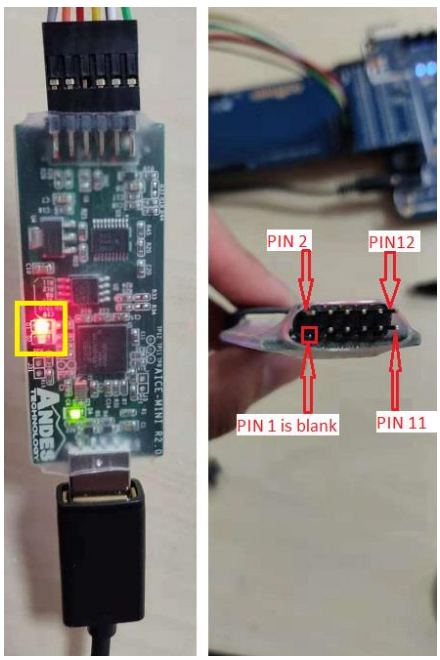
2 RISC-V Debug Cable Connection Guide

There are two RISC-V DEBUG Cables to choose, AICE MINI DEBUG CABLE or OLIMEX ARM-USB-TINY-H CABLE.

2.1 AICE-MINI DEBUG CABLE

The current version of Andesight supports AICE-MINI DEBUG CABLE by default. The appearance is shown in the below figure (left), and pin appearance is shown in the figure (right). It is a 12 pins interface. It is should be noted that the vacant pin in the figure below is PIN 1. When the connection wire is correctly connected and the Andesight is opened, the red LED light marked with yellow box in the figure will go out.

Figure 2-1 AICE MINI DEBUG CABLE and Pins



The pin definition of AICE-MINI DEBUG CABLE is as shown in Figure

2-2. It should be noted that PIN 1 is defined as NC (No Connection), corresponding to vacant PIN 1. VREF needs to connect a 3.3v power pin, and pin GND only needs to connect the pin 3 or the pin 5.

Figure 2-2 AICE Pin Definition

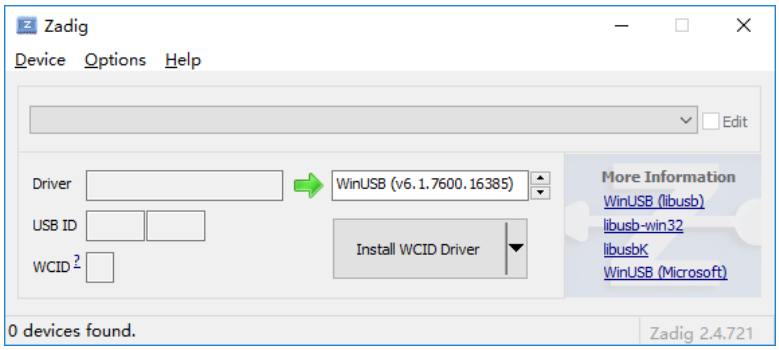
pin #	AICE mini debug cable pin
1	NC
2	TSRST_N
3	GND
4	TTMS
5	GND
6	TCK
7	VREF
8	NC
9	NC
10	TTRST_N
11	TTDO
12	TTDI

2.2 OLIMEX ARM-USB-TINY-H CABLE

The current version of Andesight supports AICE-MINI CABLE by default. If it needs to use Olimex ARM-USB-TINY-H, please install a libusbK before installing Andesight. Zadig is recommended for quick installation. The installation method is as follows:

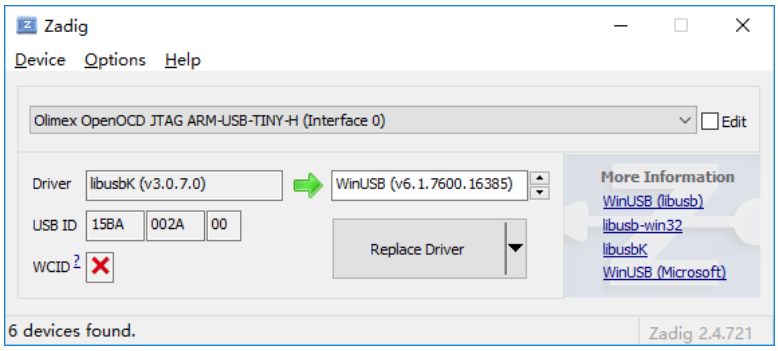
1. Download Zadig to install libusb. Download address: https://zadig.akeo.ie/
2. When the Olimex Cable is plugged into the computer, the Olimex Cable is visible in the device manager and shows no driver (yellow exclamation mark on the device);
3. Double-click zadig.exe to open the tool as shown in Figure 2-3;

Figure 2-3 Open Zading



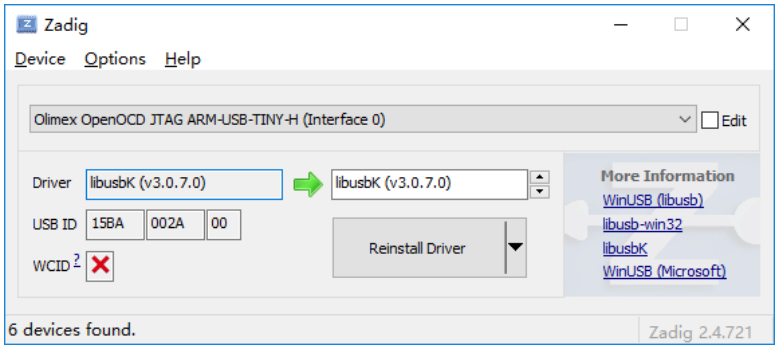
a). Click on Options>List All Devices to see all devices connected to the computer, as shown in Figure 2-4.

Figure 2-4 Device Option



b). Olimex OpenOCD JTAG ARM-USB-TINY-H (Interface 0) and Olimex OpenOCD JTAG ARM-USB-TINY-H (Interface 1) both install libusbK driver, as shown in Figure 2-5.

Figure 2-5 Install libusbK Dirver

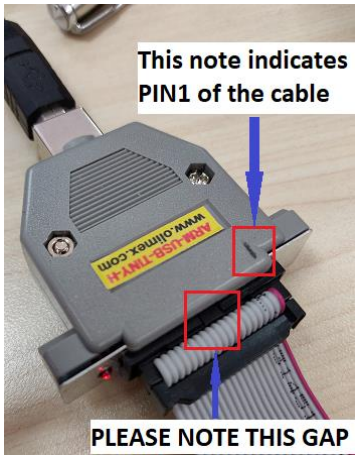


c). After installation, Olimex OpenOCD JTAG ARM-USB-TINY-H can be viewed in Windows device manager. At this time, the device exception will no longer be displayed, indicating that the driver has been installed correctly, as shown in Figure 2-6.

Figure 2-6View libusbK Installation in Device Manage

Olimex Cable appearance is as shown in Figure 2-7.

Figure 2-7 Olimex Cable Appearance



The Olimex Cable pins definition is as shown in Figure 2-8.

Figure 2-8 Olimex Cable Pins Definition

JTAG connector pinout			
PIN #	Signal name	PIN #	Signal name
1	VREF	2	VREF
3	TTRST_N	4	GND
5	TTDI	6	GND
7	TTMS	8	GND
9	TTCK	10	GND
11	TRTCK	12	GND
13	TTDO	14	GND
15	TSRST_N	16	GND
17	NOT CONNECTED	18	GND
19	TARGET SUPPLY	20	GND

Where, VREF shall be connected to 3.3v power pin, only one of the 4/6/8/... /20 GND needs to be connected (J3-2 or J3-20 can be connected.)

3 Andesight RDS User Guide

3.1 Andesight RDS Installation

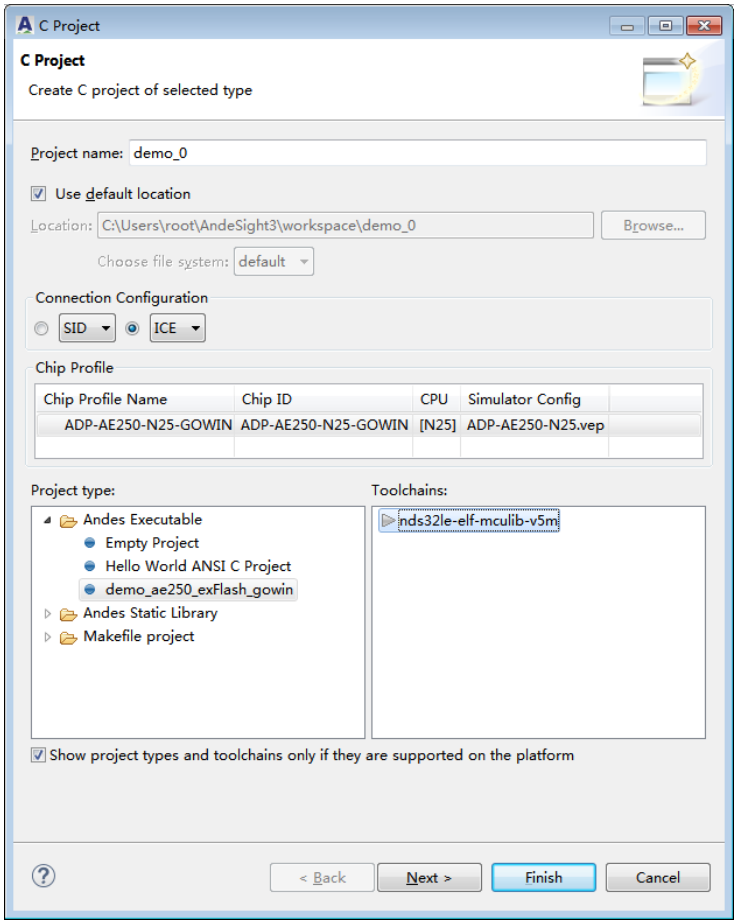
After unpacking the installation package, enter Windows/Disk1 and double-click setup.exe to install it. No special settings are required during installation. During installation, a dialog box will pop up asking whether to install the driver. Please select installation. For installation steps, please refer to AndeSight_RDS_v3.1_Installation_Guide_UM171_V1.1.pdf, which can be found in the installation package.

1. When setting the installation path and workspace path, do not include Chinese characters or space, or it will get a runtime error;
2. The current version of Andesight supports AICE-MINI Cable by default. If it needs to use Olimex ARM-USB-TINY-H, please refer to section 2.2;
3. GOWIN Programmer may be unable to connect to the development board after installing Andesight RDS, which can be fixed by uninstalling GOWIN IDE and reinstalling it;
4. The Public Key file of Andesight RDS has been placed in the installation package. For serial number and certificate files, please contact Gowin Semiconductor Corp;

3.2 Create a New Project

Click "file-> new-> project-> Andes C project->Next" on Andesight interface to enter the configuration interface of New C Project, as shown in Figure 3-1.

Figure 3-1 Create a New Project



The parameters to be configured are as follows:

1. Project name;
2. Save location, the default location is the current workspace;
3. Connection Configuration is set to ICE, indicating that the development board is connected using ICE debug cable. If using the emulator as a test platform, please select SID;
4. For Chip Profile, please select ADP-AE250-N25-GOWIN, which is optimized according to GOWIN FPGA;
5. Project Type, in addition to the Empty Project and the Hello World Project, a project template is built in: demo_ae250_exFlash_gowin. This is a project template containing BSP, bootloader, start.S, and so on;

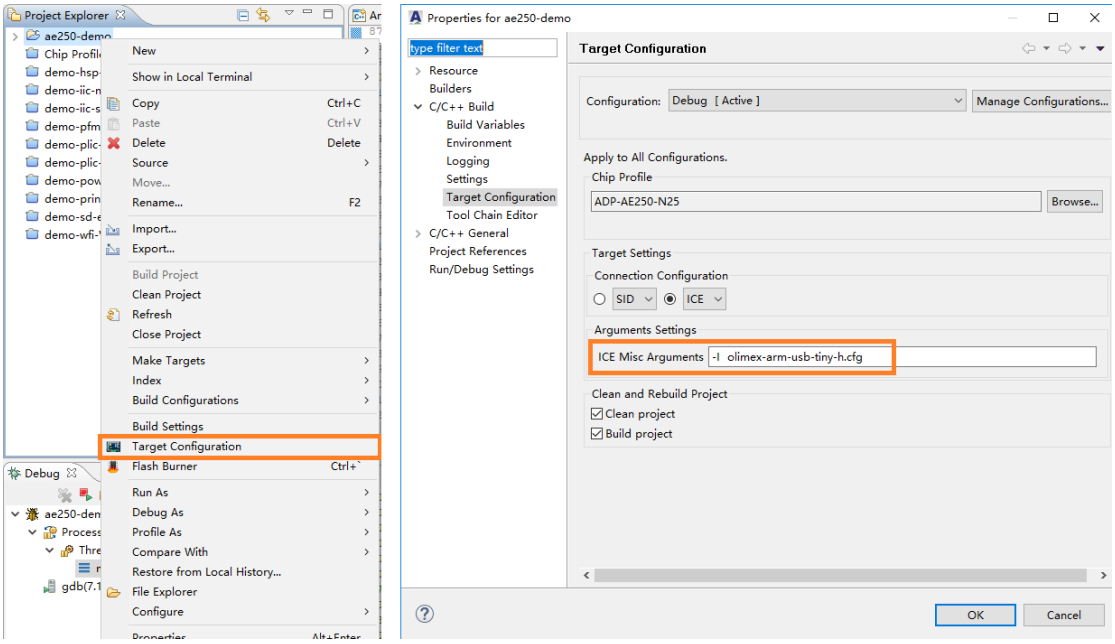
After creating a new Project, right-click on the project name in the Project Explorer, select build Project from the drop-down menu or directly

press Ctrl-B to compile and link the project. Select clean project from the drop-down menu to do make clean operation.

Note!

If using Olimex ARM-USB-TINY-H CABLE, it needs to install the driver first, please rerer to 2.2, and set the target configuration to -i Olimex arm-usb-tiny-h.cfg, as shown in Figure 3-2.

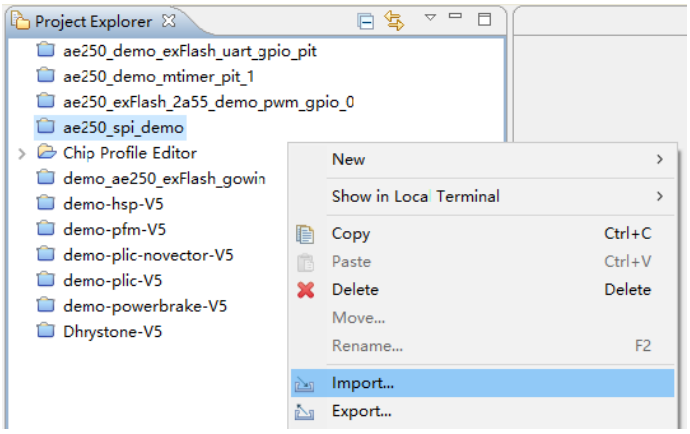
Figure 3-2 Set Debug Cable to Olimex ARM-USB-TINY-H



3.3 Import and Export Project

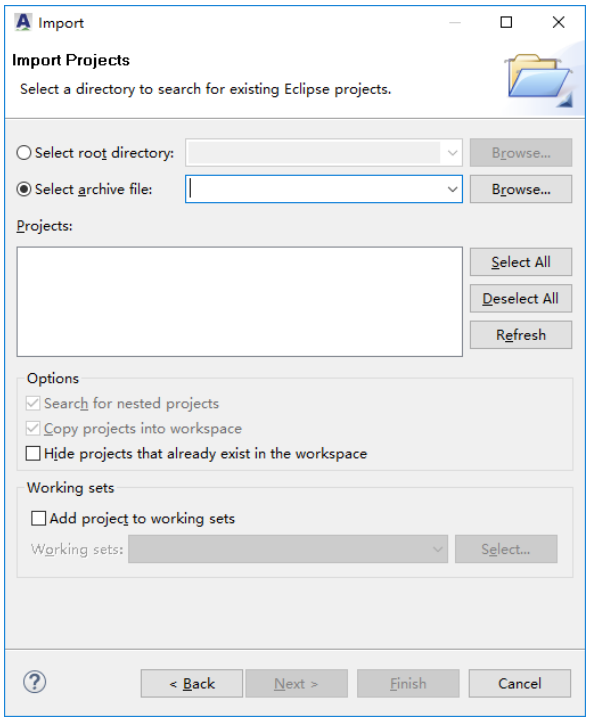
Right click in the space of Project Explorer to select Import or Export to perform the Import/Export operation of the Project, as shown in Figure 3-3.

Figure 3-3 Project Explorer



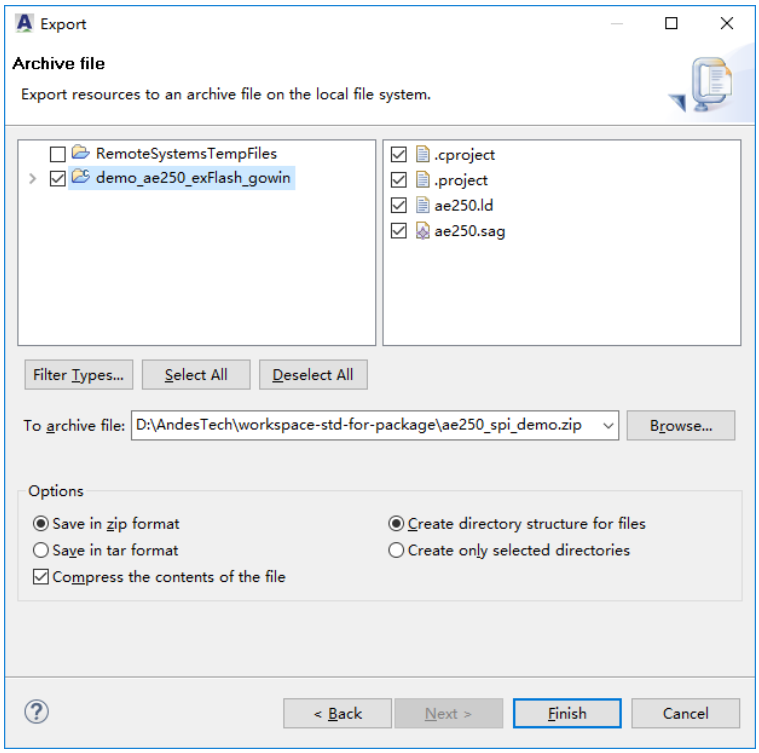
The interface for importing project is shown in Figure 3-4. Click "Import->General->Existing Project into workspace" to Import project. Import project in folder when selecting “Select root directory”; When selecting “Select archive file”, import the project in package.

Figure 3-4 Import Project Interface



The export project is shown in Figure 3-5. Select "Export...->Archive File", it can open the export project interface to select the project to be exported, compression format, save path and other parameters, and then complete the export.

Figure 3-5 Export Project Interface

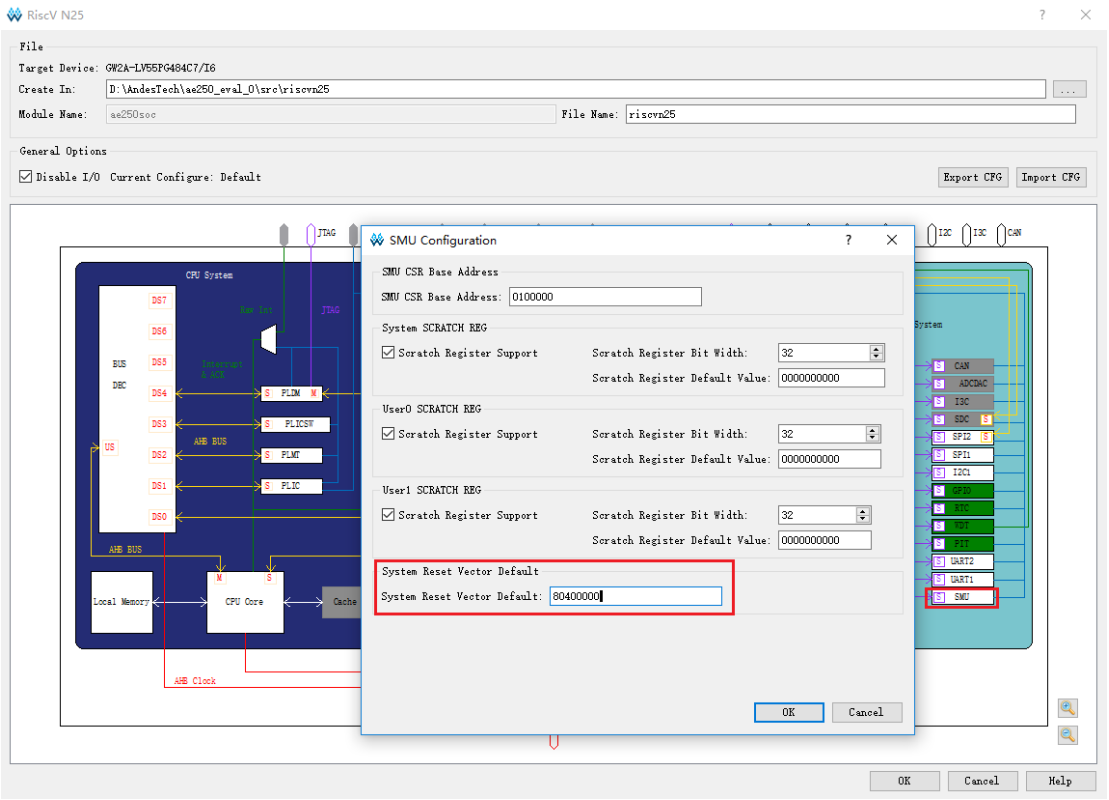


3.4 Download the Program to Flash

AE250 supports starting from flash. SPI interface is used to read the embedded program from flash and store it in the instruction memory ILM, and then the embedded program is executed. The recommended method is to reuse SPI Flash that saves FPGA bitstream, use the first half of Flash to save the FPGA bitstream, and the remaining half to save the binary files of embedded program.

1. Open the IP generator and call out the current AE250 RTL parameter settings. Double-click the SMU to open the SMU setting interface and set the System Reset Vector Default to 0x80400000, as shown in Figure 3-6. Set the space of 0~0x400000 with a total of 4M bytes as the save address of bitstream, starting from 0x400000 as the save address of binary files of embedded program. After modification, it needs to be integrated again.

Figure 3-6 Set System Reset Vector Default



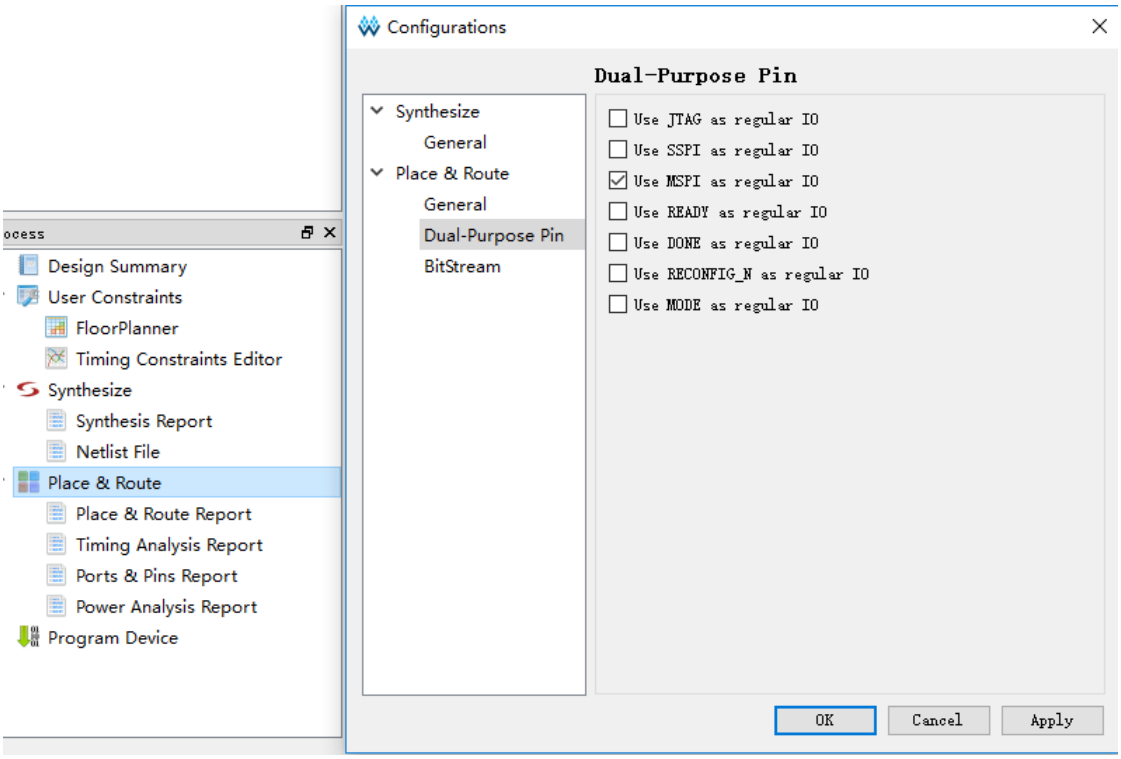
2. In the physical constraint of RTL design, theSPI1 interface should be connected to SPI Flash, and the SPI1 interface should be physically constrained according to the following table. For different FPGA chips, the location of MSPI interface is also different, and the constraint should be specific to the specific situation.

Table 3-1 SPI1 Interface Physical Constraints

AE250 SPI1 Interface	FPGA MSPI Interface
CSN	MCSN
CLK	MCLK
MISO	MSO
MOSI	MSI

3. Reuse MSPI interface as regular IO. In the Process window of Gowin IDE YunYuan software interface, right-click “Place & Route”, select “Configuration” in the pop-up menu, select “Dual Purpose Pin” tab page, check “Use MSPI as regular IO” option and click OK to conduct the correct placement and routing.

Figure 3-7 Set MSPI Interface to Regular IO



4. Modify embedded program parameters settings. First, modify the parameters of bootloader in the linker script. Since the linker script in AE250 embedded program is automatically generated by SAG file, it is modified in the SAG file. Open ae250.sag, find BOOTLOADER and modify it as the value of System Reset Vector Default in RTL design, as shown in Figure 3-8. Then modify config.h. Open src/bsp/config/config.h, find the macro definition of “BUILD_MODE” and modify it to “BUILD_BURN”.

Figure 3-8 ae250.sag bootloader Parameters Setting

```
HEAD 0x00000000
{
    BOOTLOADER 0x80400000
    {
        ADDR __flash_start
        * KEEP (.bootloader)
        LOADADDR __bootloader_lmaend
    }

}
```

Note!

The parameters should be consistent with the value of System Reset Vector Default in the RTL parameter.

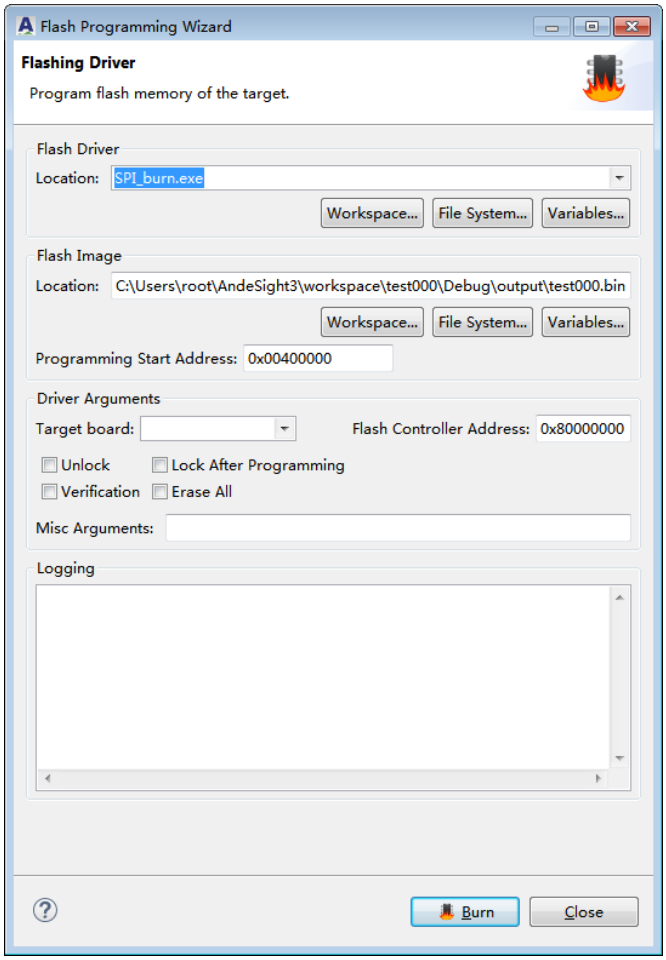
5. Modify the compilation settings, right-click the name of the embedded project, select build settings, select the "Objcopy-> General" tab, and uncheck “Disable”. (Do not auto-generate output file.).

Recompile the embedded program to generate binary files for the embedded project.

The modified RTL design is replaced and rerouted. External Flash mode is used to download it to FPGA and start it. DEBUG CABLE is well connected. Right-click the project name in Andesight, select “Flash Burner”, and the interface appears as shown in Figure 3-9.

Modify the parameters. Set the Flash Controller Address to 0x80000000, which is the total start address of SPI Flash, and set the Programming Start Address to the offset of the embedded program binary file in SPI Flash, which is the value of 0x80400000-0x80000000, which is 0x400000. Click “Burn” and program the binary files of the embedded program into Flash in about 1 minute. AE250 automatically starts to run the programmed embedded program. After the FPGA powers off and powers on again, the FPGA first reads the bitstream in Flash and configures it as AE250 MCU. Then AE250 MCU reads the embedded program in Flash through the SPI interface and automatically starts to run after reading.

Figure 3-9 Flash Burner Interface and Paremeters Setting

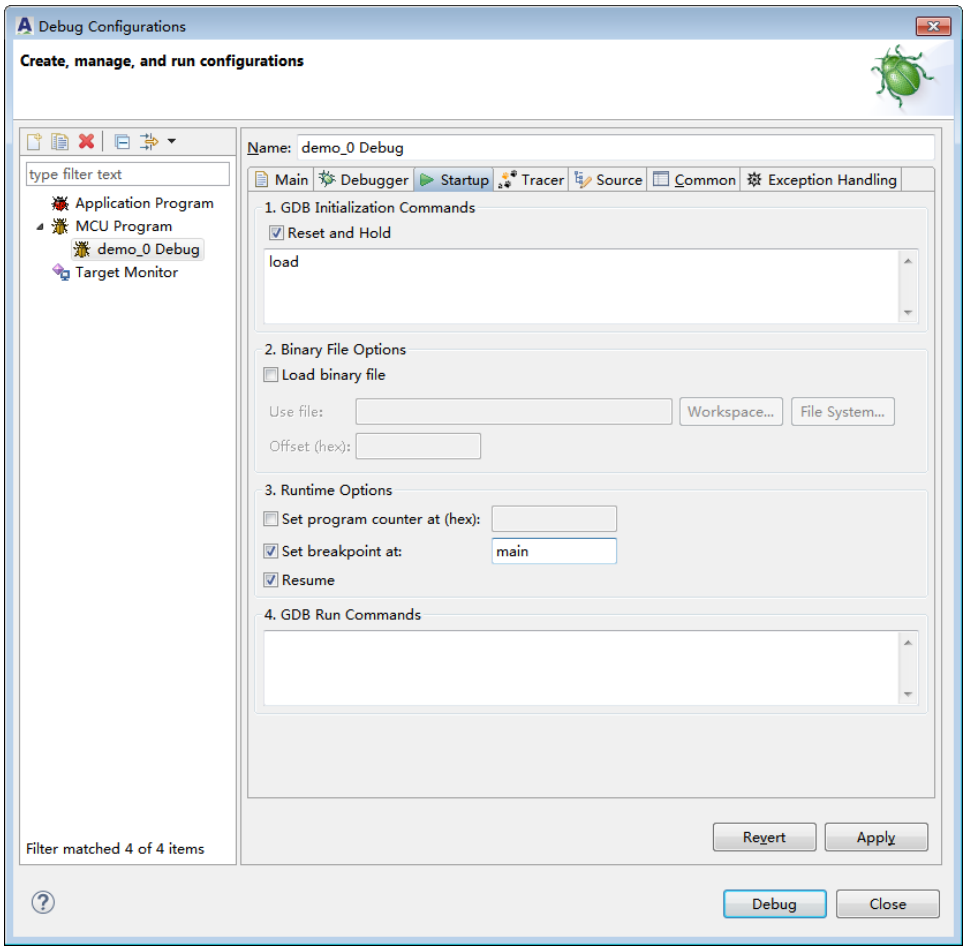


3.5 On-chip Debug

After compilation, the compilation results of the embedded project can be downloaded to the development board for on-chip debug.

Right-click on the project name in the Project Explorer, and select "Debug as-> MCU Program "from the drop-down menu. The first time it performs this operation, a dialog box pops up for setting “Debug Configuration”, as shown in Figure 3-10.

Figure 3-10 Debug Parameters Setting Interface



In the “Startup” tab, check “Reset and Hold” option to stop the program before executing the first instruction. Enter load in the parameter box below this option to download the compilation results of the embedded project into the instruction storage ILM before on-chip debug.

In Runtime Options, check “Set breakpoint at”. Enter a label (such as main) in the input box. It can set a breakpoint at the beginning of the main

function. Check “Resume”, it will start the continuous operation directly after entering on-chip debug.

When entering on-chip debug function, it automatically goes to the debug view mode and an area as shown in Figure 3-11 will be displayed. This area is the operation area for on-chip debug. In the red box are some shortcut buttons for debug. From left to right, they are: restart to DEBUG, continue to run, suspend, end, disconnect link, link to one process (the gray meaning temporarily unavailable), enter the function (step into), the execution of an instruction (step over), jump out of the function (step return, the gray meaning temporarily unavailable), a single instruction operation mode (instruction stepping mode, in this mode, each time it runs a risc - v assembly instruction, otherwise each time it runs a C language statement).

Like other Eclipse software, double click the left on the line number in the code text to quickly set breakpoints or cancel breakpoints, and right click in the code text to select run to line from the pop-up menu.

Figure 3-11 Debug Introduction

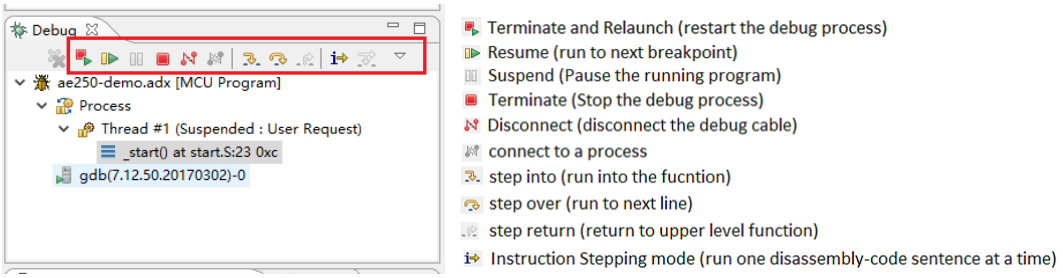
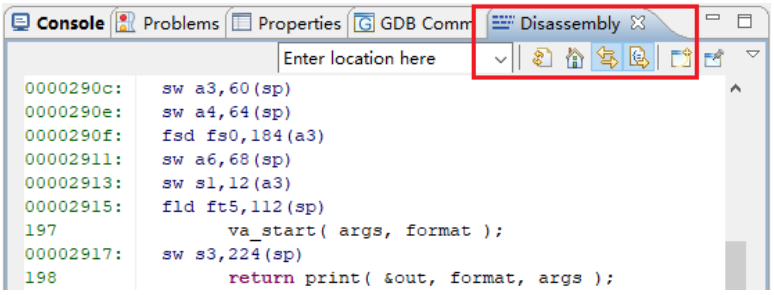


Figure 3-12 is an assembly statements window that displays the contents of assembly instructions running in real time in ILM.

Figure 3-12 Assembly Instruction Code Window

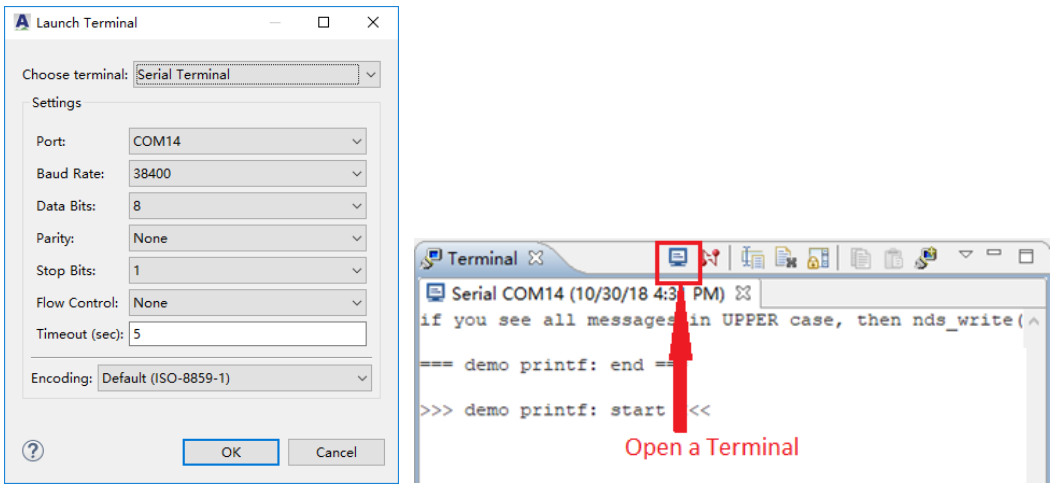


3.6 Andesight Built-in Serial Terminal Usage

Figure 3-13 shows the UART Terminal built in Andesight and the

setting interface. If it needs to use, please click "window-> Show view-> Terminal" in the top menu to open the Terminal window, and then click "open a terminal" to create a new serial terminal. After setting the port number (which can be viewed in the hardware manager), baud rate and other parameters, click “OK” to start using.

Figure 3-13 Andesight Built-in Serial Terminal



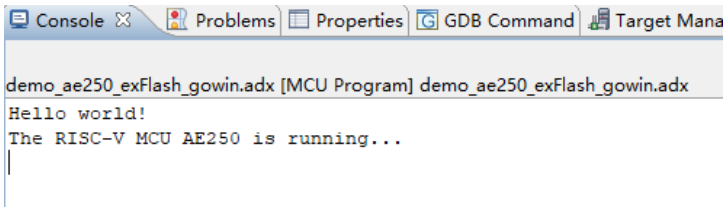
For the usage, please refer to the document: AndeSight_RDS_v3.1_User_Manual_UM170_V1.0.pdf, which can be found in the doc path of the installation directory.

3.7 VIRTUAL HOSTING

Andesight provides a way to export information -- Virtual Hosting. When debugging on chip, printf information are output through the console, as shown in Figure 3-14. It is suitable for the condition that it is inconvenient to connect to UART. To use it, open the Build Settings interface and check the Virtual Hosting option in the "Andes C linker-> General" tab. Then open the src/config/config.h file, open the macro definition #define VIRTUAL_HOSTING, and recompile it to use Virtual Hosting when debugging on chip.

It should be noted that Virtual Hosting is only supported debugging on chip. Virtual Hosting outputting information is not supported when an embedded program is running alone on the development board.

Figure 3-14 Virtual Hosting Outputting Printed Information in Console Window



4 Project Template and Example

4.1 Project Template

When creating a new project, the optional project templates are "empty project", "Hello world project", and "demo_ae250_exFlash_gowin". The demo_ae250_exFlash_gowin is recommended. This is a project template that is optimized and configured based on the GOWIN FPGA, with built-in BSP (Board Supporting Package) and some other necessary files for quick board running and debugging.

The key files in the project template are as follows:

1. SRC/BSP /ae250/ae250.h: This file contains the system clock definition, peripheral register definition, and interrupts source number definition. The clock definition must be consistent with the AE250 parameters configuration.
2. src/bsp/ae250/ae250.c: The reset_handler function is the entry to start the embedded program. In the entry, UART initialization is performed before the main function is executed. The required UART port is selected and the required baud rate is configured according to the parameters configuration of AE250.
3. SRC/BSP /config/config.h: This file contains the macro definition that control compilation method.
4. Where #define BUILD_MODE can be defined as BUILD_LOAD or BUILD_BURN,
5. BUILD_LOAD means that the program is loaded directly into the instruction storage ILM, generally used for DEBUG.
6. BUILD_BURN means that the program is downloaded into SPI Flash, and it will be run after reading the program into ILM from SPI Flash after power on. It is suitable for distribution program.

7. The second macro definition is #define VIRTUAL_HOSTING. If the virtual hosting mode is used for debugging, this macro definition needs to be opened. At this time, printf is no longer realized through UART, and the printed information will be displayed in the Console window.
8. Start. S: The start file written in assembly language.
9. Loader. C: bootloader file, which is used to start SPI Flash.
10. ae250.sag: Sag is the Scattering-and-Gathering format script defined by Jing Xin Technology. It's used to generate linker script. It should be noted that the memory map parameters in ae250.sag need to be consistent with those in AE250.
11. SRC/BSP /driver: This directory contains two folders, ae250 is AE250 driver code, include is the driver function call interface.
12. SRC/BSP /lib: It contains two files. In printf.c, the form of subfunction in C standard library is redefined to output printf information through UART. In read.c, there is a simple function to read input information through UART.

4.2 Project Example

After the completion of installation, it can find several basic project examples in the demo folder of the installation directory, which can be loaded into Andesight for trial, debugging and redevelopment by importing the project. The project examples are as follows:

1. Demo-hsp-v5: It demonstrates the function of hardware stack protection in AE250. To enable this function, please check "Andes StackSafe Extension" when setting AE250 parameters.
2. Demo-pfm-v5: It demonstrates the function of the Performance Monitor;
3. Demo-powerbrake: It demonstrates the function of PowerBrake. PoweBrake can control power consumption through software by slowing down the execution speed of instructions rather than lowering the clock frequency to slow down the running speed of programs. To enable this function, check "Andes PowerBrake Extension" option when setting the AE250 parameters.
4. Demo-plic -novector-V5: It demonstrates the response of the interrupt controller to interrupts in non-vector mode and provides demonstrations of the machine timer and pit timer.
5. Demo-plic -V5: It demonstrates the response of interrupt controller to interrupts in vector mode. If using vector mode interrupts , please check "Andes Vectored plic Extension" when setting AE250 parameters.
6. Dhrystone-v5: It is a Dhrystone program for AE250 adaptation.


GOWIN
PROGRAMMING FOR FUTURE