Fundamental Knowledge of NuCodeGenTool

For Nuvoton BSP team members

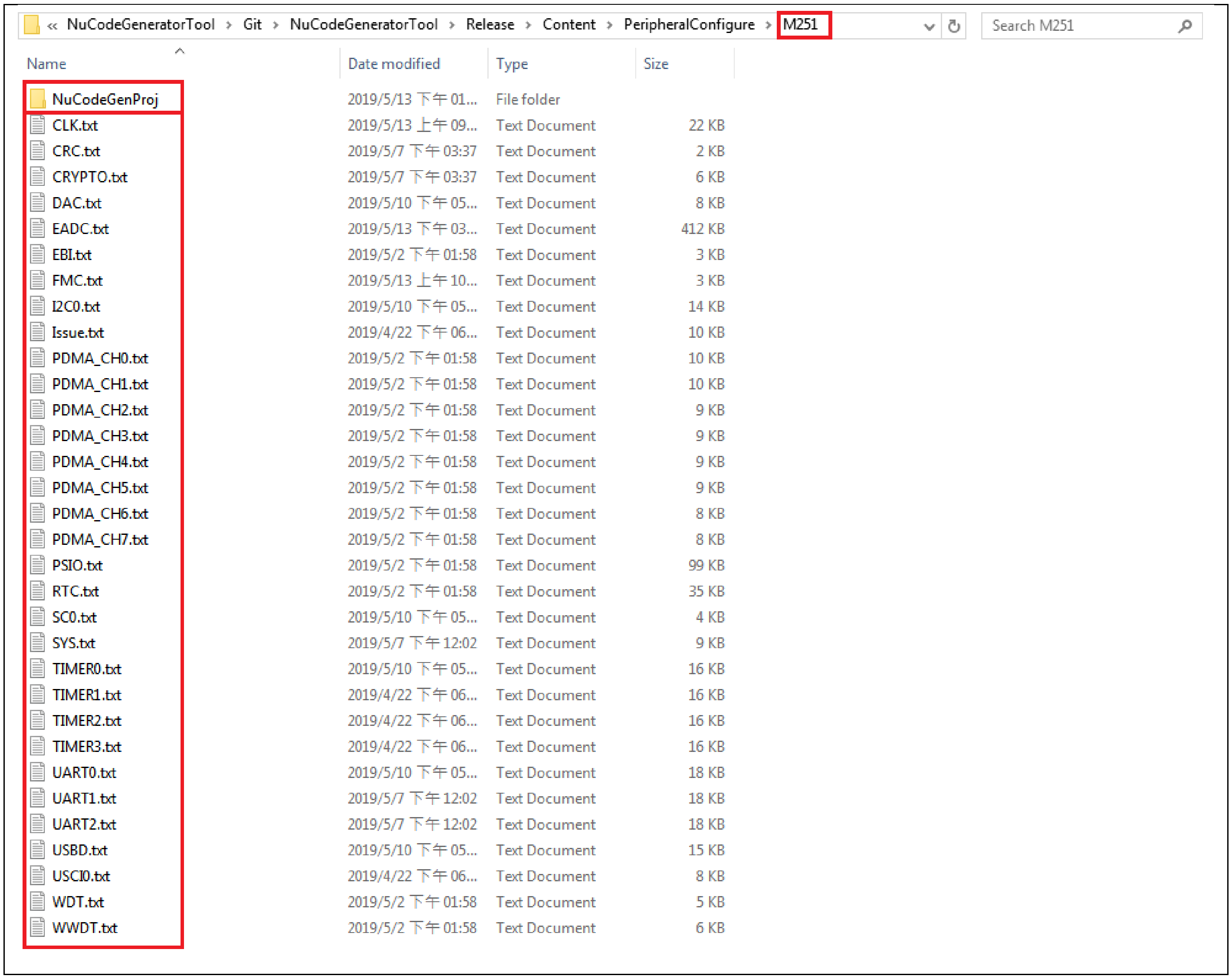
Document Information

Abstract	This document is written for BSP team members willing to contribute to NuCodeGenTool.
Apply to	NuTool – CodeGenerator_V0.42.0002

Table of Contents

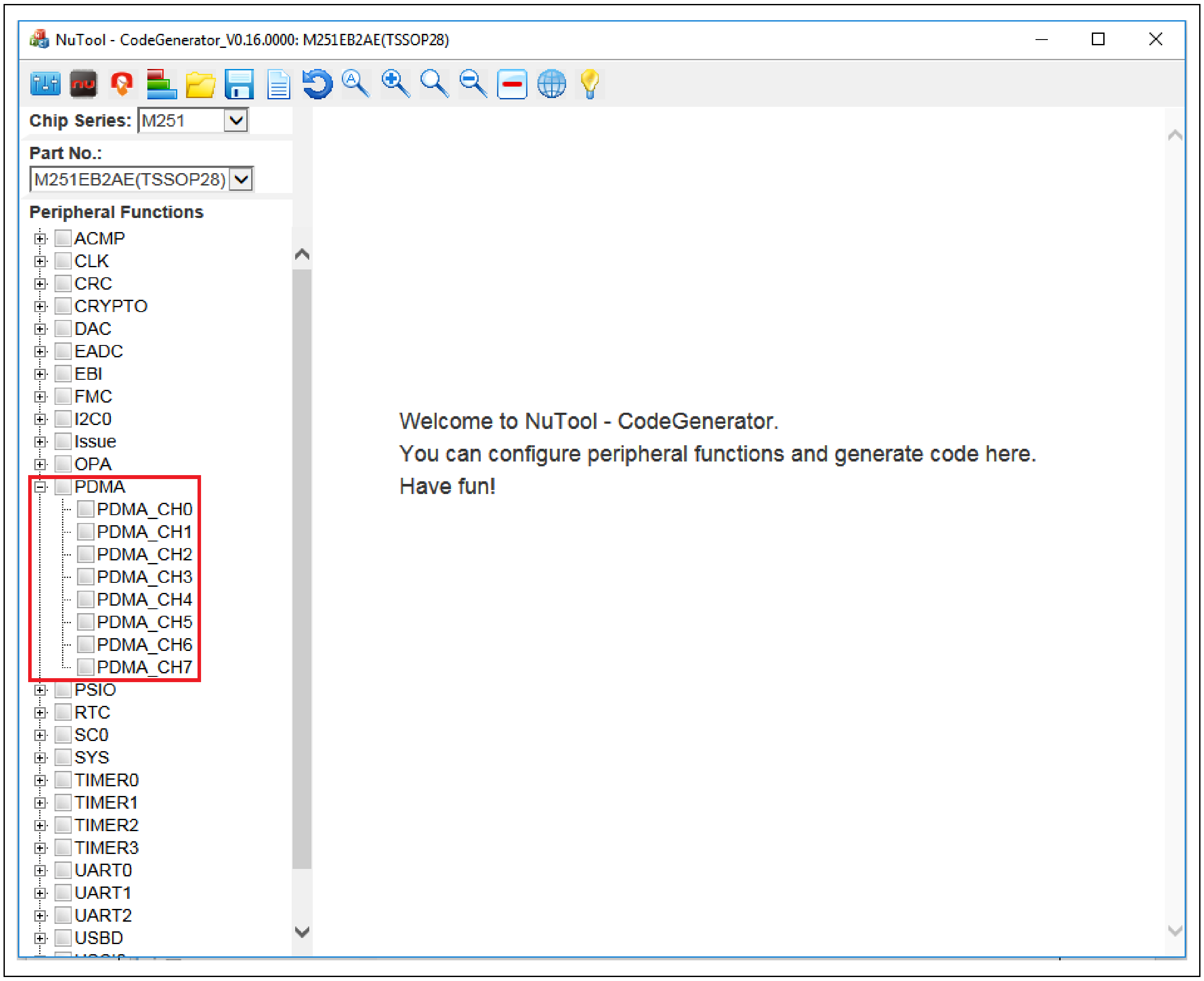
1 Introduction

NuTool – CodeGenerator, also called NuCodeGenTool, dynamically decides its UI and makes code generation. Take M251 as an example. In the folder of Content/PeripheralConfigure/M251, there are .txt files and NuCodeGenProj subfolder. When generating code, the NuCodeGenProj subfolder will be copied into the path specified by the user. The .txt files describe tags used to dynamically create the UI of NuCodeGenTool.


« NuCodeGeneratorTool › Git › NuCodeGeneratorTool › Release › Content › PeripheralConfigure › M251
Search M251

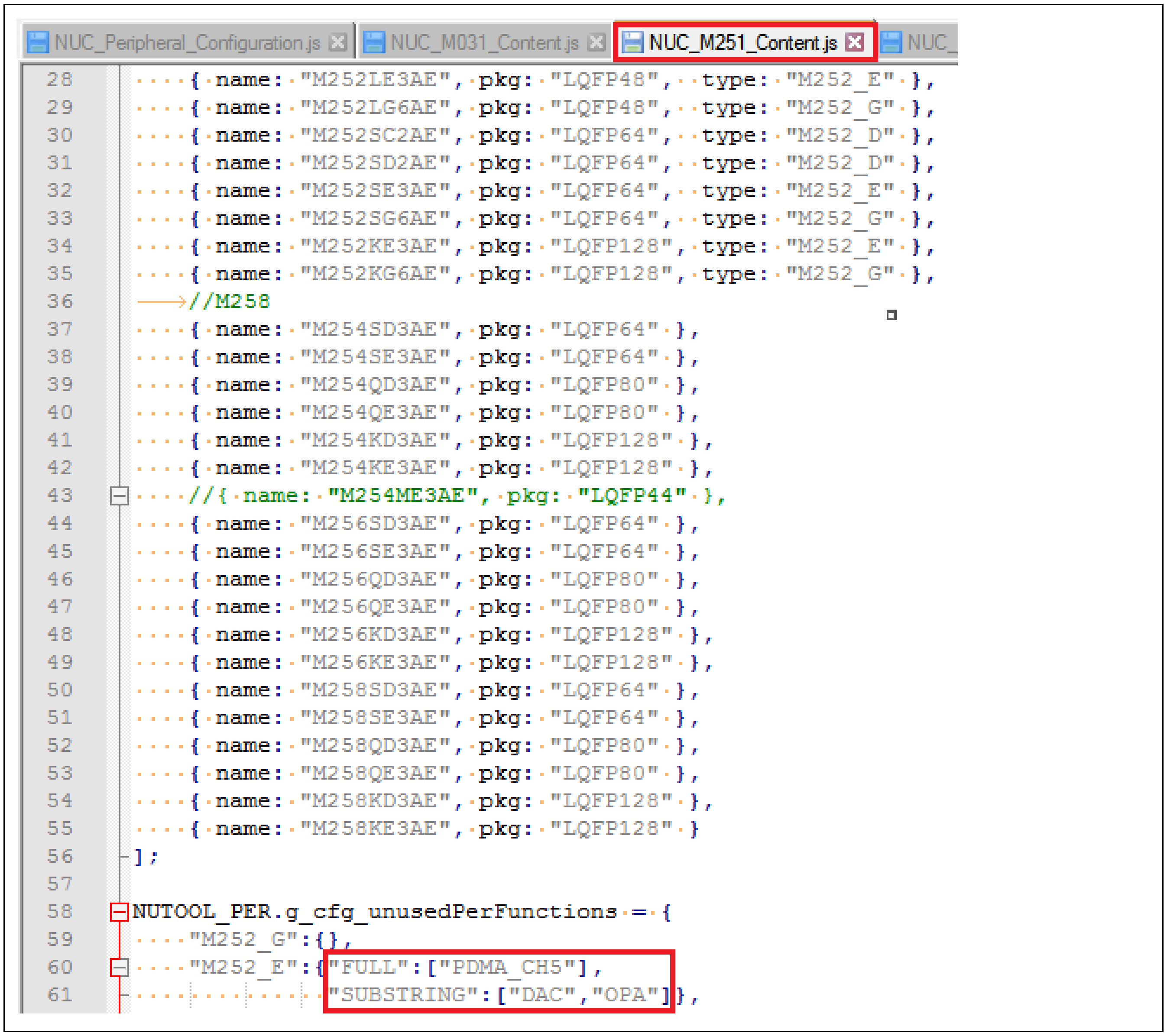
Name	Date modified	Type	Size
NuCodeGenProj	2019/5/13 下午 01...	File folder	
CLK.txt	2019/5/13 上午 09...	Text Document	22 KB
CRC.txt	2019/5/7 下午 03:37	Text Document	2 KB
CRYPTO.txt	2019/5/7 下午 03:37	Text Document	6 KB
DAC.txt	2019/5/10 下午 05...	Text Document	8 KB
EADC.txt	2019/5/13 下午 03...	Text Document	412 KB
EBI.txt	2019/5/2 下午 01:58	Text Document	3 KB
FMC.txt	2019/5/13 上午 10...	Text Document	3 KB
I2C0.txt	2019/5/10 下午 05...	Text Document	14 KB
Issue.txt	2019/4/22 下午 06...	Text Document	10 KB
PDMA_CH0.txt	2019/5/2 下午 01:58	Text Document	10 KB
PDMA_CH1.txt	2019/5/2 下午 01:58	Text Document	10 KB
PDMA_CH2.txt	2019/5/2 下午 01:58	Text Document	9 KB
PDMA_CH3.txt	2019/5/2 下午 01:58	Text Document	9 KB
PDMA_CH4.txt	2019/5/2 下午 01:58	Text Document	9 KB
PDMA_CH5.txt	2019/5/2 下午 01:58	Text Document	9 KB
PDMA_CH6.txt	2019/5/2 下午 01:58	Text Document	8 KB
PDMA_CH7.txt	2019/5/2 下午 01:58	Text Document	8 KB
PSIO.txt	2019/5/2 下午 01:58	Text Document	99 KB
RTC.txt	2019/5/2 下午 01:58	Text Document	35 KB
SC0.txt	2019/5/10 下午 05...	Text Document	4 KB
SYS.txt	2019/5/7 下午 12:02	Text Document	9 KB
TIMER0.txt	2019/5/10 下午 05...	Text Document	16 KB
TIMER1.txt	2019/4/22 下午 06...	Text Document	16 KB
TIMER2.txt	2019/4/22 下午 06...	Text Document	16 KB
TIMER3.txt	2019/4/22 下午 06...	Text Document	16 KB
UART0.txt	2019/5/10 下午 05...	Text Document	18 KB
UART1.txt	2019/5/7 下午 12:02	Text Document	18 KB
UART2.txt	2019/5/7 下午 12:02	Text Document	18 KB
USBD.txt	2019/5/10 下午 05...	Text Document	15 KB
USCI0.txt	2019/4/22 下午 06...	Text Document	8 KB
WDT.txt	2019/5/2 下午 01:58	Text Document	5 KB
WWDT.txt	2019/5/2 下午 01:58	Text Document	6 KB

If the file name of .txt files has an underline (_), NuCodeGenTool regards the name prior to the underline as the first layer name of Peripheral Functions tree. All .txt files with the same first layer name will be grouped into the second layer. For now, NuCodeGenTool only supports two-layer structure of Peripheral Functions tree.



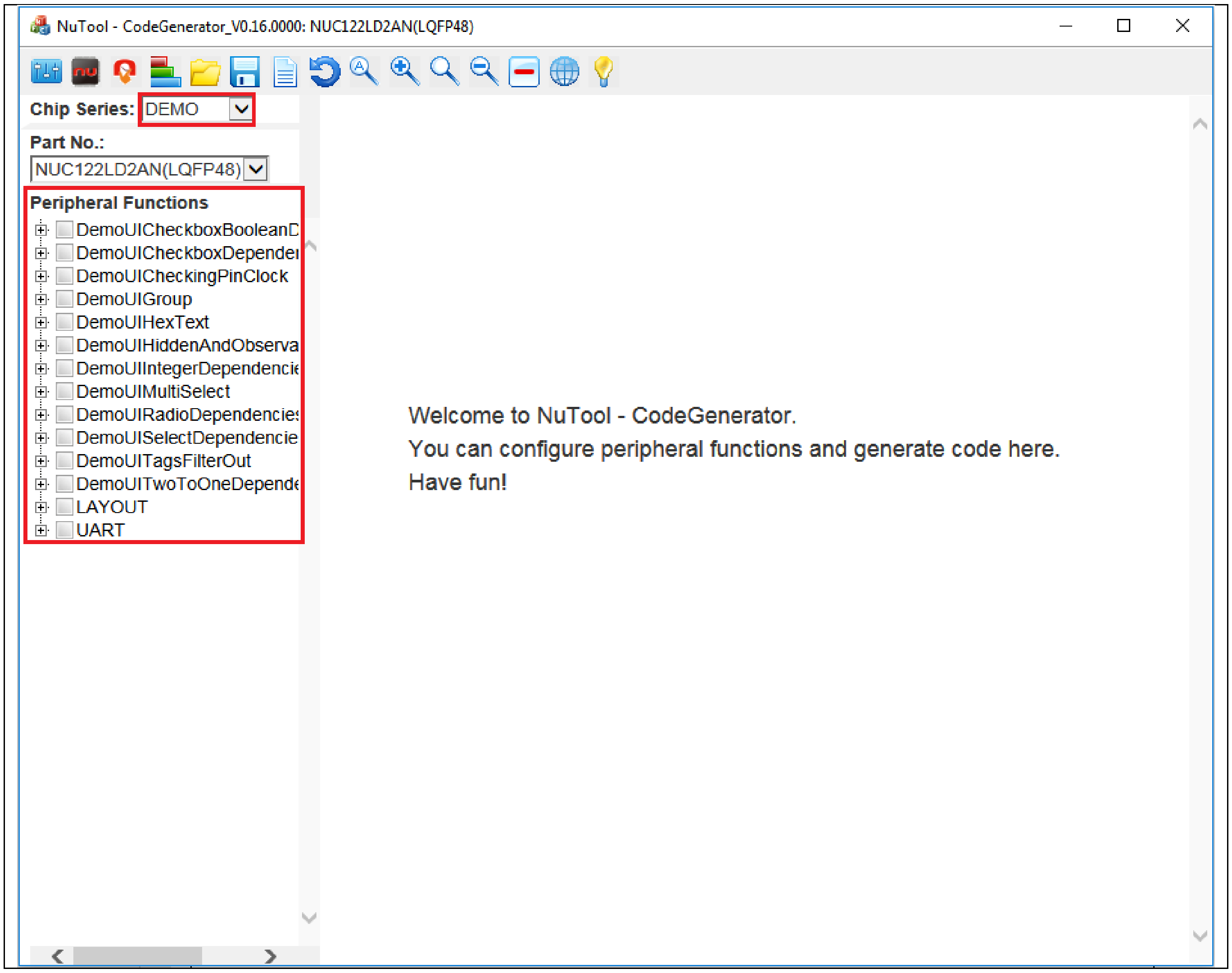
Welcome to NuTool - CodeGenerator.
You can configure peripheral functions and generate code here.
Have fun!

Some chips need not use all the .txt files because they might not support several of .txt files. Take M251 as an example. To filter out unsupported .txt files, please update the content of unusedPerFunctions in the Content/PeripheralConfigure/NUC_M251_Content.js. The SUBSTRING object defines an array of partially matched names of .txt files to remove, e.g. DACxxx and OPAxxx. The FULL object defines an array of fully matched names of .txt files to remove, e.g. PDMA_CH5. When chips whose part number or type are described in the content of unusedPerFunctions, NuCodeGenTool will filter out the related .txt files. When part number and type match twice, the tool will filter out the corresponding .txt files twice.



```
    { name: "M252LE3AE", pkg: "LQFP48",  type: "M252_E" },
    { name: "M252LG6AE", pkg: "LQFP48",  type: "M252_G" },
    { name: "M252SC2AE", pkg: "LQFP64",  type: "M252_D" },
    { name: "M252SD2AE", pkg: "LQFP64",  type: "M252_D" },
    { name: "M252SE3AE", pkg: "LQFP64",  type: "M252_E" },
    { name: "M252SG6AE", pkg: "LQFP64",  type: "M252_G" },
    { name: "M252KE3AE", pkg: "LQFP128", type: "M252_E" },
    { name: "M252KG6AE", pkg: "LQFP128", type: "M252_G" },
    //M258
    { name: "M254SD3AE", pkg: "LQFP64" },
    { name: "M254SE3AE", pkg: "LQFP64" },
    { name: "M254QD3AE", pkg: "LQFP80" },
    { name: "M254QE3AE", pkg: "LQFP80" },
    { name: "M254KD3AE", pkg: "LQFP128" },
    { name: "M254KE3AE", pkg: "LQFP128" },
    //{ name: "M254ME3AE", pkg: "LQFP44" },
    { name: "M256SD3AE", pkg: "LQFP64" },
    { name: "M256SE3AE", pkg: "LQFP64" },
    { name: "M256QD3AE", pkg: "LQFP80" },
    { name: "M256QE3AE", pkg: "LQFP80" },
    { name: "M256KD3AE", pkg: "LQFP128" },
    { name: "M256KE3AE", pkg: "LQFP128" },
    { name: "M258SD3AE", pkg: "LQFP64" },
    { name: "M258SE3AE", pkg: "LQFP64" },
    { name: "M258QD3AE", pkg: "LQFP80" },
    { name: "M258QE3AE", pkg: "LQFP80" },
    { name: "M258KD3AE", pkg: "LQFP128" },
    { name: "M258KE3AE", pkg: "LQFP128" }
];

NUTOOL_PER.g_cfg_unusedPerFunctions = {
    "M252_G":{},
    "M252_E":{"FULL":["PDMA_CH5"],
              "SUBSTRING":["DAC","OPA"]},
```

If you want to know what UI or other functions NuCodeGenTool has, please select DEMO from the select field of Chips Series. From DEMO, each UI and function will be introduced in detail. Therefore, it is highly recommended that every newcomer to NuCodeGenTool thoroughly study all examples in DEMO for future development of NuCodeGenTool.


NuTool - CodeGenerator_V0.16.0000: NUC122LD2AN(LQFP48)

Chip Series: DEMO

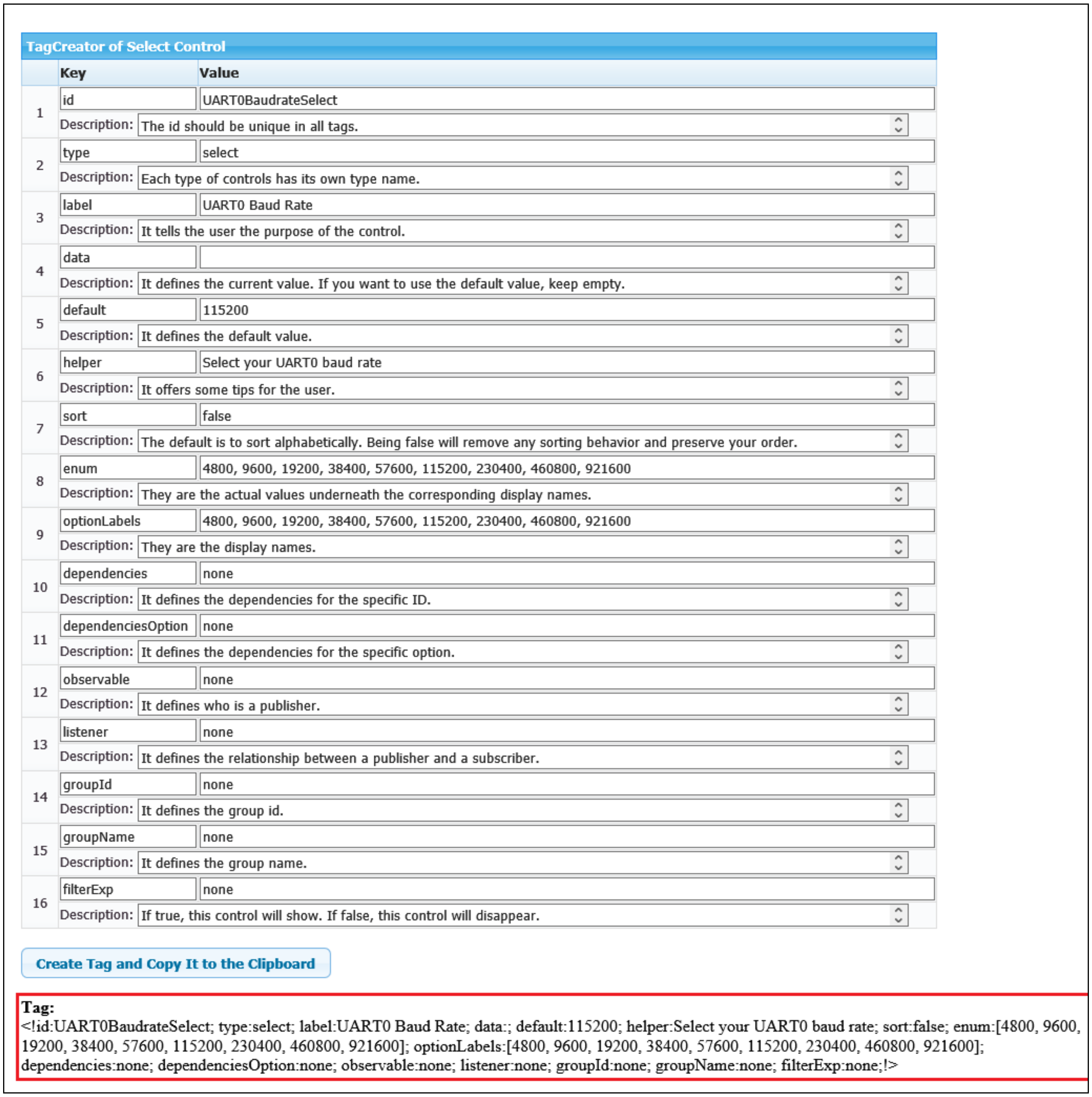
Part No.: NUC122LD2AN(LQFP48)

Peripheral Functions
- DemoUICheckboxBooleanD
- DemoUICheckboxDepender
- DemoUICheckingPinClock
- DemoUIGroup
- DemoUIHexText
- DemoUIHiddenAndObserva
- DemoUIIntegerDependencie
- DemoUIMultiSelect
- DemoUIRadioDependencies
- DemoUISelectDependencie
- DemoUITagsFilterOut
- DemoUITwoToOneDepende
- LAYOUT
- UART

Welcome to NuTool - CodeGenerator.
You can configure peripheral functions and generate code here.
Have fun!

2 TagCreator

For those not familiar with tag syntax, please utilize TagCreator to create tags. The tag syntax generated by TagCreator should be correct, so newcomers can check the correctness of tag syntax by using TagCreator. Each tag represents a UI. When generating code, tags will be replaced with data value or default value (referring to 3.1 Dependences).



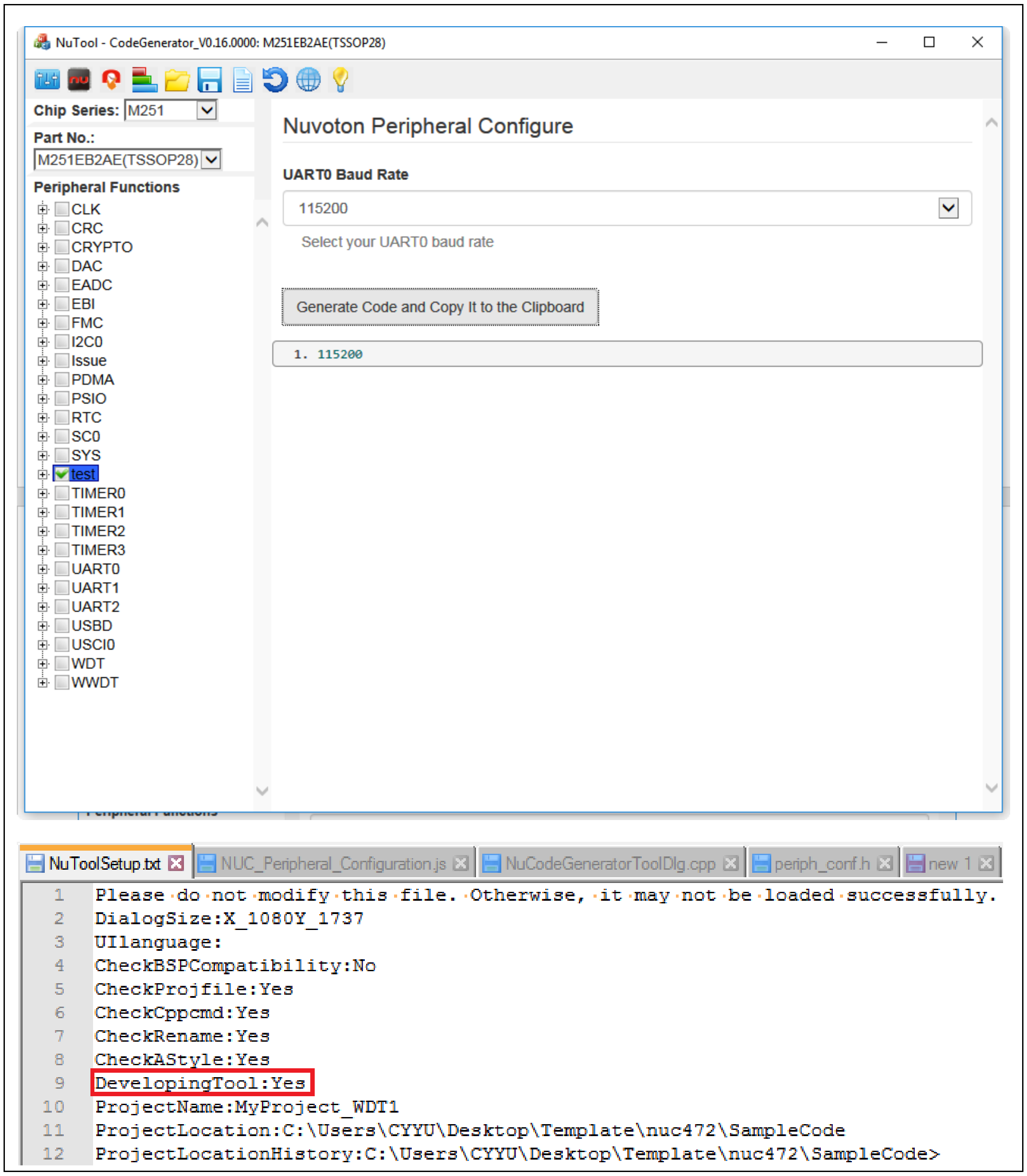
TagCreator of Select Control

	Key	Value
1	id	UART0BaudrateSelect
	Description:	The id should be unique in all tags.
2	type	select
	Description:	Each type of controls has its own type name.
3	label	UART0 Baud Rate
	Description:	It tells the user the purpose of the control.
4	data	
	Description:	It defines the current value. If you want to use the default value, keep empty.
5	default	115200
	Description:	It defines the default value.
6	helper	Select your UART0 baud rate
	Description:	It offers some tips for the user.
7	sort	false
	Description:	The default is to sort alphabetically. Being false will remove any sorting behavior and preserve your order.
8	enum	4800, 9600, 19200, 38400, 57600, 115200, 230400, 460800, 921600
	Description:	They are the actual values underneath the corresponding display names.
9	optionLabels	4800, 9600, 19200, 38400, 57600, 115200, 230400, 460800, 921600
	Description:	They are the display names.
10	dependencies	none
	Description:	It defines the dependencies for the specific ID.
11	dependenciesOption	none
	Description:	It defines the dependencies for the specific option.
12	observable	none
	Description:	It defines who is a publisher.
13	listener	none
	Description:	It defines the relationship between a publisher and a subscriber.
14	groupId	none
	Description:	It defines the group id.
15	groupName	none
	Description:	It defines the group name.
16	filterExp	none
	Description:	If true, this control will show. If false, this control will disappear.

Create Tag and Copy It to the Clipboard

Tag:
<!id:UART0BaudrateSelect; type:select; label:UART0 Baud Rate; data:; default:115200; helper:Select your UART0 baud rate; sort:false; enum:[4800, 9600, 19200, 38400, 57600, 115200, 230400, 460800, 921600]; optionLabels:[4800, 9600, 19200, 38400, 57600, 115200, 230400, 460800, 921600]; dependencies:none; dependenciesOption:none; observable:none; listener:none; groupId:none; groupName:none; filterExp:none;!>

After clicking on the button of Create Tag and Copy It to the Clipboard in TagCreator, paste the result onto a test.txt file. Move the test.txt file into Content/PeripheralConfigure/M251 folder, so that NuCodeGenTool can display the UI described in the test.txt file. If NuCodeGenTool is in release mode, the button of Create Tag and Copy It to the Clipboard will be hidden. To reveal it, open Content/NuToolSetup.txt file, input DevelopingTool:Yes and then relaunch the tool.



UART0 Baud Rate

115200

Select your UART0 baud rate

```
1. 115200
```

```
Please do not modify this file. Otherwise, it may not be loaded successfully.
DialogSize:X_1080Y_1737
UIlanguage:
CheckBSPCompatibility:No
CheckProjfile:Yes
CheckCppcmd:Yes
CheckRename:Yes
CheckAStyle:Yes
DevelopingTool:Yes
ProjectName:MyProject_WDT1
ProjectLocation:C:\Users\CYYU\Desktop\Template\nuc472\SampleCode
ProjectLocationHistory:C:\Users\CYYU\Desktop\Template\nuc472\SampleCode>
```

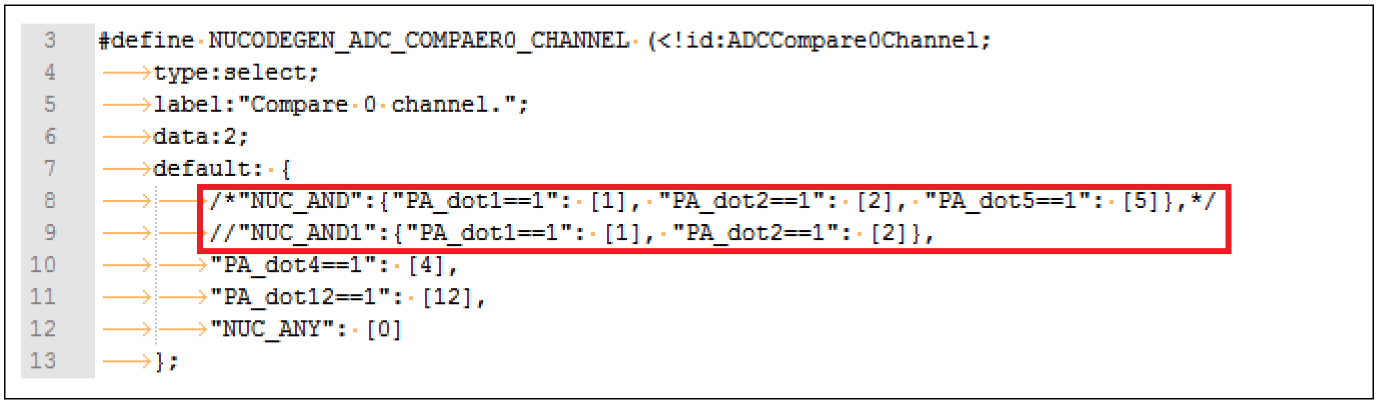
3 Tag Syntax

1. The start and end symbols of tags are <! and !>, and the content in the middle is free to edit.
2. Field of id must be unique.
3. Field of type determines the type of UI.
4. Field of label determines the description text shown around the UI.
5. Field of data stores the current value of UI. If updated, new value will be written back.
6. Field of default stores the default value of UI.
7. Field of helper determines the helper text following UI.
8. Field of sort determines the order of optionLabels. If yes, alphabetical order; If no, array order.
9. Field of enum determines the option data array.
10. Field of optionLabels determines the option label array which should corresponds to enum.
11. Field of vertical determines the UI layout. If yes, vertical layout. If not, horizontal layout.
12. Field of maximum determines the maximum input value.
13. Field of minimum determines the minimum input value.
14. Field of validate determines whether or not to validate input value. If not, empty input would be replaced with the default value.

Each field name (e.g. optionLabels) of tag is fixed and case sensitive. Please use TagCreator to check the correctness. Each field should end with semicolon. If the value of field is array, it should be enclosed in brackets. If field is unused, the input should be none.

```
<!id:UART0BaudrateSelect;
type:select;
label:UART0 Baud Rate;
data:;
default:115200;
helper:Select your UART0 baud rate;
sort:false;
enum:[4800, 9600, 19200, 38400, 57600, 115200, 230400, 460800, 921600];
optionLabels:[4800, 9600, 19200, 38400, 57600, 115200, 230400, 460800, 921600];
dependencies:none;
dependenciesOption:none;
observable:none;
listener:none;
groupId:none;
groupName:none;
filterExp:none;!>
```

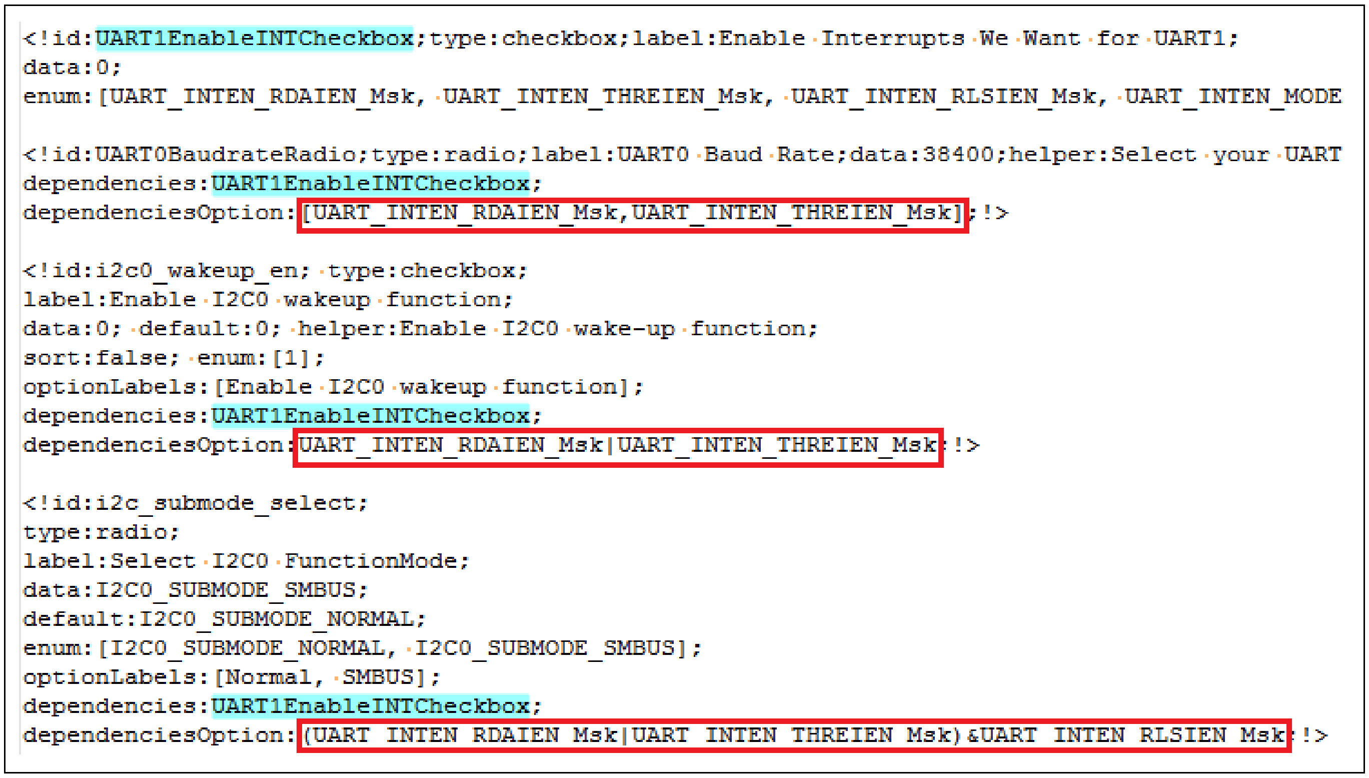
To comment out one or more lines in a tag, both of the leading characters // and block comment multiple lines of code using the characters /* */ are supported, as follows.



```
#define NUCODEGEN_ADC_COMPAER0_CHANNEL (<!id:ADCCompare0Channel;
	type:select;
	label:"Compare 0 channel.";
	data:2;
	default: {
		/*"NUC_AND":{"PA_dot1==1": [1], "PA_dot2==1": [2], "PA_dot5==1": [5]},*/
		//"NUC_AND1":{"PA_dot1==1": [1], "PA_dot2==1": [2]},
		"PA_dot4==1": [4],
		"PA_dot12==1": [12],
		"NUC_ANY": [0]
	};
```

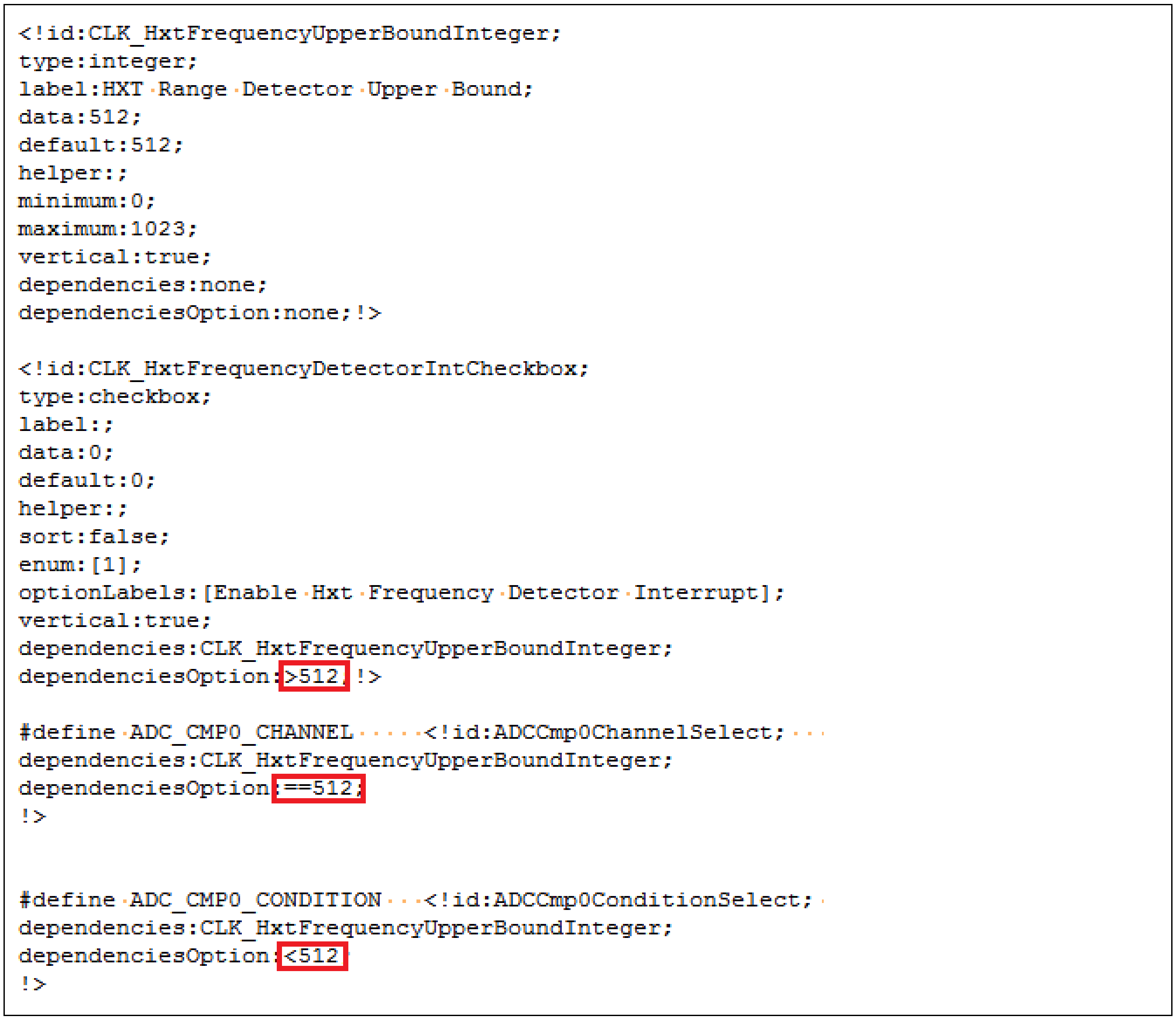
3.1 Dependencies

UI renders only when data of another UI meets some kind of conditions. These conditions are called Dependencies. Firstly, input the id of dependent UI in the dependencies field, and then input expression in the dependenciesOption field. If the expression is an array and the data of dependent UI completely match the array, the UI renders. If the expression has bitwise operation and the data of dependent UI makes the bitwise operation satisfied, the UI renders. When UI renders, the tool uses the data to generate code. When UI hides and its dependenciesDefault is true, the tool uses the default to generate code. When UI hides and its dependenciesDefault is false, the tool simply removes the tag in the generated code.



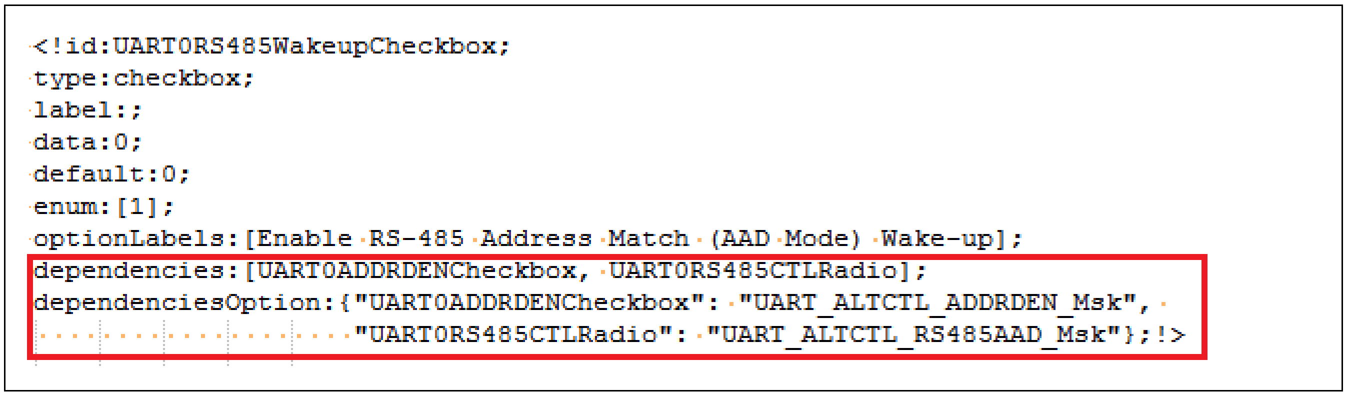
```
<!id:UART1EnableINTCheckbox;type:checkbox;label:Enable Interrupts We Want for UART1;
data:0;
enum:[UART_INTEN_RDAIEN_Msk, UART_INTEN_THREIEN_Msk, UART_INTEN_RLSIEN_Msk, UART_INTEN_MODE

<!id:UART0BaudrateRadio;type:radio;label:UART0 Baud Rate;data:38400;helper:Select your UART
dependencies:UART1EnableINTCheckbox;
dependenciesOption:[UART_INTEN_RDAIEN_Msk,UART_INTEN_THREIEN_Msk];!>

<!id:i2c0_wakeup_en; type:checkbox;
label:Enable I2C0 wakeup function;
data:0; default:0; helper:Enable I2C0 wake-up function;
sort:false; enum:[1];
optionLabels:[Enable I2C0 wakeup function];
dependencies:UART1EnableINTCheckbox;
dependenciesOption:UART_INTEN_RDAIEN_Msk|UART_INTEN_THREIEN_Msk;!>

<!id:i2c_submode_select;
type:radio;
label:Select I2C0 FunctionMode;
data:I2C0_SUBMODE_SMBUS;
default:I2C0_SUBMODE_NORMAL;
enum:[I2C0_SUBMODE_NORMAL, I2C0_SUBMODE_SMBUS];
optionLabels:[Normal, SMBUS];
dependencies:UART1EnableINTCheckbox;
dependenciesOption:(UART_INTEN_RDAIEN_Msk|UART_INTEN_THREIEN_Msk)&UART_INTEN_RLSIEN_Msk;!>
```

Expression supports Greater than, Equal to, and Less than only when the operator belongs to number.



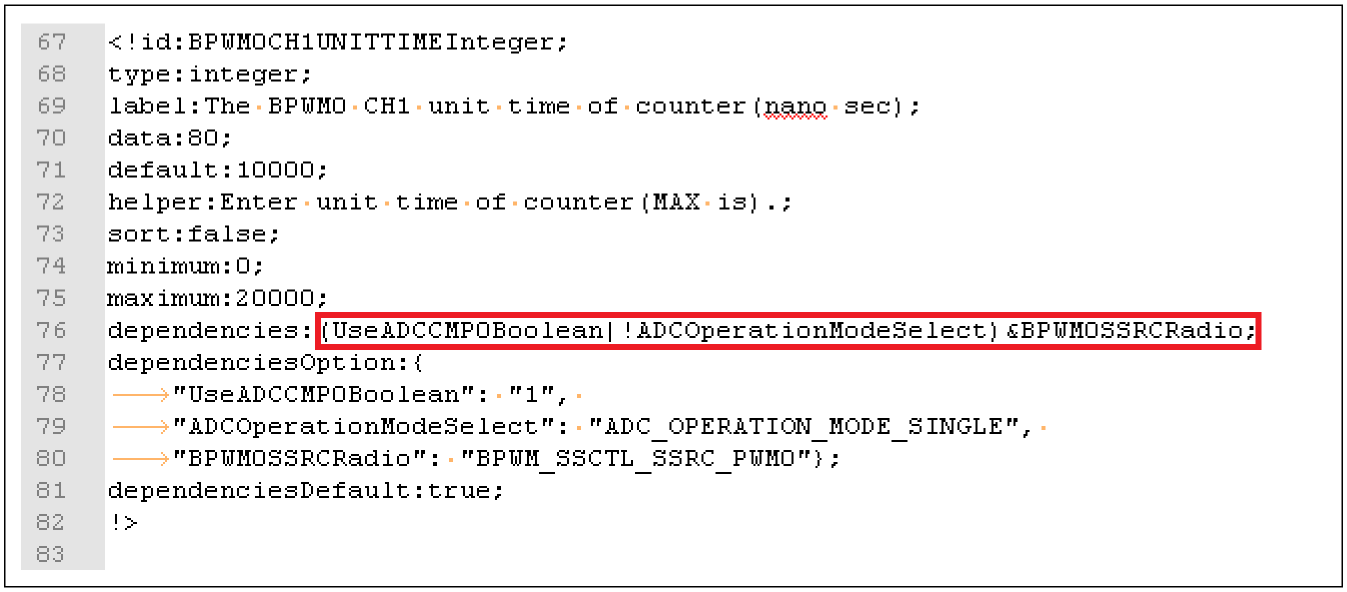
```
<!id:CLK_HxtFrequencyUpperBoundInteger;
type:integer;
label:HXT Range Detector Upper Bound;
data:512;
default:512;
helper:;
minimum:0;
maximum:1023;
vertical:true;
dependencies:none;
dependenciesOption:none;!>

<!id:CLK_HxtFrequencyDetectorIntCheckbox;
type:checkbox;
label:;
data:0;
default:0;
helper:;
sort:false;
enum:[1];
optionLabels:[Enable Hxt Frequency Detector Interrupt];
vertical:true;
dependencies:CLK_HxtFrequencyUpperBoundInteger;
dependenciesOption:>512;!>

#define ADC_CMP0_CHANNEL     <!id:ADCCmp0ChannelSelect;   
dependencies:CLK_HxtFrequencyUpperBoundInteger;
dependenciesOption:==512;
!>


#define ADC_CMP0_CONDITION   <!id:ADCCmp0ConditionSelect; 
dependencies:CLK_HxtFrequencyUpperBoundInteger;
dependenciesOption:<512;
!>
```

When the number of dependent UI is more than one and the dependencies field is an array, the dependenciesOption field needs JavaScript object to specify. When all dependencies conditions are met, the UI renders.



```
<!id:UART0RS485WakeupCheckbox;
type:checkbox;
label:;
data:0;
default:0;
enum:[1];
optionLabels:[Enable RS-485 Address Match (AAD Mode) Wake-up];
dependencies:[UART0ADDRDENCheckbox, UART0RS485CTLRadio];
dependenciesOption:{"UART0ADDRDENCheckbox": "UART_ALTCTL_ADDRDEN_Msk",
                    "UART0RS485CTLRadio": "UART_ALTCTL_RS485AAD_Msk"};!>
```

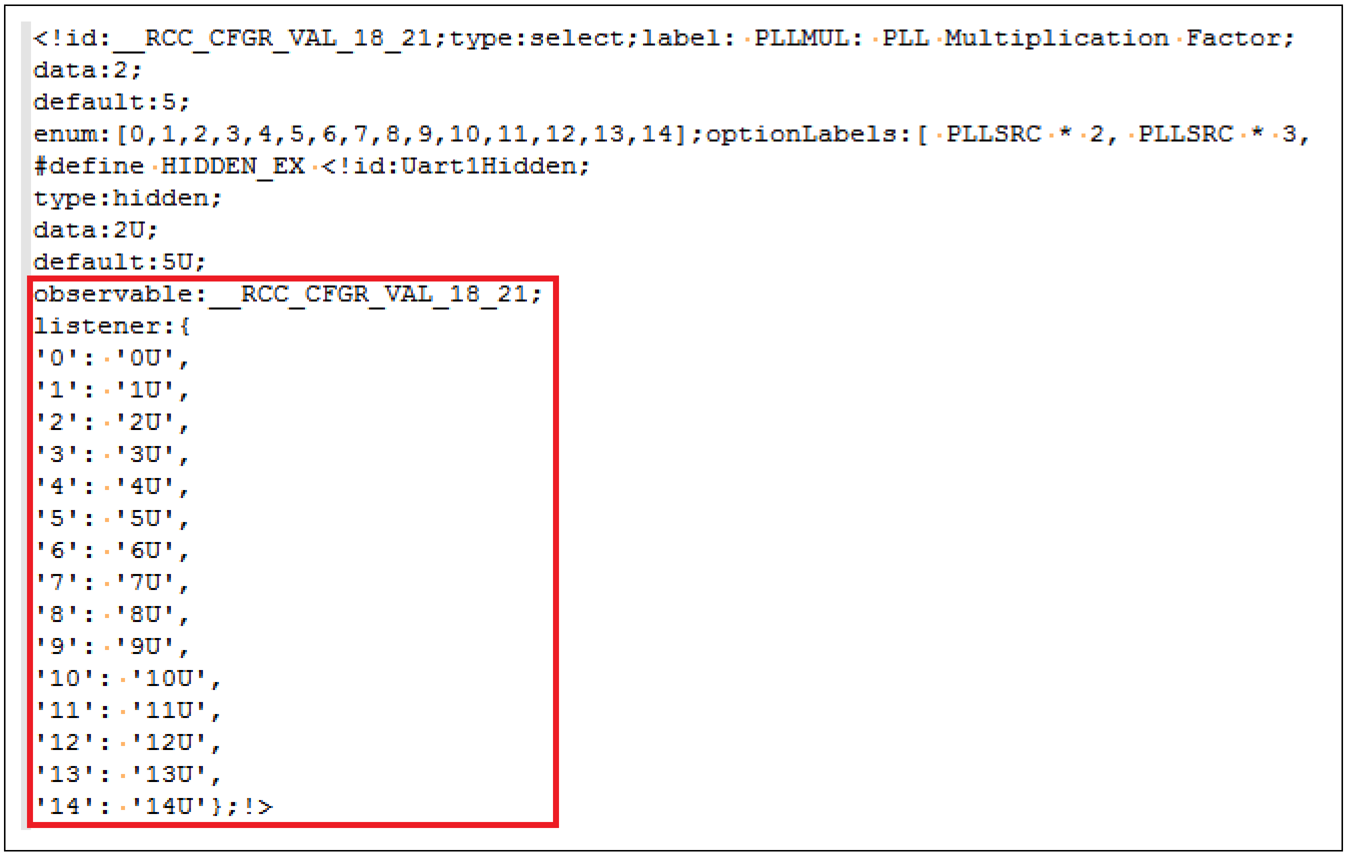
When the number of dependent UI is more than one and the dependencies field has bitwise operation, the dependenciesOption field needs JavaScript object to specify. When the bitwise operation is met, the UI renders. The supported bitwise operation has Boolean or (|), Boolean and (&) and Boolean not (!).



```
<!id:BPWM0CH1UNITTIMEInteger;
type:integer;
label:The BPWM0 CH1 unit time of counter(nano sec);
data:80;
default:10000;
helper:Enter unit time of counter(MAX is).;
sort:false;
minimum:0;
maximum:20000;
dependencies:(UseADCCMP0Boolean|!ADCOperationModeSelect)&BPWM0SSRCRadio;
dependenciesOption:{
	"UseADCCMP0Boolean": "1", 
	"ADCOperationModeSelect": "ADC_OPERATION_MODE_SINGLE", 
	"BPWM0SSRCRadio": "BPWM_SSCTL_SSRC_PWM0"};
dependenciesDefault:true;
!>
```

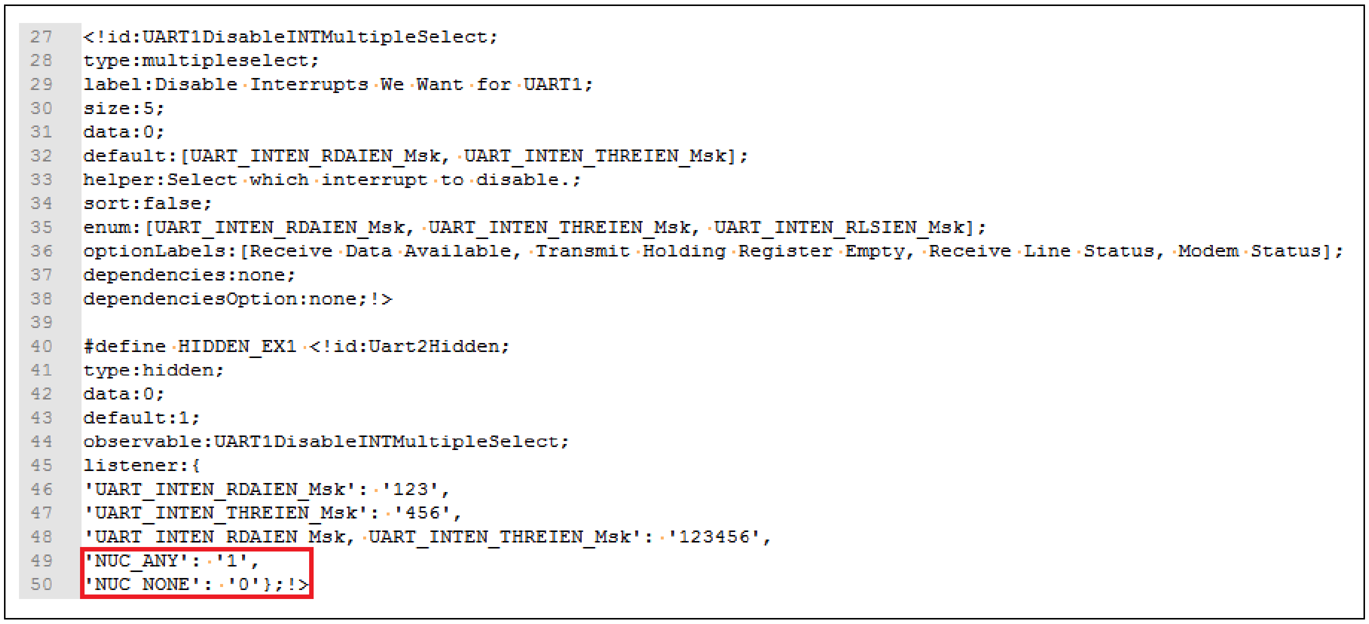
3.2 Observable

The data of UI could come from that of another UI by means of so-called Observable. Take the following figure as an example. Although a tag of hidden type does not render, it still can be used to generate code. The tag of hidden type must utilize observable, or its data is always fixed. Input the id of observable in the observable field, and input the observable relationship table in the listener field. In addition to data, the default of tag follows the observable relationship table to change, so we need not input the default by ourselves. Please note that tags of all types can utilize observable.



```
<!id:__RCC_CFGR_VAL_18_21;type:select;label: PLLMUL: PLL Multiplication Factor;
data:2;
default:5;
enum:[0,1,2,3,4,5,6,7,8,9,10,11,12,13,14];optionLabels:[ PLLSRC * 2, PLLSRC * 3,
#define HIDDEN_EX <!id:Uart1Hidden;
type:hidden;
data:2U;
default:5U;
observable:__RCC_CFGR_VAL_18_21;
listener:{
'0': '0U',
'1': '1U',
'2': '2U',
'3': '3U',
'4': '4U',
'5': '5U',
'6': '6U',
'7': '7U',
'8': '8U',
'9': '9U',
'10': '10U',
'11': '11U',
'12': '12U',
'13': '13U',
'14': '14U'};!>
```

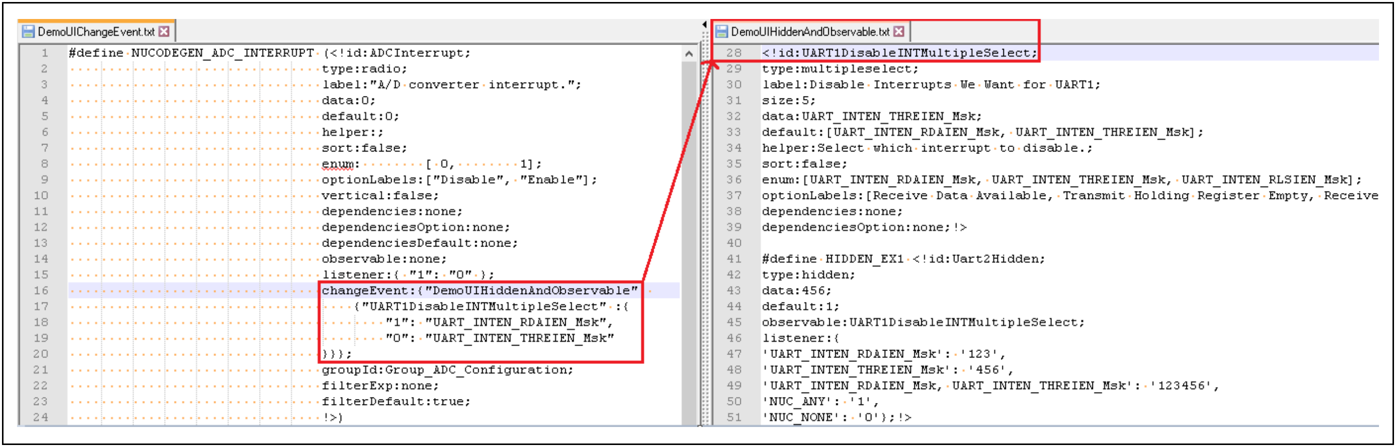
The observable relationship table in the listener field has two preserved key, i.e. NUC_ANY and NUC_NONE. When the data of observable is missing in the observable relationship table, the tool will take the value of NUC_ANY if existed. When observable does not generate any data, the tool will take the value of NUC_NONE if existed. Please note that NUC_ANY and NUC_NONE should be the last two properties in the observable relationship table.



```
<!id:UART1DisableINTMultipleSelect;
type:multipleselect;
label:Disable Interrupts We Want for UART1;
size:5;
data:0;
default:[UART_INTEN_RDAIEN_Msk, UART_INTEN_THREIEN_Msk];
helper:Select which interrupt to disable.;
sort:false;
enum:[UART_INTEN_RDAIEN_Msk, UART_INTEN_THREIEN_Msk, UART_INTEN_RLSIEN_Msk];
optionLabels:[Receive Data Available, Transmit Holding Register Empty, Receive Line Status, Modem Status];
dependencies:none;
dependenciesOption:none;!>

#define HIDDEN_EX1 <!id:Uart2Hidden;
type:hidden;
data:0;
default:1;
observable:UART1DisableINTMultipleSelect;
listener:{
'UART_INTEN_RDAIEN_Msk': '123',
'UART_INTEN_THREIEN_Msk': '456',
'UART_INTEN_RDAIEN_Msk, UART_INTEN_THREIEN_Msk': '123456',
'NUC_ANY': '1',
'NUC_NONE': '0'};!>
```

3.3 ChangeEvent

A tag can change the data of target tag in another .txt file by means of so-called ChangeEvent. Take the following figure as an example. Input the .txt file name and id of target tag in the changeEvent field, and specify the changeEvent relationship table. In addition to the target UI, its observable tag follows the observable relationship table to change automatically, so we need not input the changeEvent relationship table between the current tag and target tag's observable by ourselves. The changeEvent relationship table has two preserved key, i.e. NUC_ANY and NUC_NONE. Their functionality is the same as that of observable relationship table.

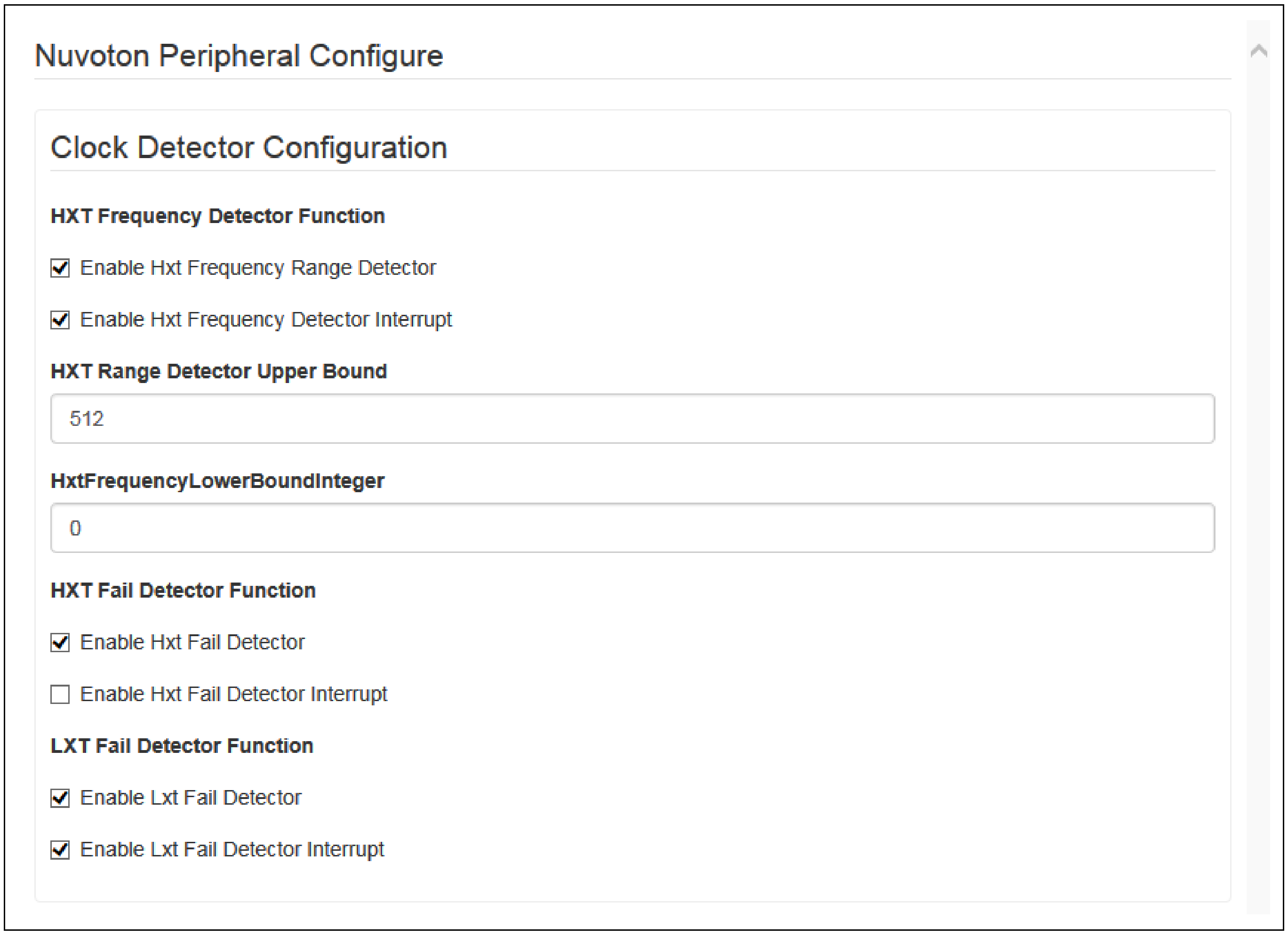


```
#define NUCODEGEN_ADC_INTERRUPT (<!id:ADCInterrupt;
                                type:radio;
                                label:"A/D converter interrupt.";
                                data:0;
                                default:0;
                                helper:;
                                sort:false;
                                enum:        [ 0,        1];
                                optionLabels:["Disable", "Enable"];
                                vertical:false;
                                dependencies:none;
                                dependenciesOption:none;
                                dependenciesDefault:none;
                                observable:none;
                                listener:{ "1": "0" };
                                changeEvent:{"DemoUIHiddenAndObservable"
                                    {"UART1DisableINTMultipleSelect" :{
                                        "1": "UART_INTEN_RDAIEN_Msk",
                                        "0": "UART_INTEN_THREIEN_Msk"
                                }}};
                                groupId:Group_ADC_Configuration;
                                filterExp:none;
                                filterDefault:true;
                                !>)
```

```
<!id:UART1DisableINTMultipleSelect;
type:multipleselect;
label:Disable Interrupts We Want for UART1;
size:5;
data:UART_INTEN_THREIEN_Msk;
default:[UART_INTEN_RDAIEN_Msk, UART_INTEN_THREIEN_Msk];
helper:Select which interrupt to disable.;
sort:false;
enum:[UART_INTEN_RDAIEN_Msk, UART_INTEN_THREIEN_Msk, UART_INTEN_RLSIEN_Msk];
optionLabels:[Receive Data Available, Transmit Holding Register Empty, Receive
dependencies:none;
dependenciesOption:none;!>

#define HIDDEN_EX1 <!id:Uart2Hidden;
type:hidden;
data:456;
default:1;
observable:UART1DisableINTMultipleSelect;
listener:{
'UART_INTEN_RDAIEN_Msk': '123',
'UART_INTEN_THREIEN_Msk': '456',
'UART_INTEN_RDAIEN_Msk, UART_INTEN_THREIEN_Msk': '123456',
'NUC_ANY': '1',
'NUC_NONE': '0'};!>
```

3.4 Group

UI could be grouped together as follows. To do that, input the same id in the groupID field. The first UI in the group determines the group name in the groupName field.



Nuvoton Peripheral Configure

Clock Detector Configuration

HXT Frequency Detector Function

- [✓] Enable Hxt Frequency Range Detector
- [✓] Enable Hxt Frequency Detector Interrupt

HXT Range Detector Upper Bound

512

HxtFrequencyLowerBoundInteger

0

HXT Fail Detector Function

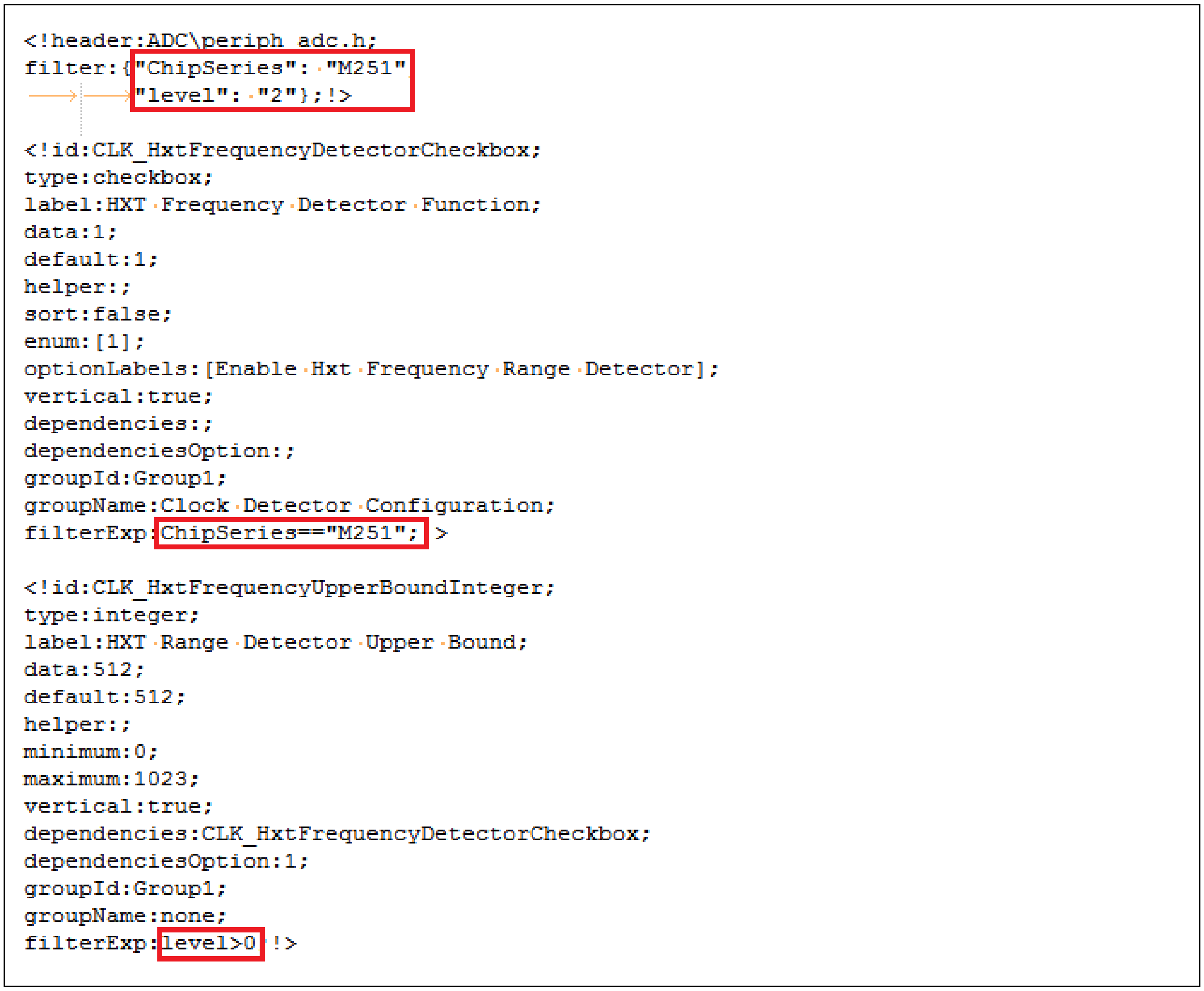
- [✓] Enable Hxt Fail Detector
- [ ] Enable Hxt Fail Detector Interrupt

LXT Fail Detector Function

- [✓] Enable Lxt Fail Detector
- [✓] Enable Lxt Fail Detector Interrupt

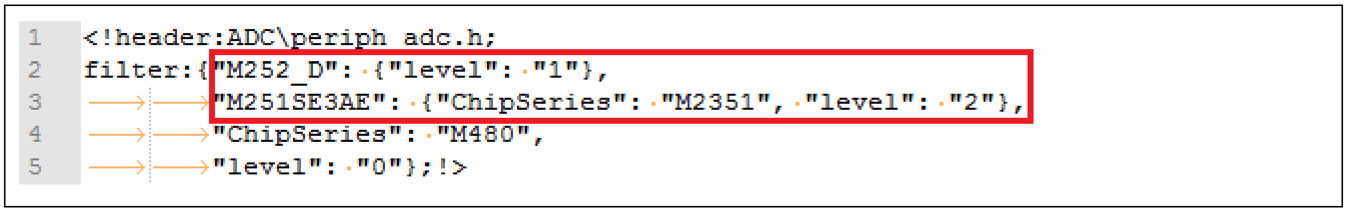
3.5 FilterExp

In .txt files, we can use filter tag to filter out unnecessary UI, as follows. If the filterExp expression exists and meets the condition, the UI renders. The general tag (referring to Q&A 4th question) including filter tag must be in the topmost place of whole txt file. When UI hides and its its filterDefault is true, the tool uses the default to generate code. When UI hides and its filterDefault is false, the tool simply removes the tag in the generated code.



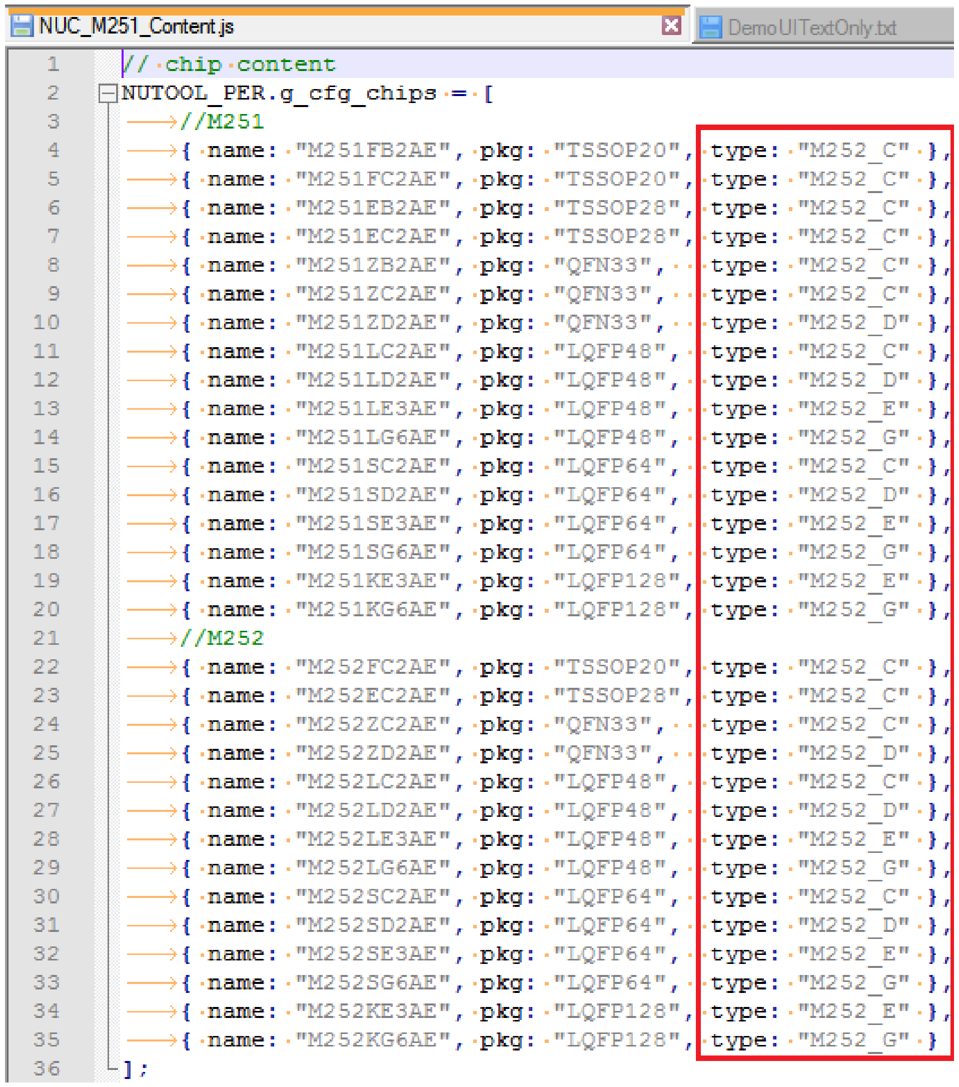
```
<!header:ADC\periph_adc.h;
filter:{"ChipSeries": "M251"
		"level": "2"};!>

<!id:CLK_HxtFrequencyDetectorCheckbox;
type:checkbox;
label:HXT Frequency Detector Function;
data:1;
default:1;
helper:;
sort:false;
enum:[1];
optionLabels:[Enable Hxt Frequency Range Detector];
vertical:true;
dependencies:;
dependenciesOption:;
groupId:Group1;
groupName:Clock Detector Configuration;
filterExp:ChipSeries=="M251"; !>

<!id:CLK_HxtFrequencyUpperBoundInteger;
type:integer;
label:HXT Range Detector Upper Bound;
data:512;
default:512;
helper:;
minimum:0;
maximum:1023;
vertical:true;
dependencies:CLK_HxtFrequencyDetectorCheckbox;
dependenciesOption:1;
groupId:Group1;
groupName:none;
filterExp:level>0 !>
```

In filter tag, we can use chip type or name to define filter conditions, as follows.



```
<!header:ADC\periph_adc.h;
filter:{"M252_D": {"level": "1"},
		"M251SE3AE": {"ChipSeries": "M2351", "level": "2"},
		"ChipSeries": "M480",
		"level": "0"};!>
```

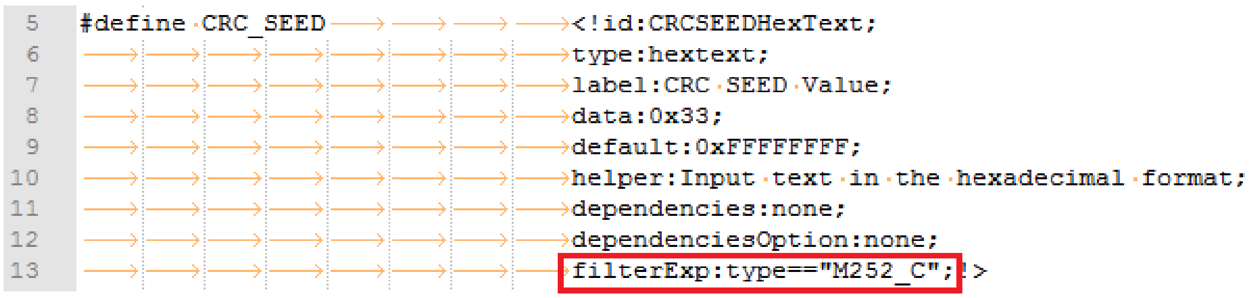
Take M251 as an example, the filter tag has built-in conditions which can be defined in Content/PeripheralConfigure/NUC_M251_Content.js, such as type=M252_C and name=M251FB2AE. In addition, GPIO pins have built-in condition called Px_dotx. If PA.0 exists, PA_dot0=1; if not, PA_dot0=0.



NUC_M251_Content.js

DemoUITextOnly.txt

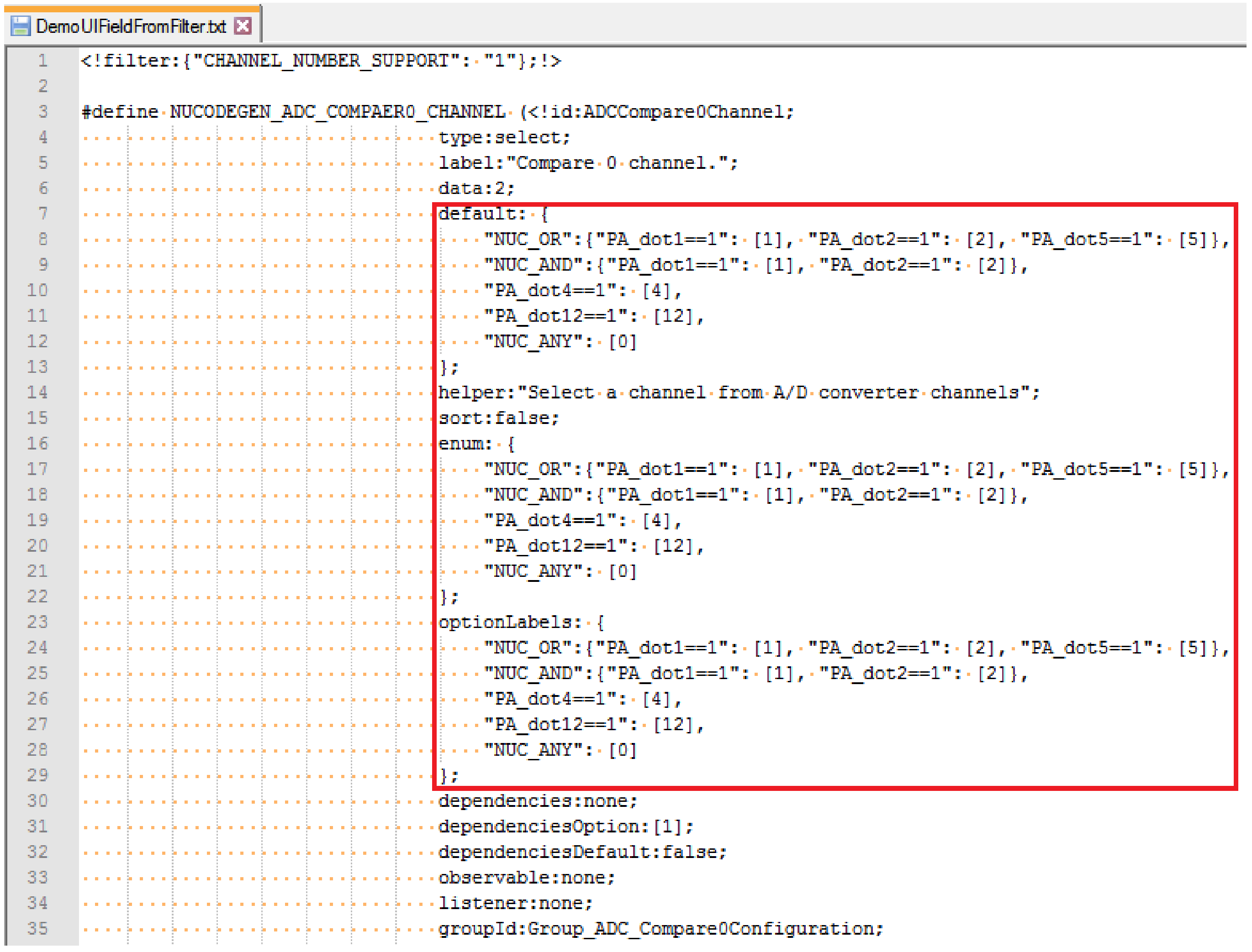
```
// chip content
NUTOOL_PER.g_cfg_chips = [
    //M251
    { name: "M251FB2AE", pkg: "TSSOP20", type: "M252_C" },
    { name: "M251FC2AE", pkg: "TSSOP20", type: "M252_C" },
    { name: "M251EB2AE", pkg: "TSSOP28", type: "M252_C" },
    { name: "M251EC2AE", pkg: "TSSOP28", type: "M252_C" },
    { name: "M251ZB2AE", pkg: "QFN33",   type: "M252_C" },
    { name: "M251ZC2AE", pkg: "QFN33",   type: "M252_C" },
    { name: "M251ZD2AE", pkg: "QFN33",   type: "M252_D" },
    { name: "M251LC2AE", pkg: "LQFP48",  type: "M252_C" },
    { name: "M251LD2AE", pkg: "LQFP48",  type: "M252_D" },
    { name: "M251LE3AE", pkg: "LQFP48",  type: "M252_E" },
    { name: "M251LG6AE", pkg: "LQFP48",  type: "M252_G" },
    { name: "M251SC2AE", pkg: "LQFP64",  type: "M252_C" },
    { name: "M251SD2AE", pkg: "LQFP64",  type: "M252_D" },
    { name: "M251SE3AE", pkg: "LQFP64",  type: "M252_E" },
    { name: "M251SG6AE", pkg: "LQFP64",  type: "M252_G" },
    { name: "M251KE3AE", pkg: "LQFP128", type: "M252_E" },
    { name: "M251KG6AE", pkg: "LQFP128", type: "M252_G" },
    //M252
    { name: "M252FC2AE", pkg: "TSSOP20", type: "M252_C" },
    { name: "M252EC2AE", pkg: "TSSOP28", type: "M252_C" },
    { name: "M252ZC2AE", pkg: "QFN33",   type: "M252_C" },
    { name: "M252ZD2AE", pkg: "QFN33",   type: "M252_D" },
    { name: "M252LC2AE", pkg: "LQFP48",  type: "M252_C" },
    { name: "M252LD2AE", pkg: "LQFP48",  type: "M252_D" },
    { name: "M252LE3AE", pkg: "LQFP48",  type: "M252_E" },
    { name: "M252LG6AE", pkg: "LQFP48",  type: "M252_G" },
    { name: "M252SC2AE", pkg: "LQFP64",  type: "M252_C" },
    { name: "M252SD2AE", pkg: "LQFP64",  type: "M252_D" },
    { name: "M252SE3AE", pkg: "LQFP64",  type: "M252_E" },
    { name: "M252SG6AE", pkg: "LQFP64",  type: "M252_G" },
    { name: "M252KE3AE", pkg: "LQFP128", type: "M252_E" },
    { name: "M252KG6AE", pkg: "LQFP128", type: "M252_G" }
];
```



```
#define CRC_SEED                        <!id:CRCSEEDHexText;
                                        type:hextext;
                                        label:CRC SEED Value;
                                        data:0x33;
                                        default:0xFFFFFFFF;
                                        helper:Input text in the hexadecimal format;
                                        dependencies:none;
                                        dependenciesOption:none;
                                        filterExp:type=="M252_C";!>
```

3.6 FieldFromFilter

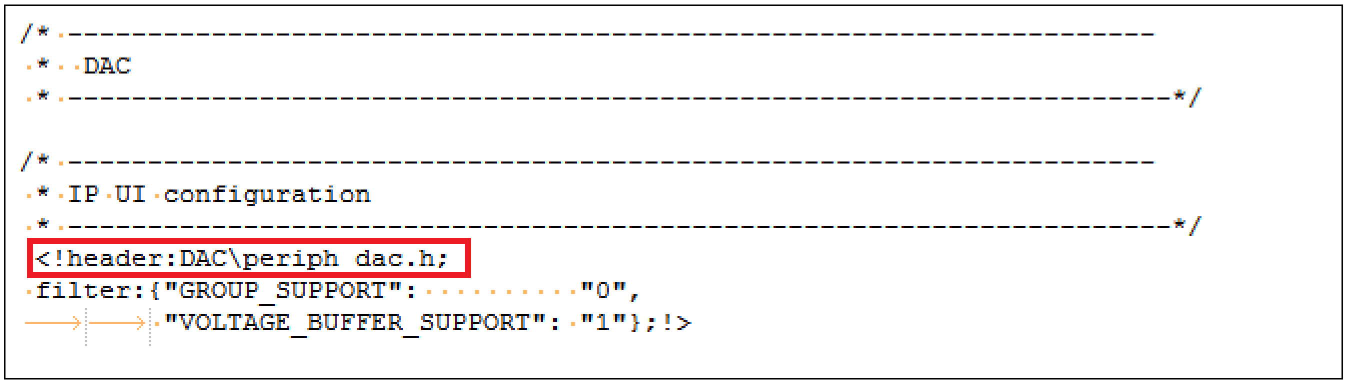
The content of default, enum and optionLabels field can be dynamically decided by filter conditions using JavaScript object, (i.e., FieldFromFilter) to specify, as follows. When a name:value pair in FieldFromFilter object meets the filter condition, the tool will take its value as the content of the corresponding field. Please note that first meets first takes. If all pairs fail to meet, the tool will take the value of NUC_ANY if existed. If any pair of the NUC_ORx object meets the condition, the tool will concatenate their value. If all pairs of the NUC_ANDx object meet the condition, the tool will concatenate their value. The x could be blank or number.



```
<!filter:{"CHANNEL_NUMBER_SUPPORT": "1"};!>

#define NUCODEGEN_ADC_COMPARE0_CHANNEL (<!id:ADCCompare0Channel;
                                type:select;
                                label:"Compare 0 channel.";
                                data:2;
                                default: {
                                    "NUC_OR":{"PA_dot1==1": [1], "PA_dot2==1": [2], "PA_dot5==1": [5]},
                                    "NUC_AND":{"PA_dot1==1": [1], "PA_dot2==1": [2]},
                                    "PA_dot4==1": [4],
                                    "PA_dot12==1": [12],
                                    "NUC_ANY": [0]
                                };
                                helper:"Select a channel from A/D converter channels";
                                sort:false;
                                enum: {
                                    "NUC_OR":{"PA_dot1==1": [1], "PA_dot2==1": [2], "PA_dot5==1": [5]},
                                    "NUC_AND":{"PA_dot1==1": [1], "PA_dot2==1": [2]},
                                    "PA_dot4==1": [4],
                                    "PA_dot12==1": [12],
                                    "NUC_ANY": [0]
                                };
                                optionLabels: {
                                    "NUC_OR":{"PA_dot1==1": [1], "PA_dot2==1": [2], "PA_dot5==1": [5]},
                                    "NUC_AND":{"PA_dot1==1": [1], "PA_dot2==1": [2]},
                                    "PA_dot4==1": [4],
                                    "PA_dot12==1": [12],
                                    "NUC_ANY": [0]
                                };
                                dependencies:none;
                                dependenciesOption:[1];
                                dependenciesDefault:false;
                                observable:none;
                                listener:none;
                                groupId:Group_ADC_Compare0Configuration;
```

3.7 Header

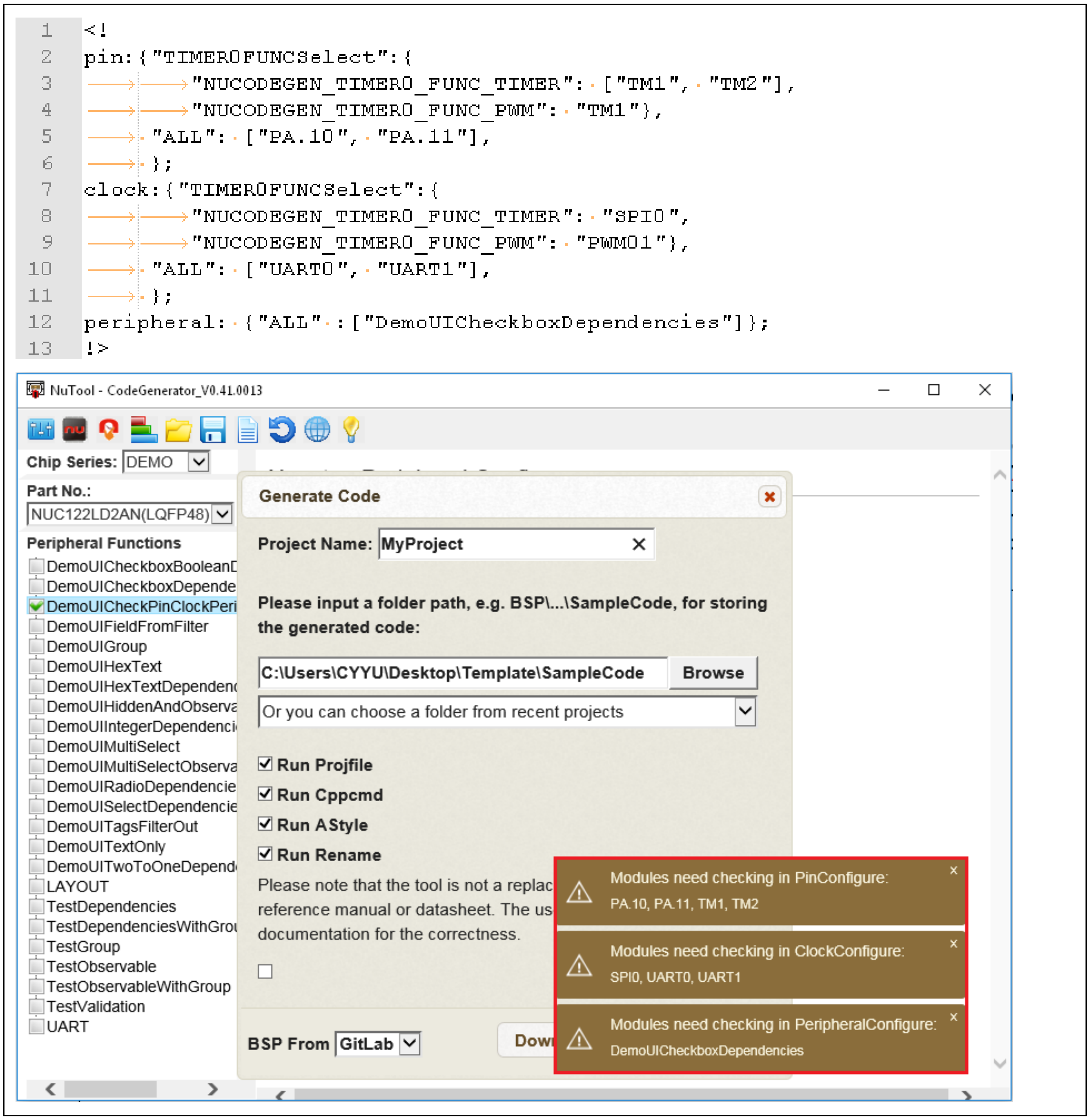
In general, the generated code will be placed in the periph_conf.h file. We can assign it to a different header by defining the header tag, as follows.



```
/* -------------------------------------------------------------------
 *  DAC
 * ---------------------------------------------------------------------*/

/* -------------------------------------------------------------------
 * IP UI configuration
 * ---------------------------------------------------------------------*/
<!header:DAC\periph_dac.h;
 filter:{"GROUP_SUPPORT":          "0",
		 "VOLTAGE_BUFFER_SUPPORT": "1"};!>
```

3.8 Pin/Clock/Peripheral

The general tag (referring to Q&A 4th question) also includes pin, clock and peripheral tags, as follows. **Pin** tag describes the relationship between PeripheralConfigure and PinConfigure. For example, when the data of TIMER0FUNCSelect is NUCODEGEN_TIMER0_FUNC_TIMER, PinConfigure should enable TM1 and TM2. **Clock** tag describes the relationship between PeripheralConfigure and ClockConfigure. **Peripheral** tag describes the relationship among .txt files in PeripheralConfigure. Regarding the module name of pin and clock, please switch to PinConfigure and ClockConfigure and follow their module names. In addition, the keyword of ALL means that if the .txt file is selected, which module of PinConfigure or ClockConfigure should be enabled.



```
<!
pin:{"TIMER0FUNCSelect":{
		"NUCODEGEN_TIMER0_FUNC_TIMER": ["TM1", "TM2"],
		"NUCODEGEN_TIMER0_FUNC_PWM": "TM1"},
	"ALL": ["PA.10", "PA.11"],
	};
clock:{"TIMER0FUNCSelect":{
		"NUCODEGEN_TIMER0_FUNC_TIMER": "SPI0",
		"NUCODEGEN_TIMER0_FUNC_PWM": "PWM01"},
	"ALL": ["UART0", "UART1"],
	};
peripheral: {"ALL" :["DemoUICheckboxDependencies"]};
!>
```

NuTool - CodeGenerator_V0.41.0013

Chip Series: DEMO

Part No.: NUC122LD2AN(LQFP48)

Peripheral Functions
- DemoUICheckboxBooleanD
- DemoUICheckboxDepende
- ✓ DemoUICheckPinClockPeri
- DemoUIFieldFromFilter
- DemoUIGroup
- DemoUIHexText
- DemoUIHexTextDependen
- DemoUIHiddenAndObserva
- DemoUIIntegerDependenci
- DemoUIMultiSelect
- DemoUIMultiSelectObserva
- DemoUIRadioDependencie
- DemoUISelectDependencie
- DemoUITagsFilterOut
- DemoUITextOnly
- DemoUITwoToOneDepend
- LAYOUT
- TestDependencies
- TestDependenciesWithGrou
- TestGroup
- TestObservable
- TestObservableWithGroup
- TestValidation
- UART

Generate Code

Project Name: MyProject

Please input a folder path, e.g. BSP\...\SampleCode, for storing the generated code:

C:\Users\CYYU\Desktop\Template\SampleCode   Browse

Or you can choose a folder from recent projects

☑ Run Projfile
☑ Run Cppcmd
☑ Run AStyle
☑ Run Rename

Please note that the tool is not a replac reference manual or datasheet. The us documentation for the correctness.

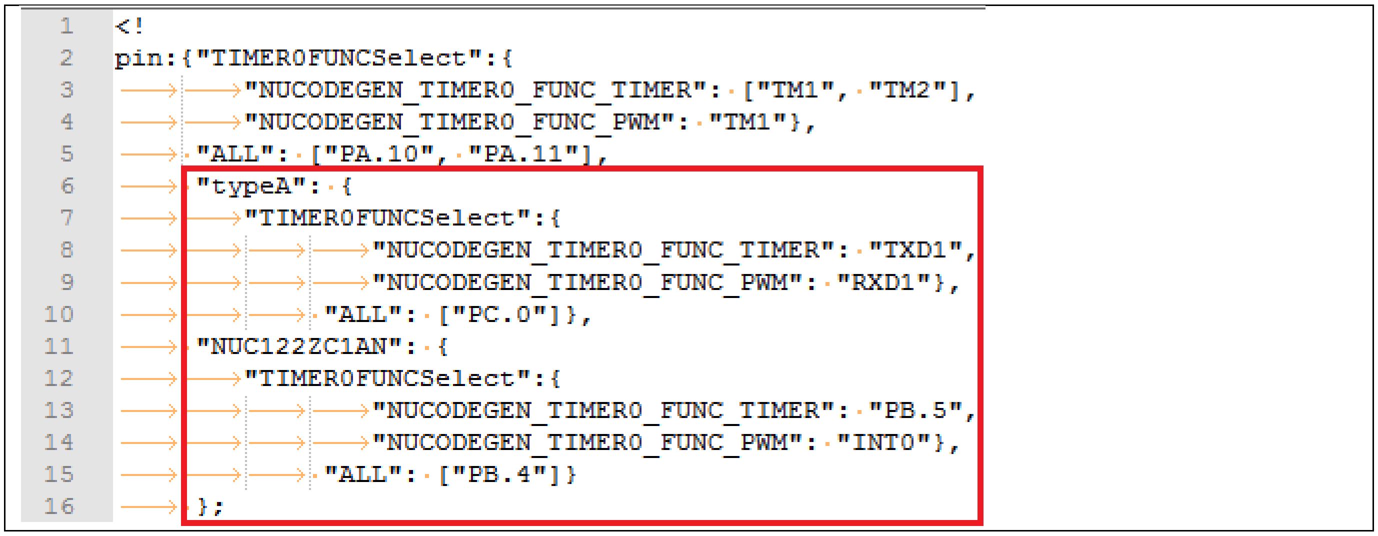
BSP From GitLab   Dowi

Modules need checking in PinConfigure:
PA.10, PA.11, TM1, TM2

Modules need checking in ClockConfigure:
SPI0, UART0, UART1

Modules need checking in PeripheralConfigure:
DemoUICheckboxDependencies

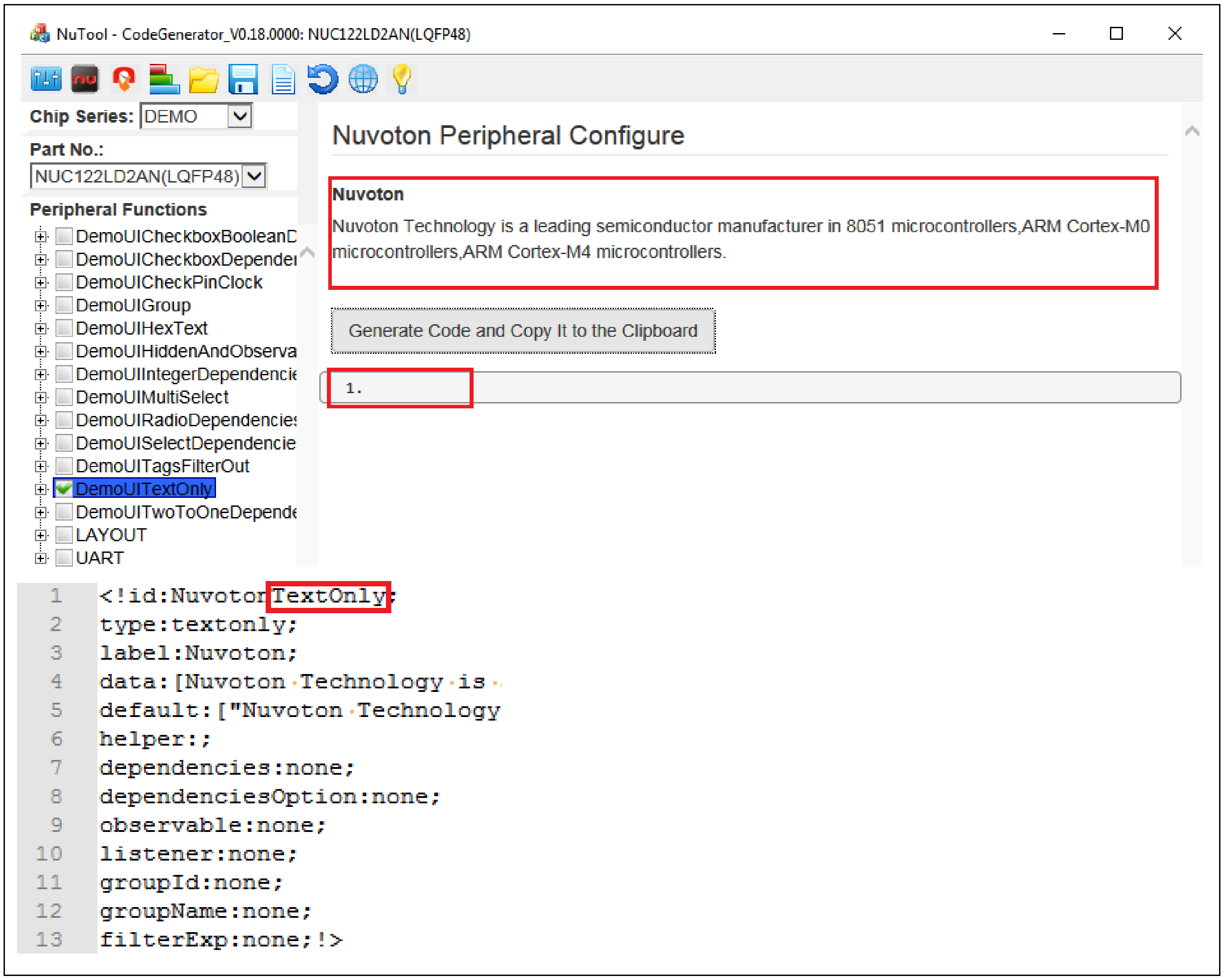
Sometimes the pin or clock tag might have a different description for a different chip type or chip name, as follows. When both of chip type and name matches the current chip, the description of chip name will take effect.



```
<!
pin:{"TIMER0FUNCSelect":{
        "NUCODEGEN_TIMER0_FUNC_TIMER": ["TM1", "TM2"],
        "NUCODEGEN_TIMER0_FUNC_PWM": "TM1"},
     "ALL": ["PA.10", "PA.11"],
     "typeA": {
        "TIMER0FUNCSelect":{
                "NUCODEGEN_TIMER0_FUNC_TIMER": "TXD1",
                "NUCODEGEN_TIMER0_FUNC_PWM": "RXD1"},
             "ALL": ["PC.0"]},
     "NUC122ZC1AN": {
        "TIMER0FUNCSelect":{
                "NUCODEGEN_TIMER0_FUNC_TIMER": "PB.5",
                "NUCODEGEN_TIMER0_FUNC_PWM": "INT0"},
             "ALL": ["PB.4"]}
     };
```

3.9 TextOnly

TextOnly tag only renders text on UI, but does not generate any code. Please note that the id of TextOnly tag must contain the keyword of TextOnly. Otherwise, the tool will replace the tag with its data in the generated code.



Nuvoton

Nuvoton Technology is a leading semiconductor manufacturer in 8051 microcontrollers,ARM Cortex-M0 microcontrollers,ARM Cortex-M4 microcontrollers.

```
1.
```

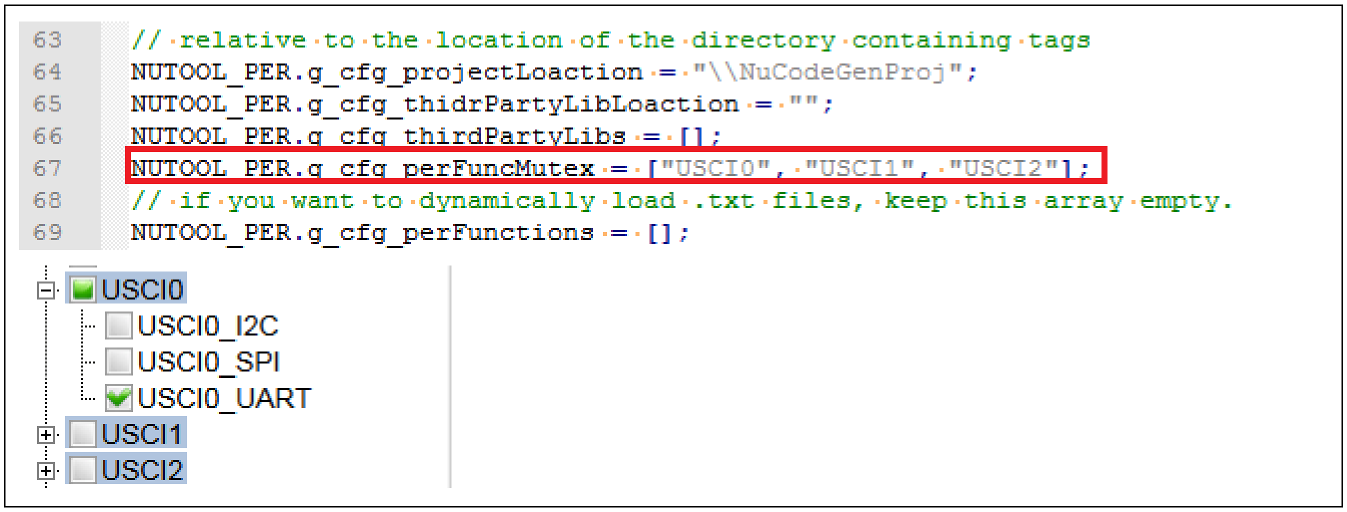
```
<!id:NuvotonTextOnly;
type:textonly;
label:Nuvoton;
data:[Nuvoton Technology is 
default:["Nuvoton Technology
helper:;
dependencies:none;
dependenciesOption:none;
observable:none;
listener:none;
groupId:none;
groupName:none;
filterExp:none;!>
```

4 Chip Content

In addition to .txt files, each chip has an javascript file to describe its content of PeripheralConfigure, such as Content\PeripheralConfigure\NUC_M251_Conttent.js.

4.1 Mutex

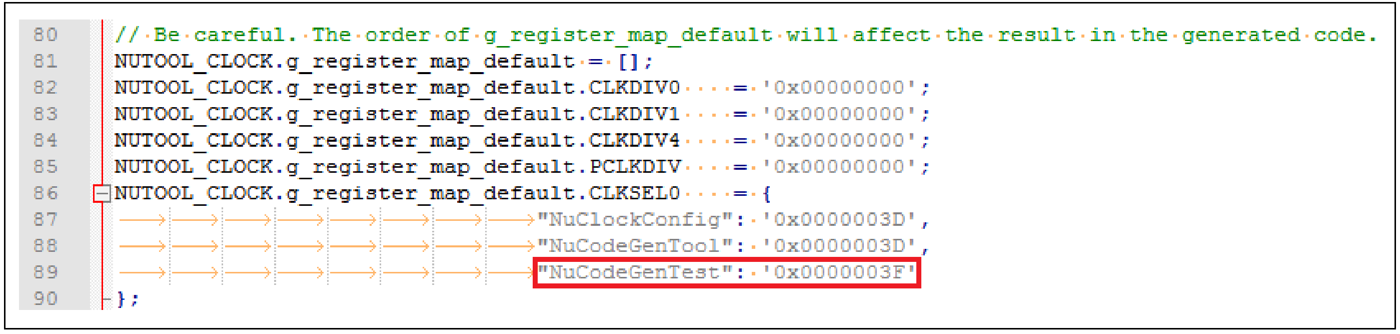
For some reason, one .txt file is split into several ones, and the tool only allows one of them to be selected at a time. To do that, we need to describe the first layer of Peripheral Functions tree in the perFuncMutex, which resides in NUC_M251_Content.js when the chip is M251. After that, these peripheral functions are grouped under a module highlighted with the steel blue color.



```
// relative to the location of the directory containing tags
NUTOOL_PER.g_cfg_projectLoaction = "\\NuCodeGenProj";
NUTOOL_PER.g_cfg_thidrPartyLibLoaction = "";
NUTOOL_PER.g_cfg_thirdPartyLibs = [];
NUTOOL_PER.g_cfg_perFuncMutex = ["USCI0", "USCI1", "USCI2"];
// if you want to dynamically load .txt files, keep this array empty.
NUTOOL_PER.g_cfg_perFunctions = [];
```

4.2 Default of clock registers

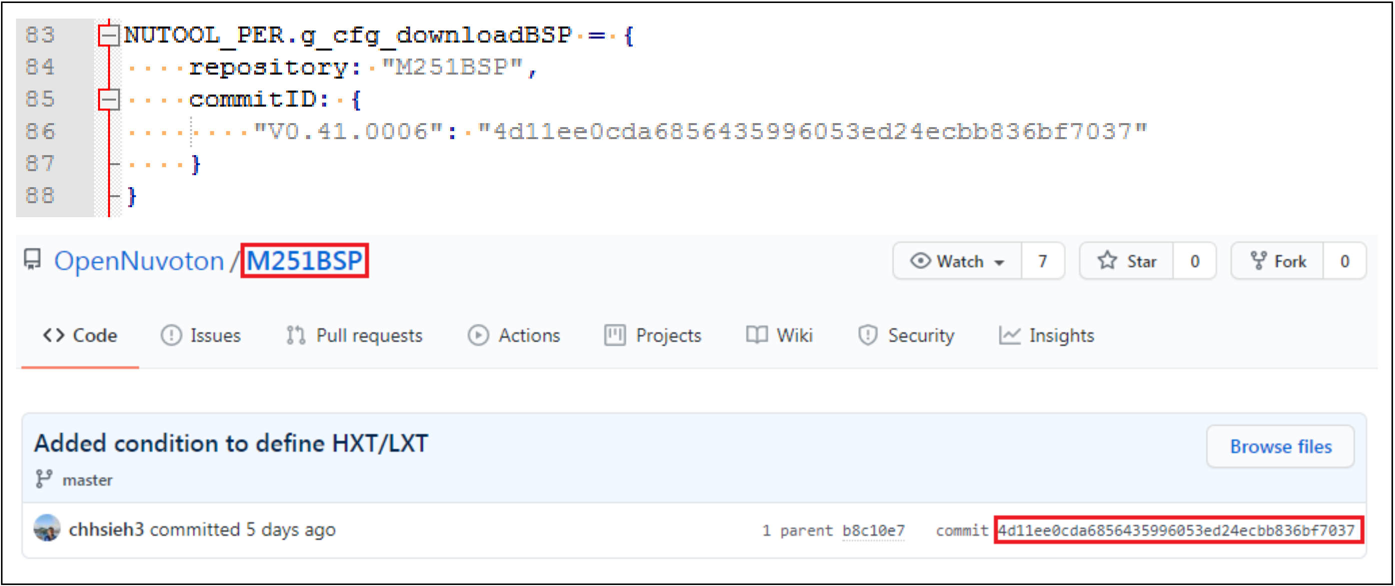
For auto-testing, the default of clock registers may need varying, as follows. To do that, input the varied default value into the NuCodeGenTest key of clock-regiseter default map, which resides in Content\ClockConfigure\NUC_M251_Content.js when the chip is M251. The default value of clock registers in the NuClockConfig and NuCodeGenTool keys should follow that in the TRM, so BSP team members leave them for the tool developer.



```
// Be careful. The order of g_register_map_default will affect the result in the generated code.
NUTOOL_CLOCK.g_register_map_default = [];
NUTOOL_CLOCK.g_register_map_default.CLKDIV0    = '0x00000000';
NUTOOL_CLOCK.g_register_map_default.CLKDIV1    = '0x00000000';
NUTOOL_CLOCK.g_register_map_default.CLKDIV4    = '0x00000000';
NUTOOL_CLOCK.g_register_map_default.PCLKDIV    = '0x00000000';
NUTOOL_CLOCK.g_register_map_default.CLKSEL0    = {
								"NuClockConfig": '0x0000003D',
								"NuCodeGenTool": '0x0000003D',
								"NuCodeGenTest": '0x0000003F'
};
```

4.3 DownloadBSP

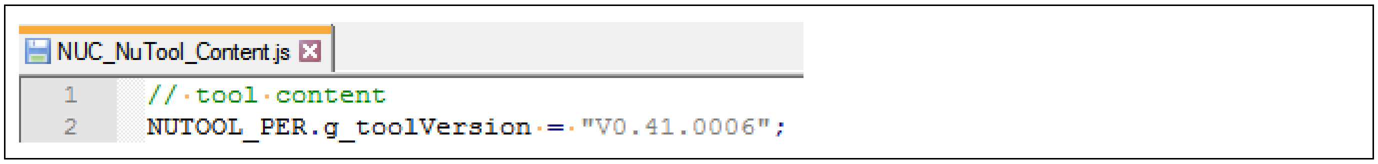
NuCodeGenTool can directly download BSP from GitHub and GitLab. To do that, input the relationship table between the tool version and git commit SHA key in the downloadBSP, which resides in NUC_M251_Content.js when the chip is M251. As the tool version increase, the relationship table returns the largest number less than or equal to the current tool version to download the corresponding BSP. Please note that BSP teams should test NuCodeGenTool using the BSP specified in the downloadBSP for the consistency.



```
NUTOOL_PER.g_cfg_downloadBSP = {
    repository: "M251BSP",
    commitID: {
        "V0.41.0006": "4d11ee0cda6856435996053ed24ecbb836bf7037"
    }
}
```

Added condition to define HXT/LXT
master
chhsieh3 committed 5 days ago
1 parent b8c10e7 commit 4d11ee0cda6856435996053ed24ecbb836bf7037

To change the tool version, modify toolVersion, which resides in NUC_M251_Content.js when the chip is M251. Please increment the third code of tool number when changing the tool version.



```
// tool content
NUTOOL_PER.g_toolVersion = "V0.41.0006";
```

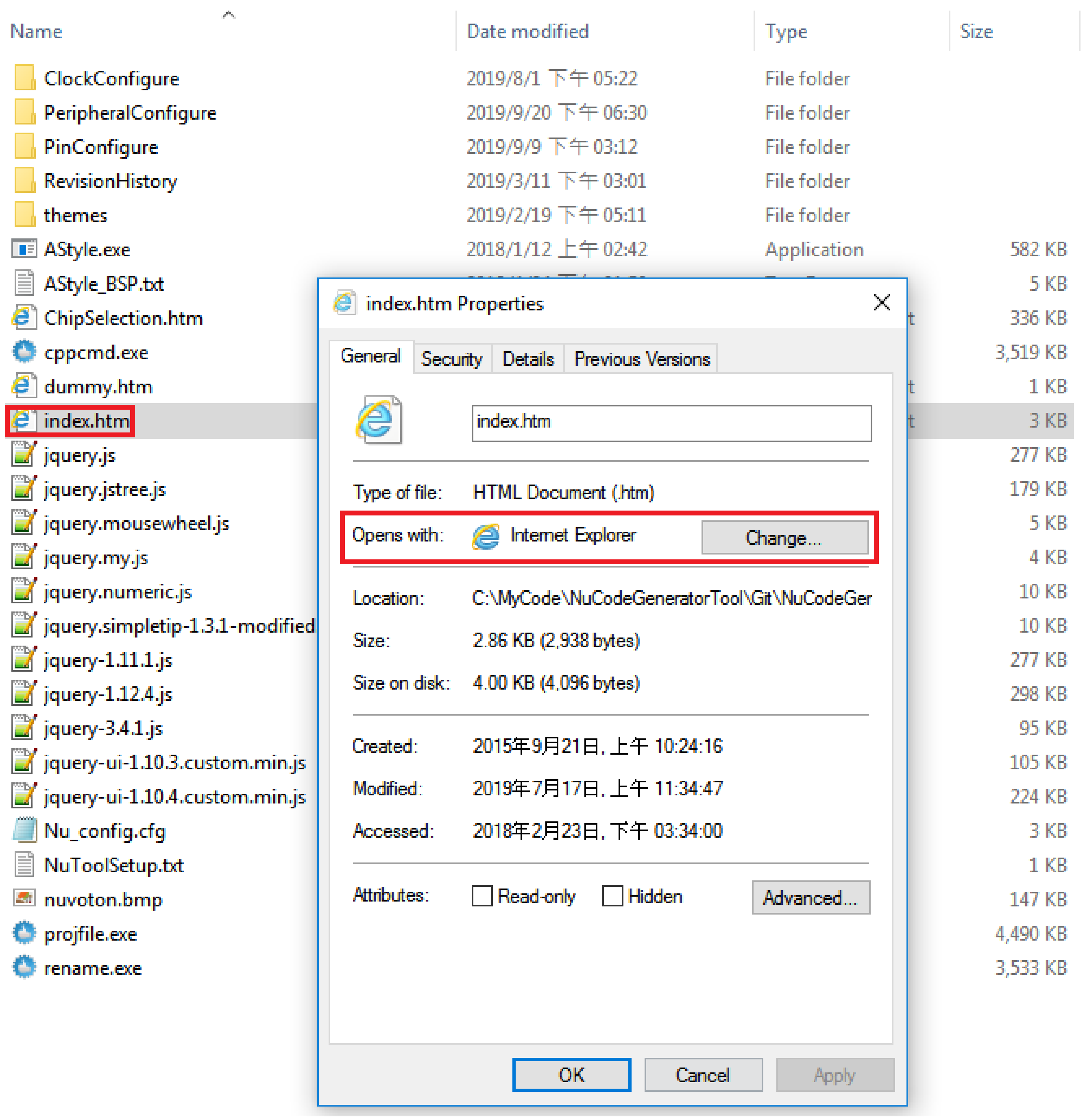
5 Functional Tests

We have to run functional tests to ensure the correctness of .txt files and generated code.

5.1 Prerequisite

Before running functional tests, several IE settings are required:

a. Set IE as the default program to open .html files.



Name	Date modified	Type	Size
ClockConfigure	2019/8/1 下午 05:22	File folder	
PeripheralConfigure	2019/9/20 下午 06:30	File folder	
PinConfigure	2019/9/9 下午 03:12	File folder	
RevisionHistory	2019/3/11 下午 03:01	File folder	
themes	2019/2/19 下午 05:11	File folder	
AStyle.exe	2018/1/12 上午 02:42	Application	582 KB
AStyle_BSP.txt			5 KB
ChipSelection.htm			336 KB
cppcmd.exe			3,519 KB
dummy.htm			1 KB
index.htm			3 KB
jquery.js			277 KB
jquery.jstree.js			179 KB
jquery.mousewheel.js			5 KB
jquery.my.js			4 KB
jquery.numeric.js			10 KB
jquery.simpletip-1.3.1-modified			10 KB
jquery-1.11.1.js			277 KB
jquery-1.12.4.js			298 KB
jquery-3.4.1.js			95 KB
jquery-ui-1.10.3.custom.min.js			105 KB
jquery-ui-1.10.4.custom.min.js			224 KB
Nu_config.cfg			3 KB
NuToolSetup.txt			1 KB
nuvoton.bmp			147 KB
projfile.exe			4,490 KB
rename.exe			3,533 KB

index.htm Properties

index.htm

Type of file: HTML Document (.htm)

Opens with: Internet Explorer

Location: C:\MyCode\NuCodeGeneratorTool\Git\NuCodeGer

Size: 2.86 KB (2,938 bytes)

Size on disk: 4.00 KB (4,096 bytes)

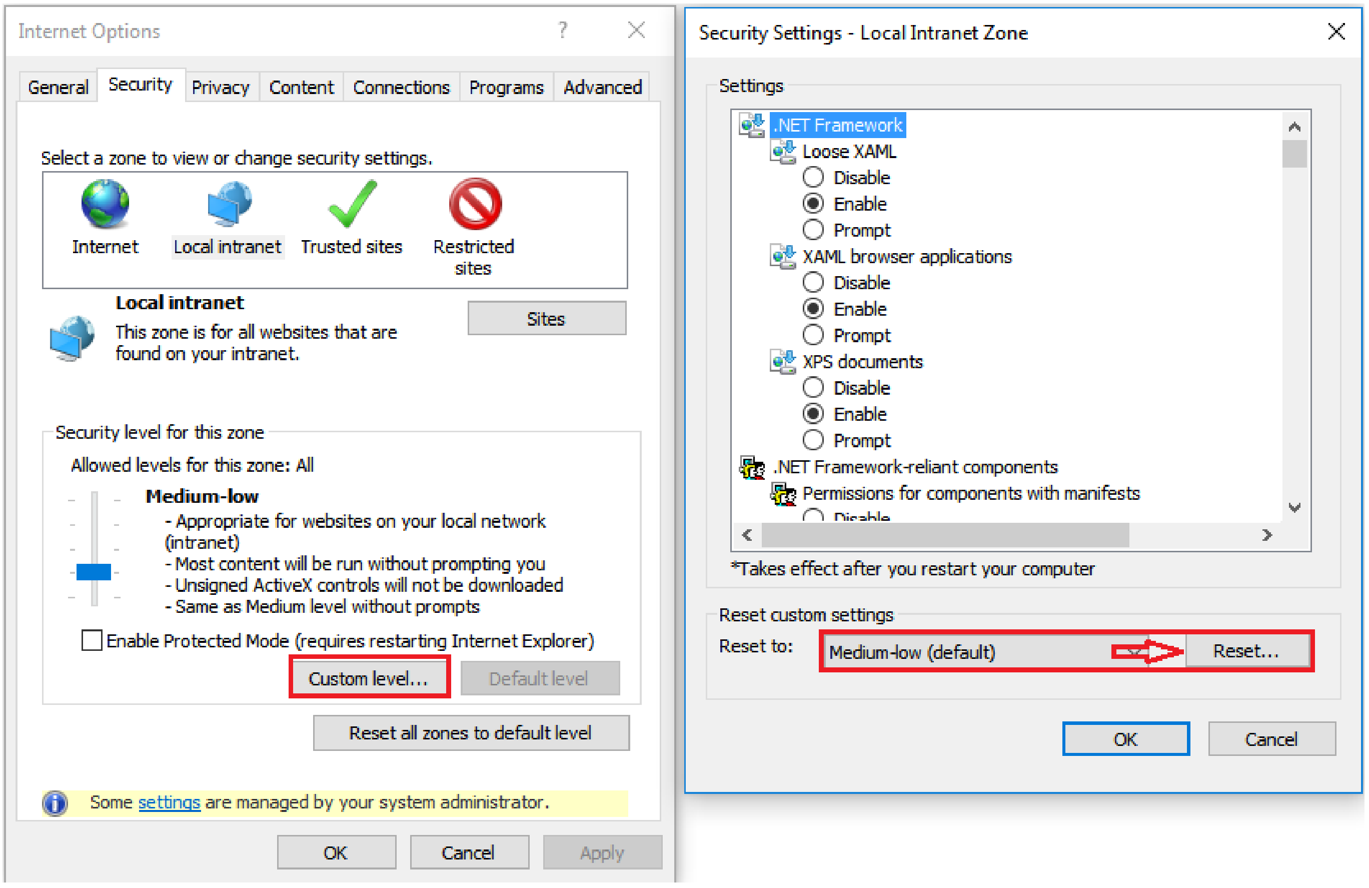
Created: 2015年9月21日, 上午 10:24:16

Modified: 2019年7月17日, 上午 11:34:47

Accessed: 2018年2月23日, 下午 03:34:00

Attributes: ☐ Read-only ☐ Hidden

b. Reset IE. Open **Internet Options**, as follows. Click on **Custom level…**, ant then click on **Medium-low (default)**. Finally click on **Reset….**


Internet Options

Select a zone to view or change security settings.

Local intranet
This zone is for all websites that are found on your intranet.

Security level for this zone
Allowed levels for this zone: All
Medium-low
- Appropriate for websites on your local network (intranet)
- Most content will be run without prompting you
- Unsigned ActiveX controls will not be downloaded
- Same as Medium level without prompts

[ ] Enable Protected Mode (requires restarting Internet Explorer)

Some settings are managed by your system administrator.

Security Settings - Local Intranet Zone

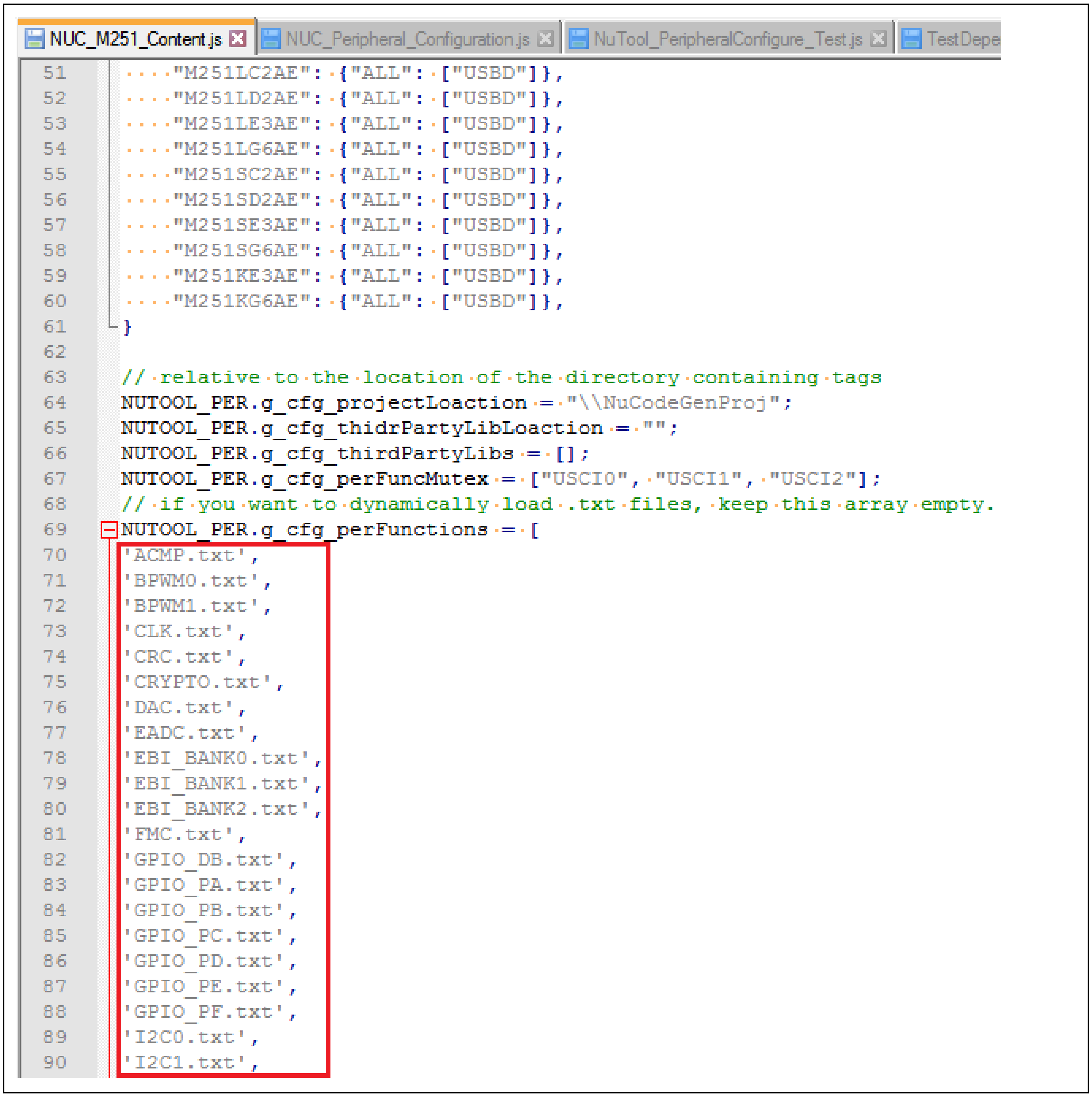
Settings

```
.NET Framework
    Loose XAML
        ○ Disable
        ● Enable
        ○ Prompt
    XAML browser applications
        ○ Disable
        ● Enable
        ○ Prompt
    XPS documents
        ○ Disable
        ● Enable
        ○ Prompt
.NET Framework-reliant components
    Permissions for components with manifests
        ○ Disable
```

*Takes effect after you restart your computer

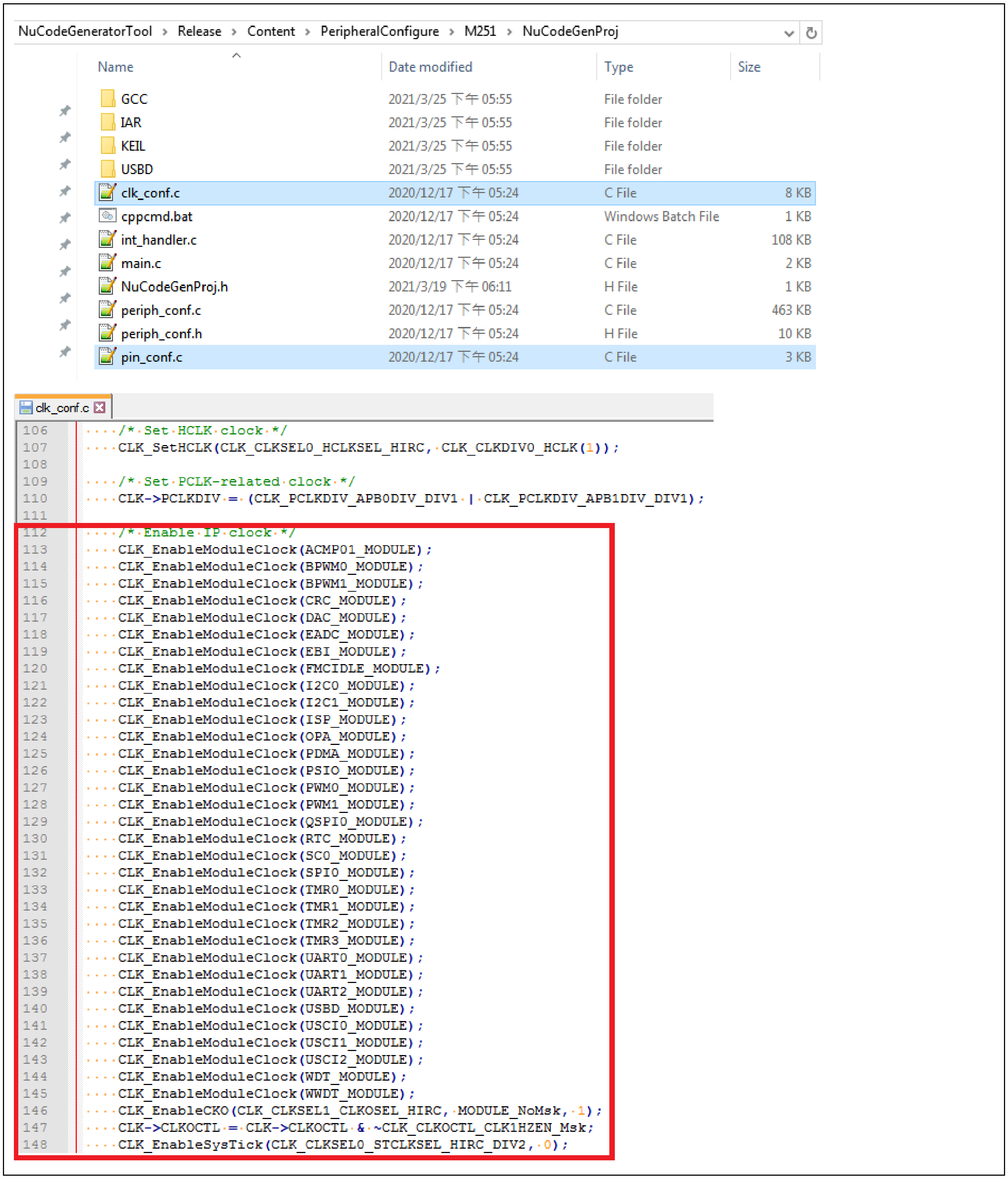
Reset custom settings
Reset to: Medium-low (default)

c. Input .txt file names in the perFunctions, which resides in Content\PeripheralConfigure\NUC_M251_Content.js when the chip is M251.


NUC_M251_Content.js | NUC_Peripheral_Configuration.js | NuTool_PeripheralConfigure_Test.js | TestDepe

```
    "M251LC2AE": {"ALL": ["USBD"]},
    "M251LD2AE": {"ALL": ["USBD"]},
    "M251LE3AE": {"ALL": ["USBD"]},
    "M251LG6AE": {"ALL": ["USBD"]},
    "M251SC2AE": {"ALL": ["USBD"]},
    "M251SD2AE": {"ALL": ["USBD"]},
    "M251SE3AE": {"ALL": ["USBD"]},
    "M251SG6AE": {"ALL": ["USBD"]},
    "M251KE3AE": {"ALL": ["USBD"]},
    "M251KG6AE": {"ALL": ["USBD"]},
}

// relative to the location of the directory containing tags
NUTOOL_PER.g_cfg_projectLoaction = "\\NuCodeGenProj";
NUTOOL_PER.g_cfg_thidrPartyLibLoaction = "";
NUTOOL_PER.g_cfg_thirdPartyLibs = [];
NUTOOL_PER.g_cfg_perFuncMutex = ["USCI0", "USCI1", "USCI2"];
// if you want to dynamically load .txt files, keep this array empty.
NUTOOL_PER.g_cfg_perFunctions = [
'ACMP.txt',
'BPWM0.txt',
'BPWM1.txt',
'CLK.txt',
'CRC.txt',
'CRYPTO.txt',
'DAC.txt',
'EADC.txt',
'EBI_BANK0.txt',
'EBI_BANK1.txt',
'EBI_BANK2.txt',
'FMC.txt',
'GPIO_DB.txt',
'GPIO_PA.txt',
'GPIO_PB.txt',
'GPIO_PC.txt',
'GPIO_PD.txt',
'GPIO_PE.txt',
'GPIO_PF.txt',
'I2C0.txt',
'I2C1.txt',
```

d. Prepare clk_conf.c and pin_conf.c in the Content\PeripheralConfigure\M251\NuCodeGenProj when the chip is M251. Don't forget to enable all the clock sources in the clk_conf.c.



NuCodeGeneratorTool › Release › Content › PeripheralConfigure › M251 › NuCodeGenProj

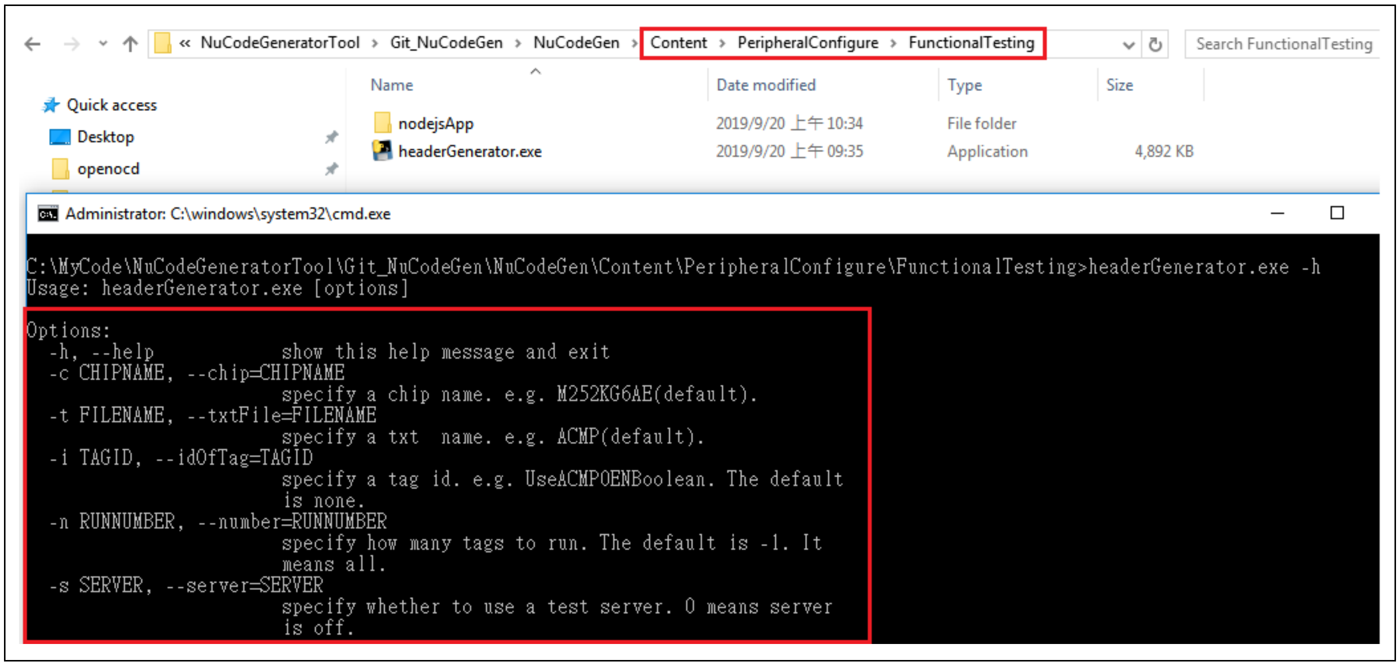
Name	Date modified	Type	Size
GCC	2021/3/25 下午 05:55	File folder	
IAR	2021/3/25 下午 05:55	File folder	
KEIL	2021/3/25 下午 05:55	File folder	
USBD	2021/3/25 下午 05:55	File folder	
clk_conf.c	2020/12/17 下午 05:24	C File	8 KB
cppcmd.bat	2020/12/17 下午 05:24	Windows Batch File	1 KB
int_handler.c	2020/12/17 下午 05:24	C File	108 KB
main.c	2020/12/17 下午 05:24	C File	2 KB
NuCodeGenProj.h	2021/3/19 下午 06:11	H File	1 KB
periph_conf.c	2020/12/17 下午 05:24	C File	463 KB
periph_conf.h	2020/12/17 下午 05:24	H File	10 KB
pin_conf.c	2020/12/17 下午 05:24	C File	3 KB

clk_conf.c

```c
    /* Set HCLK clock */
    CLK_SetHCLK(CLK_CLKSEL0_HCLKSEL_HIRC, CLK_CLKDIV0_HCLK(1));

    /* Set PCLK-related clock */
    CLK->PCLKDIV = (CLK_PCLKDIV_APB0DIV_DIV1 | CLK_PCLKDIV_APB1DIV_DIV1);

    /* Enable IP clock */
    CLK_EnableModuleClock(ACMP01_MODULE);
    CLK_EnableModuleClock(BPWM0_MODULE);
    CLK_EnableModuleClock(BPWM1_MODULE);
    CLK_EnableModuleClock(CRC_MODULE);
    CLK_EnableModuleClock(DAC_MODULE);
    CLK_EnableModuleClock(EADC_MODULE);
    CLK_EnableModuleClock(EBI_MODULE);
    CLK_EnableModuleClock(FMCIDLE_MODULE);
    CLK_EnableModuleClock(I2C0_MODULE);
    CLK_EnableModuleClock(I2C1_MODULE);
    CLK_EnableModuleClock(ISP_MODULE);
    CLK_EnableModuleClock(OPA_MODULE);
    CLK_EnableModuleClock(PDMA_MODULE);
    CLK_EnableModuleClock(PSIO_MODULE);
    CLK_EnableModuleClock(PWM0_MODULE);
    CLK_EnableModuleClock(PWM1_MODULE);
    CLK_EnableModuleClock(QSPI0_MODULE);
    CLK_EnableModuleClock(RTC_MODULE);
    CLK_EnableModuleClock(SC0_MODULE);
    CLK_EnableModuleClock(SPI0_MODULE);
    CLK_EnableModuleClock(TMR0_MODULE);
    CLK_EnableModuleClock(TMR1_MODULE);
    CLK_EnableModuleClock(TMR2_MODULE);
    CLK_EnableModuleClock(TMR3_MODULE);
    CLK_EnableModuleClock(UART0_MODULE);
    CLK_EnableModuleClock(UART1_MODULE);
    CLK_EnableModuleClock(UART2_MODULE);
    CLK_EnableModuleClock(USBD_MODULE);
    CLK_EnableModuleClock(USCI0_MODULE);
    CLK_EnableModuleClock(USCI1_MODULE);
    CLK_EnableModuleClock(USCI2_MODULE);
    CLK_EnableModuleClock(WDT_MODULE);
    CLK_EnableModuleClock(WWDT_MODULE);
    CLK_EnableCKO(CLK_CLKSEL1_CLKOSEL_HIRC, MODULE_NoMsk, 1);
    CLK->CLKOCTL = CLK->CLKOCTL & ~CLK_CLKOCTL_CLK1HZEN_Msk;
    CLK_EnableSysTick(CLK_CLKSEL0_STCLKSEL_HIRC_DIV2, 0);
```

5.2 HeaderGenerator

Browse Content\PeripheralConfigure\FunctionalTesting folder and launch Windows command line tool. Input headerGenerator.exe –h and see its options, as follows.



```
C:\MyCode\NuCodeGeneratorTool\Git_NuCodeGen\NuCodeGen\Content\PeripheralConfigure\FunctionalTesting>headerGenerator.exe -h
Usage: headerGenerator.exe [options]

Options:
  -h, --help            show this help message and exit
  -c CHIPNAME, --chip=CHIPNAME
                        specify a chip name. e.g. M252KG6AE(default).
  -t FILENAME, --txtFile=FILENAME
                        specify a txt  name. e.g. ACMP(default).
  -i TAGID, --idOfTag=TAGID
                        specify a tag id. e.g. UseACMP0ENBoolean. The default
                        is none.
  -n RUNNUMBER, --number=RUNNUMBER
                        specify how many tags to run. The default is -1. It
                        means all.
  -s SERVER, --server=SERVER
                        specify whether to use a test server. 0 means server
                        is off.
```

For instances,

To test M251SD2AE, input headerGenerator.exe –c M251SD2AE.

To test CRC.txt, input headerGenerator.exe –t crc (case insensitive).

To test RCModeRadio tag in CRC.txt, input headerGenerator –t crc –i CRCModeRadio.

To test the topmost five tags in CRC.txt, input headerGenerator –t crc –n 5.

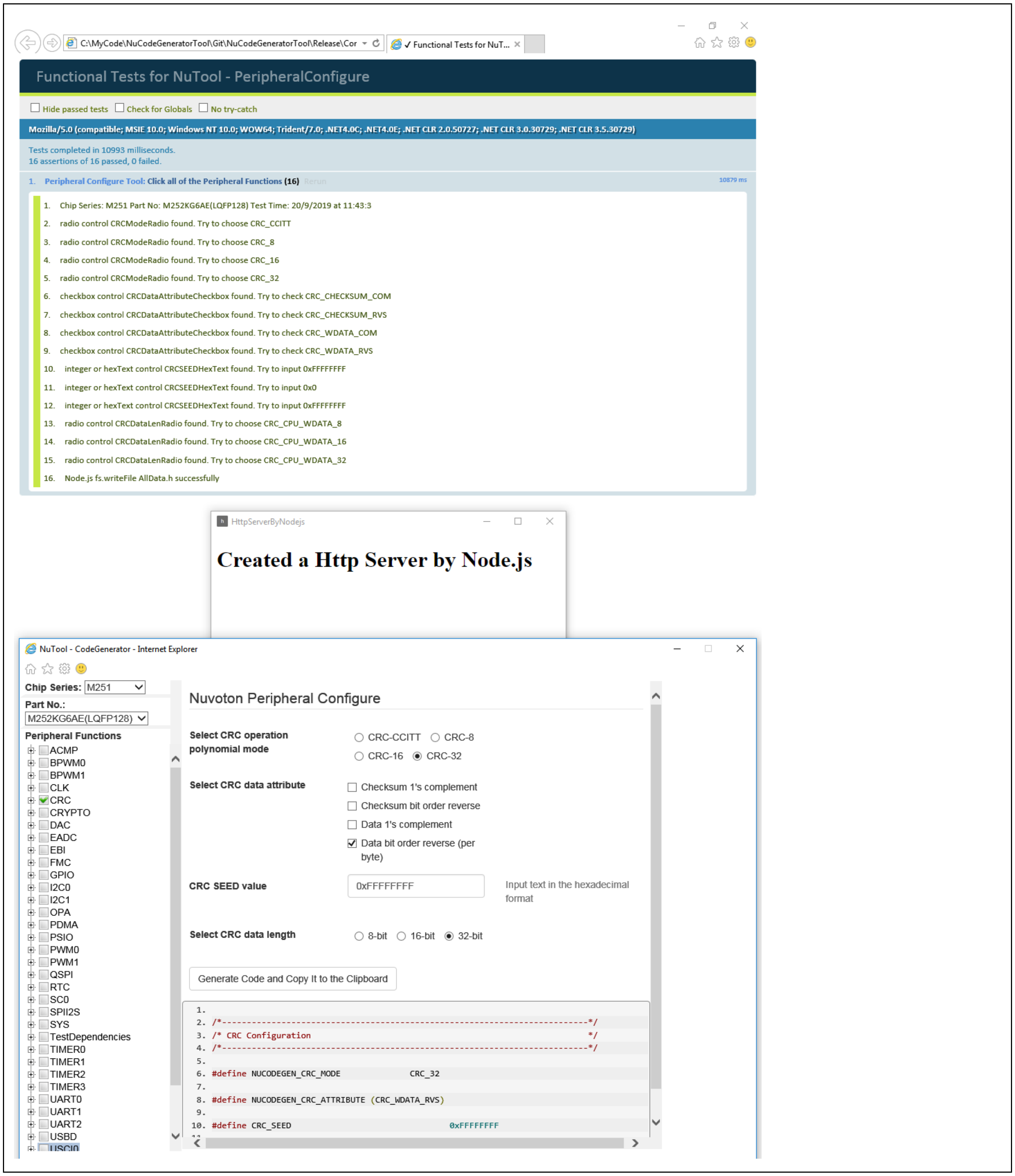
To test tree tags starting from CRCModeRadio, input headerGenerator –t crc –i CRCModeRadio –n 3.

If a .txt file has too many tags (over one hundred), we need to use advanced server mode to run the test. To do that, input headerGenerator –t EADC –s 1.

Use advanced server mode and start from UseRadio, input headerGenerator –t EADC -i UseRadio –s 1.

Use advanced server mode and run ACMP.txt and CRC.txt, input headerGenerator.exe –t acmp+crc.

While the headerGenerator is running, three windows show up and we should not touch them. After the test is over, they will be closed. If some problems occur, there will be a failure report by inputting the command with –s d0 or –s d1. Please keep the report and start debugging (call out MS60 CYYu or MS60 CCLi).


C:\MyCode\NuCodeGeneratorTool\Git\NuCodeGeneratorTool\Release\Cor
✓ Functional Tests for NuT...

Functional Tests for NuTool - PeripheralConfigure

☐ Hide passed tests ☐ Check for Globals ☐ No try-catch

Mozilla/5.0 (compatible; MSIE 10.0; Windows NT 10.0; WOW64; Trident/7.0; .NET4.0C; .NET4.0E; .NET CLR 2.0.50727; .NET CLR 3.0.30729; .NET CLR 3.5.30729)

Tests completed in 10993 milliseconds.
16 assertions of 16 passed, 0 failed.

1. Peripheral Configure Tool: Click all of the Peripheral Functions (16) Rerun 10879 ms

1. Chip Series: M251 Part No: M252KG6AE(LQFP128) Test Time: 20/9/2019 at 11:43:3
2. radio control CRCModeRadio found. Try to choose CRC_CCITT
3. radio control CRCModeRadio found. Try to choose CRC_8
4. radio control CRCModeRadio found. Try to choose CRC_16
5. radio control CRCModeRadio found. Try to choose CRC_32
6. checkbox control CRCDataAttributeCheckbox found. Try to check CRC_CHECKSUM_COM
7. checkbox control CRCDataAttributeCheckbox found. Try to check CRC_CHECKSUM_RVS
8. checkbox control CRCDataAttributeCheckbox found. Try to check CRC_WDATA_COM
9. checkbox control CRCDataAttributeCheckbox found. Try to check CRC_WDATA_RVS
10. integer or hexText control CRCSEEDHexText found. Try to input 0xFFFFFFFF
11. integer or hexText control CRCSEEDHexText found. Try to input 0x0
12. integer or hexText control CRCSEEDHexText found. Try to input 0xFFFFFFFF
13. radio control CRCDataLenRadio found. Try to choose CRC_CPU_WDATA_8
14. radio control CRCDataLenRadio found. Try to choose CRC_CPU_WDATA_16
15. radio control CRCDataLenRadio found. Try to choose CRC_CPU_WDATA_32
16. Node.js fs.writeFile AllData.h successfully

HttpServerByNodejs

Created a Http Server by Node.js

NuTool - CodeGenerator - Internet Explorer

Chip Series: M251
Part No.: M252KG6AE(LQFP128)

Peripheral Functions

Nuvoton Peripheral Configure

Select CRC operation polynomial mode: ○ CRC-CCITT ○ CRC-8 ○ CRC-16 [✓ CRC-32]

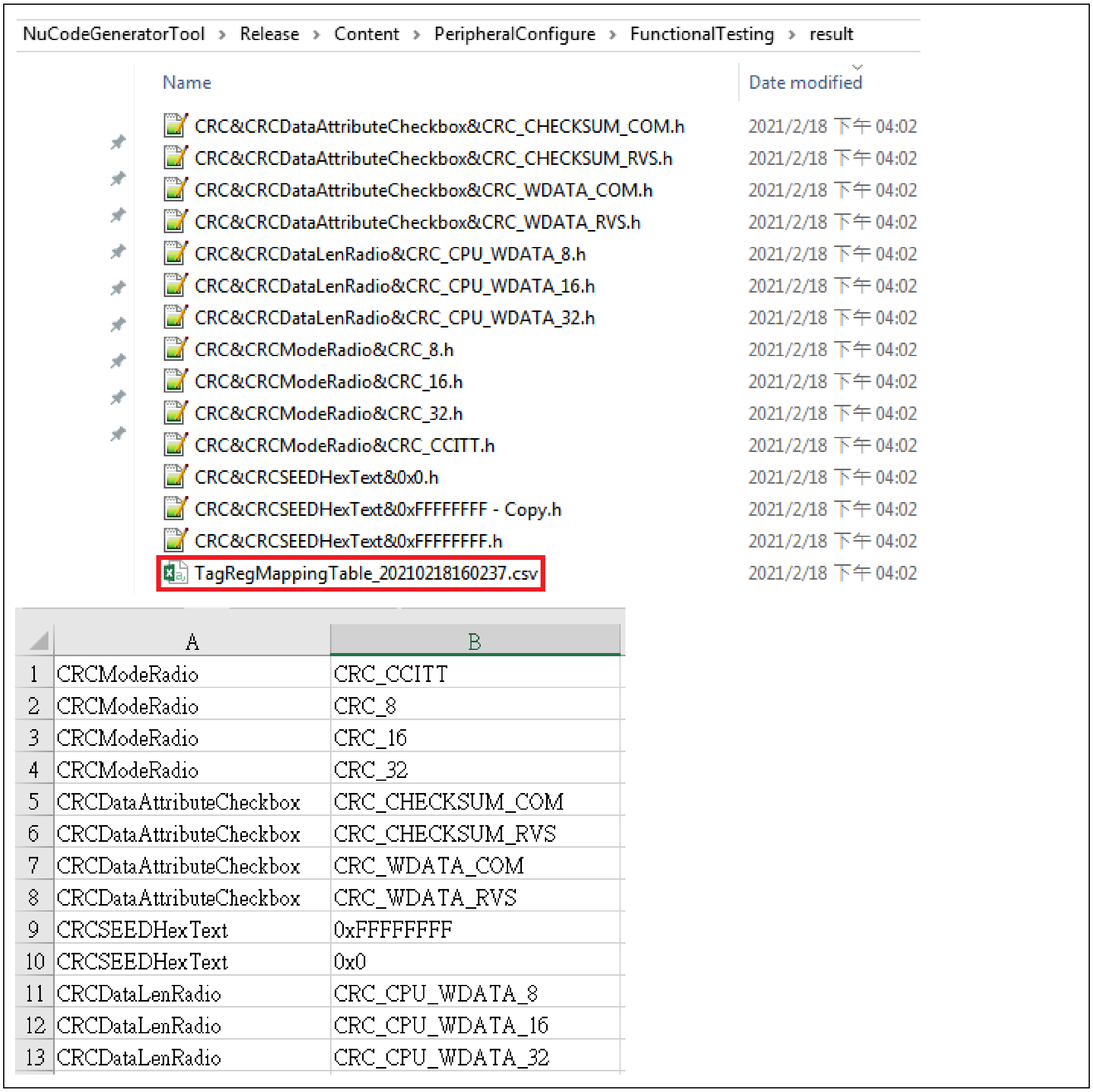
Select CRC data attribute:
☐ Checksum 1's complement
☐ Checksum bit order reverse
☐ Data 1's complement
☑ Data bit order reverse (per byte)

CRC SEED value: 0xFFFFFFFF — Input text in the hexadecimal format

Select CRC data length: ○ 8-bit ○ 16-bit [✓ 32-bit]

```

/*----------------------------------------------------------------------------*/
/* CRC Configuration                                                          */
/*----------------------------------------------------------------------------*/

#define NUCODEGEN_CRC_MODE                CRC_32

#define NUCODEGEN_CRC_ATTRIBUTE (CRC_WDATA_RVS)

#define CRC_SEED                                    0xFFFFFFFF
```

The headerGenerator will traverse all the UI options in the .txt file and generate each option with a header file (if dependencies exist, headerGenerator will make dependencies evaluate to be true). The generated header files are placed in Content\PeripheralConfigure\FunctionalTesting\result folder. The name of header file has a meaningful format, i.e., (txt name)&(tag id name)&(option name).h. In addition, the .csv file of TagRegMappingTable stores all the tag ids and corresponding options. BSP team members can use the .csv file to create the TagReg Mapping Table for auto testing.

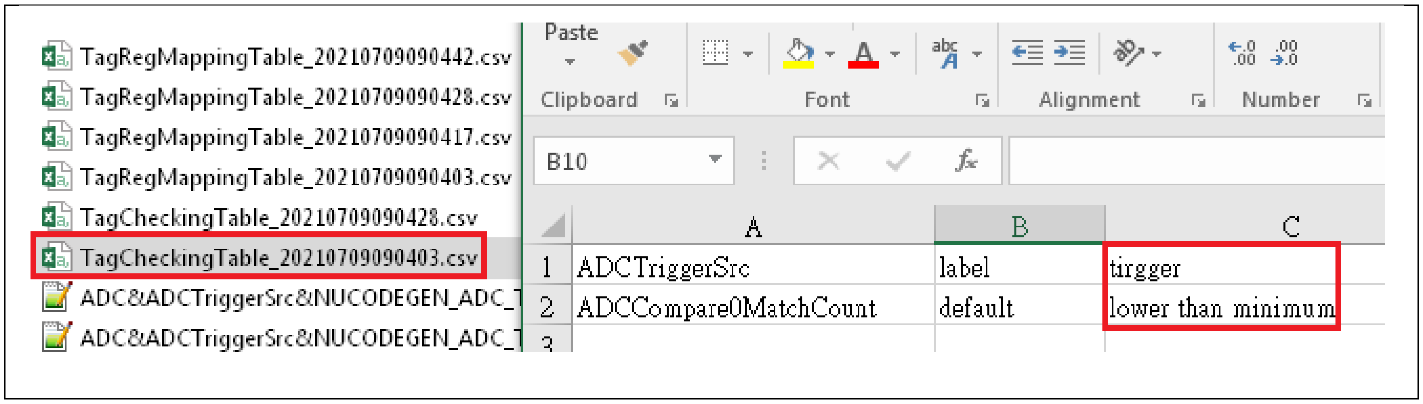

NuCodeGeneratorTool › Release › Content › PeripheralConfigure › FunctionalTesting › result

Name	Date modified
CRC&CRCDataAttributeCheckbox&CRC_CHECKSUM_COM.h	2021/2/18 下午 04:02
CRC&CRCDataAttributeCheckbox&CRC_CHECKSUM_RVS.h	2021/2/18 下午 04:02
CRC&CRCDataAttributeCheckbox&CRC_WDATA_COM.h	2021/2/18 下午 04:02
CRC&CRCDataAttributeCheckbox&CRC_WDATA_RVS.h	2021/2/18 下午 04:02
CRC&CRCDataLenRadio&CRC_CPU_WDATA_8.h	2021/2/18 下午 04:02
CRC&CRCDataLenRadio&CRC_CPU_WDATA_16.h	2021/2/18 下午 04:02
CRC&CRCDataLenRadio&CRC_CPU_WDATA_32.h	2021/2/18 下午 04:02
CRC&CRCModeRadio&CRC_8.h	2021/2/18 下午 04:02
CRC&CRCModeRadio&CRC_16.h	2021/2/18 下午 04:02
CRC&CRCModeRadio&CRC_32.h	2021/2/18 下午 04:02
CRC&CRCModeRadio&CRC_CCITT.h	2021/2/18 下午 04:02
CRC&CRCSEEDHexText&0x0.h	2021/2/18 下午 04:02
CRC&CRCSEEDHexText&0xFFFFFFFF - Copy.h	2021/2/18 下午 04:02
CRC&CRCSEEDHexText&0xFFFFFFFF.h	2021/2/18 下午 04:02
TagRegMappingTable_20210218160237.csv	2021/2/18 下午 04:02

	A	B
1	CRCModeRadio	CRC_CCITT
2	CRCModeRadio	CRC_8
3	CRCModeRadio	CRC_16
4	CRCModeRadio	CRC_32
5	CRCDataAttributeCheckbox	CRC_CHECKSUM_COM
6	CRCDataAttributeCheckbox	CRC_CHECKSUM_RVS
7	CRCDataAttributeCheckbox	CRC_WDATA_COM
8	CRCDataAttributeCheckbox	CRC_WDATA_RVS
9	CRCSEEDHexText	0xFFFFFFFF
10	CRCSEEDHexText	0x0
11	CRCDataLenRadio	CRC_CPU_WDATA_8
12	CRCDataLenRadio	CRC_CPU_WDATA_16
13	CRCDataLenRadio	CRC_CPU_WDATA_32

When some issues are found in .txt files, the headerGenerator will record them in TagCheckingTable file. BSP team members should fix them before making a commit. The checking items include:

1. Checking typo.
2. Checking whether the data value be the same with the default value.
3. Checking the correctness of the default value.



TagRegMappingTable_20210709090442.csv
TagRegMappingTable_20210709090428.csv
TagRegMappingTable_20210709090417.csv
TagRegMappingTable_20210709090403.csv
TagCheckingTable_20210709090428.csv
TagCheckingTable_20210709090403.csv
ADC&ADCTriggerSrc&NUCODEGEN_ADC_T
ADC&ADCTriggerSrc&NUCODEGEN_ADC_T

B10

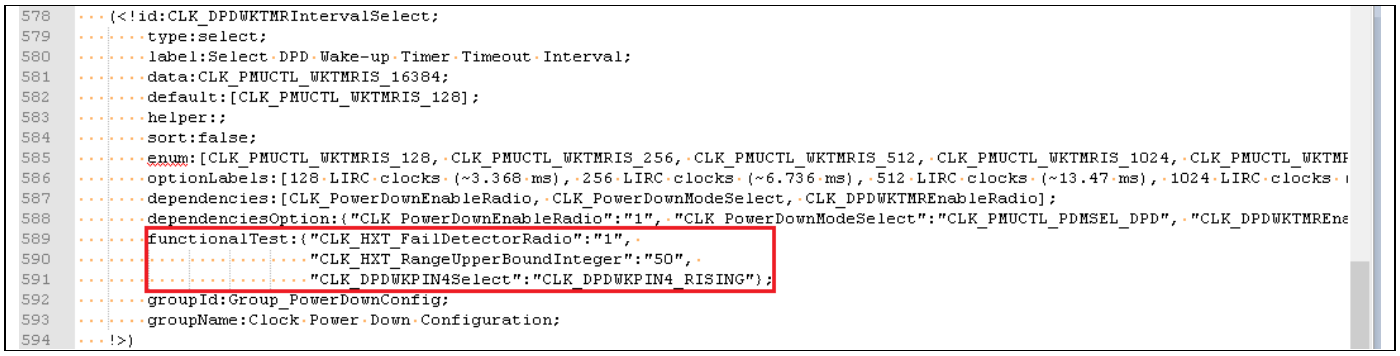
	A	B	C
1	ADCTriggerSrc	label	tirgger
2	ADCCompare0MatchCount	default	lower than minimum
3			

Regarding how to fix them,

To fix “data should be the same with the default”, click the Return to Default Settings button on the toolbar or manually modify data to keep it with the same with the default value.

To fix special abbreviation term, write the whole name and keep it inside parentheses, i.e., Auto Address Detection Operation Mode (AAD).

The field of **functionalTest** is used to assign some tags with the specific data value while the headerGenerator is generating its header files. Please note that only the tag where the field of functionalTest is defined will take effect while the headerGenerator is running and has nothing to do with the normal usage of NuCodeGen tool.



```
   (<!id:CLK_DPDWKTMRIntervalSelect;
       type:select;
       label:Select DPD Wake-up Timer Timeout Interval;
       data:CLK_PMUCTL_WKTMRIS_16384;
       default:[CLK_PMUCTL_WKTMRIS_128];
       helper:;
       sort:false;
       enum:[CLK_PMUCTL_WKTMRIS_128, CLK_PMUCTL_WKTMRIS_256, CLK_PMUCTL_WKTMRIS_512, CLK_PMUCTL_WKTMRIS_1024, CLK_PMUCTL_WKTMF
       optionLabels:[128 LIRC clocks (~3.368 ms), 256 LIRC clocks (~6.736 ms), 512 LIRC clocks (~13.47 ms), 1024 LIRC clocks (
       dependencies:[CLK_PowerDownEnableRadio, CLK_PowerDownModeSelect, CLK_DPDWKTMREnableRadio];
       dependenciesOption:{"CLK_PowerDownEnableRadio":"1", "CLK_PowerDownModeSelect":"CLK_PMUCTL_PDMSEL_DPD", "CLK_DPDWKTMREna
       functionalTest:{"CLK_HXT_FailDetectorRadio":"1", 
                       "CLK_HXT_RangeUpperBoundInteger":"50", 
                       "CLK_DPDWKPIN4Select":"CLK_DPDWKPIN4_RISING"};
       groupId:Group_PowerDownConfig;
       groupName:Clock Power Down Configuration;
   !>)
```

5.3 AutoTestProgram

Before running the automatic test, explain the purpose of the files related to the automatic test.

basic_test.exe：The main execution file of the automatic test.

cppcmd.exe：The execution file required for automatic testing are used to filter and delete redundant define content to generate the final project content.

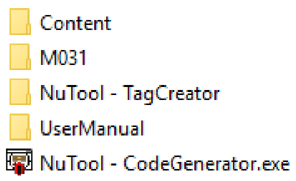
TagReg Mapping Table.xlsx：The automatic test is used to compare the correctness of the register info. excel table.

parsingexcel.exe：The execution file of parsing Mapping Table excel content.

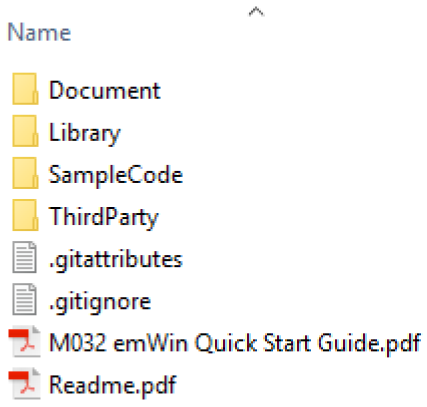
projfile.exe：The execution file is used to delete redundant project setting library content.

Some files need to be prepared and confirmed before running the automatic test.

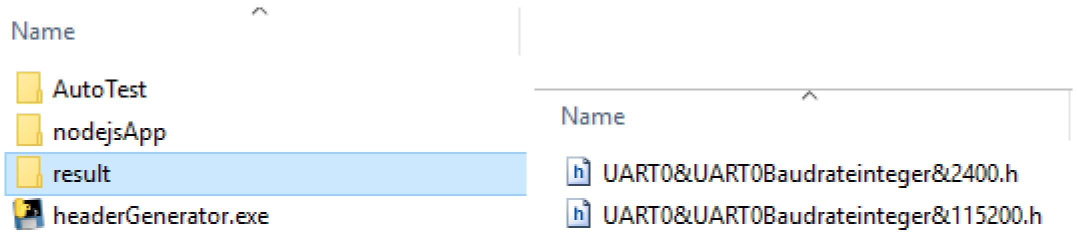
1. Download the complete NuCodeGen tool package from git and place the BSP folder in the same directory as the tool executable file. (e.g. M031)


Content
M031
NuTool - TagCreator
UserManual
NuTool - CodeGenerator.exe

Please note that the directory structure of the relative position of the BSP. After entering the M031 folder in the above figure, it will be as shown below.


Name
Document
Library
SampleCode
ThirdParty
.gitattributes
.gitignore
M032 emWin Quick Start Guide.pdf
Readme.pdf

2. HeaderGenerator has been used to generate the header file to be tested in the result folder, which is on the right as shown in the figure below. (UART for example)

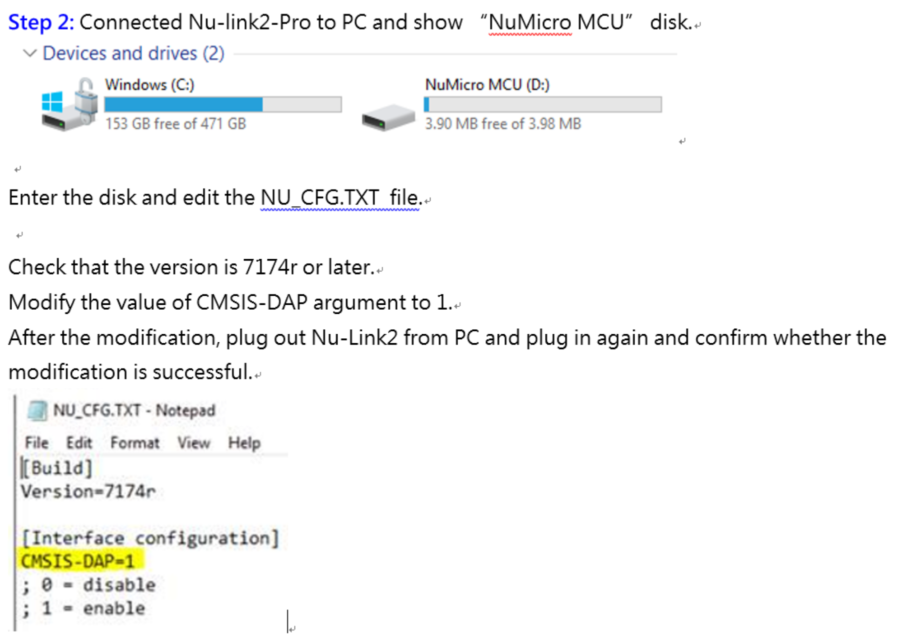

Name
AutoTest
nodejsApp
result
headerGenerator.exe

Name
UART0&UART0Baudrateinteger&2400.h
UART0&UART0Baudrateinteger&115200.h

3. Mapping table has the latest content. (e.g. M251 TagReg Mapping Table.xlsx)
   https://docs.google.com/spreadsheets/d/1Wt9tC0e_MxuhX2TvhRjW-BohnA_ciNHMUIF6q3SFSGQ/edit#gid=1935424765

4. Nu-Link2 ICE firmware has been updated to version 7174 or later version, and the target chip is connected.
5. Install Nu-Link command tool of 7174 or later version

The firmware can be upgraded by installing a new version of Keil or ICP tool. After upgrading the firmware, follow the steps below to enable the CMSIS-DAP function. (don't care about the version of the screenshot, just change the CMSIS-DAP to 1)



Step 2: Connected Nu-link2-Pro to PC and show "NuMicro MCU" disk.

Devices and drives (2)

Windows (C:) 153 GB free of 471 GB

NuMicro MCU (D:) 3.90 MB free of 3.98 MB

Enter the disk and edit the NU_CFG.TXT file.

Check that the version is 7174r or later.
Modify the value of CMSIS-DAP argument to 1.
After the modification, plug out Nu-Link2 from PC and plug in again and confirm whether the modification is successful.

```
[Build]
Version=7174r

[Interface configuration]
CMSIS-DAP=1
; 0 = disable
; 1 = enable
```

Next, explain how to run automatic test.
First open the command line windows and execute parsingexcel.exe to generate Test_database.txt. This file is needed for automatic testing, so please be sure to generate it. The usage is as follows.
parsingexcel.exe [full path of excel file name] [sheet name of each module]
Ex: parsingexcel.exe "C:\M251 TagReg Mapping Table.xlsx" FMC DAC
Get the contents of the sheet name as FMC and DAC in the mapping table.
Ex: parsingexcel.exe "C:\utest\M251 TagReg Mapping Table.xlsx" all

Get the contents after the third sheets in the mapping table.

After generating Test_database.txt, user can execute basic_test.exe to run automatic tests. The usage is as follows.

basic_test.exe [chip series name] –keil [full path of Keil IDE executable file] –download [full path of Nu-Link command tool executable file]

For example, if user just want to run keil build and download bin file test first, user can use the following command.

basic_test.exe m031 -keil "C:\Keil\UV4\UV4.exe" -download "C:\Program Files (x86)\Nuvoton Tools\NuLink Command Tool\NuLink.exe"

User can use basic_test.exe without adding any parameters, and the complete description of the parameters will be listed. User can increase or decrease the parameters according to the actual situation.

Here is a simple example with M031, suppose user want to run the automatic test of FMC

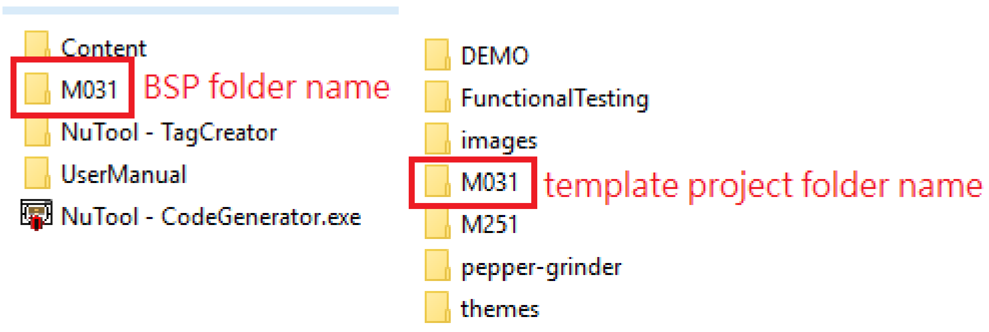
(A) HeaderGenerator.exe -c M031KIAAE -t FMC

(B) parsingexcel.exe "C:\AutoTest\NuCodeGen\Content\PeripheralConfigure\FunctionalTesting\M031 TagReg Mapping Table.xlsx" FMC

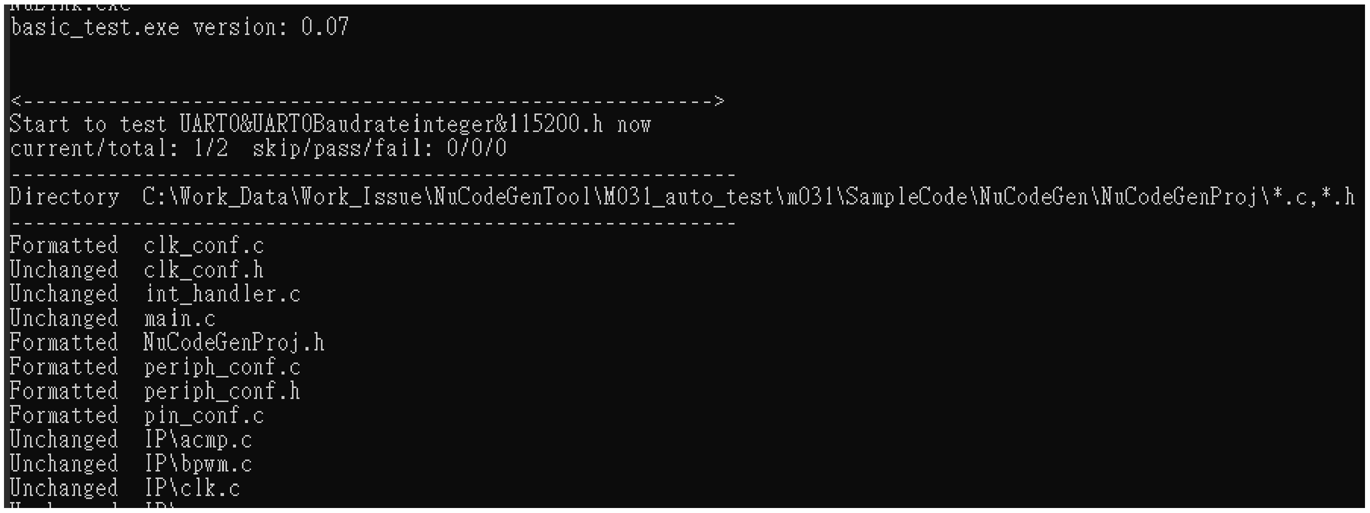
Note: The mapping table is not the excel file generated by HeaderGenerator.exe, please refer to https://docs.google.com/spreadsheets/d/1Wt9tC0e_MxuhX2TvhRjW-BohnA_ciNHMUIF6q3SFSGQ/edit#gid=1935424765

(C) basic_test.exe m031 -keil C:"\Keil_v5\UV4\UV4.exe" -download "C:\Program Files (x86)\Nuvoton Tools\NuLink Command Tool\NuLink.exe"

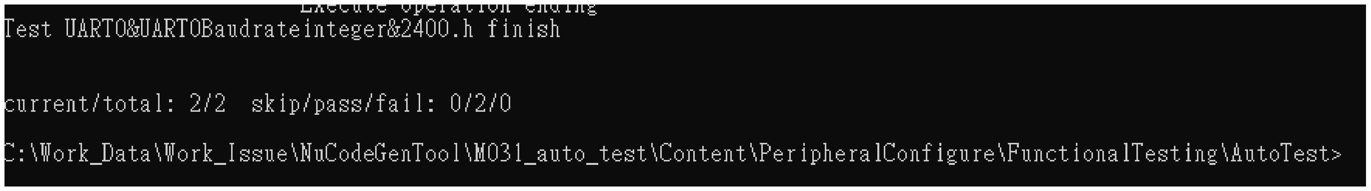
The M031 parameters here correspond to the two folder names in NuCodeGen.



Part of snapshot that the automatic test starts to run is as follows.

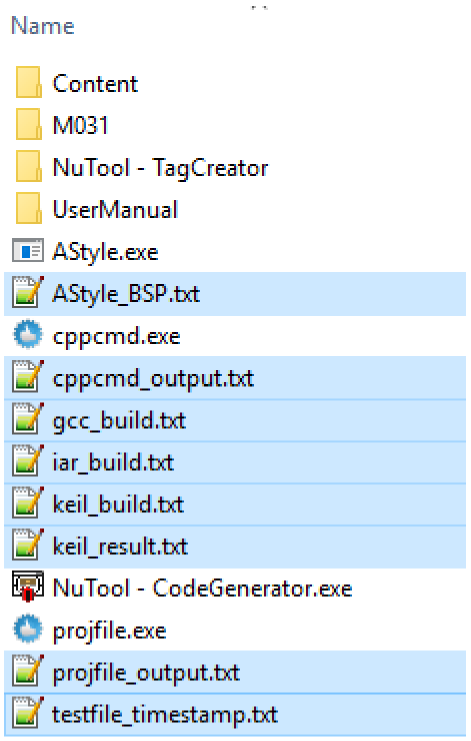


```
basic_test.exe version: 0.07


<----------------------------------------------------------->
Start to test UART0&UART0Baudrateinteger&115200.h now
current/total: 1/2  skip/pass/fail: 0/0/0
-------------------------------------------------------------
Directory  C:\Work_Data\Work_Issue\NuCodeGenTool\M031_auto_test\m031\SampleCode\NuCodeGen\NuCodeGenProj\*.c,*.h
-------------------------------------------------------------
Formatted  clk_conf.c
Unchanged  clk_conf.h
Unchanged  int_handler.c
Unchanged  main.c
Formatted  NuCodeGenProj.h
Formatted  periph_conf.c
Formatted  periph_conf.h
Formatted  pin_conf.c
Unchanged  IP\acmp.c
Unchanged  IP\bpwm.c
Unchanged  IP\clk.c
```



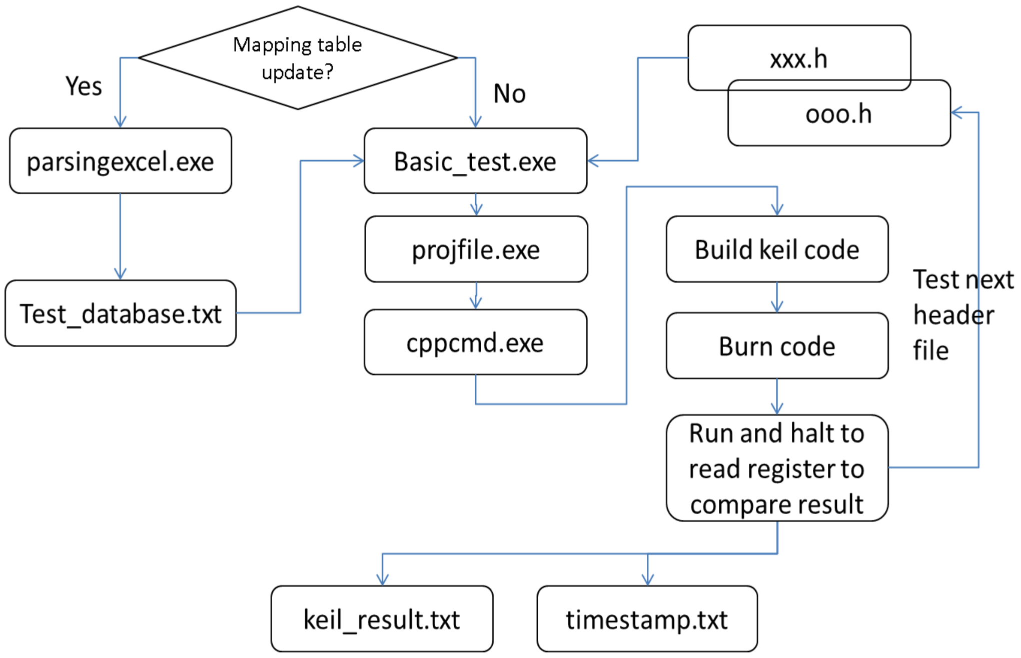
```
         Execute operation ending
Test UART0&UART0Baudrateinteger&2400.h finish

current/total: 2/2  skip/pass/fail: 0/2/0

C:\Work_Data\Work_Issue\NuCodeGenTool\M031_auto_test\Content\PeripheralConfigure\FunctionalTesting\AutoTest>
```

The root directory of the NuCodeGen tool will create the test log after running. For example, the test result can be found in keil_result.txt



```
Name
Content
M031
NuTool - TagCreator
UserManual
AStyle.exe
AStyle_BSP.txt
cppcmd.exe
cppcmd_output.txt
gcc_build.txt
iar_build.txt
keil_build.txt
keil_result.txt
NuTool - CodeGenerator.exe
projfile.exe
projfile_output.txt
testfile_timestamp.txt
```

The following figure shows the automatic test process and the final test result file.



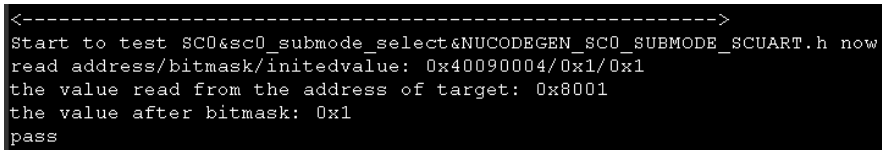
The files required for automatic testing are Test_database.txt. When the content of parsingexcel.exe or Mapping table is updated, user need to run parsingexcel.exe to generate a new Test_database.txt. After the automatic test reads the contents of Test_database.txt, copy the files from the backup_file directory to restore NuCodeGenProj to the default template project. Grab a header file from header directory at a time and replace the template project's periph_conf.h. After executing projfile.exe and cppcmd.exe, Start to build the code and the Keil IDE path must be correct, and make sure that the Keil IDE version can build code. (please check the license and whether it supports Cortex M23, etc.) After building the code, start to burn the code to the target chip and run code. Wait for the target chip to stop and then read the register for comparison. Finally, record the test result to keil_result.txt and record the modification time of each tested header file in timestamp.txt.

The purpose of timestamp.txt is to continue the last unfinished test. When -restart parameter is not added and timestamp.txt file exists, the automatic test will find the last time of the record, and the file after this time in the header file directory will continue to test.

The content of keil_result.txt will record the result information of each test header file, and a comprehensive result is listed at the end of the file

The following figure shows the test result message of one of the header files.

It contains the result of pass or fail, the read position, bitmask, and expected result.



```
<---------------------------------------------------------->
Start to test SC0&sc0_submode_select&NUCODEGEN_SC0_SUBMODE_SCUART.h now
read address/bitmask/initedvalue: 0x40090004/0x1/0x1
the value read from the address of target: 0x8001
the value after bitmask: 0x1
pass
```

Each run will generate a keil_result.txt to store the current test results, the number will increase in order, keil_result_1.txt, keil_result_2.txt…

6 Q&A

1. Q: What should the data value in the tag be when the tool is being released?

   A: The data value should be the same with the default. To do that, click the Return to Default Values button on the toolbar.

2. Q: What is difference between tag type of checkbox and checkboxBoolean?

   A: When unchecked, the data value of checkbox would be empty, but that of checkboxBoolean is false. If you want to use unchecked state as dependencies, please use checkboxBoolean tags.

3. Q: What is difference between HexText and Integer?

   A: HexText can input hexadecimal number. Integer can input decimal number.

4. Q: What is general tag?

   A: One .txt file only has one general tag which serves the whole .txt file and must be the first tag.

5. Q: Does Hidden tag serve as another tag's dependencies?

   A: No, it cannot do that.

6. Q: Dose the content of tags have any preserve keyword?

   A: Yes, the preserved keywords are NUC_OR, NUC_AND, NUC_NULL, NUC_NONE and NUC_ANY.

7. Q: How to place a tag which has no label on the second row?

   A: Input the data value as <br> to make the label become whitespace.