ARM® Cortex®-M
32-bit Microcontroller

NuTool – CodeGenerator
User Manual

For additional information or questions, please contact: Nuvoton Technology Corporation.

www.nuvoton.com

Table of Contents

List of Figures

1 Introduction

The **NuTool - CodeGenerator** is used to generate initialization projects of Nuvoton NuMicro® Family. Its features are listed below:

- **Configuring peripherals**: All the supported peripherals are collected and listed in the Peripheral Functions TreeView. The user can choose them in the tree and configure the detailed settings in the FormView.
- **Configuring pins by PinConfigure**: The PinConfigure tool is integrated into the tool. The user can configure GPIO multi-functions intuitively and efficiently.
- **Configuring clocks by ClockConfigure**: The ClockConfigure tool is integrated into the tool. The user can configure system and peripheral clocks intuitively and efficiently.
- **Generating code**: After doing the above actions, the user can create an initialization project. The project can be opened and built in Keil, IAR or NuEclipse.

Through the application, the user can create initialization projects of the NuMicro® Family correctly and handily.

2 Starting to Use the NuTool - CodeGenerator

2.1 System Requirements

The following table lists system requirements for the user to run **NuTool - CodeGenerator**.

	Minimum Requirements	Recommended Specifications
Operating System	Windows® 7 with latest service pack	Windows® 10 with latest service pack
Input	Keyboard and mouse required	

3 Supported Chips

To see the list of supported chips, please refer to **Supported_chips.htm** in the folder of user manual. The alternative way is to click the **Read User Manual** button on the toolbar.

3.1 Running the NuTool - CodeGenerator

To run **NuTool - CodeGenerator**, double-click the **NuTool - CodeGenerator.exe**.

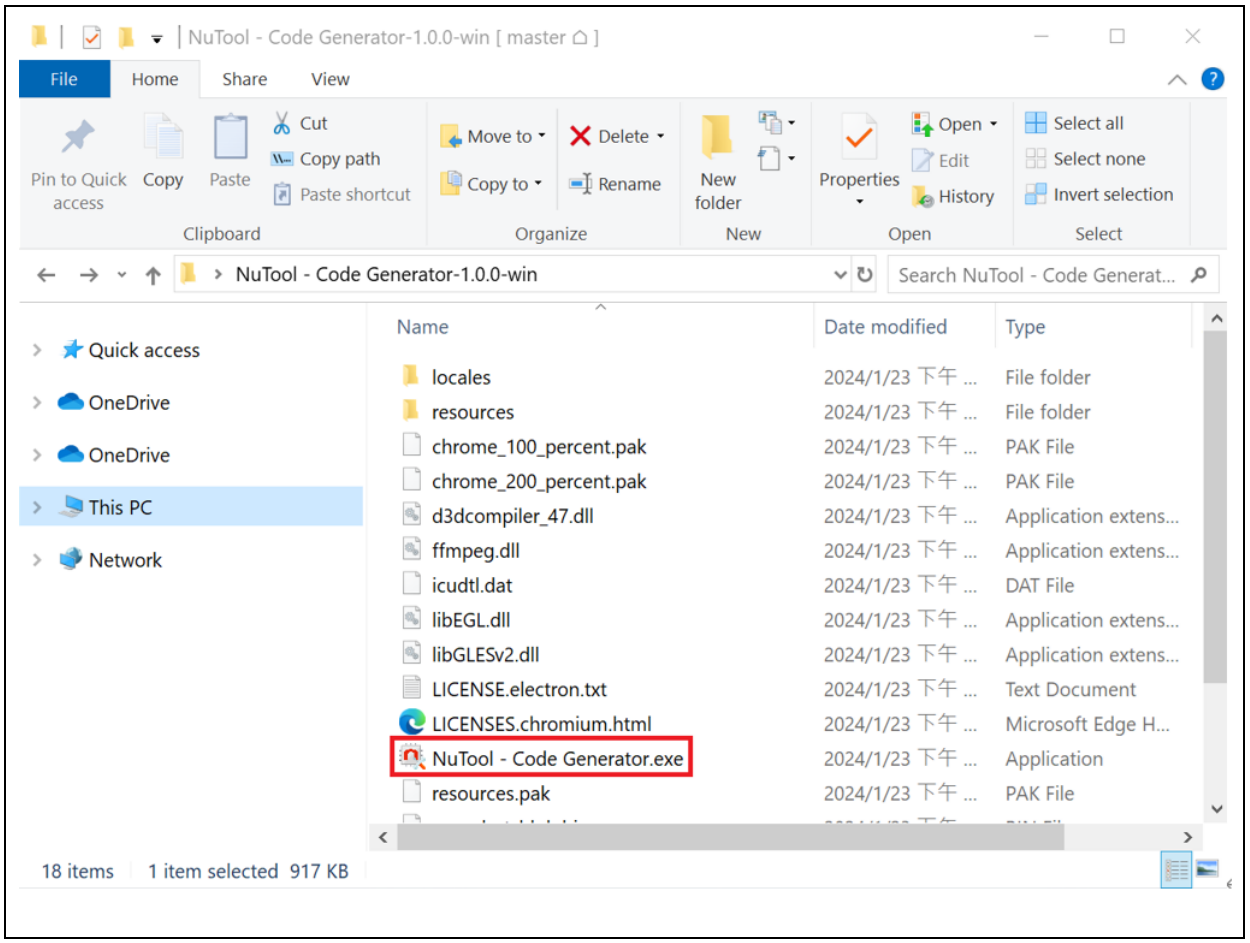

Figure 3-1 NuTool - CodeGenerator.exe and Related Folders

4 Configuring Peripherals

To create a feasible iniailization project, it is recommendated to **configure peripherals, pins and clocks in order**. Firstly, click the **Configure Peripheral** button on the toolbar.

4.1 GUI Overview

The CodeGenerator Window includes a variety of components to configure peripherals. The name of each component is described in Figure 4-1.

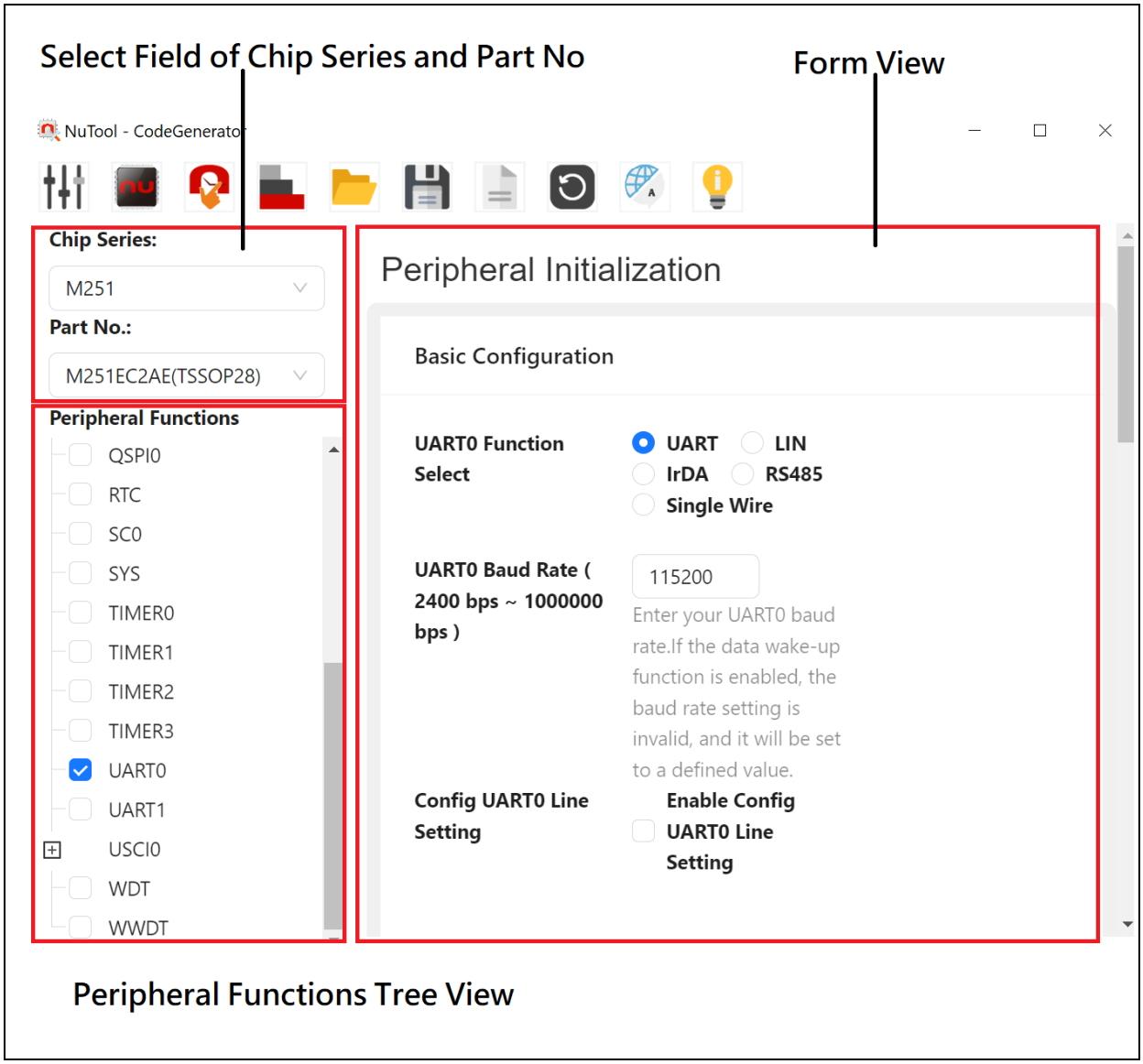

Figure 4-1 CodeGenerator Window

4.2 Select Field of Chip Series and Part No.

The user can select the expected chip series and part number from the upper-left select field (referring to Figure 4-2). If the select field is hidden, please click the **Switch the Left Panel** to show it. In the following discussion, the chip series and part number selected are M251 and M251LC2AE, respectively.

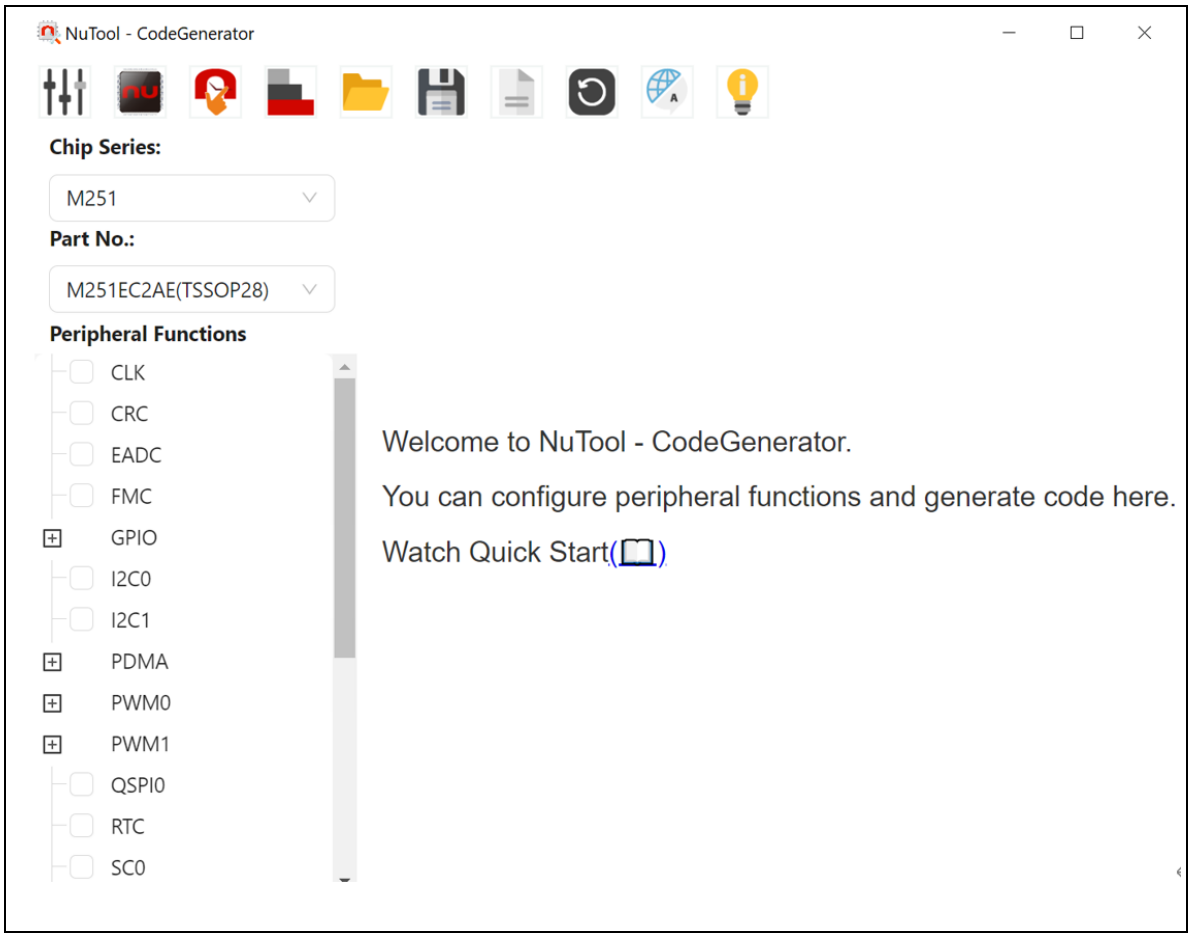

Figure 4-2 Selecting Part Number

4.3 Peripheral Functions TreeView and FormView

Within the Peripheral Functions - TreeView, the user can select which peripheral function to configure. Each time a peripheral function is checked in the check boxes, the corresponding settings shown in the FormView is ready to be configured more detailedly. Besides, for some peripheral functions (e.g., USCI0_I2C, USCI0_SPI, and USCI0_UART) only one is permitted to select at one time. These peripheral functions are grouped under a module highlighted with the steel blue color.

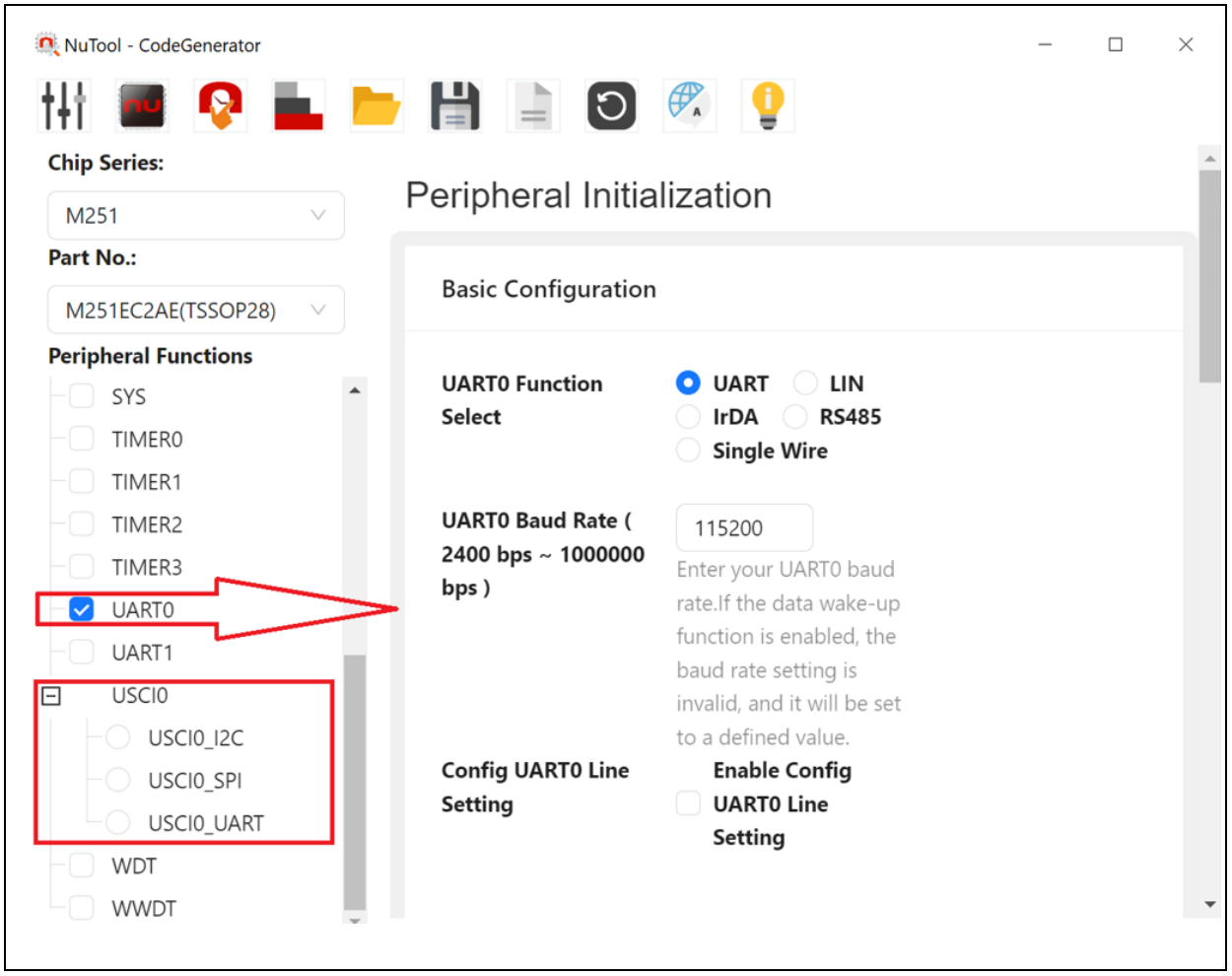

Figure 4-3 Selecting a Peripheral Function

4.4 Toolbar

4.4.1 Switch the Left Panel

To show the select field and the Peripheral Functions TreeView, click the **Switch the Left Panel**  button on the toolbar.

4.4.2 Load Configuration

The user can browse the previously saved configuration files (*.ncfg, *.cfg) and select one of them to restore the configured MCU chip.To load the configuration, click the **Load Configuration**  button on the toolbar, select the directory preserving the expected configuration file and click the Open button.

4.4.3 Save Configuration

To save the current configuration, follow the steps below:

1. Click the **Save Configuration**  button on the toolbar.
2. Browse a user-defined location and give a proper name to the configuration file (*.ncfg).
3. Click the Save button. The current configuration will be saved as a .ncfg file with a given name. The configuration file can be used to restore the configured MCU chip in the future.

4.4.4 Generate Code

Please refer to Chapter 7 for more details.

4.4.5 Return to Default Values

To return to default peripheral values, click the **Return to Default Values**  button on the toolbar.

4.4.6 Settings

To select UI language, click the **Settings**  button on the toolbar. There are three languages supported in the application, including English, Simplified Chinese, and Traditional Chinese.

4.4.7 Read User Manual

To read the user manual, click the **Read User Manual**  button on the toolbar.

5 Configuring Pins

Secondly, click the **Configure Pin** button on the toolbar.

5.1 GUI Overview

The CodeGenerator Window includes a variety of components to configure pins. The name of each component is described in Figure 5-1.

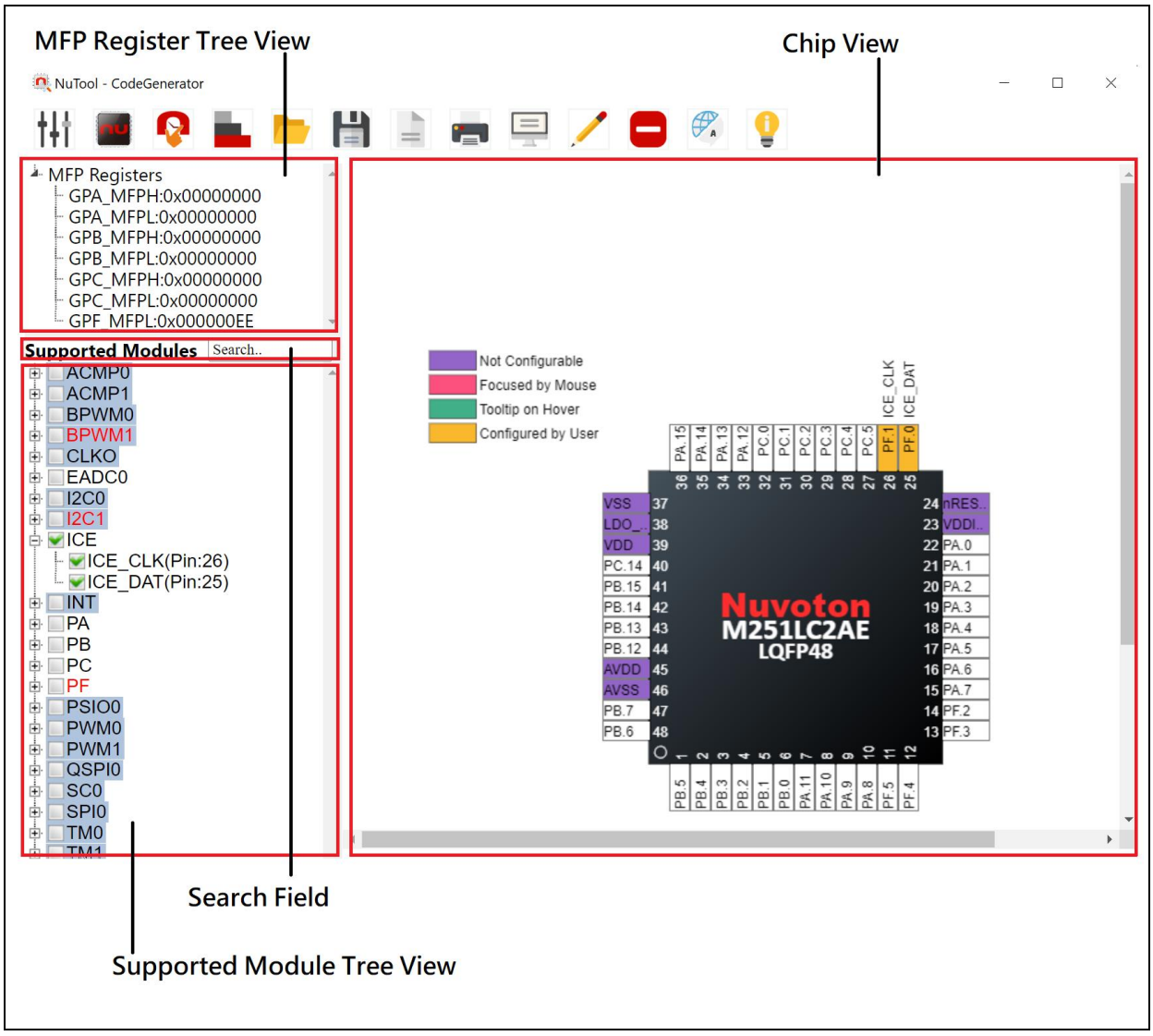

Figure 5-1 PinConfigure Window

5.2 MFP Registers TreeView

The current values of MFP registers are displayed in this TreeView. Moreover, the user can edit them directly by double-clicking on the expected one and enter a new value (referring to Figure 5-2). After editing, the corresponding check boxes of the Supported Modules - TreeView and the chip view will be updated immediately. Some chips require two different MFP registers to configure GPIO multi-functions, and thus the user cannot edit the values of MFP registers by double-clicking these chips.

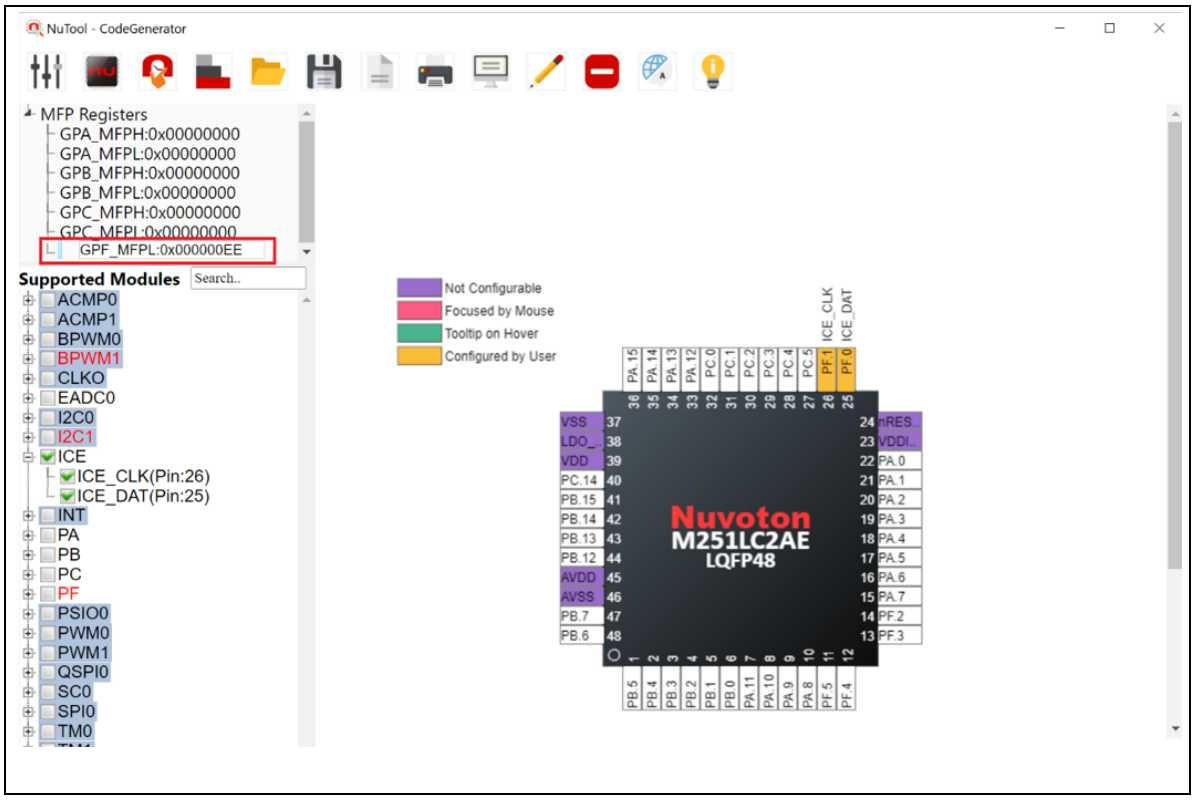

Figure 5-2 Editing a MFP Register

5.3 Supported Module – TreeView

5.3.1 Usage

Within the Supported Module - TreeView, the user can configure the peripheral pin(s). Each time a module or its individual GPIO multi-function is checked in the check boxes, the chip view shown in the right window will display the new state of the pin(s). Besides, the corresponding value of MFP register will be updated at the same time. For example, the user configures PSIO0 and the results are shown as Figure 5-3 displays.

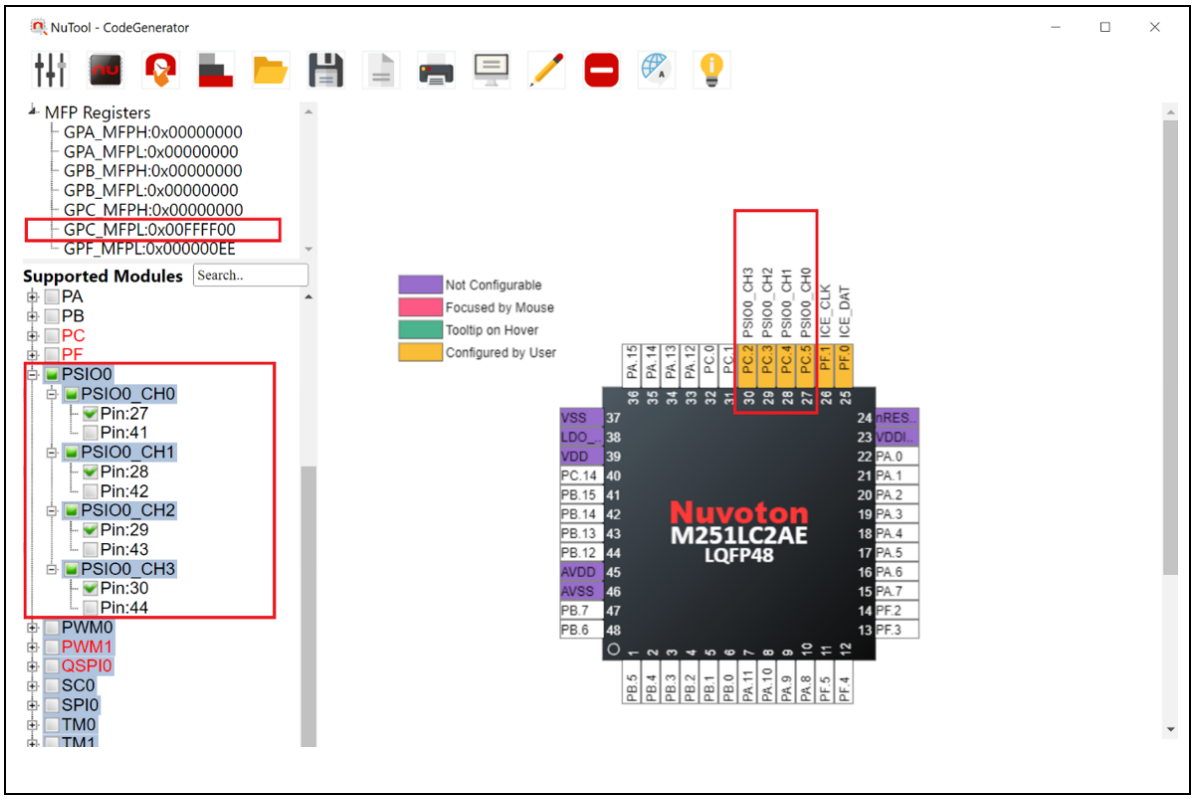

Figure 5-3 Results of Configuring PSIO0 by the TreeView

5.3.2 Conflict

When the pins have been configured to a module, the related texts in the check boxes will be marked in red. If the user obliviously wants to configure the pins again through the TreeView, this case is called as a conflict. A dialog box which lists the relevant pins and their configured modules will be invoked (referring to Figure 5-4) and offers two options for the user to decide the next step. If the user clicks the Yes button, the tool will make the **adjustment of conflicts**. If the user clicks the No button, the tool will only configure the remaining pins.

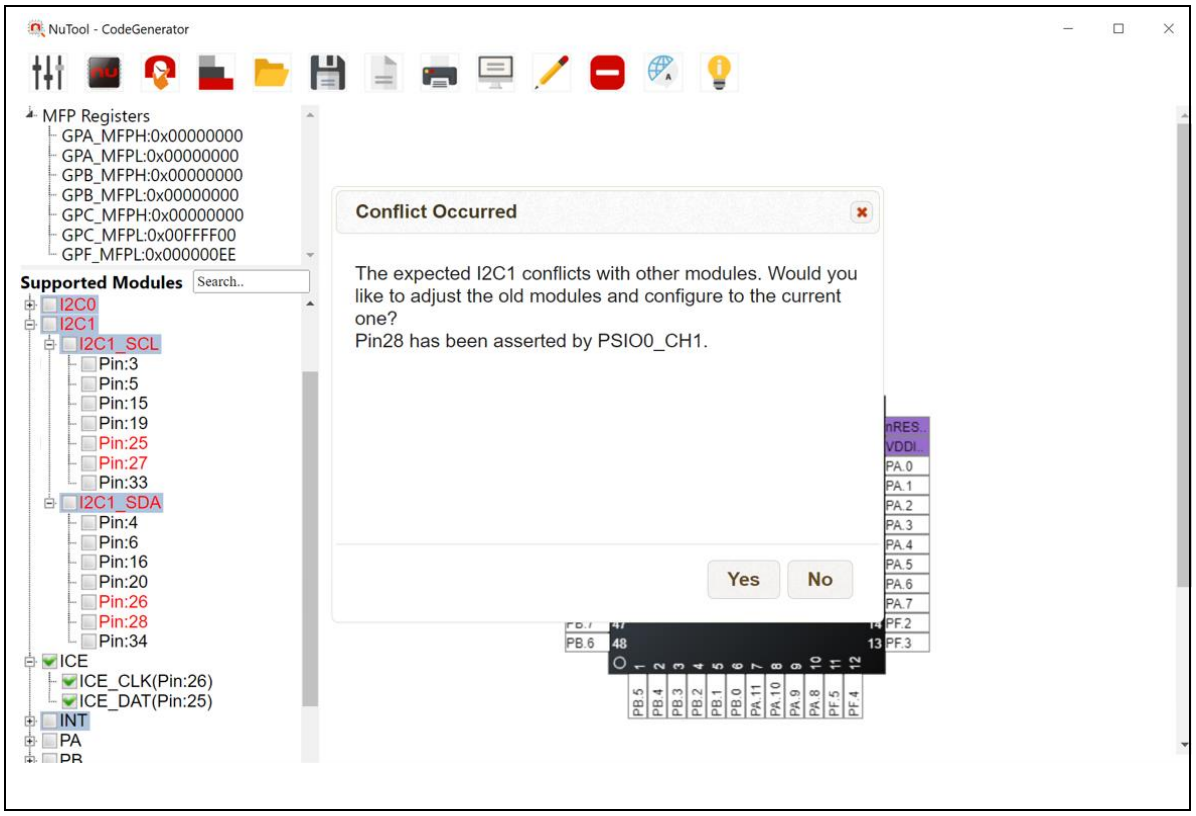

Figure 5-4 “Conflict Occurred” Dialog Box

5.3.3 Adjustment of Conflicts

To resolve conflicts, the tool recursively adjusts configured modules if possible. For instance, if the user wants to configure I2C0_SCL, the tool will try to adjust ACMP0_O to another pin (Pin47). However, Pin47 is occupied by BPWM1_CH4. Fortunately, BPWM1_CH4 has a configurable pin (Pin35) to configure.

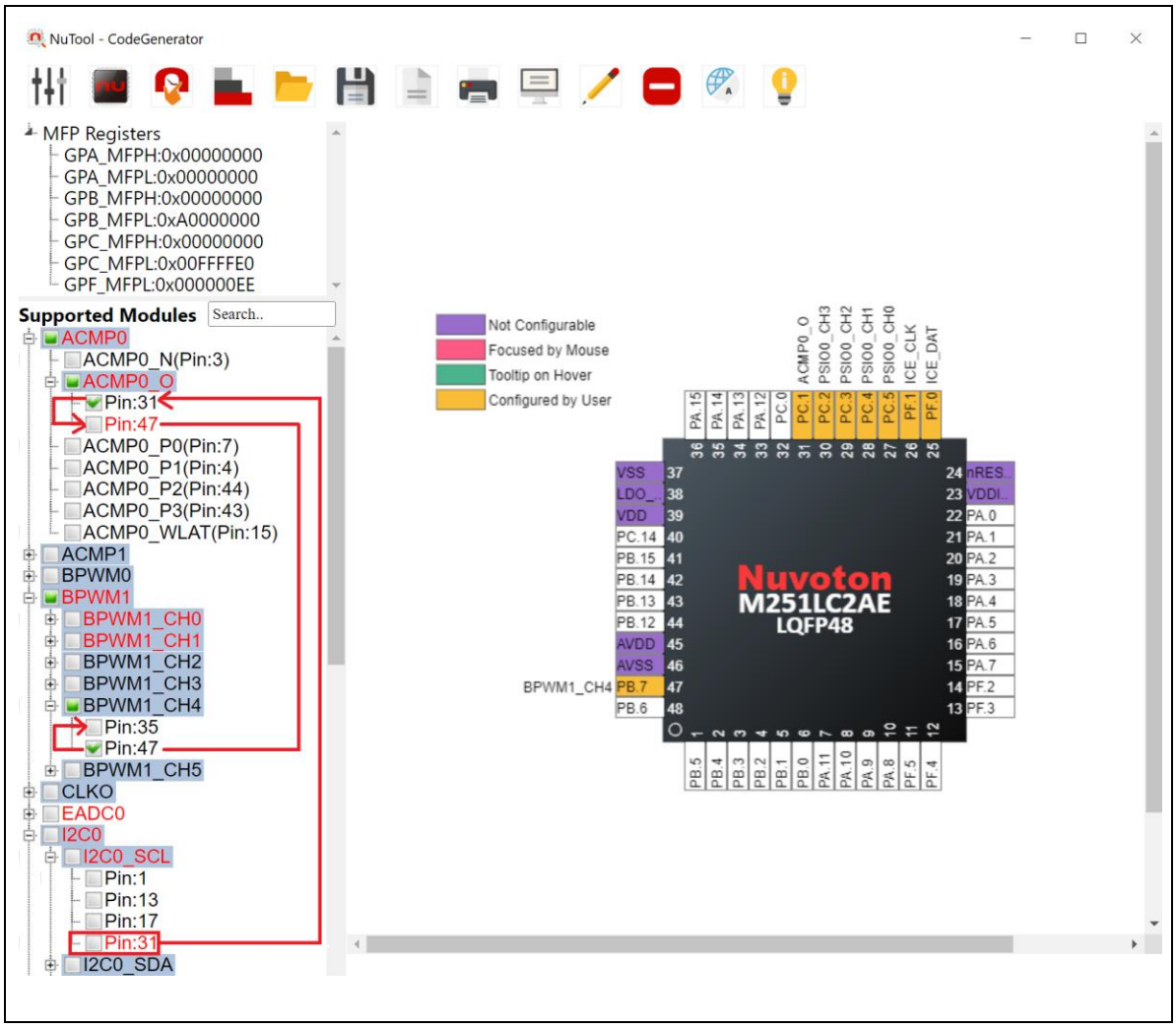

Figure 5-5 Recursive Adjustment

As a result, the tool finds the way to adjust the conflict and I2C0_SCL is configured. At the same time, ACMP0_O and BPWM1_CH4 are kept. A dialog shows up to tell the adjustment details. If the user wants to undo the adjustment of the conflicts, please click the Undo button.

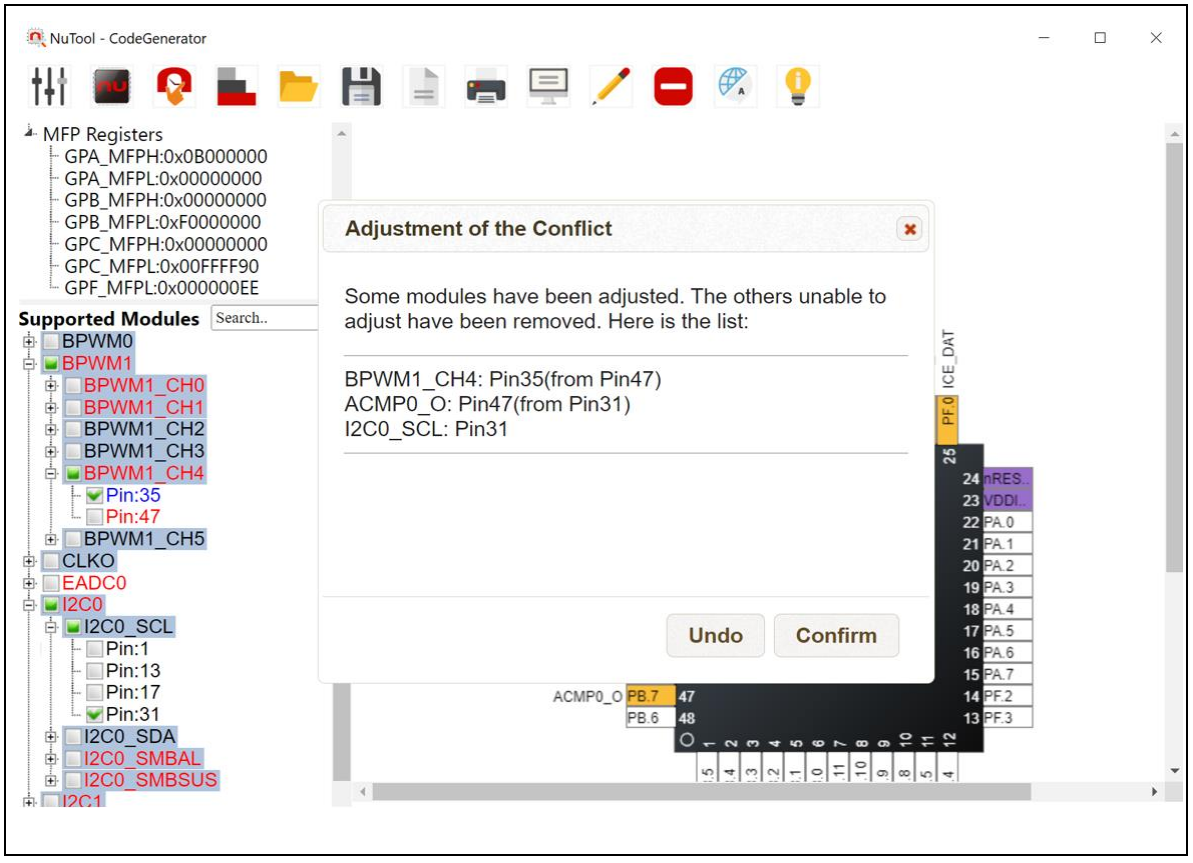

Figure 5-6 "Adjustment of the Conflict" Dialog Box

Sometimes, the tool could find several modules unable to adjust. For instance, Pin26 is occupied by ICE_CLK. ICE_CLK has only one option. Thus, if the user wants to configure I2C1_SDA (Pin26), the tool is unable to adjust ICE_CLK. That is why when configuring I2C1_SDA (Pin26), ICE_CLK has to be removed.

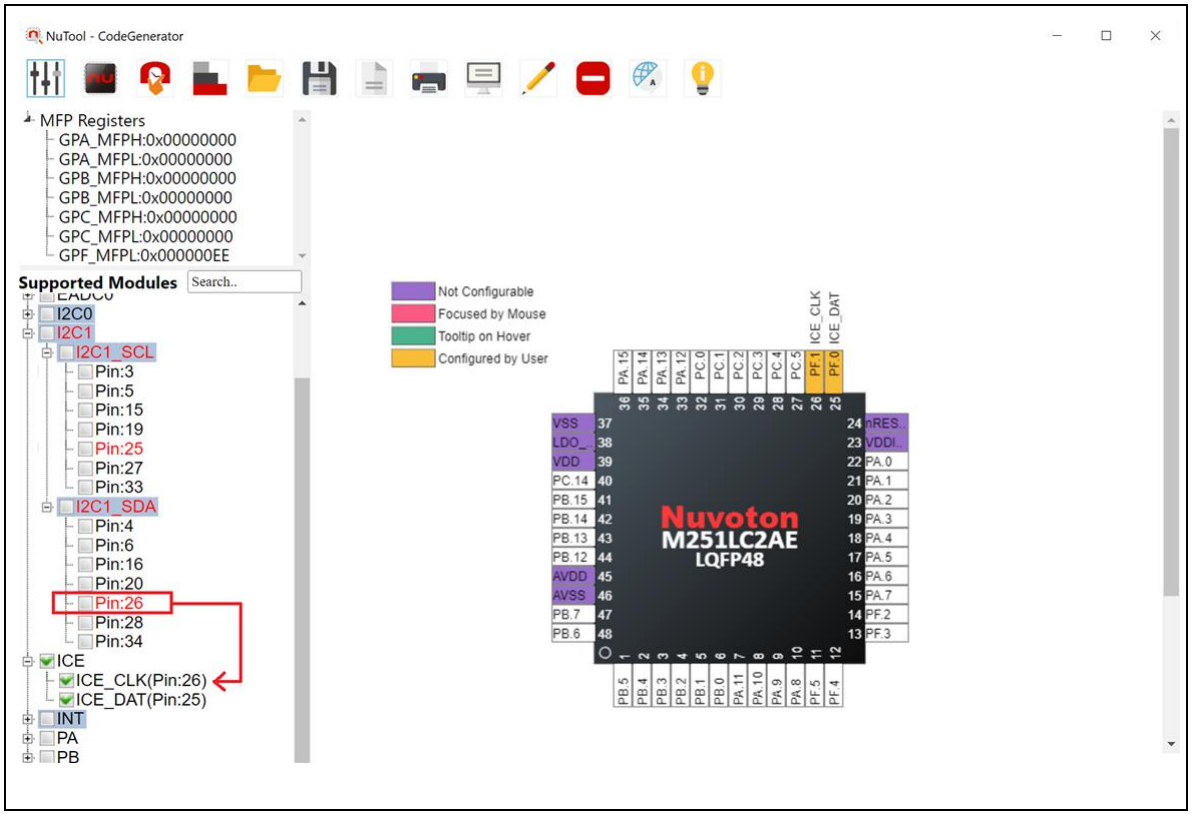

Figure 5-7 Adjustment Based on Removal

5.3.4 Multiple Selections

There are some modules whose GPIO functions have multiple selections of pins to the same function. In this case, the related check boxes are highlighted with the steel blue color. The user is only permitted to select one of pins. For example, in the ACMP0 module, its GPIO function of ACMP0_O has two options, pin 31 and 47, but only one of them can be occupied by ACMP0_O (referring to Figure 5-8).

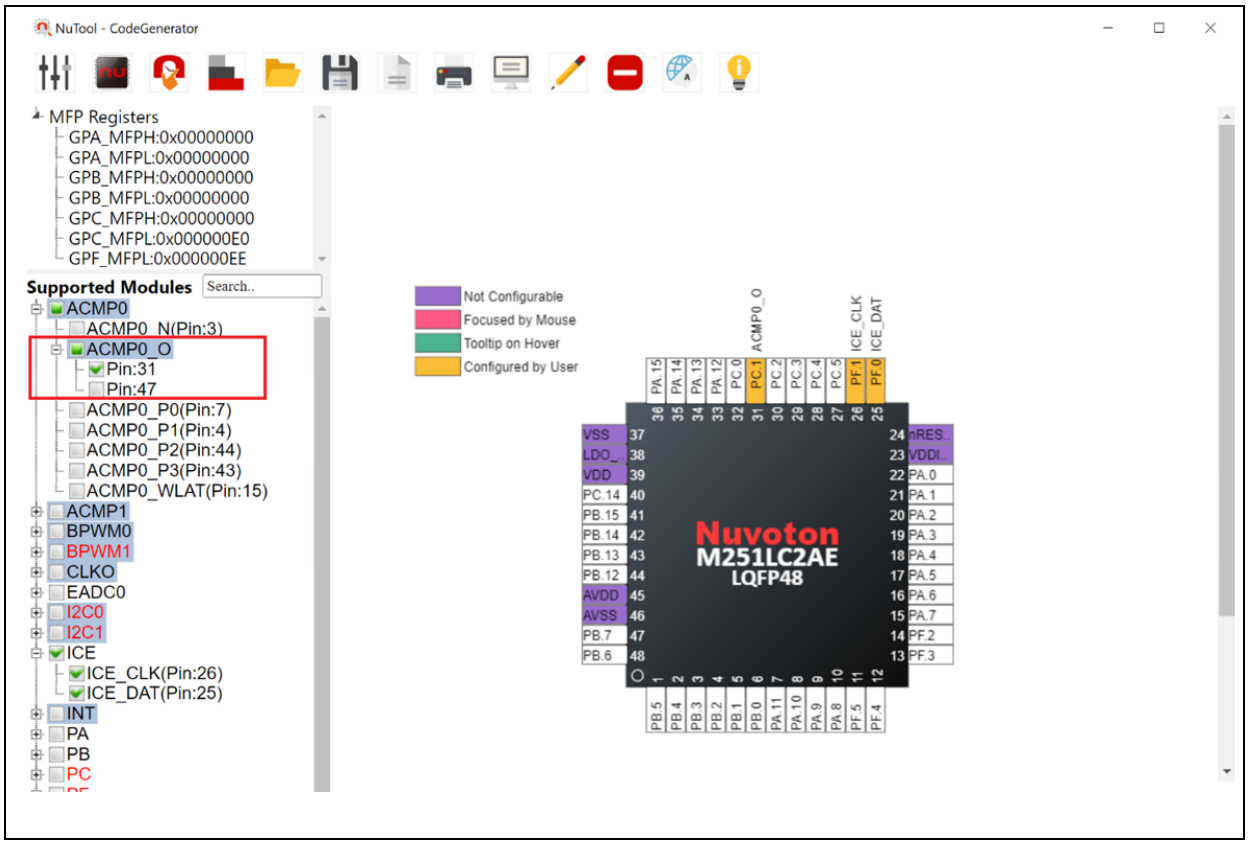

Figure 5-8 Multiple Selections of ACMP0_O

5.3.5 Search

To find a specific module in the Supported Modules - TreeView, the user can input the expected module name in the search field. After input, the matched texts in the check boxes will be marked in bold and italics. Note that the search adopts the partial match, not exact match (referring to Figure 5-9). The minimum number of input characters is two.

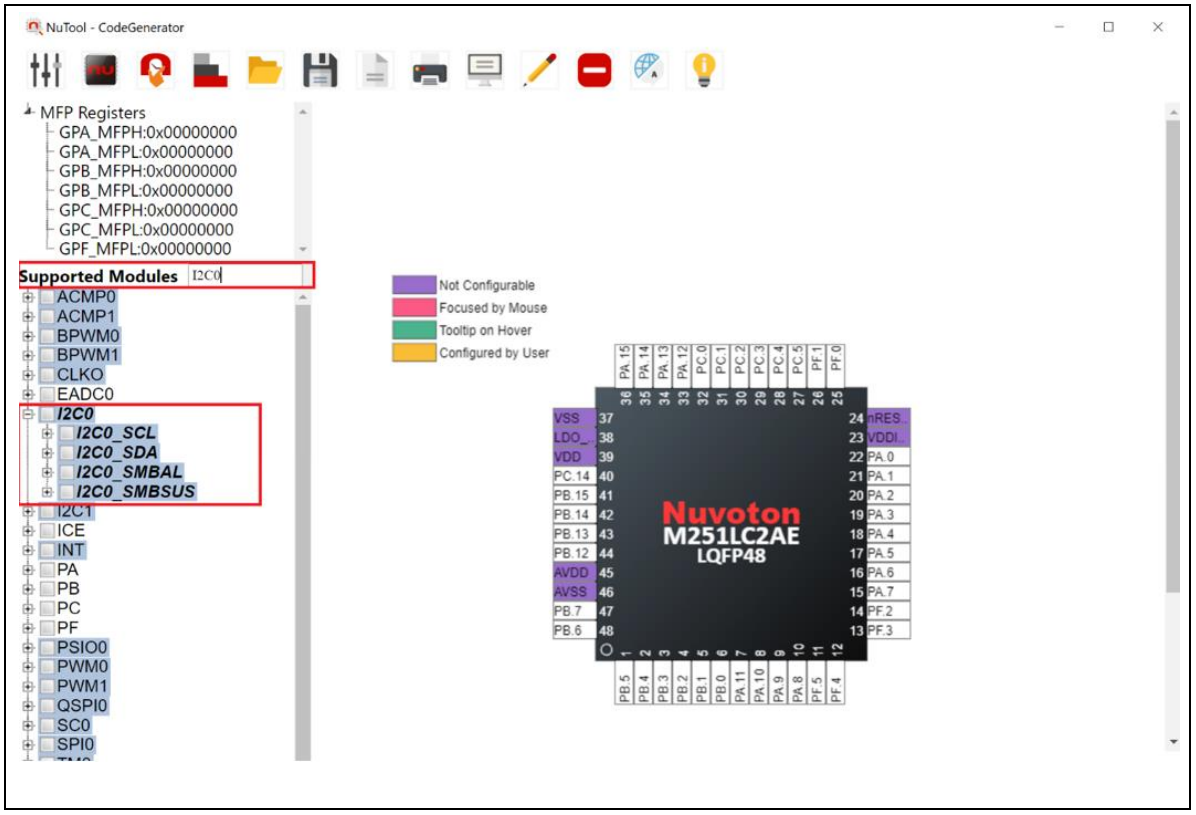

Figure 5-9 Matched Search Results

5.4 ChipView

The chip view, which is in the right pane of the window, depicts a graphical chip involving its pins. Each pin possesses its own information of the current pin assignment. The pins which are highlighted with the purple color denote that they do not belong to the configurable pins. If a pin is being configured to a GPIO multi-function, the corresponding function name will emerge in the vicinity of the pin. Meantime, the pin will be highlighted with the green color if it is configured by the TreeView, or with the orange color if it is configured by the individual pin.

To configure by the individual pin, follow the steps below:

1. Move the mouse cursor to the expected pin and click on the left button of the mouse. Then the list of all the related GPIO multi-functions will emerge in the vicinity of the pin (referring to Figure 5-10).
2. Move the mouse cursor into the list and select the expected GPIO function and click on it. Configuring by the individual pin is accomplished. At the same time, the TreeView and the value of the MFP register will be updated correspondingly (referring to Figure 5-11).

The difference between configuring by individual pins and TreeView is that the user can arbitrarily configure any pin by the individual pins without considering the occurrence of a conflict. To disable the configured pins by individual pins, move the mouse cursor to the expected pins and left-click. Select the last row of the list which is named as Reset (referring to Figure 5-12). Then the disable operation is completed. To configure user-defined functions mentioned in Section 7.4, double click on the chip view.

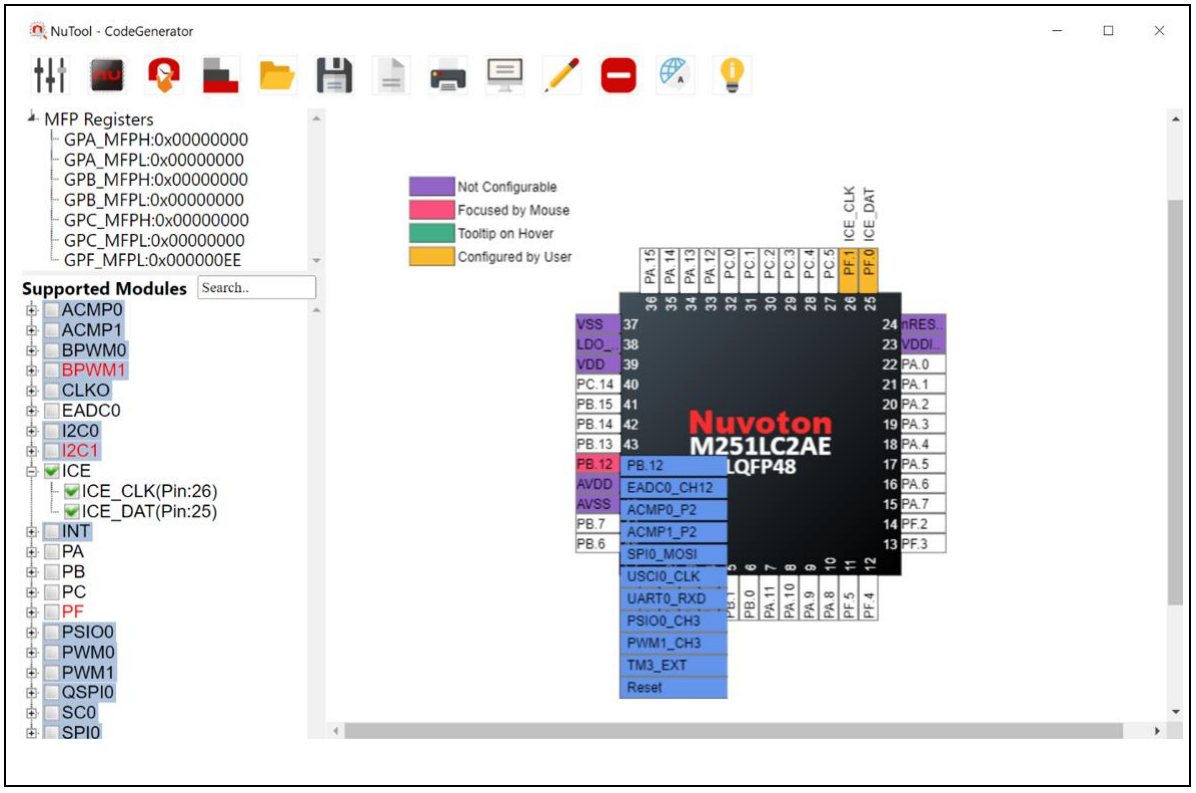

Figure 5-10 List of All the Related GPIO Multi-functions

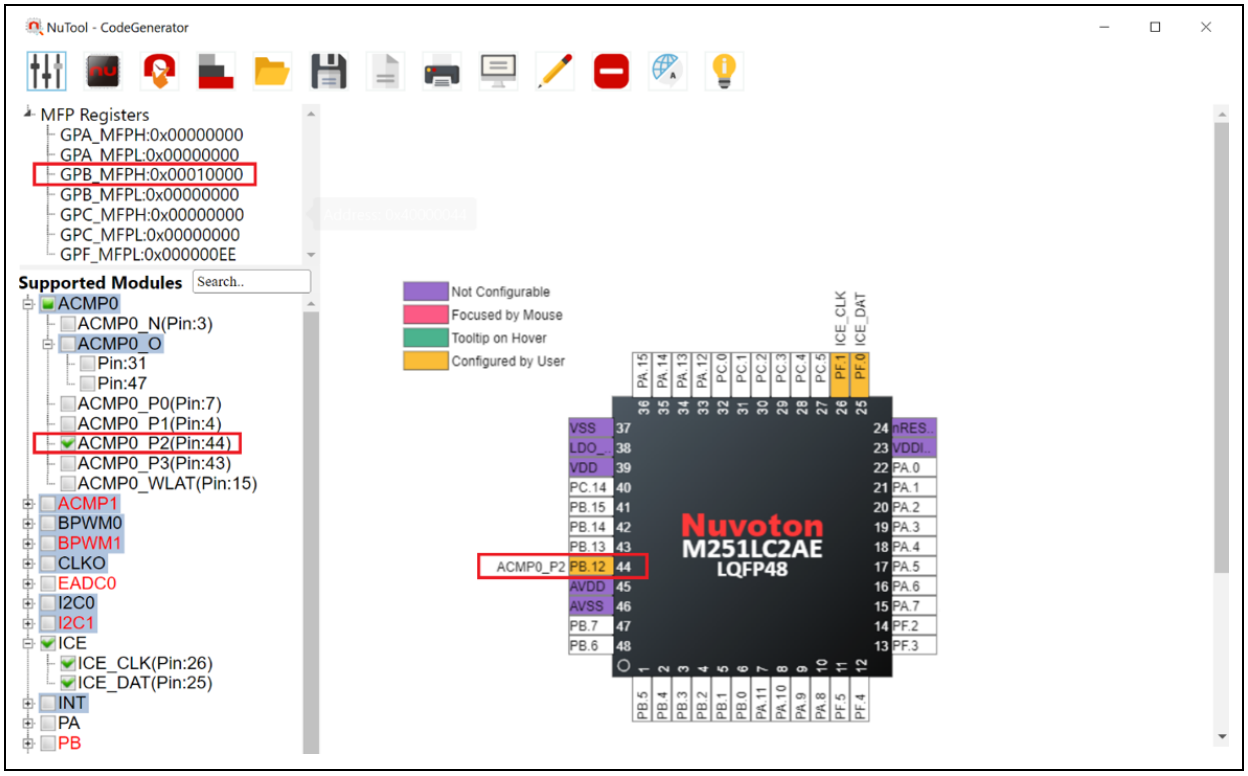

Figure 5-11 Results of Configuring ACMP0_P2 by the Individual Pin

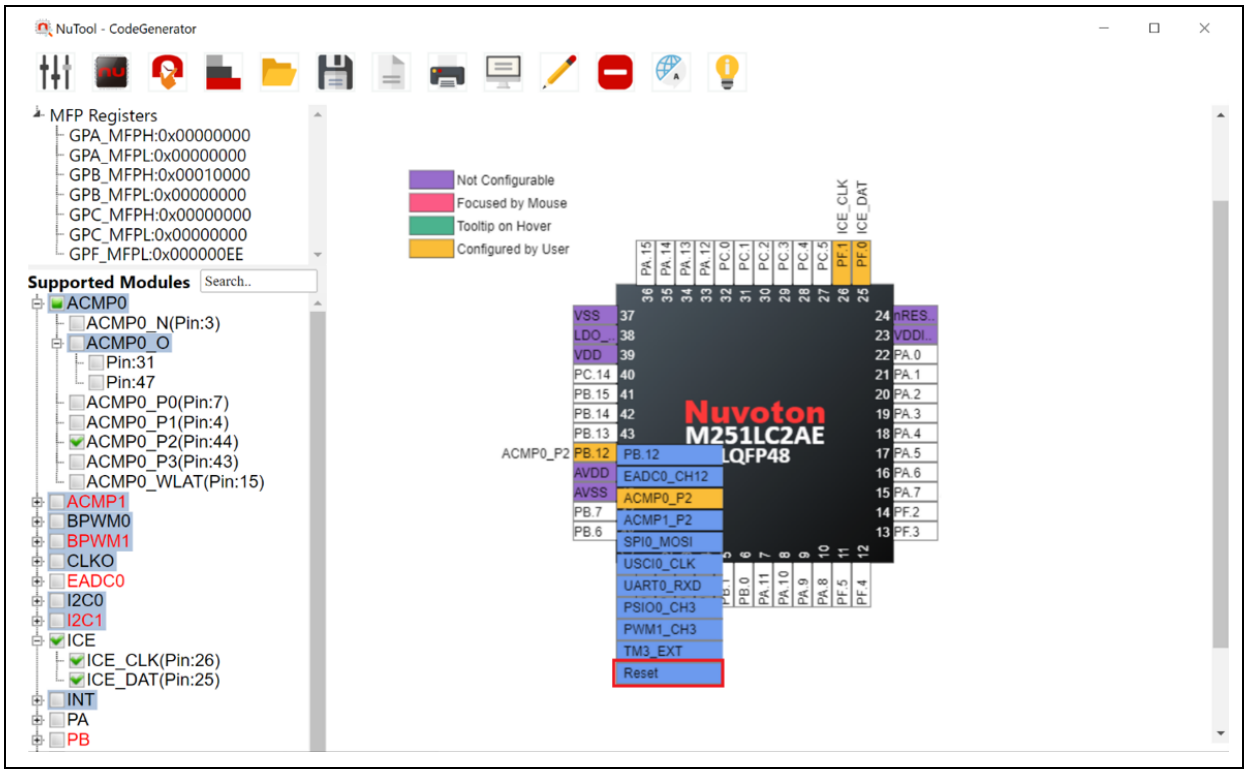

Figure 5-12 Disabling the Configured Pin

5.5 Toolbar

Some duplicate buttons have been described in Section 4.4.

5.5.1 Switch MFP-Registers TreeView

To show the MFP Registers TreeView, click the **Switch MFP-Registers TreeeView**  button on the toolbar.

5.5.2 Print Report

To print a report, click the **Print Report**  button on the toolbar. After inputting the project name and selecting the expected criteria, click on the Confirm button to print the report.

5.5.3 Generate Report of Pin Description

To generate report of pin description, click the **Generate Report of Pin Description**  button on the toolbar.

5.5.4 Switch Pin Description

To show pin description, click the **Switch Pin Description**  button on the toolbar. The whole description will be expanded around the chip.

5.5.5 Disable All Checked Modules

To disable all checked modules, click the **Disable All Checked Modules**  button on the toolbar.

6 Configuring Clocks

Thirdly, click the **Configure Clock** button on the toolbar.

6.1 GUI Overview

The CodeGenerator Window includes a variety of components to configure clocks. The name of each component is described in Figure 6-1.

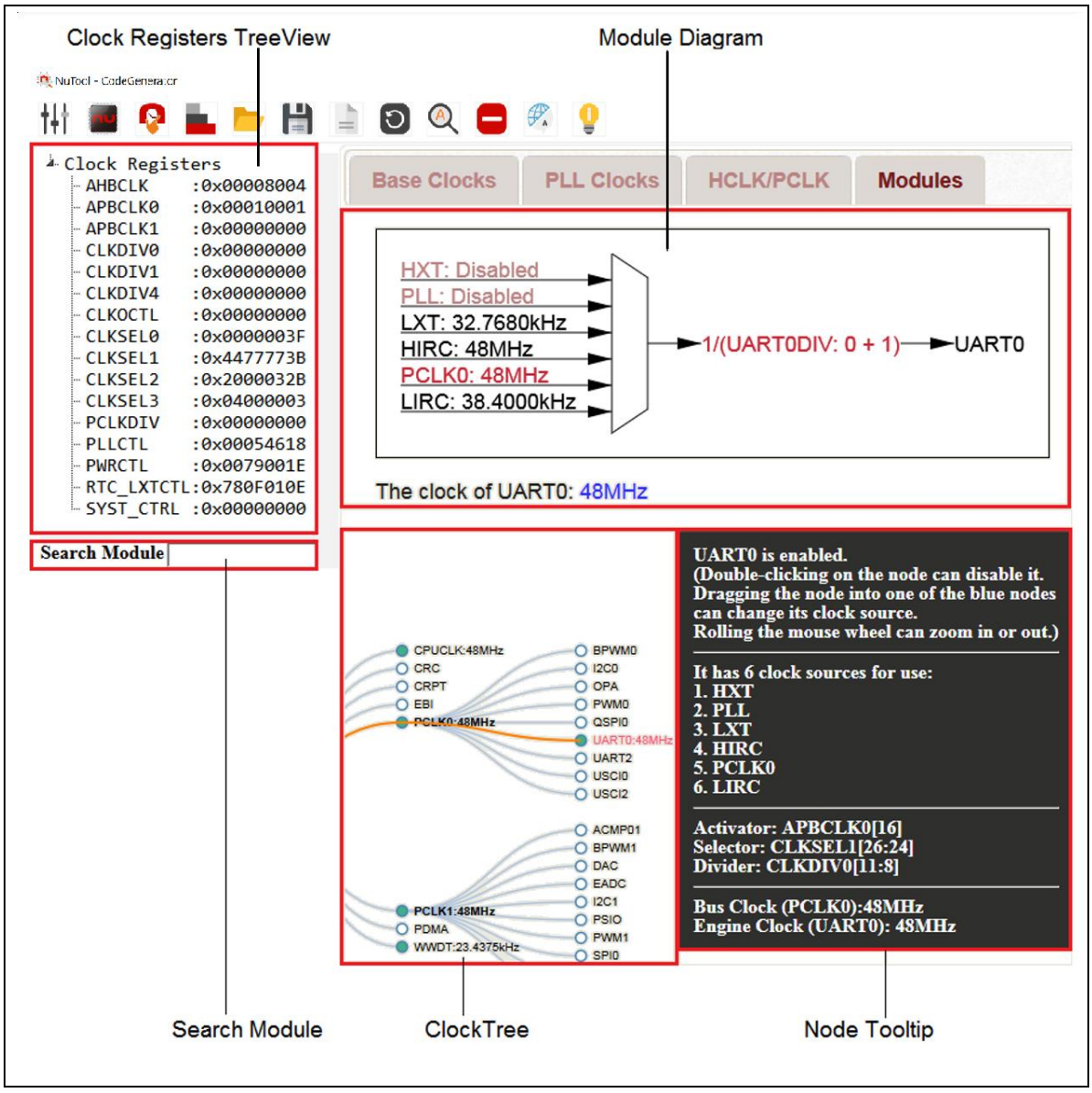

Figure 6-1 ClockConfigure Window

6.2 Clock Registers TreeView

The current values of clock registers are displayed in the upper-left TreeView. Moreover, the user can edit them directly by double-clicking on the expected one and enter a new value (referring to Figure 6-2). After editing, the corresponding result will be updated immediately.

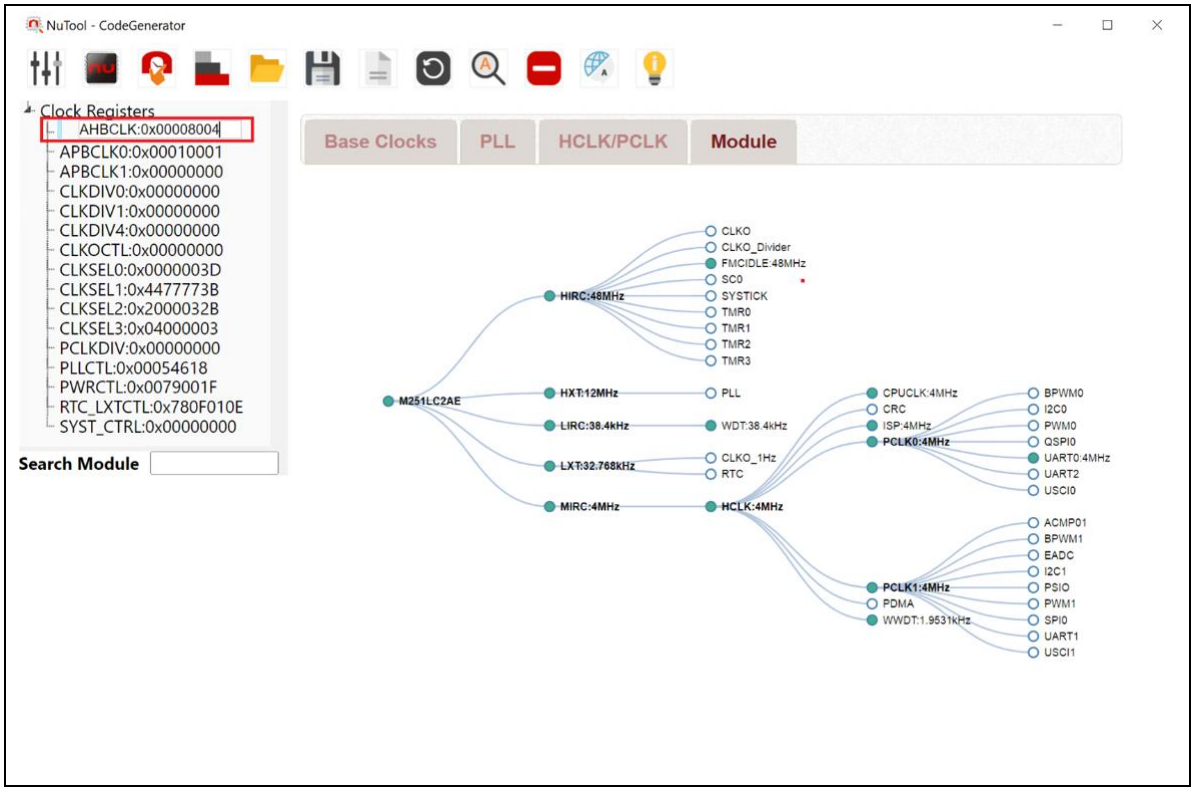

Figure 6-2 Editing a Clock Register

6.3 Search Module

To search a specific module in the ClockTree, the user can input the expected module name in the search field. After input, the matched node will be emphasized with an orange path from the root.

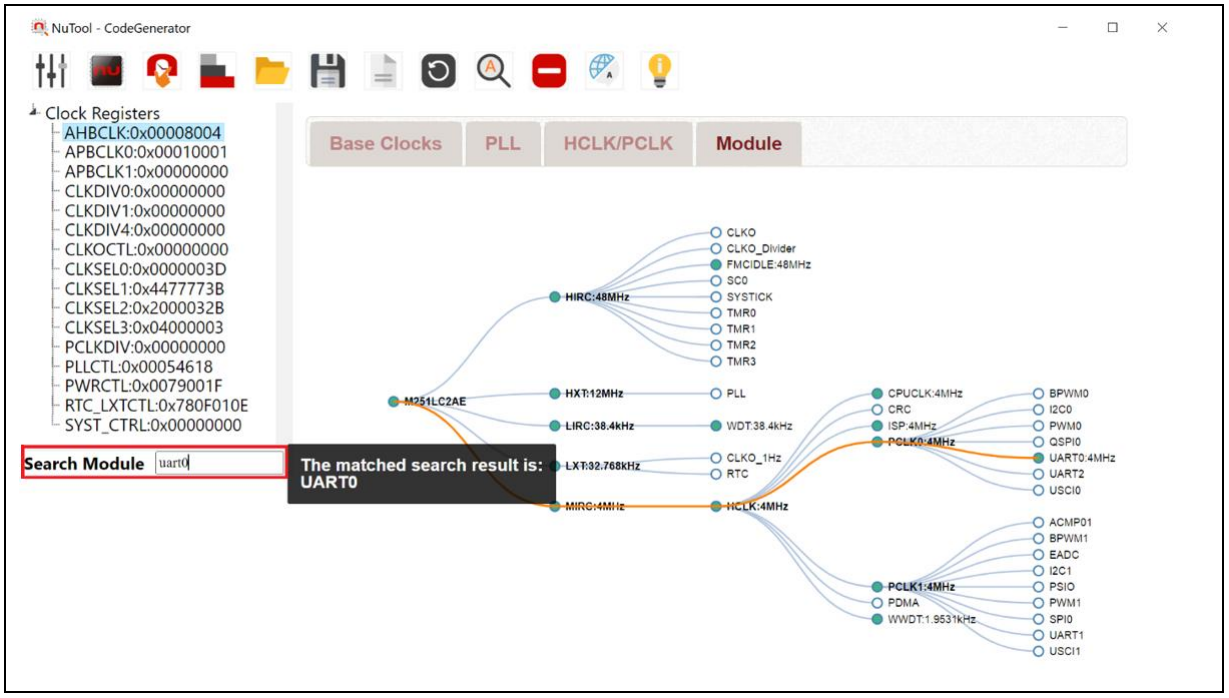

Figure 6-3 Matched Search Result

6.4 Clock Configuration Flow

6.4.1 Overview

The corresponding clock registers are displayed in the upper-left Clock Registers TreeView region. In the following discussion, the chip series and part number selected are M251 and M251LC2AE, respectively. Other chip may have a slight difference in the flow, but the basic logic is the same. For M251, there are four steps to complete the clock configuration, i.e., Base Clocks, PLL Clocks, HCLK/PCLK and Modules. Even if the four steps are set, the user can adjust them as wished. In fact, the clock configuration of the tool is set to the final step automatically.

6.4.2 Step 1: Base Clocks

In step 1, the user can enable or disable the base clocks of LIRC, HIRC, MIRC, LXT and HXT (referring to Figure 6-4).

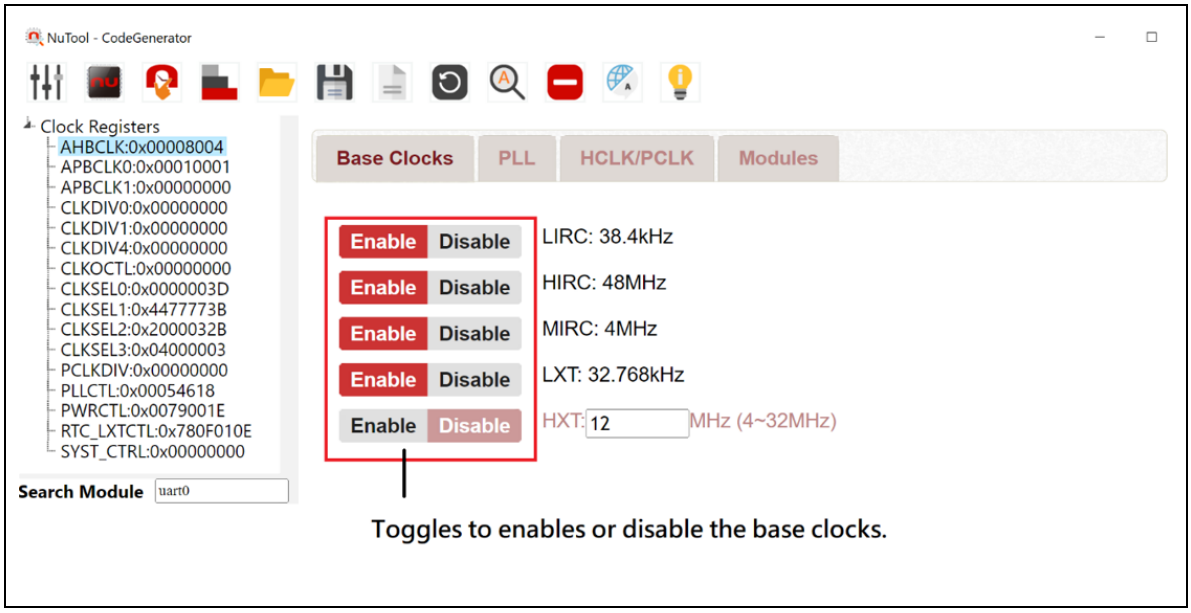

Figure 6-4 Step 1: Base Clocks

6.4.3 Step 2: PLL Clocks

The user can input his expectation value to PLL frequency. All the possible candidates sorted by the inaccuracy will be listed in the table. If the expected clock input is not in the table, please **increase the inaccuracy** or change the expectation value of PLL frequency to another value. Move the mouse into the table and choose one of checkboxes. The clock of PLL will be shown below the table. Note that all the manipulations will update the content of clock registers simultaneously (referring to Figure 6-5).

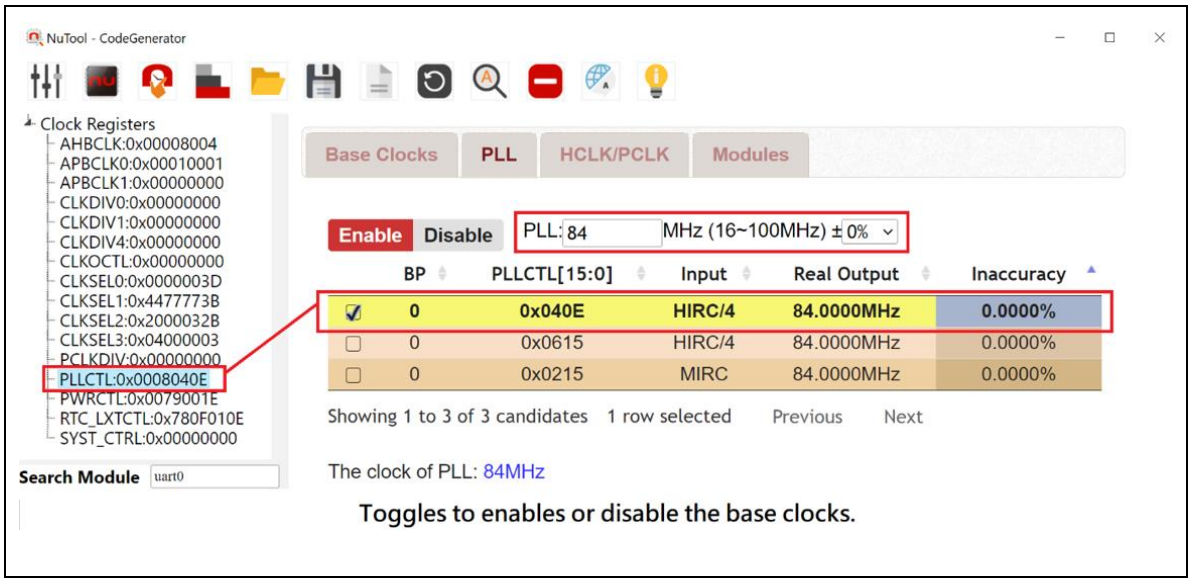

Figure 6-5 Step 2: PLL Clocks

6.4.4 Step 3: HCLK/PCLK

In step 3, the feasible clock sources and HCLK's divider will be drawn in the diagram used to configure HCLK. The user can choose one of clock sources by moving the mouse into the diagram and directly clicking on the expected clock source (referring to Figure 6-6). The chosen one will be highlighted with an India red color.

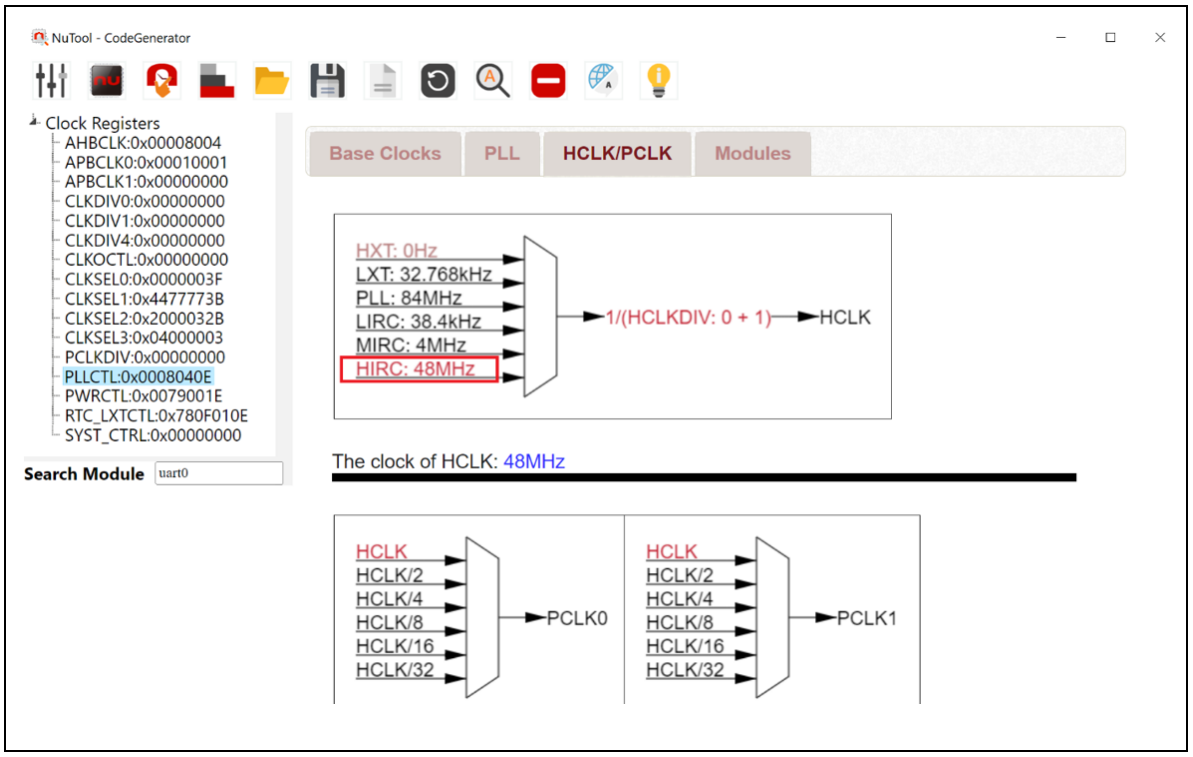

Figure 6-6 Step 3: Choosing the Clock Source of HCLK

To configure the value of HCLK's divider, move the mouse into the diagram and click on the divider region. A dialog will pop up to allow the user to input a value to HCLK's divider (referring to Figure 6-7). For instance, the user inputs 0 to HCLKDIV. After pressing the confirm button, the clock of HCLK will be calculated and shown below the diagram. In this case, it would be 48MHz.

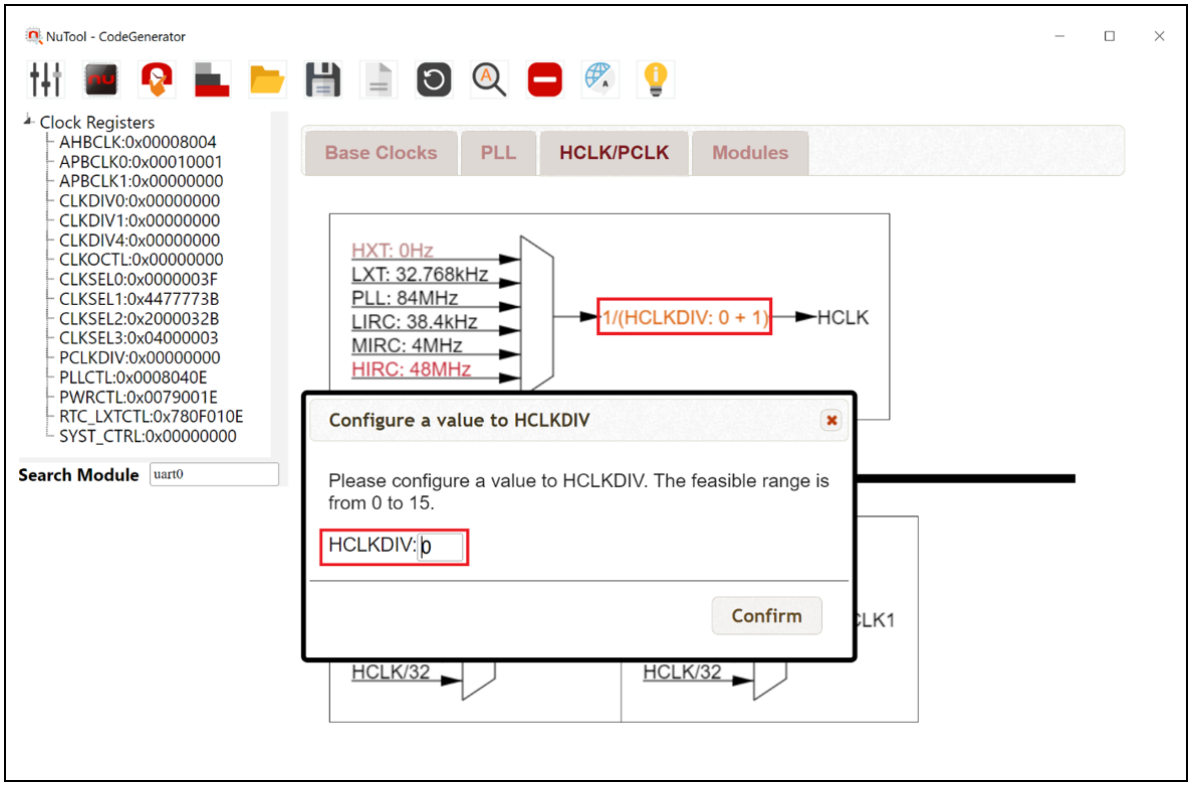

Figure 6-7 Step 3: Setting a Value to HCLK's Divider

6.4.5 Step 4: Modules Using ClockTree

In ClockTree, the user can enable or disable modules by double-clicking on the corresponding nodes of the ClockTree. In addition, dragging a node into one of the blue nodes can change its clock source (referring to Figure 6-8). The **red connection line** means that the module belongs to the target clock source. Only when the red connection line appears, the new change of the clock source could happen after dropping the node. However, the user is unable to configure the divider of any module here. The operation is allowed in the Module Diagram. Besides, double-clicking on the background of ClockTree can review the configured report mentioned in Section 7.4.

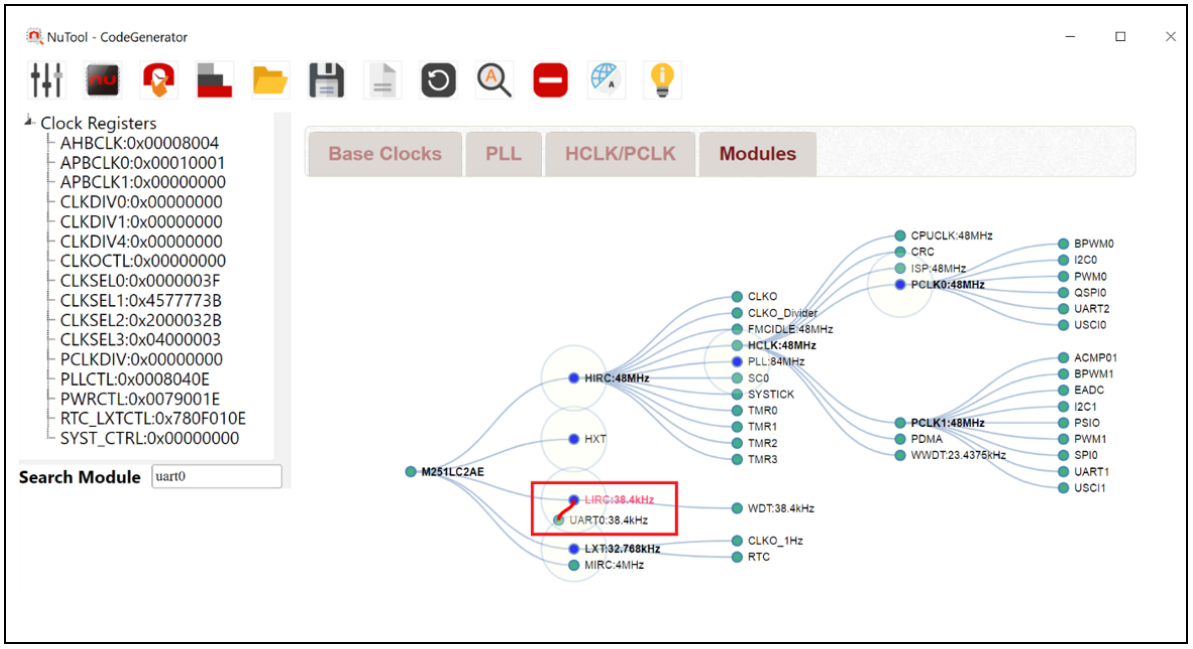

Figure 6-8 Step 4: Dragging UART0 Node into LIRC Node

6.4.6 Step 4: Modules Using Module Diagram

When the module is enabled, single-clicking on the node can show the module diagram. The manipulation of module diagram is the same as HCLK diagram mentioned in Section 6.4.4.

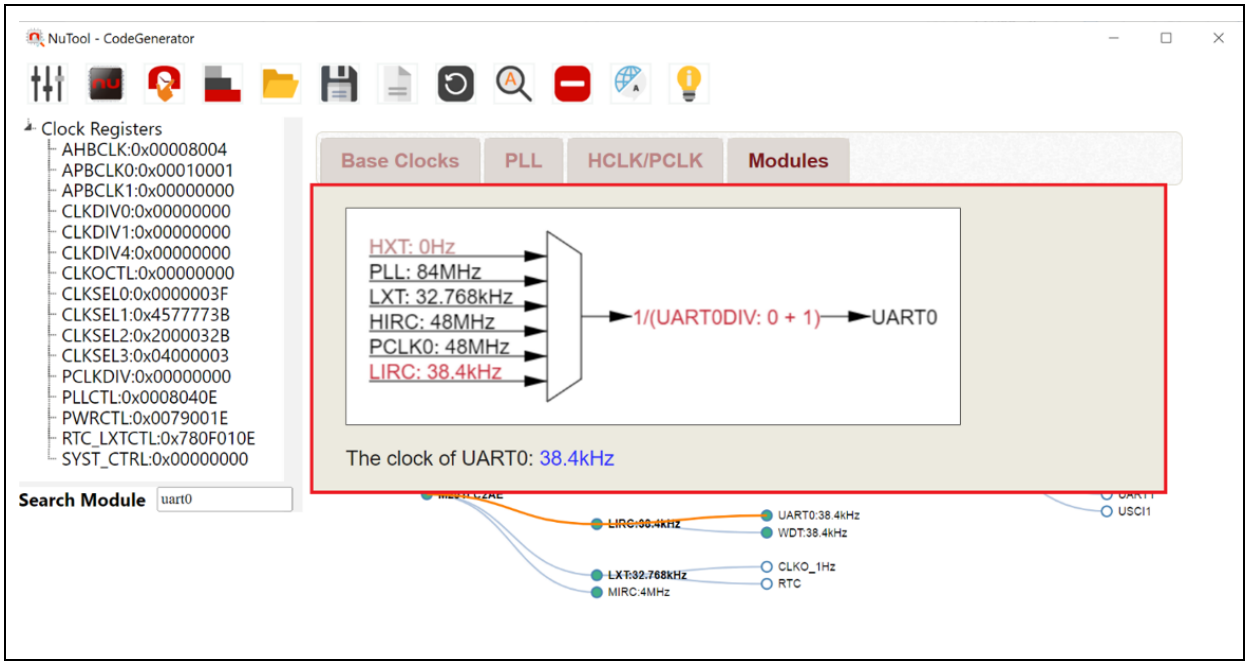

Figure 6-9 Step 4: UART0 Diagram

6.5 Toolbar

Some duplicate buttons have been described in Section 4.4.

6.5.1 Switch the Left Panel

To show the Clock Registers TreeView, click the **Switch the Left Panel**  button on the toolbar.

6.5.2 Return to Default Settings

To return to default clock settings, click the **Return to Default Settings**  button on the toolbar.

6.5.3 Switch Clock Tree

To show the clock tree which only contains the enabled modules, click the **Switch Clock Tree**  button on the toolbar. As a result, a compact tree will show up.

6.5.4 Disable All Enabled Modules

To disable all enabled modules, click the **Disable All Enabled Modules**  button on the toolbar.

7 Generating Code

Finally, click the **Generate Code** button on the toolbar. Please click **Download BSP** to download the corresponding BSP tag on github. This can help avoid compatibility issues between versions.

7.1 Modules Need Checking

When the user should forget to configure corresponding modules in PinConfigure, ClockConfigure or PeripheralConfigure, the tool would list them in the lower-right corner as a reminder (referring to Figure 7-1). As long as the user configures them correctly, the tool can create a feasible initialization project.

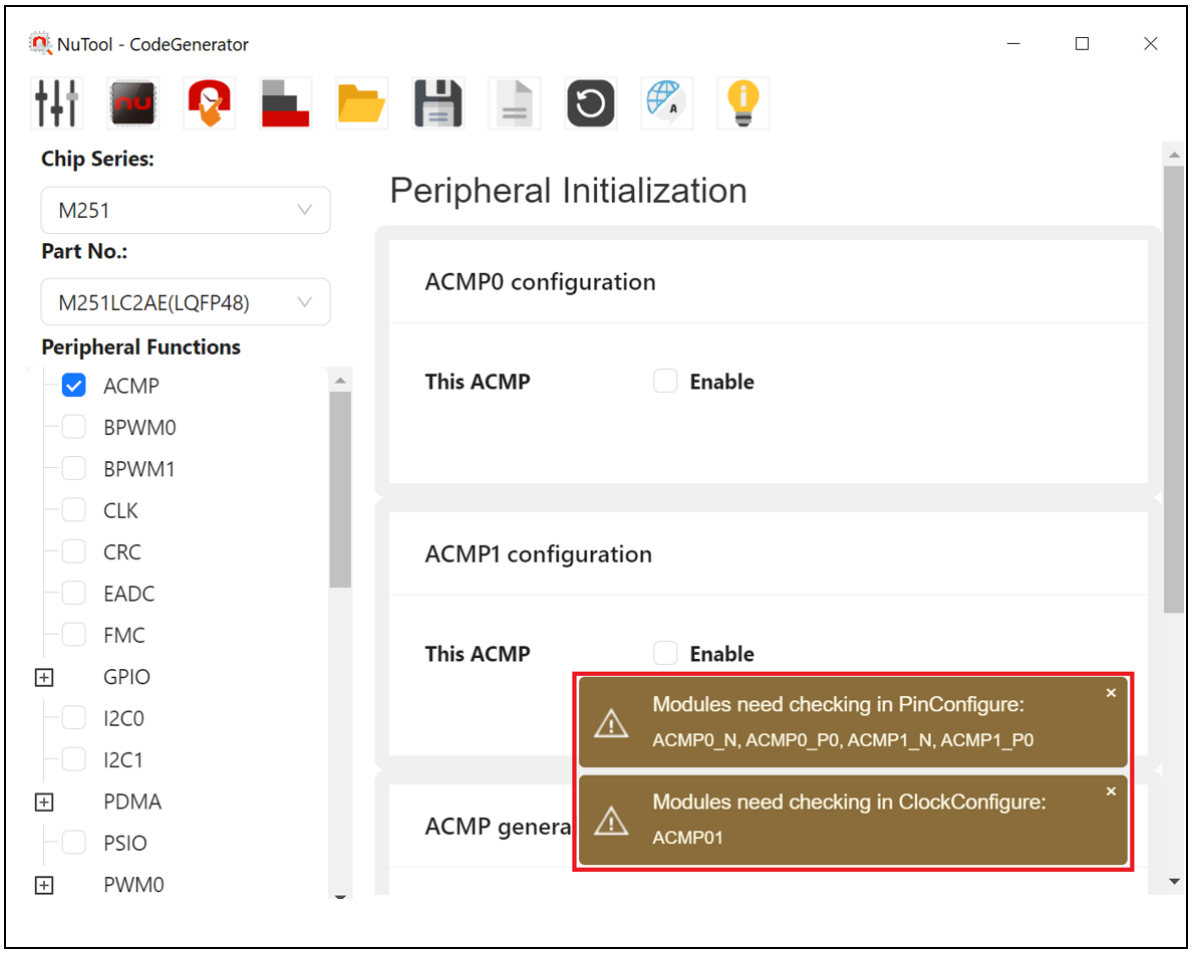

Figure 7-1 List of Modules Need Checking

7.2 Folder Path for Storing the Generated Code

The folder path for storing the generated code must be located in the **SampleCode** folder of Nuvoton BSP. If not, the iniailization project may fail to build. After the iniailization project is created, the user can develop it with Keil, IAR or NuEclipse. Moreover, the project configuration file is saved within the iniailization project. The user can load it to restore the project configuration to the saved one (referring to Section 4.4.2).

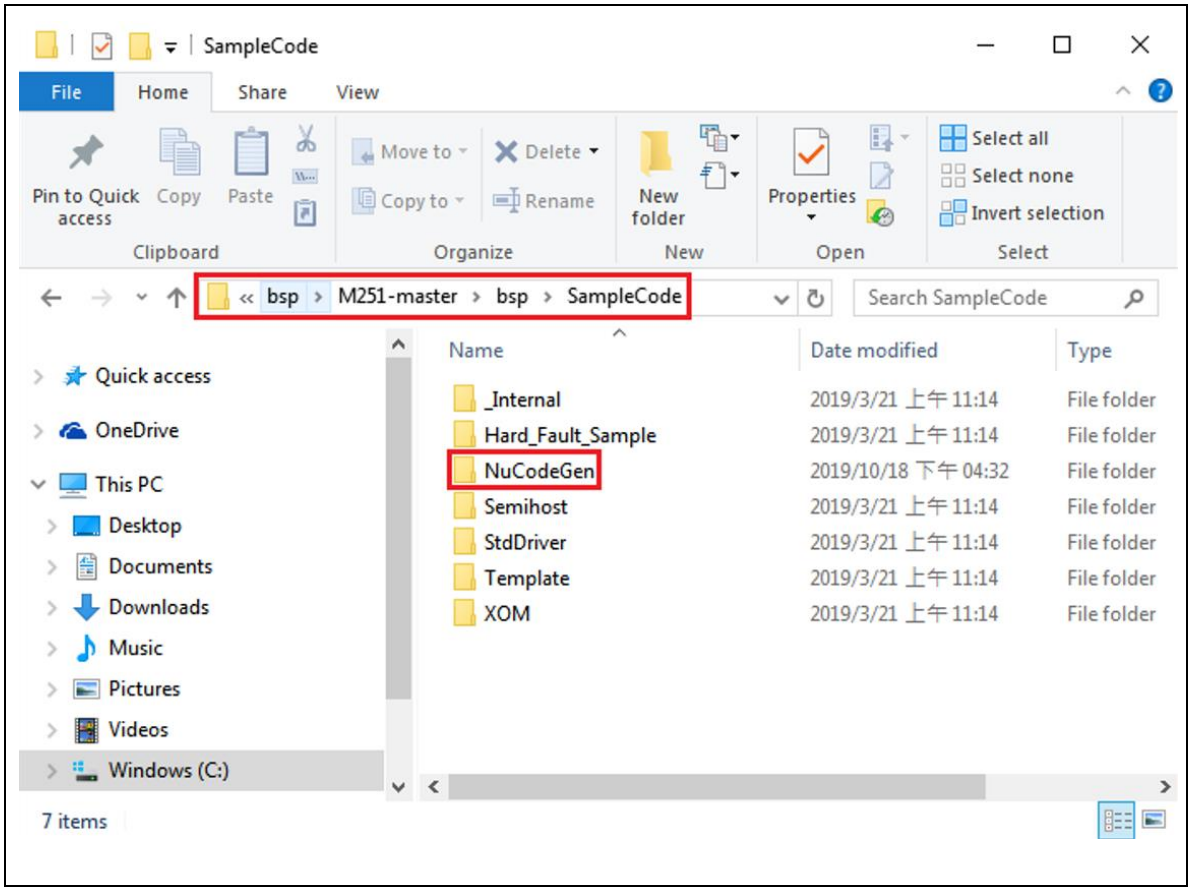

Figure 7-2 Locating to BSP SampleCode Folder

7.3 Generated Project

In **SampleCode/NuCodeGen**, the user can find the generated project, e.g., MyProject. Its content is listed below:

1. KEIL/IAR/GCC folders, which contain IDE related files.
2. A main.c file, which defines a System_Init function for initializing pins, clocks, and peripherals. Peripheral API functions (e.g., UART_READ) should be called after the System_Init function is completed.
3. An int_handler.c file, which specifies all required IRQHandler functions.

7.4 Review Report

To review the configured report, click on the check box of Review Reort in the Generate Code dialog. From there, the user can obtain the configured information and give configured pins with user-defined names. According to #define macro of C language, input characters can only be letters, digits and underscores. In addition, the maximum length is 30.

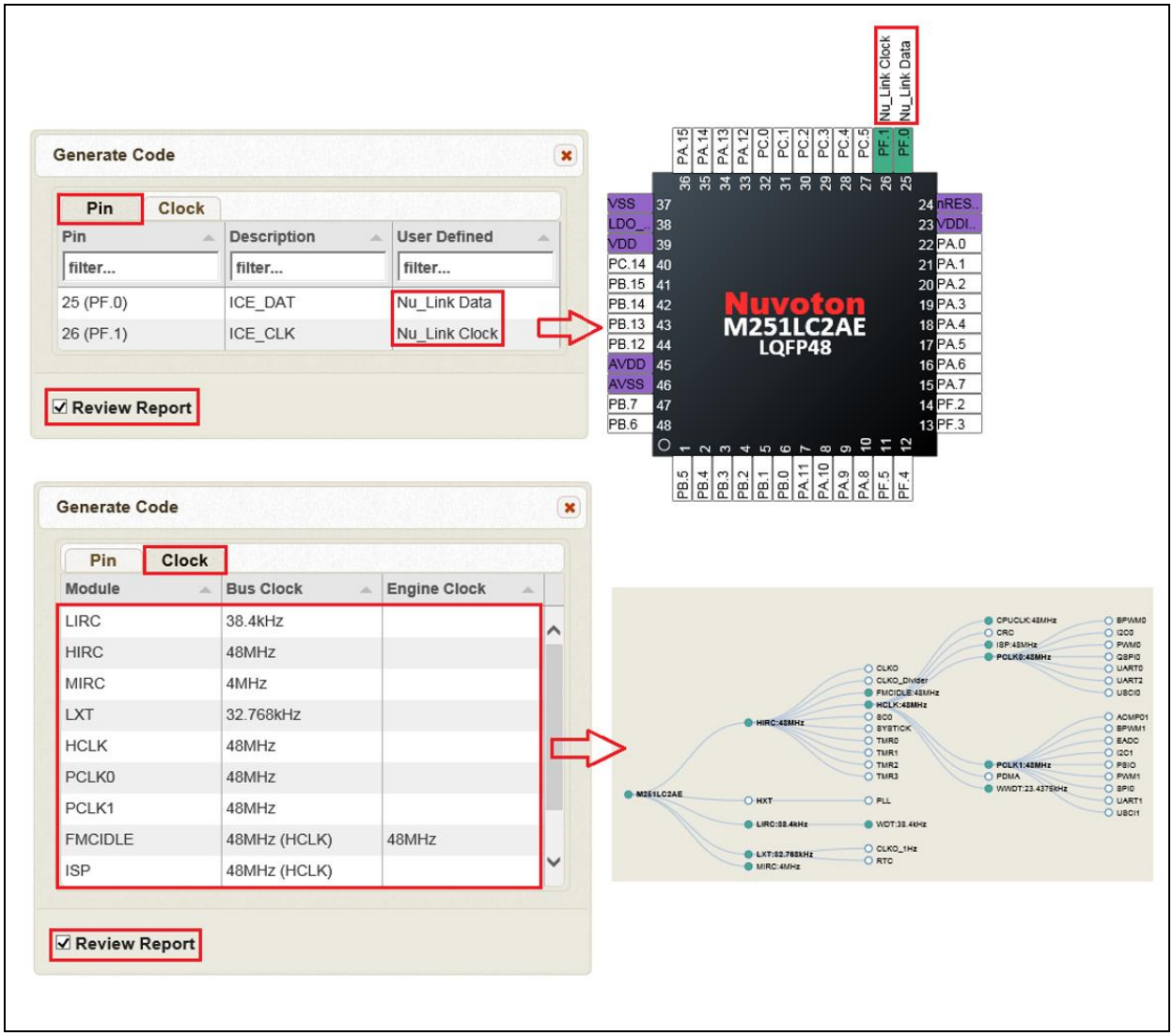

Figure 7-3 Review Report

Notice: Using this software indicates your acceptance of the disclaimer hereunder:

THIS SOFTWARE IS FOR YOUR REFERENCE ONLY AND PROVIDED "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. YOUR USING THIS SOFTWARE/FIRMWARE IS BASED ON YOUR OWN DISCRETION, IN NO EVENT SHALL THE COPYRIGHT OWNER OR PROVIDER BE LIABLE TO ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.