TASKING

Using the TASKING Pin Mapper for ARM

Table of Contents

Chapter 1. Introduction

Nowadays many microcontrollers are equipped with a large number of on-chip peripheral modules. These microcontrollers are made available in surface mount packages with various numbers of physical pins. The number of pins usually does not allow all peripheral modules to be used simultaneously. Hardware registers on the microcontroller allow for configurable assignment of peripheral module signals to physical pins. This means that you have to select the proper device for your application and properly initialize hardware registers from software. The purpose of the TASKING Pin Mapper is to assist you in performing those tasks.

1.1. Port Pin

The basic entity of the TASKING Pin Mapper is the port pin. Port pins are physical pins of the chip. A port pin name typically consists of a port identifier and, if the port identifier is shared by multiple pins, a sequential pin number, for example PA1. A port pin name is not package specific. Each package has its own specific assignment of port pin names to package pin numbers.

The basic configuration item of a port pin is its function (which in most cases is determined by the signal assigned to it). Some ports have dedicated functions, mostly related to power supply or chip infrastructure. Other ports have configurable functionality (ultimately defined by run-time register settings). The required function is application specific.

Port functions are listed by their symbolic names (which are mostly signal names, for example RX, TX).

Port pins can have a number of other chip specific properties. Port pins are visible in the Package view. You can assign a user-defined symbol name to the port pin and you can make the symbol names visible in the Package view. You can use filters in pin selection views or you can use property values as background color codes in the Package view. You can change the configurable property values.

1.2. Peripheral

ARM devices, such as the STM32F2xx, have support for several on-chip peripherals, such as serial controllers, timer units, and analog-to-digital converters.

1.3. Connection

The TASKING Pin Mapper allows you to define connections between modules, where a module is either a port pin or a peripheral.

You can connect port pin modules to peripheral modules by selecting the pin's Mode and its Chip input function or its Chip output function. The peripheral module side of a connection is referred to as a virtual pin. Virtual pins are internal to the chip and are not visible in the Package view.

If you have defined a connection (either valid or invalid) the "follow the connection" button ( ) becomes available and you can navigate from one side of the connection to the other side. This applies to virtual

pins as well as physical pins. Virtual pins usually do not have a configurable direction. They are either Input or Output (and some virtual pins are bidirectional).

See Section 2.3, *Configuring the Pins* for a step-by-step example.

1.4. Package

When you create a new pin mapper document, as explained in Section 2.1, *Adding a Pin Mapper Document to an Existing Project*, you first select a processor (family), for example STM32F217I(E-G)Hx, and a package, for example UFBGA176. This selection defines the package pin numbers and the linking of package pins to port pins.

For QFP (Quad Flat Package) type packages the pin number consists of the package identifier and a sequential pin number. For BGA (Ball Grid Array) type packages the pin number consists of the package identifier and a matrix cell number.

Chapter 2. Getting Started with the TASKING Pin Mapper

The TASKING Pin Mapper is the graphical interface to configure the pins and modules in a package.

2.1. Adding a Pin Mapper Document to an Existing Project

To configure the pins and modules of a package you need to create a project and add a Pin Mapper document.

1. If you have no existing C project, follow the steps to create a new C project, as explained in the *Getting Started with the TASKING VX-toolset for ARM*.
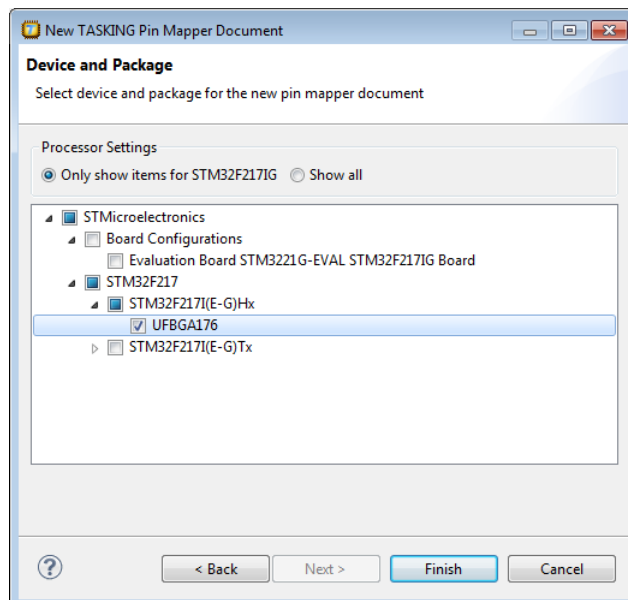
   In the following steps we assume you have an existing STM32F217IG C project named `myproject`.

2. From the **File** menu, select **New » TASKING Pin Mapper Document**.

   The New TASKING Pin Mapper Document wizard appears.

3. Select the **Project** folder for the Pin Mapper document: type the name of your project (`myproject`) or click the **Browse** button to select a project.

4. In the **File name** field, enter a name for the Pin Mapper document, for example `myproject.pincfg` and click **Next**.

   The Device and Package page appears.

5. Select a board configuration or select a processor, for example STM32F217I(E-G)Hx, and package, for example UFBGA176, for which you want to make a pin configuration and click **Finish**.
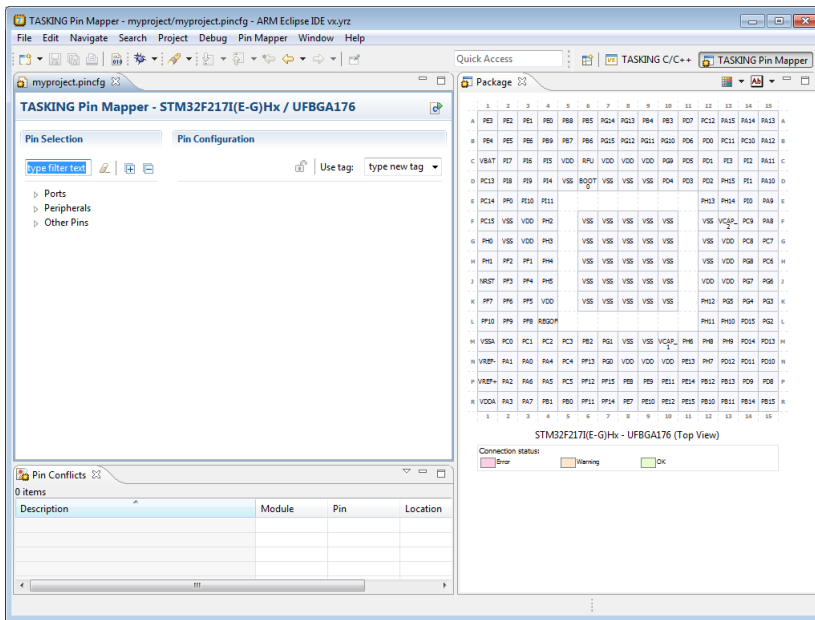
   When you select a board configuration, the pins are pre-configured. A pin configuration file with extension `.pincfg` *is added to the existing project. Because the TASKING C/C++ perspective is still active, Eclipse asks to open the TASKING Pin Mapper perspective.*

6. Optionally, enable the option **Remember my decision** and click **Yes**.

   The TASKING Pin Mapper perspective is opened.
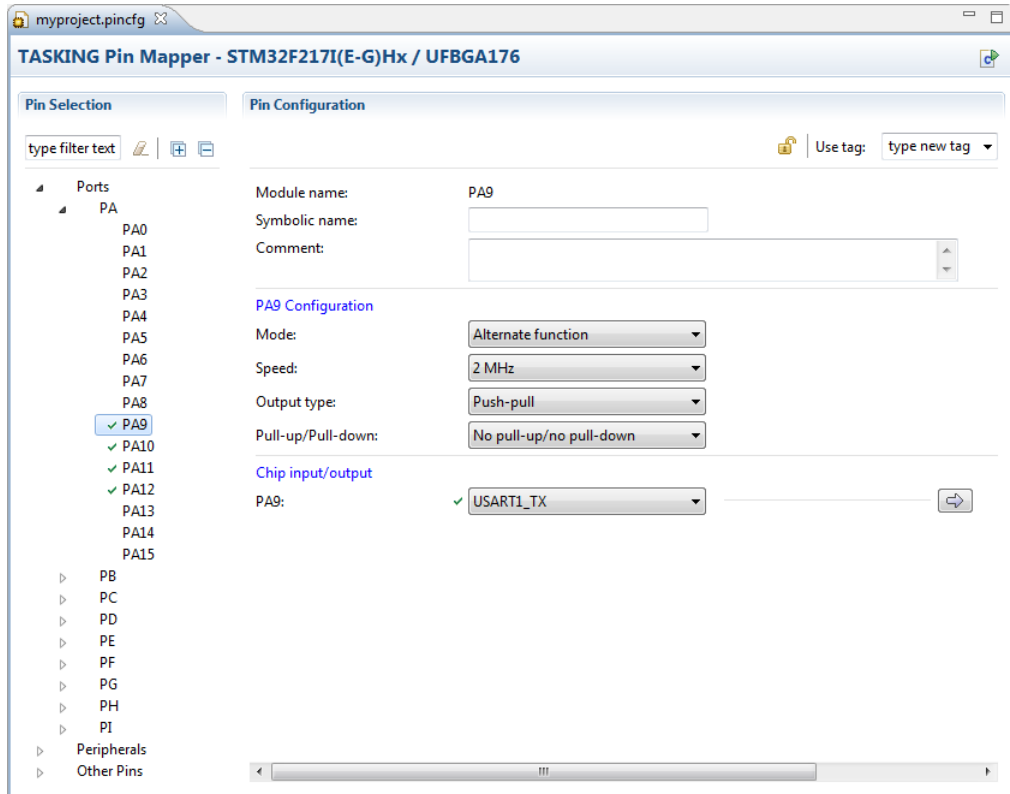
2.2. TASKING Pin Mapper Perspective

The TASKING Pin Mapper perspective contains several views. Each of the views are discussed in the following sections.

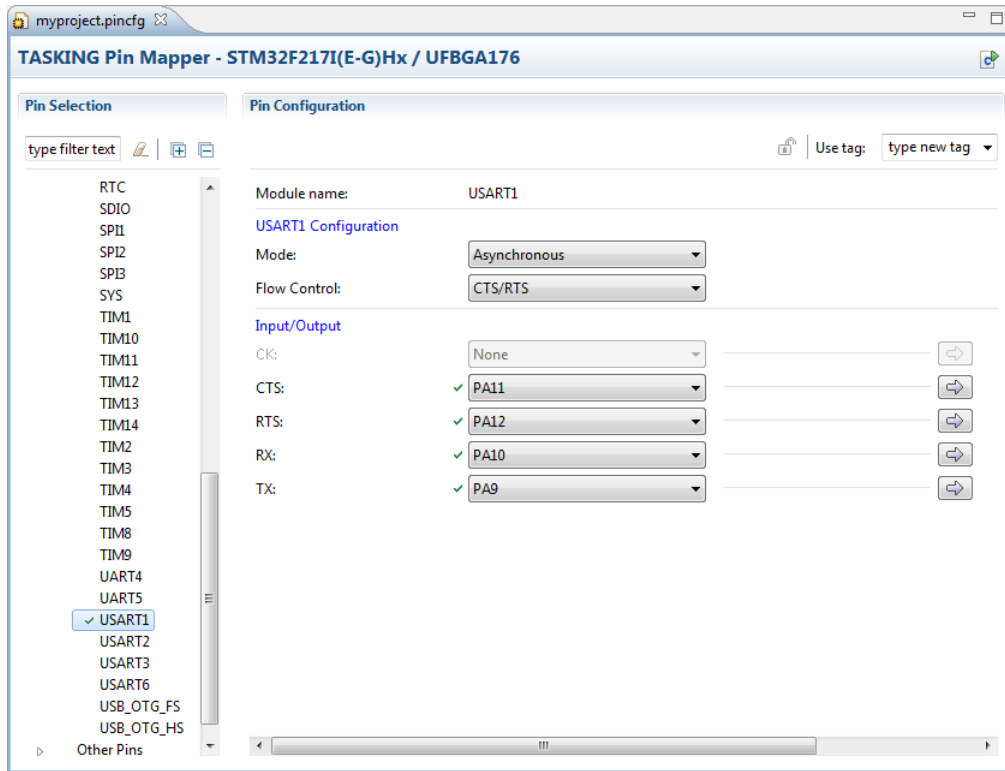## 2.2.1. TASKING Pin Mapper Editor View

The TASKING Pin Mapper Editor view is the main area where you can make changes to your configuration.

Port example

Peripheral example

The following toolbar icon is available:

Icon	Action	Description
	Generate Code	Generates the source code and adds it to your project.

Pin Selection

In the left pane the port pins, peripherals and other pins are listed. Port pins can be logically grouped in different ways. Each of these groupings defines a tree view. Groups can be logically grouped into higher level groups as long as the overall structure can be represented as a mathematical tree. The leafs of the tree are references to port pins, peripherals or peripheral sub-modules.

The tree structure of the grouping allows you to expand and collapse the view.

In the `type filter text` edit field, you can add a port pin selection filter in order to reduce the number of visible port pins. The filter is case insensitive. For example, type `adc` to only show pins that have "adc" in their name. Wildcards are allowed.

Pin Configuration - Port pin

When you click on a port in the left pane, the configuration appears in the right pane. The following information can be present:

- **Module name** - The name of the selected module or port pin.

- **Symbolic name** - You can assign a user-defined symbol name to the port pin. You can make the symbol names visible in the Package view.

- **Comment** - Any user comments you can add here.

- **Mode** - This field sets the port mode register. You can configure each of the pins as output (push-pull or open-drain, with or without pull-up or pull-down), as input (floating, with or without pull-up or pull-down) or as peripheral alternate function. The analog mode is available for ADC and DAC. Pins have different functions. Normally a pin refers to GPIO, General Purpose I/O, so that you can use the pin directly by writing to and reading from the relevant registers. For alternate functions you can refer to other peripherals.

- **Speed** - This field sets the port output speed register (2, 25, 50 or 100 MHz).

- **Output type** - This field sets the port output type register (push-pull or open drain).

- **Pull-up/Pull-down** - This field sets the port pull-up/pull-down register (no pull-up/pull-down, pull-up or pull-down).

- **Chip input/output**. Here you can make a pin connection. See Section 2.3, *Configuring the Pins* for more information. From the drop-down list select the appropriate function.

  If you set the mode to input, select the GPIO function or an external interrupt function EXTI_xxxx as chip input. If you set the mode to output, select the GPIO function as output. For the ADC and DAC set the mode to analog mode. For other peripherals, set the mode to alternate function.

For more information about the different configurations and port registers, see chapter *General-purpose I/Os (GPIO)* in the *STM32Fxxxxx Reference manual* for your device.

When you make a selection a '*' can appear in front of other selectable pin names. This indicates that when you select a pin name with a '*', this may result in an error.

Pin Configuration - Peripheral

When you click on a peripheral in the left pane, the configuration appears in the right pane. The following information can be present:

- **Module name** - The name of the selected peripheral.

- **Configuration**. The configuration depends on the peripheral you selected. In most peripherals you can set a function mode, which connects a predefined list of I/O functions (virtual pins) to port pins. When you set the mode to Custom, you can connect virtual pins to port pins manually.

- **Input/Output** - A list of port I/O functions (virtual pins). Based on the configuration setting, you can connect virtual pins to port pins.

Pin Configuration - Other pins

For other pins the following information can be present:

- **Module name** - The name of the selected other pin.

- **Boot domain** - The boot domain the pin uses.

- **Power domain** - The power domain the pin uses.

- **Reset domain** - The reset domain the pin uses.

Lock/Unlock

When you want to be certain that the settings cannot be changed anymore, you can click the 🔓 button to lock the current settings. You can always click the 🔒 button to unlock the settings again.

Use tag

You can assign a tag name to a connection. The name you enter in the **Use tag** field will be used for the following connections you make. So, change the field before you make a connection. The tag name is user defined. You can use it the way you want. For example, to tag connections of the same type.

Undo/Redo

You can undo (Ctrl+Z) or redo (Ctrl+Y) one or more actions.

Back/Forward

With the ⇦ (Back) and ⇨ (Forward) buttons you can navigate to a previously opened pin configuration.

Save

From the **File** menu, select **Save** (Ctrl+S) or click 💾 to save the configuration.
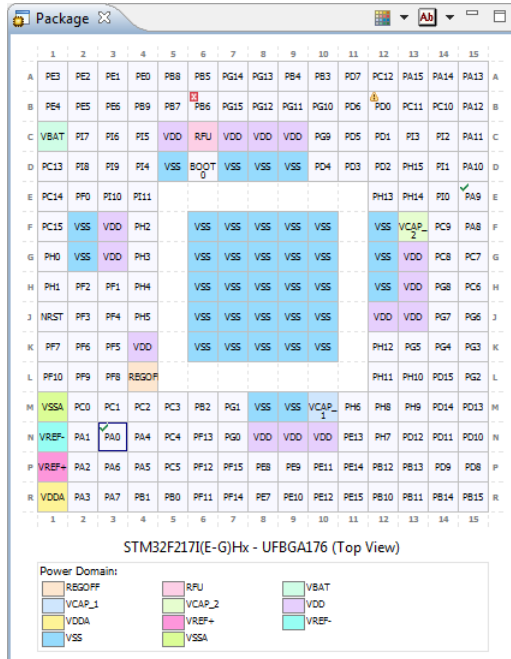
2.2.2. Package View

The Package view shows a graphical representation of the package. For example, it shows the individual pins in a Ball Grid Array.

A square around a pin marks the selected pin. In the package above PA0 is selected. A green check mark indicates that the pin has a valid connection; PA0 and PA9 in the package above. A red cross indicates an error; PB6 in the package above. A triangle with exclamation mark indicates a warning; PD0 in the package above. When you click on a pin, the pin appears in the editor. If the pin contains an error, you can see what is wrong in the Pin Conflicts view.

By default, the background colors of the pins indicate the connection status. But you can change the color properties of the view to show Boot Domain, Power Domain or Reset Domain. Use the drop-down menu next to ▦ to switch color properties.
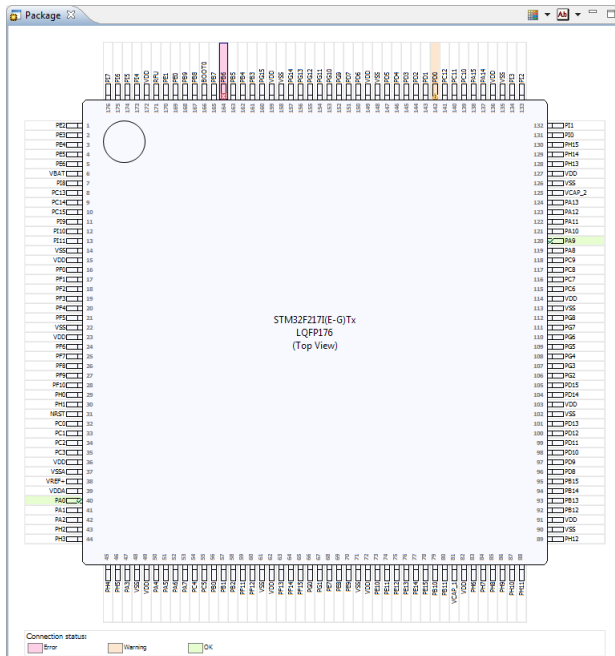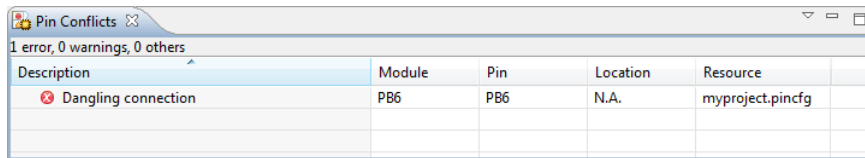
Using the TASKING Pin Mapper for ARM

STM32F217I(E-G)Hx - UFBGA176 (Top View)

Boot Domain:
BOOT0

STM32F217I(E-G)Hx - UFBGA176 (Top View)

Power Domain:
REGOFF · RFU · VBAT
VCAP_1 · VCAP_2 · VDD
VDDA · VREF+ · VREF-
VSS · VSSA

10

With the drop-down menu next to you can switch between module names and symbol names.

If you have selected a Quad Flat Package, the Package view may look similar to:

2.2.3. Pin Conflicts View

Potentially conflicting pin assignments are reported in the Pin Conflicts view. For example, if multiple virtual pins are assigned to the same port pin then there is a potential conflict. Also, if the same virtual pin is assigned to multiple port pins then there is a potential conflict.

When you double-click on a conflict the corresponding pin configuration will be visible in the Editor view.

See Section 2.7, *Errors and Warnings* for a list of errors and warnings.

2.3. Configuring the Pins

Basically you can connect pins in two different ways. The first way is to select a port pin and then assign a peripheral virtual pin (function) to it. The second way is to select a peripheral virtual pin (function) and then assign a port pin to it.

You can choose between input or output mode. If an application is going to use a port pin in both directions then you should select the initial run-time direction of the port. This is necessary because the pin mapper will generate port pin initialization code and needs to know whether to configure for input or for output.

To connect pins from the port pin side

1. In the Project Explorer view double-click on a configuration file (`myproject.pincfg`) to view its contents in the other views.

2. In the Pin Selection pane, select a port pin. For example, **Ports » PA » PA0**.

3. In the Pin Configuration pane select the **Mode**. In this example we select **Alternate function**.

4. Select a port output **Speed** (2, 25, 50 or 100 MHz).

5. Select a port **Output type** (push-pull or open drain).

6. Specify the I/O **Pull-up/Pull-down**.

7. Connect the input pin to a peripheral virtual pin:

    • From the **PA0** drop-down box select a virtual pin, for example TIM2_CH1.

    A green check mark ✓ *appears if the connection is successful.*

8. (Optional) When you want to prevent that these settings can be changed, click 🔓 to lock the settings.

   A little lock 🔒 appears next to the port pin.

9. From the **File** menu, select **Save** (Ctrl+S) or click 💾 to save the configuration.

When you click ⇨ you follow the connection from source to destination or vice versa. The corresponding selection is shown in a light blue color.

To connect pins from the peripheral side

1. In the Project Explorer view double-click on a configuration file (`myproject.pincfg`) to view its contents in the other views.

2. In the Pin Selection pane, select a module. For example, **Peripherals » USART1**.

3. In the Pin Configuration pane select a function **Mode** or select Custom and select the pins you want to assign to the virtual pins.

   A green check mark ✓ appears if the connection is successful.

4.
   Click ⇨ to follow the connection to the port pin side. The corresponding selection is shown in a light blue color.

5. Because we selected a peripheral in this example, set the **Mode** to **Alternate function**. If you selected an ADC or DAC, set the mode to **Analog mode**. If you selected an external interrupt function (EXTI), set the mode to **Input mode**.

6. Specify the **Speed**, **Output type** and **Pull-up/Pull-down**.

7. (Optional) When you want to prevent that these settings can be changed, click 🔓 to lock the settings.

   A little lock 🔒 appears next to the port pin.

8. From the **File** menu, select **Save** (Ctrl+S) or click 💾 to save the configuration.
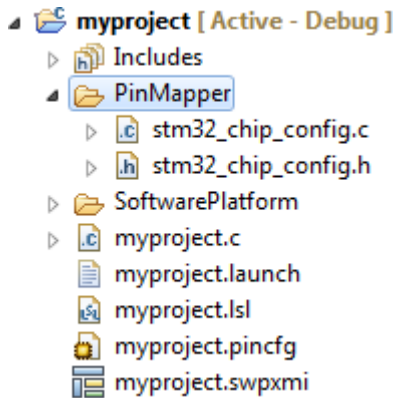
2.4. Generating and Using the Source Code

When you have configured the pins you are ready to add the Pin Mapper sources to your project.

Generate source code

• From the **Pin Mapper** menu, select **Generate Code** or click 🔧.

   The Pin Mapper sources are generated and are added to your project in the folder PinMapper.

Using the Pin Mapper sources

The Pin Mapper sources contain the initialization functions for all peripherals configured in the Pin Mapper. The generated code is meant to work closely together with the STM32 Access Peripheral Library.

> You can easily add the STM32 Peripheral Access Library to your project by adding a TASKING Software Platform document, add the STM32 Peripheral Access Library plug-in and generate the code.

If you are also using Software Platform's Peripheral plug-ins, the Software Platform will automatically call the functions generated by the Pin Mapper to initialize the peripherals. If you do not use Peripheral plug-ins, you can also access the Pin Mapper sources from your application. To use the generated files, you must add a `#include` statement into your top-level source file (typically `main.c`):

```
#include "PinMapper/stm32_chip_config.h"
```

You can now use the functions of the Pin Mapper in your sources. For example,

```
int main( void )
{

    /* configure TIM2 and USART1 */

    pincfg_tim2_pin_config();
    pincfg_usart1_pin_config();
    ...
}
```

2.5. Generate CSV File

You can save the pin mapper configuration in a file with comma-separated values (CSV).

- From the **Pin Mapper** menu, select **Generate CSV File** ( ).

    The CSV file (pincfg_*.csv) is generated in the folder `PinMapper` of your project.

The following is an example of such a file:

```
"Name","Pin","Function","Mode"
"PA4","N4","ADC1_IN4","Analog mode"
```

You can open the file with any text editor or in a spreadsheet program.

2.6. Pin Mapper Preferences

You can use the Preferences dialog in Eclipse to specify how the Pin Mapper should operate.

To set preferences

1. From the **Window** menu, select **Preferences**.

    The Preferences dialog appears.

2. Select **TASKING » Pin Mapper**.

    The Pin Mapper page appears.

3. Set your preferences and click **OK**.

You can set the following preferences:

Generate code on save

By default the TASKING Pin Mapper asks if you want to generate code when you save a document (**Prompt**). You can choose to do this automatically (**Always**) or **Never**.

2.7. Errors and Warnings

The TASKING pin mapper reports errors and warnings in the Pin Conflicts view. When you encounter an error, you can double-click on the error message to go to the conflicting situation.

A '*' can appear in front of selectable pin names. This indicates that when you select a pin name with a '*', this may result in an error.

Error: Dangling connection

A connection consists of two ends; a source and a destination. If the source is connected to a destination while the destination is not connected to the source then the connection is reported to be dangling. There are two situations in which this error can appear.

- Situation 1. When dealing with a 1-to-N connection (one source, many destinations), on the right-hand side of the drop-down box a list of destinations is shown. Each destination has a check box and you can make one or more selections. Initially nothing is selected and at that point the error appears. For example,

  1. Select **Ports » PD » PD0**.

  2. Change the **Mode** to **Alternate function**.

  3. Change the chip input/output **PD0** from **None** to **FSMC_D2**.

     Several destinations appear, all unselected. The error appears to notify you to make a selection. Once you made a selection, the error disappears. To make a correct connection, it can be necessary to follow the connection and enable the virtual pin at the peripheral side..

- Situation 2. When you attempt to connect a source to a destination that already has a connection. In this case the source is connected to the destination, while the destination keeps its old connection. For example,

  1. Select **Ports » PA » PA2**.

  2. Change the **Mode** to **Output mode**

  3. Change the chip input/output **PA2** from **None** to **GPIO**.

  4. Select **Peripherals » ETH**.

  5. Set the **Mode** to **Custom**.

  6. Change the input/output **MDIO** from **None** to * **PA2**.

     The error appears since ETH_MDIO is now connected to PA2, while PA2 is still connected to GPIO.

Error: Incorrect settings for *name*, expected pins *name1* and *name2* to be connected

This error appears when you connect one pin, where multiple pins need to be connected.

For example,

1. Select **Ports » PA » PA9**.

2. Change the **Mode** to **Alternate function**

3. Change the chip input/output **PA9** from **None** to **USART1_TX**.

4. Follow the direction (click ⇨).

5. Change the **Mode** of USART1 from **Disabled** to **Synchronous**.

The error will disappear when you also connect input Input/Output **TX** from **None** to **PA9**.

Warning: Multiple selections for a 1-to-N connection

This warning appears when you make more than one selection for a 1-to-N connection.

For example,

select **Ports » P00 » P00_0** and change the **Chip input** configuration **IN** from **None** to **Select**. Several destinations appear, all unselected. Now select more than one destination.

1. Select **Ports » PD » PD0**.

2. Change the **Mode** to **Alternate function**.

3. Change the chip input/output **PD0** from **None** to **FSMC_D2**.

   Several destinations appear, all unselected..

4. Select more than one destination.

The warning disappears if you only select one destination.