TASKING®

Using the TASKING Pin Mapper for AURIX

MA160-870 (v1.0) February 12, 2016

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 TMS, AN4 or P10_2. 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.

A port function definition consists of a symbolic name, a direction and a description. The TASKING Pin Mapper distinguishes between input functions and output functions. Functions are listed by their symbolic names (which are mostly signal names, for example ASCLK3).

Port pins can have a number of other chip specific properties. Such properties can be fixed (for example the type of pad the port is connected to) or configurable (for example the pad state). 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

AURIX devices, such as the TC27xC, 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 Direction and its Chip input function or its Chip output function. The peripheral module side of a connection is referred to as a virtual pin. Some peripheral modules can also be connected to other peripheral modules. Such a connection

is made between two virtual pins. Virtual pins are internal to the chip and are not visible in the Package view. However, virtual pin conflicts are reported in the Pin Conflicts view, for example when you try to connect a GTM virtual output pin to a physical port pin and at the same time to a virtual input pin of an ADC peripheral module.

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 TC27XC, and a package, for example BGA292. This selection defines the package pin numbers and the linking of package pins to port pins. It is possible that a port pin is linked to a single package pin multiple times by means of port pin aliases, for example AN39 and P40_9.

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 TriCore*.

   In the following steps we assume you have an existing TriCore TC27xC C project named `myproject`.

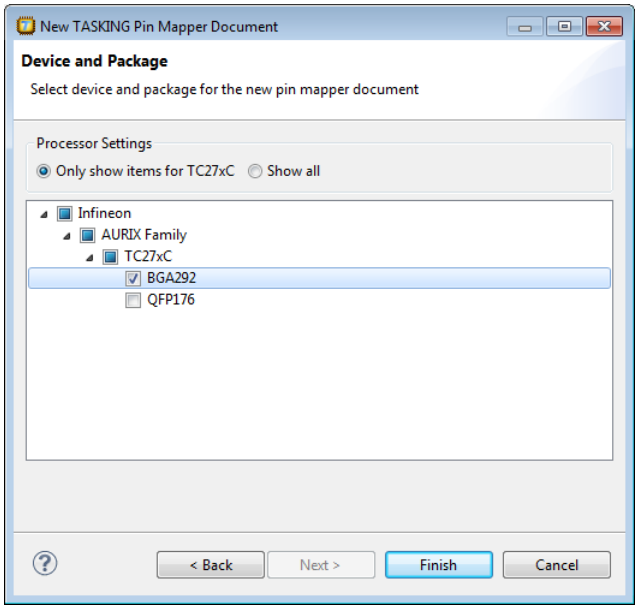
2. From the **File** menu, select **New » TASKING Pin Mapper Document**.

   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 the processor, for example TC27xC, and package, for example BGA292, for which you want to make a pin configuration and click **Finish**.

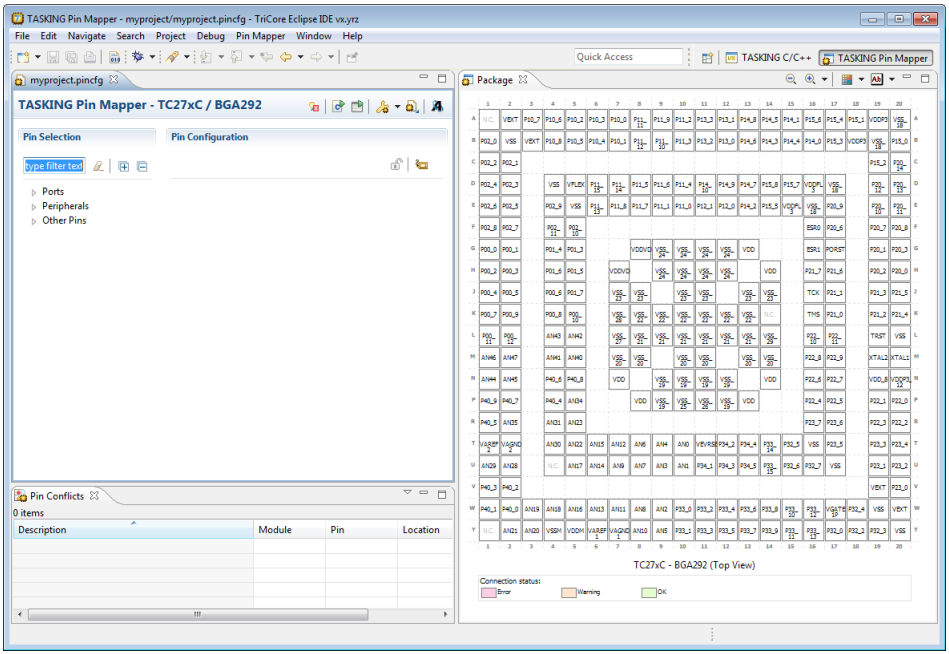
   A pin configuration file with extension `.pincfg` is added to the existing project. Because the TASKING C/C++ perspective is still active, Eclipse asks to open the TASKING Pin Mapper perspective.

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

   The TASKING Pin Mapper perspective is opened.

2.2. TASKING Pin Mapper Perspective

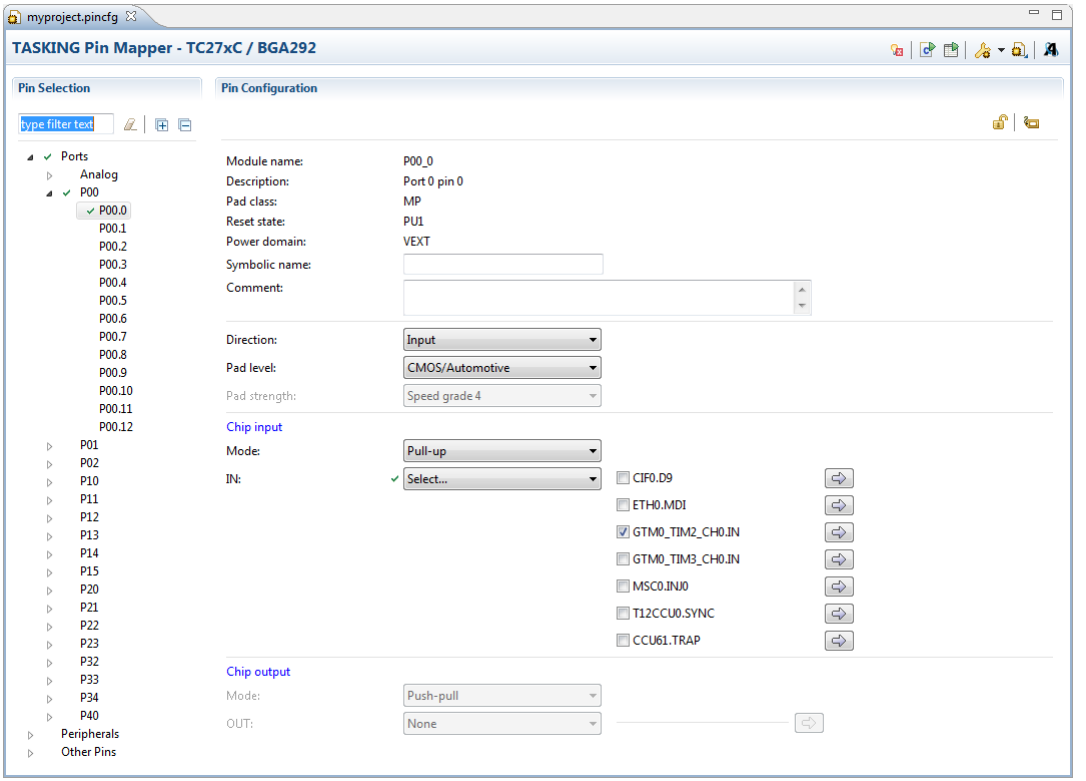
The TASKING Pin Mapper perspective contains several views. Each of the views are discussed in the following sections.



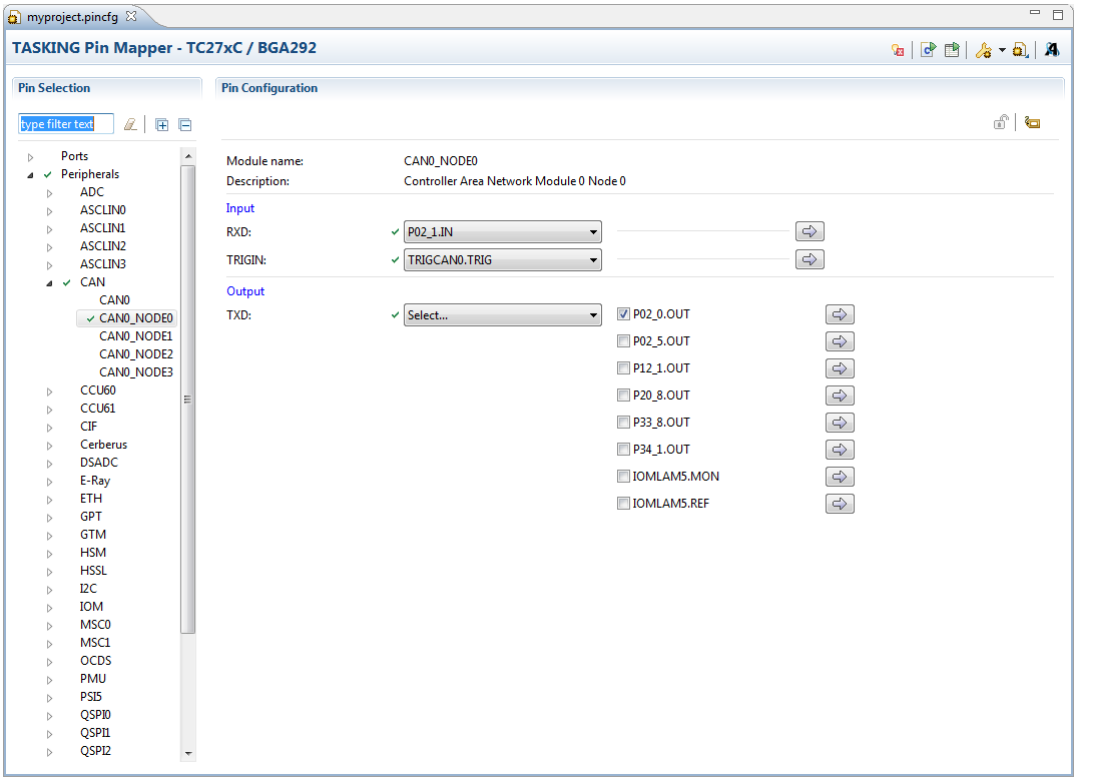
2.2.1. TASKING Pin Mapper Editor View

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

Port example



Peripheral example

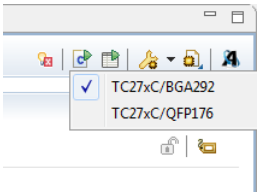


The following toolbar icons are available:

Icon	Action	Description
	Solve Pin Conflicts	Solve conflicts with pin mappings. See Section 2.2.3, *Pin Conflicts View*.
	Generate Code	Generate the source code and add it to your project. See Section 2.5, *Generating and Using the Source Code*.
	Generate CSV File	Save the pin mapper configuration in a file with comma-separated values (CSV). See Section 2.6, *Generate CSV File*.
	Select Device/Package	Select another package for your device. For example, from BGA to QFP or vice versa.
	Import Pin Configuration	Import pin configurations from a file and merge it with the current configuration.
	Launch Altium Designer	Launch Altium Designer with the pin configurations. See Section 2.7, *Launch Altium Designer*.

Select Device/Package

With the Select Device/Package drop-down menu ( ) you change the package for your device. For example:



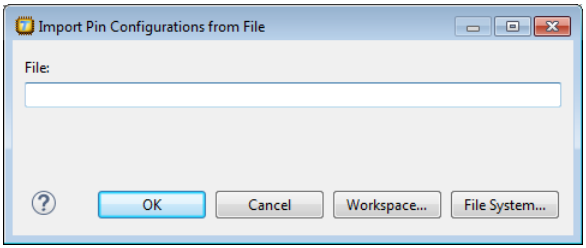
After the package switch you still keep the current configuration settings. If the same pin mapping can be used on the new device/package, the pin mapper automatically applies the mapping. For mapping parts which cannot be applied, the pin mapper issues a warning message.

Import Pin Configuration

You can import saved pin configurations and merge it with the current configuration. This can be useful, for example, when you have a set of configurations for ports and peripherals that you want to use in several projects.

1. Click the Import Pin Configuration button ( ).

   The Import Pin Configurations from File dialog appears.



2. Type the full path name of the configuration file (`.pincfg`), or use the **Workspace** button to select a configuration from one of your project directories, or use the **File System** button to select one from any directory.

   The imported configuration will be merged with the current configuration.

If the current configuration already has settings for a pin or peripheral and the imported configuration has other settings, pin conflicts errors or warnings may occur.

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 `an` to only show pins that have "an" in their name. Wildcards are allowed.

Pin Configuration

When you click on a port or module 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.
- **Alias** - An alias for the module name.
- **Description** - The description of the selected module or port pin.
- **Pad class** - The assignment of a port pin to one of the pad classes.
- **Reset state** - The state of the pin after reset. For example, PU (Pull-up) or PD (Pull-down).
- **Power domain** - The power domain the pin uses.
- **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.
- **Pin function** - You can configure some pins as digital or analog.
- **Direction** - You can specify if a pin must be configured as an input pin or an output pin.
- **Pad level** - Here you can select CMOS/Automotive or TTL.
- **Pad strength** - For an output pin you can specify the Speed grade.
- **Mode** - The chip input or output mode. For chip input this can be Pull-up, Pull-down or Tri-state. For chip output this can be Push-pull or Open drain.
- Other properties allow you to make a pin connection. See Section 2.3, *Configuring the Pins* for more information.

When you make a selection a '*' can appear in front of other selectable pin names. When you select a pin name with a '*', this results in an error.

Lock/Unlock

When you want to be certain that the port 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.

Tags

You can assign a tag name to a connection. When you click the (Edit tags) button, you can edit the tag field to the right of the (Follow connection) button. 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

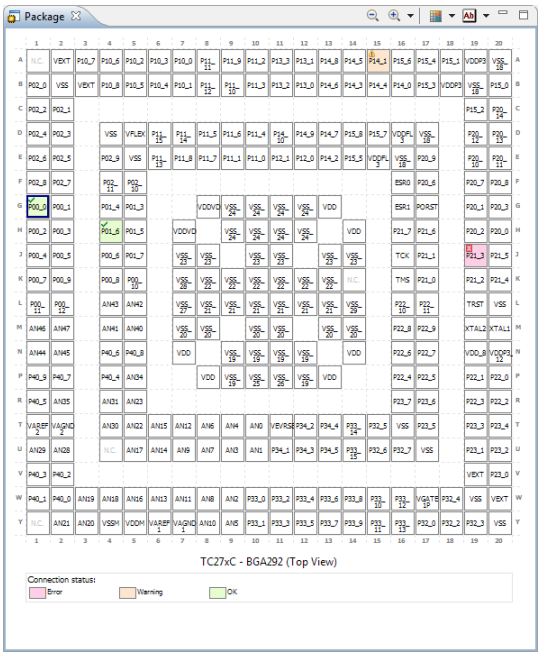
With the (Back) and (Forward) buttons you can navigate to a previously opened pin configuration.

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 P00_0 is selected. A green check mark indicates that the pin has a valid connection; P00_0 and P01_6 in the package above. A red cross

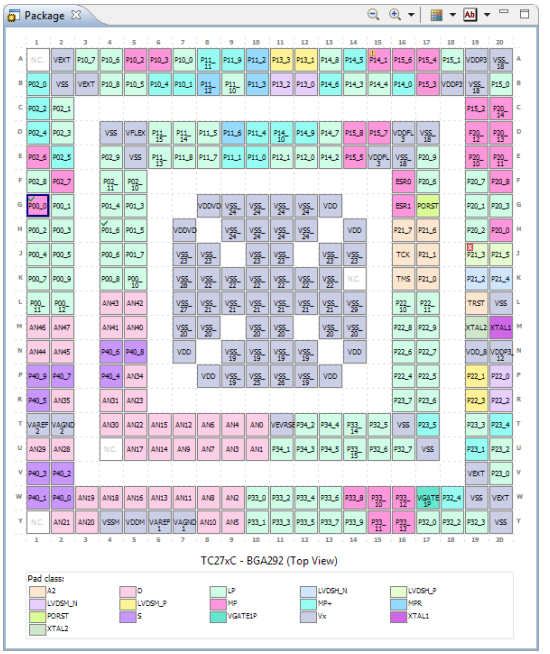
indicates an error; P21_3 in the package above. A triangle with exclamation mark indicates a warning; P14_4 in the package above. When you click on a pin, the pin appears in the editor. If the pin contains an error, you can see what is wrong in the Pin Conflicts view.

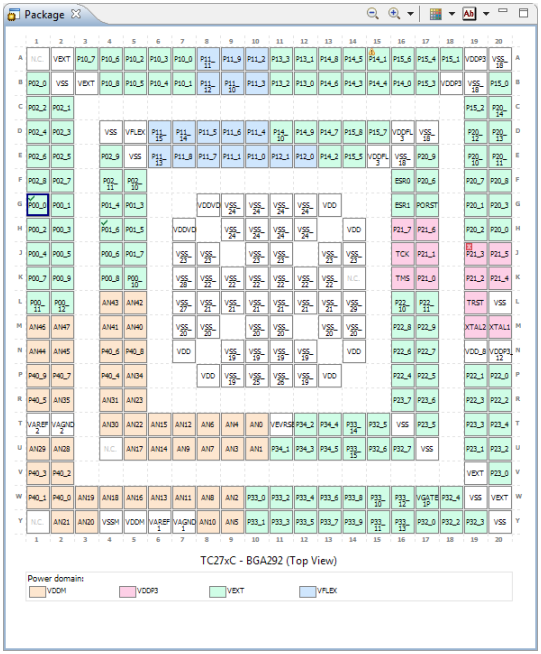
Zoom in / Zoom out

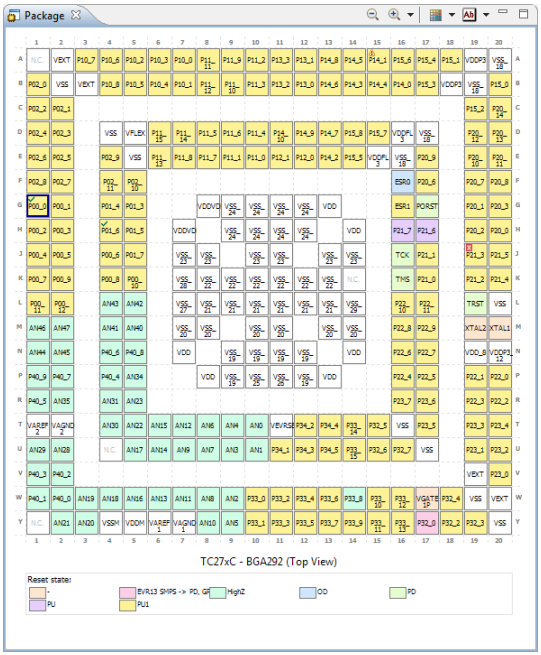
With and you can zoom in or zoom out on the package view. Use the drop-down menu next to to select a zoom factor directly.

Background Color

By default, the background colors of the pins indicate the connection status. But you can change the color properties of the view to show static configuration items (Pad class, Power domain or Reset state), or dynamic configuration items (Direction, Pad level, Pad strength, Mode or Pin function). Use the drop-down menu next to to switch color properties. Some examples:




Package
TC27xC - BGA292 (Top View)
Power domain:
VDDM
VDDP3
VEXT
VFLEX

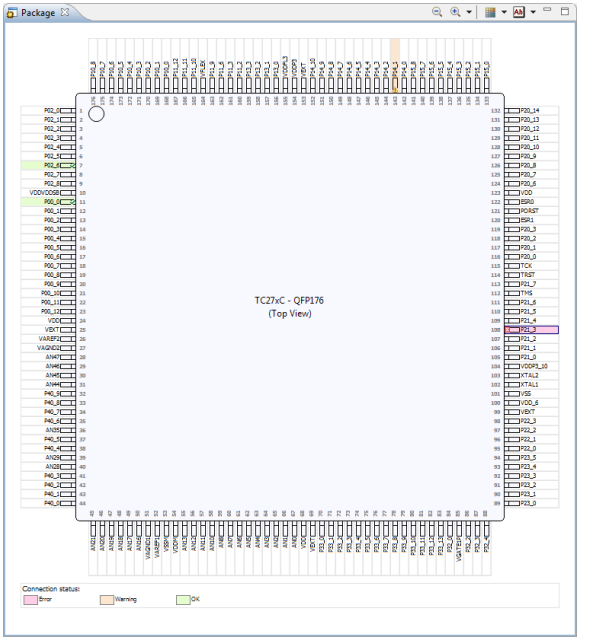

Package
TC27xC - BGA292 (Top View)
Reset state:
PU
PU1
HighZ
OD
PD

Labels

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

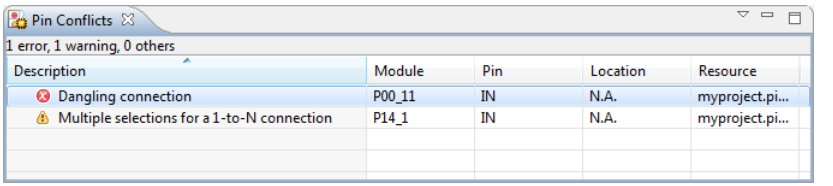
Quad Flat Package

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 2.4, *Solving Pin Conflicts* for information how to solve pin conflicts.

See Section 2.9, *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.
2. In the Pin selection pane, select a port pin. For example, **Ports » P00 » P00_0**.
3. In the Pin configuration pane select the **Direction**: **Input** or **Output**. In this example we select **Input**.
4. Select the pin **Mode**.
5. Connect the input pin to a peripheral virtual pin:
   - From the **IN** drop-down box choose **GPIO** or **Select...**

     With Select... a virtual pin appears, or with a 1-to-N connection (one source, many destinations) a list of virtual pins appears.
   - Select one (or more) pins.

     A green check mark appears if the connection is successful.
6. (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.
7. 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.
2. In the Pin selection pane, select a module. For example, **Peripherals » ETH » ETH_0**.
3. In the Pin configuration pane select the pin you want to assign to the virtual pin.

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. (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.

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

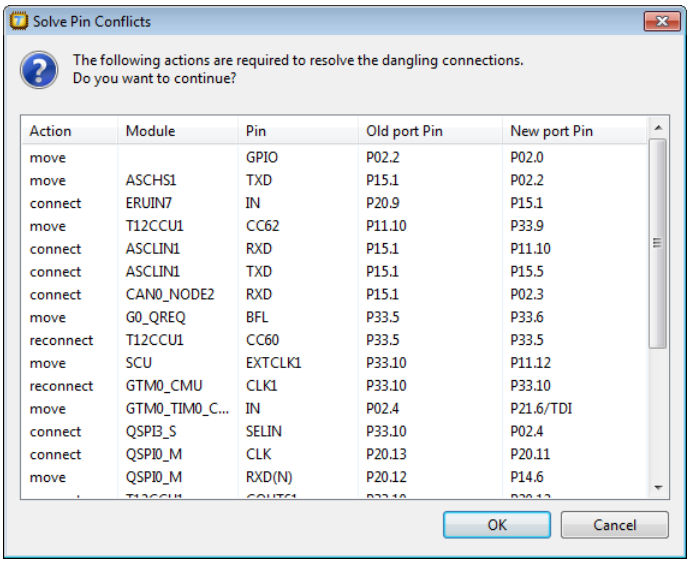
2.4. Solving Pin Conflicts

It is your responsibility to solve any pin conflicts. It is important that there are no conflicts when you use the tools to generate the code. You can solve the conflicts by hand by making other connections, but when there are many conflicts or in situations where most port pins are in use, it can be quite cumbersome and complex to solve conflicts. The Pin Conflicts Solver can automate this process and solve most conflicts for you.

To solve pin conflicts

1. From the **Pin Mapper** menu, select **Solve Pin Conflicts** ( ).

   The Solve Pin Conflicts dialog appears with a list of actions that will be taken when you click OK.



2. Click **OK** to accept the actions.

   The configuration is updated accordingly.

3. Check the pin configuration settings of the new connections and make changes if needed.

The meaning of the columns in the Solve Pin Conflicts dialog is:

Column	Description
Action	The action that takes place to solve the pin conflict. `move` - move the connection of the module pin from the Old port Pin to the New port Pin including all configuration settings. The Old port Pin is reset to the default settings. `connect` - connect the module pin from the Old port Pin to the New port Pin using the default configuration settings. You still need to check and configure the pin settings. `reconnect` - after a move of the original connection to another pin, reconnect to the same port pin. You still need to check and configure the pin settings. `reset` - reset the port pin to the default settings.
Module	The name of the selected module.
Pin	The name of the (virtual) pin of the module.
Old port Pin	The name of the port pin the module pin was connected to.
New port Pin	The name of the port pin the module pin will be connected to.

Note that the solver only tries to solve dangling connections between virtual module pins and port pins, not between virtual pins (internal connections). Also, the solver does not change peripheral instances. If, for example, a connection error with peripheral ASCLIN0 exists, the solver will not try to use ASCLIN1 instead. Try to use a different instance of the same peripheral type manually. If the solver does not find a solution for all pin conflicts, then no port pins will be swapped at all. You can try to move some connections manually or unlock some pin settings (if applicable) or, as a final resort, try to switch to another device/package.

Depending on the projected calculation time, the solver either performs an exhaustive search or applies a heuristic partial search algorithm to find a solution. In case of a partial search it is possible that a solution does exist while the solver does not find it. The solver error dialog distinguishes these situations.

Locked connections are not touched. If you have locked a port pin, its configuration stays the same. The standard undo/redo functionality is available. The menu entry and button are disabled (grayed) if there are no Pin Conflicts to resolve.

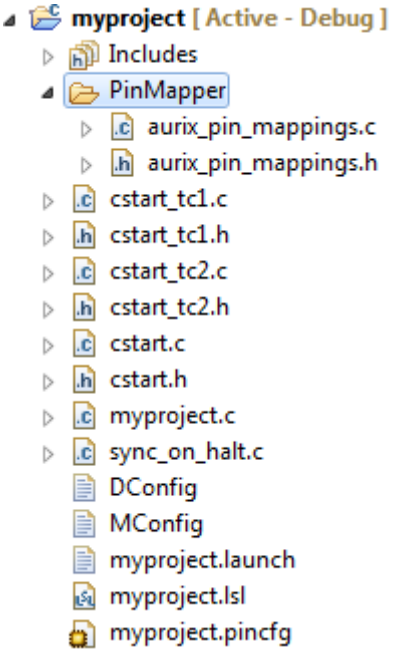
2.5. Generating and Using the Source Code

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 can generate pin mappings for peripherals, internal MUX initialization code and an initialization function for GPIO pins, depending on the configured module connections. For GPIO the following line is present in `aurix_pin_mappings.c`.

```
#include <Port/Io/IfxPort_Io.h>
```

For internal MUX initialization the Pin Mapper generates:

```
#include <_Lib/InternalMux/Ifx_InternalMux.h>
```

The generated code is meant to work closely together with the Infineon iLLD library. This library must be available to your project.

You can easily add the Infineon iLLD library to your project by adding a TASKING Software Platform document and generate the code.

The Software Platform sources are generated in the `SoftwarePlatform` folder of your project. To use the generated files, you must add a `#include` statement into your top-level source file (typically `main.c`):

```
#include "swplatform.h"
```

You can 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/aurix_pin_mappings.h"
```

You can now use the functions of the Pin Mapper in your sources. For initialization of the GPIO pins or internal MUX settings, use the initialization functions from the Pin Mapper sources. For initialization of the peripherals and other pins use the initialization functions from the iLLD library. For example,

```
#include "PinMapper/aurix_pin_mappings.h"
#include "swplatform.h"

IfxEth *drv;

int main( void )
{
    /* initialize GPIO */
    gpio_init_pins();

    /* initialize internal MUX settings */
    internal_mux_init();

    /* initialize ETH0 */
    drv = IfxEth_Phy_Pef7071_ActInit(IFXETH_PHY_PEF7071_1);
    ...

    return 0;
}
```

2.6. 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","Direction","Mode","Pad level","Pad class","Power domain","Reset state"
"P00_0","G1","GTM0_TIM2_CH0_IN","Input","Pull-up","CMOS/Automotive","MP","VEXT","PU1"
"P10_5","B5","SCU_HWCFG4","Input","Pull-up","CMOS/Automotive","LP","VEXT","PU1"
```

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

2.7. Launch Altium Designer

When you have Altium Designer installed with the TASKING Pin Mapper software extension, you can start Altium Designer from the TASKING Pin Mapper Editor view.

- From the **Pin Mapper** menu, select **Launch Altium Designer** ().

  Altium Designer launches and a wizard guides you to import the pin configurations. This results in a schematic document with the selected device, peripherals and pins.

2.8. 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**.

2.9. 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.

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, select **Ports » P00 » P00_0** and change the **Chip input** configuration **IN** from **None** to **Select**. Several destinations appear, all unselected. The error appears to notify you to make a selection. Once you made a selection, the error disappears.

- 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 » P00 » P00_0** and change the **Direction** configuration from **Input** to **Output**
  2. Change the configuration **OUT** from **None** to **GPIO**.
  3. Select **Peripherals » GTM » TOUT9** and change the output **OUT** from **None** to **P00_0.OUT**.

     The error appears since TOUT9.OUT is now connected to P00_0.OUT, while P00_0.OUT 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 » P21 » P21_3** and change the **Direction** configuration from **Input** to **Output**.
2. Change the configuration **OUT** from **None** to **ETH0.MDO3**.

The error will disappear when you follow the connection and also connect input MDI from ETH0 to P21_3.IN.

Error: Modules *name1* and *name2* cannot be used at the same time, since they are mutually exclusive

This error appears when you try two use two modules of which only one module can be used at a time.

For example,

1. Select **Peripherals » ASCLIN0 » SPI3W0** and change the input **RXD(N)** from **None** to **P14_1.IN**.

2. Select **Peripherals » ASCLIN0 » SPI4W0** and change the output **TXD** from **None** to **P14_1.OUT**.

Modules SPI4W0 and SPI3W0 cannot be used at the same time.

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. The warning disappears if you only select one destination.