UM0169
User manual

Configuration and programming software (CAPS) tool for µPSD families

Introduction

Configuration and Programming Software (CAPS) is the configuration software for the µPSD family microcontroller. The CAPS configuration tool allows you to easily configure the µPSD using simple drag-and-drop and point-and-click operations. CAPS also supports In-System-Programming through an external JTAG adapter, allowing fast In-System-Programming of µPSD in development or even in production.

This is the user manual for CAPS. The purpose of this manual is to explain CAPS software functionality. When working with the CAPS tool, you are also encouraged to download the datasheet associated with your particular device; the datasheet may provide the only source of important configuration information needed for your design.

Getting started gives an introduction to device configuration and programming, and the CAPS features. This section also describes CAPS installation procedures. Although installation may seem trivial, it is highly recommended that you carefully follow the instructions because many problems are often caused by incorrectly installing CAPS.

This is followed by the *Introduction to CAPS* section, which introduces the CAPS *Design Flow* and menu environment. An example of a simple project is included as a way to quickly familiarize you with basic concepts and the user interface.

The *Using the wizard* and *Working in advanced mode* sections give more detailed information about how to use CAPS features. These sections describe the user interface for common design functions, for the various CAPS modes of operation.

The appendices provide reference material useful for design and analysis.

1 Getting started

Before using CAPS, you must install the software on your PC and, also, connect your target device, if you plan to program the device.

This section discusses the following topics needed to begin using the CAPS software.

- CAPS software installation.
- Setting up the target hardware.
- The recommended approach to using this manual.
- Complementary documentation considered to be useful when using the CAPS software.

1.1 Installation

This section describes the requirements and procedures needed to install the CAPS software.

1.1.1 System requirements

The CAPS PC configuration minimally requires:

- PC with an Intel Pentium processor running a 32-bit Microsoft Operating System:
  - Microsoft Windows XP
  - Windows 2000
  - Windows 98
  - Windows ME
  - Windows NT with Service Pack 6
- 32 MB RAM
- 25 MB hard disk space available

Note: To use RLINK-ST, a USB port is required with a USB-supporting Windows Operating System (Win98SE, Win2000, Me, XP). Note that Win95, Win98 First Edition and NT4.0 do NOT support USB.

Caution: FlashLINK Cable: JTD driver (OD) is NOT supported on dual-processor systems or hyper-threading enabled systems. Refer to *Appendix C: FlashLINK Cable – Install fast JTAG driver*, for workaround options for both dual-processor and hyperthreading systems.

1.1.2 Installing CAPS

Follow these procedures and the on-screen instructions to install CAPS.

1. Download the compressed CAPS software from the website.
2. Extract the contents of the *.zip* file into a temporary directory.
3. Double-click the extracted executable, *setup.exe*, to initiate the installation, and follow the on-screen prompts to install CAPS in the development environment. This executable installs all the necessary files and configures the PC environment for running CAPS. You may be prompted to restart your PC before running CAPS for the first time following the installation.

CAPS installation includes a number of utilities. Documentation for the utilities is located in the subdirectory *\Docs* where CAPS is installed.

Table 1. CAPS utility programs

Utility Executable	Description
ObjFileEditor.exe	Programming data file (.OBJ) editor.
uFLink.exe	Standalone JTAG/ISP programming utility.
uMerge.exe	Merge firmware utility.
uObjOsf.exe	Program data file conversion utility. Convert obj-to-osf and osf-to-obj files.

Subdirectory *Projects* is also created, and is the default location for storing your CAPS project files. For example, if CAPS is installed in base directory *C:\CAPS*, the *Projects* directory is located at *C:\CAPS\Projects*.

1.1.3 Uninstalling CAPS

To uninstall CAPS, select **Start | Programs | STMicroelectronics - CAPS | Uninstall CAPS**. This removes all CAPS executable software and desktop references.

Note: Any project files and environment files are preserved so they are available following a CAPS software upgrade. However, it is a safe practice to backup your project files before uninstalling and reinstalling the software.

1.2 Setting up the target hardware

If you are using CAPS features that interact with the target hardware, such as programming the flash, configure and power the target hardware before starting a CAPS session. (Refer to the the User Guide for your particular target hardware, found at http://www.st.com/mcu/).

Note: 1 The device under test must be the same as the target device you select when you create your project.

2 Follow the instructions below referring to the device-specific quick start guide or design guide as needed.

1. Connect either the RLINK-ST USB cable or the FlashLINK parallel cable to your PC, and connect the other end of the JTAG interface to the target board.
2. Configure jumpers according to the quick start or design guide documentation for the target board.
3. Attach the power plug to the power jack of the target board.
4. Switch ON the target board.

1.3 How to use this manual

Use these recommendations as a guide to learning the CAPS software.

- Read *Getting started* to learn what CAPS is and to install the software for the first time.
- Read *Introduction to CAPS* to get a basic understanding of CAPS procedures and an understanding of the user interface fundamentals. The section provides a quickstart for understanding CAPS. More advanced users may skip this section.

- Read *Using the wizard* to learn how to design with CAPS, using the wizard assistant. This section provides a reference for the wizard mode of operation.
- Users with more complex design requirements beyond the scope of the wizard templates should read *Working in advanced mode*. This section provides a reference for the advanced mode of operation.

1.4 Recommended reading

You are also encouraged to download the datasheet associated with a particular device. The datasheet may be the only source of important configuration information needed for your design.

2 Introduction to CAPS

This section introduces you to CAPS, presenting the following topics:

- Project development steps and considerations.
- A reference for the CAPS user interface.
- Beginning steps needed to start a project.
- An example of a simple project, using the wizard mode of operation.

2.1 Project development

The CAPS software guides the designer through the process of configuring a target device for a particular application, using the following project development steps.

1. Create a unique project for each device under test/application combination.
2. Automatically generate the Hardware Description Language (HDL) for the PLD logic, given address decoding, paging, segmentation, chip selects, general-purpose logic and I/O pin assignments.
3. “Fit” the design to the silicon architecture, merging the design with your Intel hex-format firmware file.
4. Program the resulting programming data file into the device under test.
5. Save your project to a file for later use.

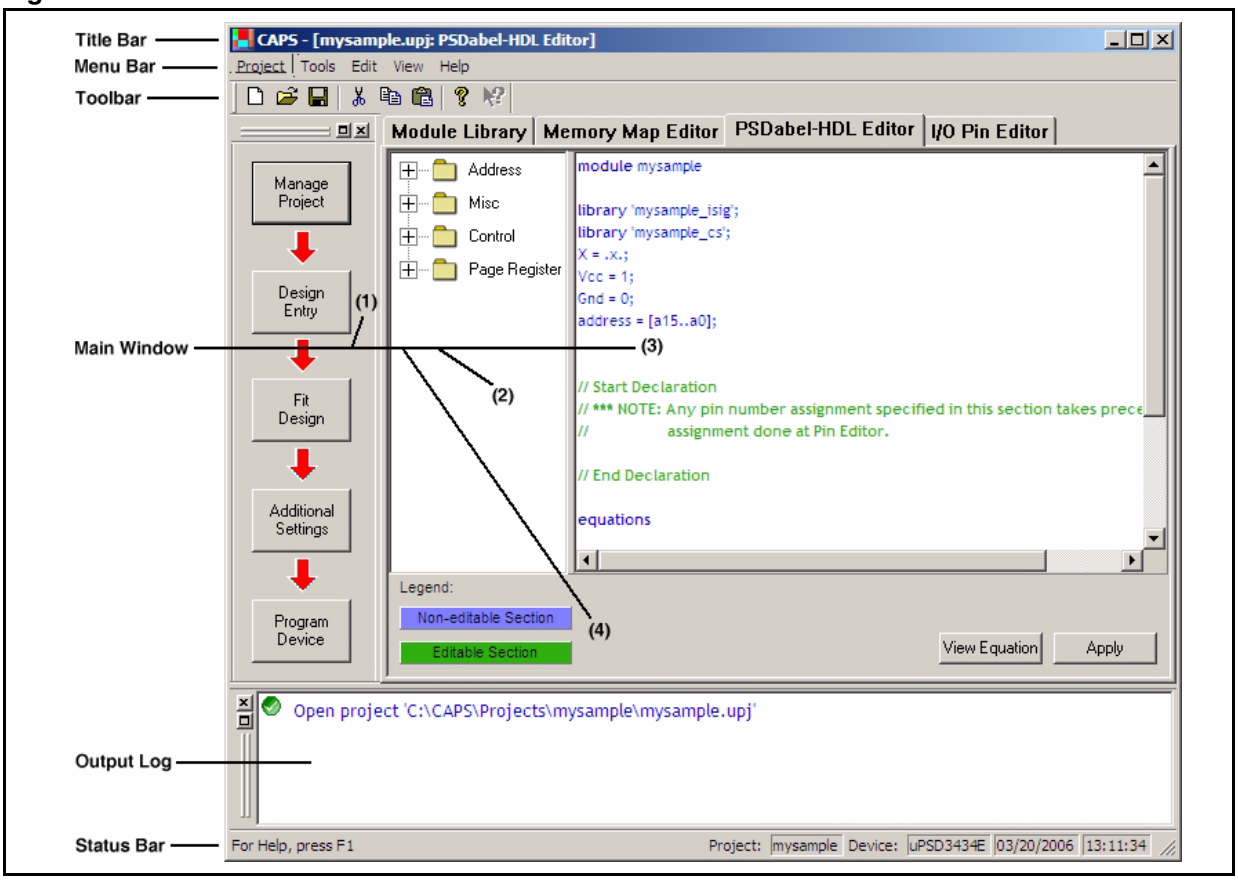
2.2 The user interface

This section is a reference for the CAPS user interface. To learn how to begin using the interface to work with a project, see *Starting a project*.

Figure 1: CAPS user interface shows the basic CAPS user interface components, followed by a brief description of each component. Following subsections describe the menu bar selections in more detail. Components and functionality specific to the different modes of operation are described in *Using the wizard* and *Working in advanced mode*.

The CAPS software is a Windows-based program. As such, the user interface implements basic interface functionality commonly found in Windows programs.

Figure 1. CAPS user interface



Menu bar

Use the menu bar to access these CAPS design functions: *The project menu*, *The tools menu*, *The edit menu*, *The view menu* and *The help menu*. (These functions are described in more detail beginning with *2.2.1: The project menu*.

Toolbar

The toolbar provides quick access to common menu bar functions, including:

- Create a new project.
- Open an existing project.
- Save a project.
- File editing functions: cut, copy and paste.
- Help using CAPS.

Main window

The main window displays the CAPS design entry forms. The window may consist of multiple panes, depending on the current design function.

Figure 1: CAPS user interface is an example of a window with four panes, from the advanced mode of operation:

1. A navigation pane to the left.
2. A design function selection pane to the right.
3. A file contents window on the far right.
4. A legend and command window on the bottom.

Some modes of operation may also have function tabs across the top of the window, as shown in the example figure.

Output log window

The output log window echoes all commands executed by CAPS along with informational and progress messages.

Note: This window is made visible by checking the Output Log option in the View menu.

Status bar

The status bar displays:

- Current project name.
- Target device.
- Today's date. (MM/DD/YYYY)
- Current time (HH:MM:SS)

The CAPS user interface also provides the following aids to using the software.

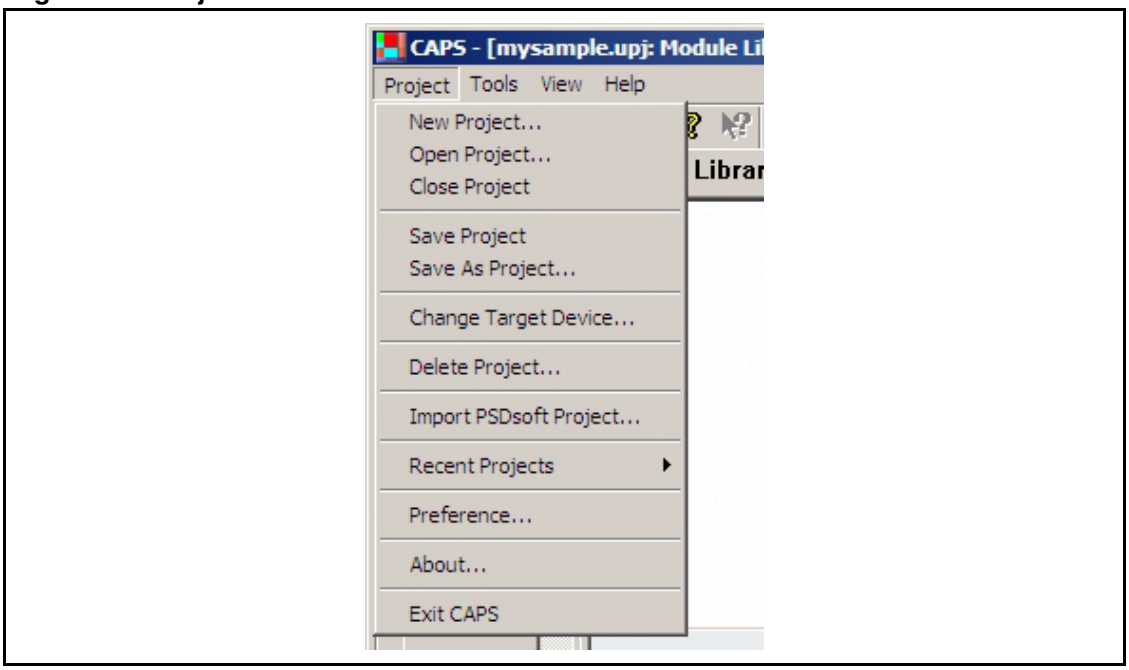
- Descriptive pop-up error messages.
- Explicit design flow sequencing that models the actual design process.
- Detailed directions on most forms that describe input field formats and how to use the form.
- An output log to provide user interface feedback.

2.2.1 The project menu

The project menu allows users to manage the project life cycle, set project preferences and exit the CAPS program.

This section describes the operations available in the project menu.

Figure 2. Project menu



New Project ... Creates a new project.

Open Project ... Opens an existing project.

Close Project Closes the currently active project. Other projects previously opened within the same CAPS session remain open.

Save Project Saves the currently active project.

Save as Project ... Saves the currently active project to a different file name.

1. Enter the new project name (see *2.3.1: Creating a new project* for project name constraints).
2. Optionally, enter or modify the project description
3. Click the **Save** button.

Change Target Device ... Selects another target device ONLY within the same product family, from the expandable device tree list.

Changing the package type invalidates the pin re-assignment done in the Pin Editor window and also requires a refitting of the design. (Note that the target device may only be changed when working in advanced mode).

Delete Project ... Deletes a project and all files associated with the project.

Import PSDsoft Project ... Imports an existing project created using the PSDsoft tool.

Recent Projects Lists the most recently opened projects. A project may be opened by double-clicking on a list entry.

Preference ... Sets the CAPS design environment. Select either “Single device view” or “Multiple Device view” for the JTAG/ISP property. Preferences may only be changed while in advanced mode.

About Displays details about a project:

- Project name.
- Project folder.
- Device family.
- Part number.
- Voltage.
- Project description.

Note: The project must already be open in either wizard or advanced mode.

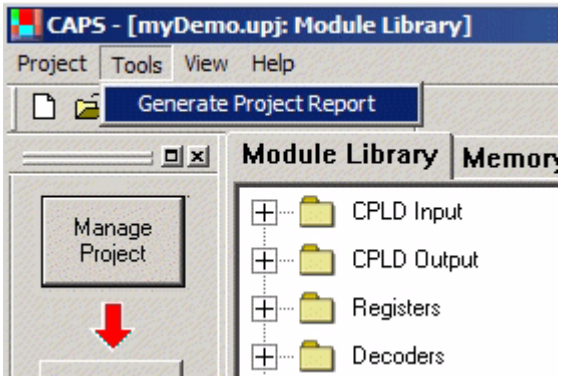
Exit CAPS Exit the CAPS program.

2.2.2 The tools menu

The tools menu provides access to useful utilities for working with projects.

This section describes the utilities available in the tools menu.

Figure 3. Tools menu



Generate Project Report	Produces a text file report of your design, describing the project and showing the fitting results. Once the fitting process is complete and without errors, a summary of the fitting result is displayed as a text file in a separate window. The result shows detailed resource utilization, I/O pin and configuration information, and address and data bus assignments. See *Project Report* for an example.

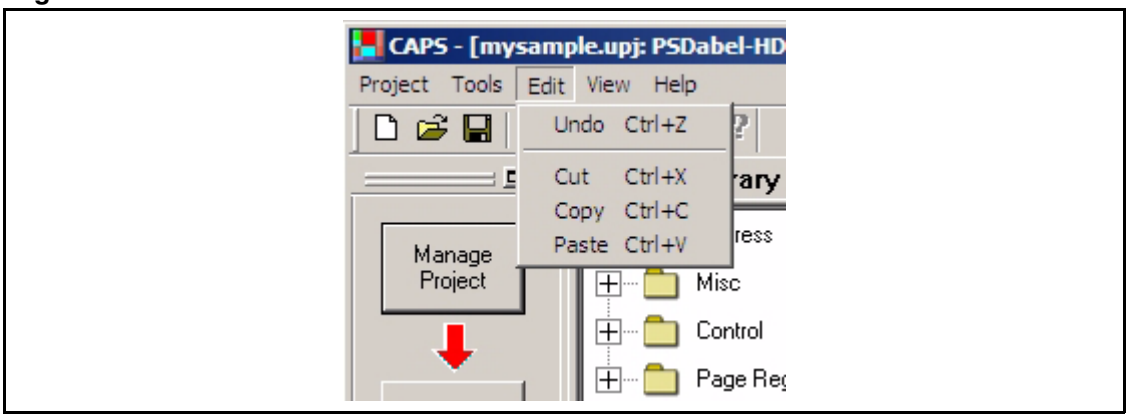
2.2.3 The edit menu

The edit menu provides basic editing functions. These are the standard Windows *cut*, *copy* and *paste* editing functions.

Note: The edit menu item is context sensitive, and is only available when the PSDabel-HDL Editor feature is selected (4.6: Editing PSDabel-HDL code).

This section describes the operations available in the edit menu.

Figure 4. Edit menu



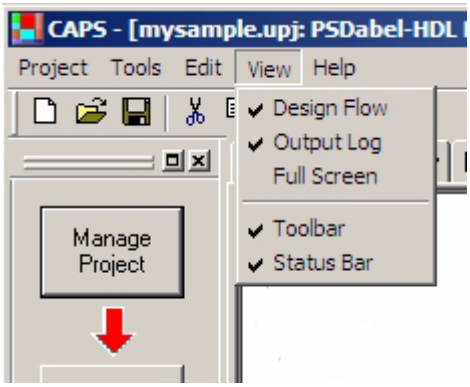
Undo	Reverses the last editing command.
Cut	Removes the selected text, saving it in the clipboard.
Copy	Copies the selected text to the clipboard.
Paste	Copies the last text buffer in the clipboard to the current cursor location.

2.2.4 The view menu

The view menu selection allows the user to specify the user interface panes to be displayed while working on a project. Check the menu item to enable the display.

This section describes the operations available in the view menu.

Figure 5. View menu



Design Flow — Displays the CAPS design flow pane when working in advanced mode.

Note: This option is unavailable in wizard mode.

Output Log — Display the output log pane that shows commands executed by CAPS, informational and status messages (*Figure 1: CAPS user interface*).

Full Screen — Maximizes the CAPS display interface on the screen.

Toolbar — Displays the toolbar (*Figure 1: CAPS user interface*).

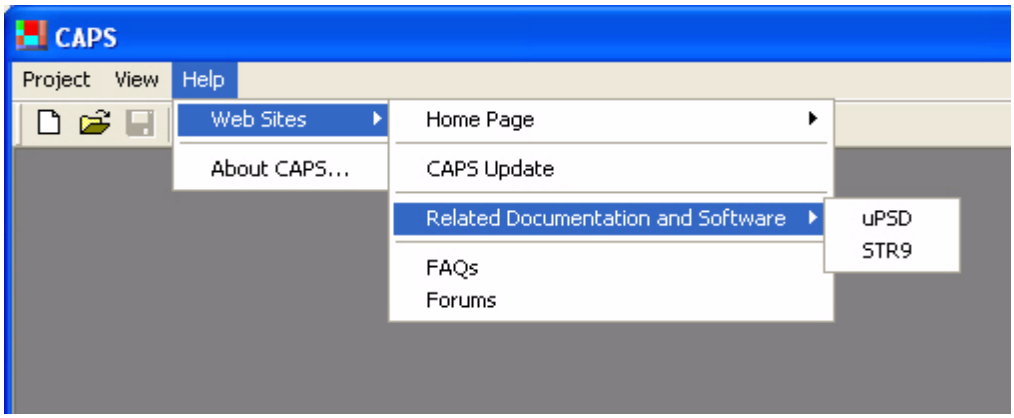
Status Bar — Displays the status bar (*Figure 1: CAPS user interface*).

2.2.5 The help menu

The help menu provides access to various links to obtain technical information about STMicroelectronics Microcontroller products and to report questions or issues related to CAPS.

This section describes the operations available in the help menu.

Figure 6. Help menu



Web Sites Displays links to obtain technical information about STMicroelectronics Microcontroller products and to report questions or issues regarding CAPS. A link to Frequently Asked Questions (FAQs) is also provided. (See *Figure 6.: Help menu* for a list of the links provided).

About CAPS ... Displays CAPS software version, copyright, contact and licensing information.

2.3 Starting a project

Every CAPS session begins by opening a project. You may either create a new project or open an existing project. Once the project is opened and the basic project preferences are selected, you may choose to use either wizard mode or advanced mode to configure your project.

This section describes the beginning steps common to all projects.

2.3.1 Creating a new project

Follow these steps to create a new project.

1. Use either the **New** icon in the toolbar or the **Project** | **New Project ...** menu item (*Figure 2.: Project menu*) to create a new project.
2. In the *Create Project* dialog, use the **Browse** button to specify the name of your project and directory where the project files are to be created. (The default project name is *new.upj* and the default directory is *c:\CAPS\Projects\new*; note that the default project name is the same as the directory name).
3. Enter an optional project description in the *Description* text box.
4. To accept the default project name and directory, click **Open** in the file browser dialog. To choose another directory, use the file browser to navigate to the desired directory. Then, either use the default *Project Name* or enter a new name for your project. After a new directory and project name are entered, click **Open**.

Note: The project is saved to disk only after the wizard or advanced mode procedures are successfully completed.

A valid project name,

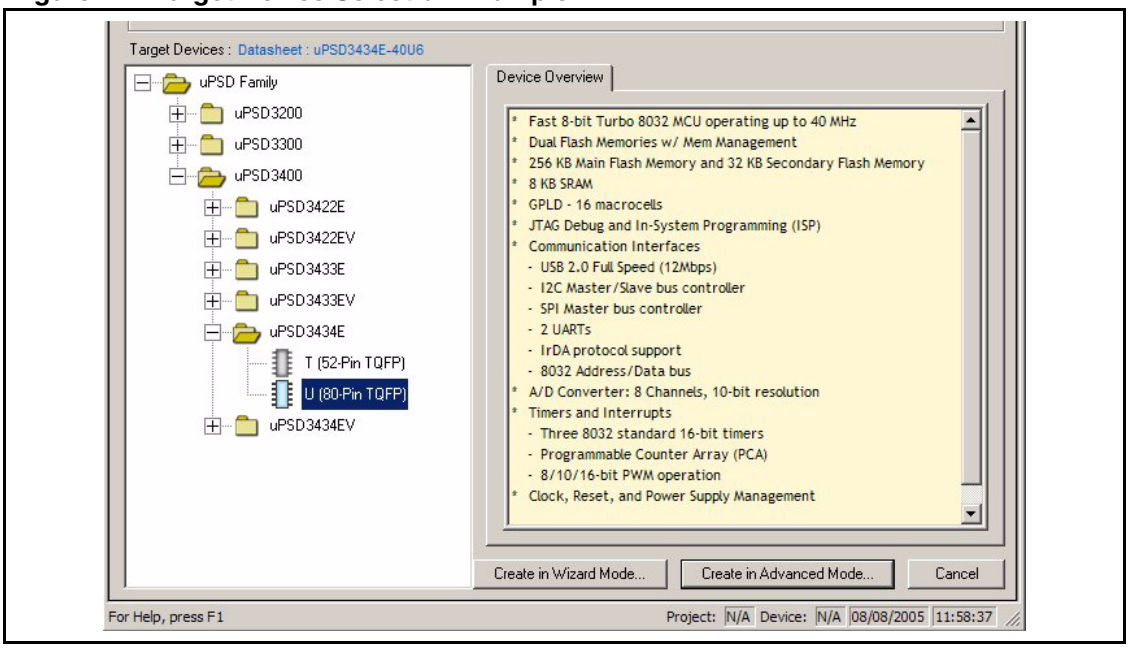
- Can have a maximum number of 45 characters.
- Must begin with an alphanumeric character, an underscore (_) or a tilde (~). Names are not case-sensitive.
- Cannot include symbols or punctuation marks.

5. For new projects, a target device must also be specified. Using the device list tree in the *Target Devices* dialog, expand the tree until your target device appears, then click on the device icon. A description of the device appears in the *Device Overview* window to the right, and the name of the datasheet associated with the device appears in the *Target Devices* label line.

   Available devices include all devices currently supported by CAPS software.

   The following figure shows an example in which the 80-pin uPSD3434E device is selected.

Figure 7. Target Device Selection Example



Continue configuring your project using either wizard mode or advanced mode, according to the guidelines discussed in *2.3.3: Choosing a design method*.

2.3.2 Opening an existing project

Follow these steps to open an existing project.

1. Use either the **Open** icon in the toolbar or the **Project** | **Open Project ...** menu item (*Figure 2.: Project menu*) to open an existing project.
2. In the Open Project dialog, use the **Browse** button to use the file browser to locate the directory where your existing project file resides. A project file name with extension *.upj* is displayed. Click on the file name then click **Open** to open the file. Your previously saved project *Description* and *Target Device* are displayed.

Note: You may edit the project description, however, notice that the target device may not be changed.

Continue configuring your project using either wizard mode or advanced mode, according to the guidelines discussed in *2.3.3: Choosing a design method*.

2.3.3 Choosing a design method

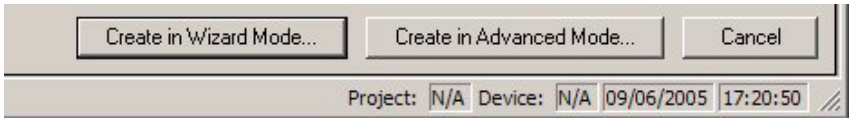
CAPS provides two modes for working with projects: wizard mode and advanced mode. Wizard mode offers a simple step-by-step approach to entering the design. It is intended for new users who are unfamiliar with many CAPS design features. Use advanced mode to access the full range of CAPS features.

One indication that advanced mode should be used rather than wizard mode is that one of the available memory maps does not meet your system requirements, for example.

For a discussion of the design facilities offered using wizard mode, refer to *Using the wizard*. For a discussion of the complete CAPS feature set, refer to *Working in advanced mode*.

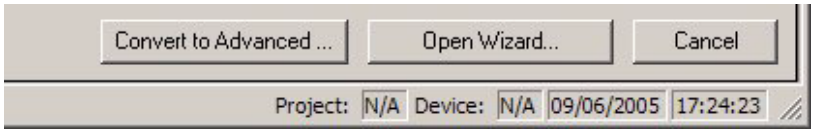
To create a project using the CAPS wizard facilities, click the **Create in Wizard Mode** button (*Figure 8.: Initial design mode option buttons*). Alternatively, click the **Create in Advanced Mode** button to create a project using advanced mode.

Figure 8. Initial design mode option buttons



For an existing wizard mode project, you may choose to continue to work with the project in wizard mode or convert the project to advanced mode by clicking the appropriate button, as shown in *Figure 9.: Wizard design mode option selection buttons*.

Figure 9. Wizard design mode option selection buttons



*Note: A project originally created using the wizard can be converted to advanced mode by opening the project and clicking **Convert to Advanced**. However, once the project is converted, it is not possible to revert to the wizard design mode at a later time. You may wish to first make a copy of your project before switching to advanced mode.*

For an existing advanced mode project, you may only continue to work with the project in advanced mode by clicking the **Open** button, as shown in *Figure 10.: Advanced design mode option selection buttons*.

Figure 10. Advanced design mode option selection buttons

Click **Cancel** in any of the design modes to quit the project.

2.4 An example using the DK3400

The purpose of this example is to demonstrate the basic design steps using CAPS. The example uses the wizard mode of operation to show how easy it is to design a project for the most common types of target device configurations.

We begin by creating a new project, which we name *blink_led*. Because we intend to program the µPSD DK3400 development board, we choose the µPSD3434E, 80-pin TQFP, as the target device.

Note: This demonstration uses an example provided with the RIDE development environment. The same procedures can be used for your in-house project, choosing the target device that matches your hardware.

Next, we use the CAPS wizard to define the memory map and the firmware files to download into flash. For this example, we omit external chip selects, security and sector protection specifications.

The wizard maps the contents of the firmware files to the physical memory, translating absolute addresses to physical addresses used to program the target device. We call this “fitting” the design to the silicon architecture, which the wizard does transparently.

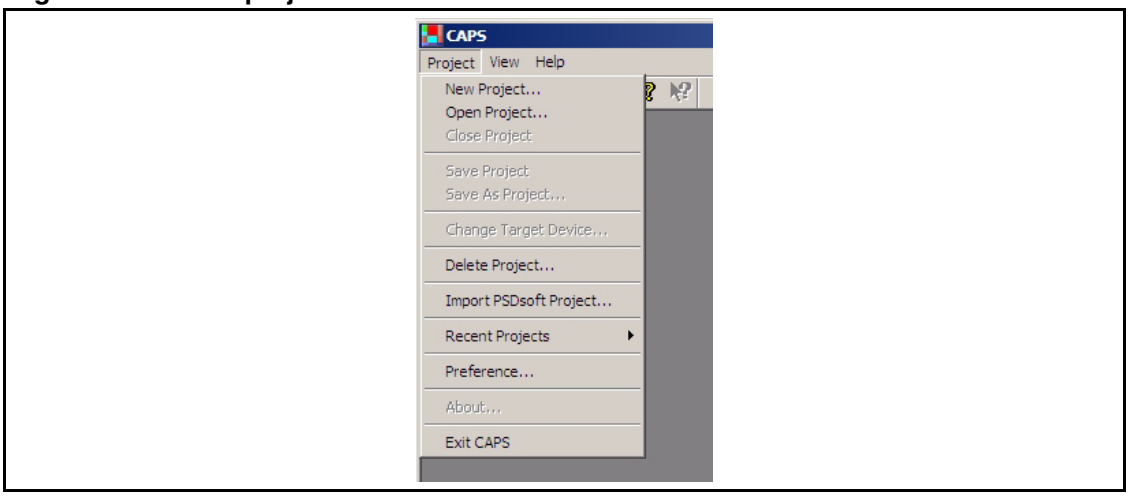
We then verify the communication link and operation of the target development board before programming the flash.

Finally, we close the CAPS session and save our project.

2.4.1 Creating a new project

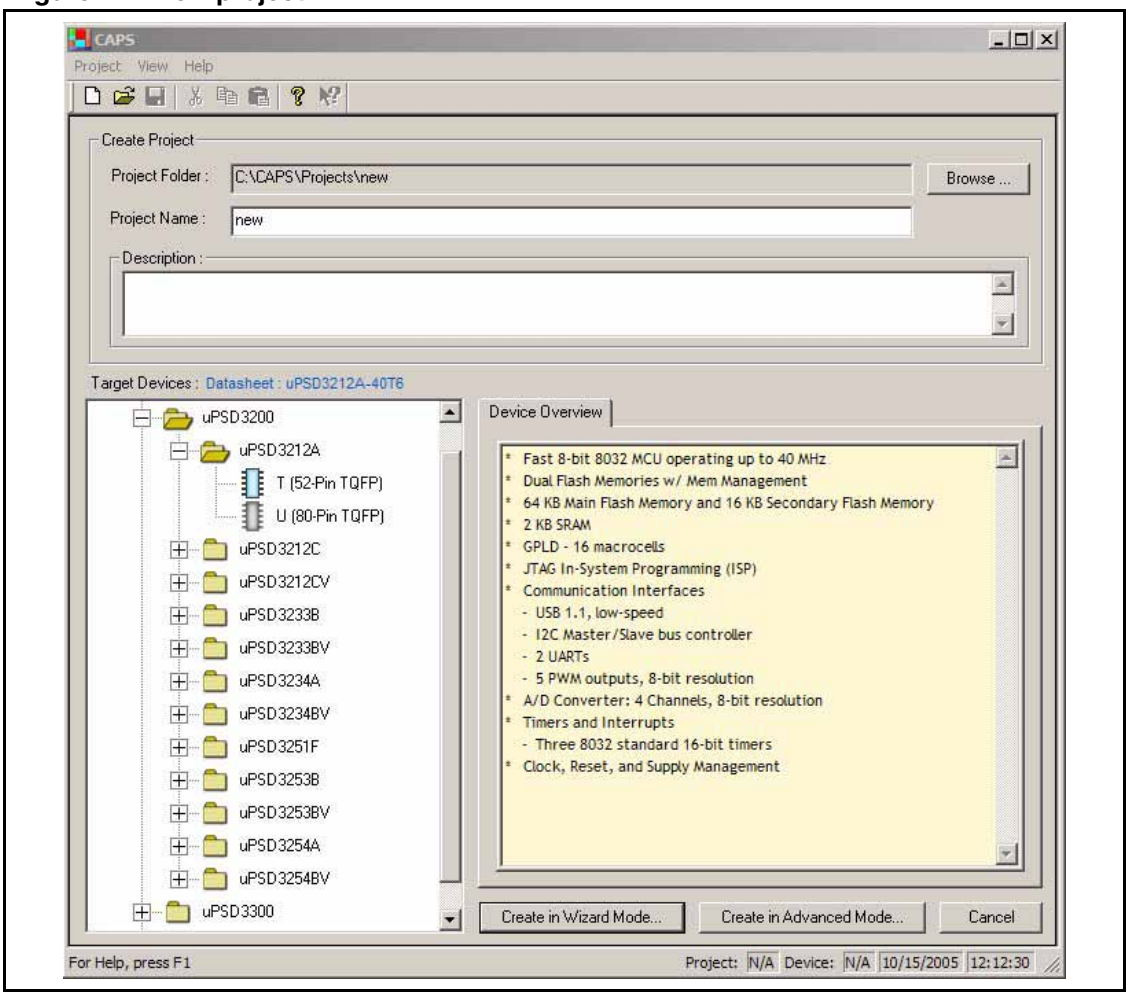
Because this is a new project, we first need to create the project. From the **Project** menu click on **New Project ...** , as shown in *Figure 11*.

Figure 11. Create project



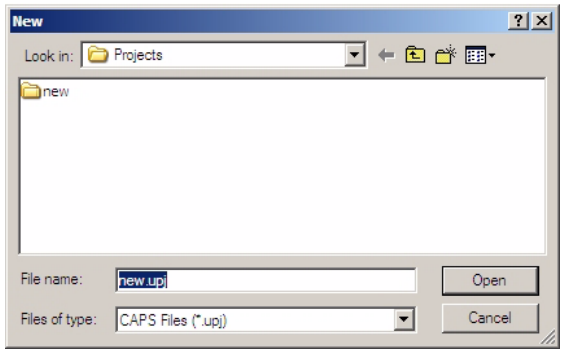
This displays the project dialog, shown in *Figure 12*.

Figure 12. New project



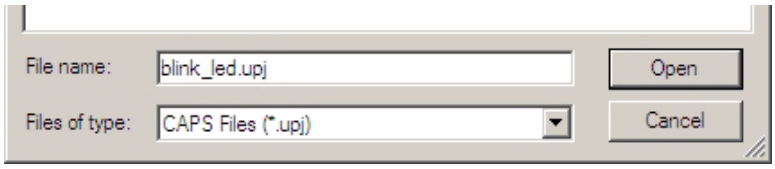
Click on the **Browse ...** button to enter a project name and choose a directory location for the project, using the dialog shown in *Figure 13*.

Figure 13. Project name dialog



You may choose another directory or accept the default project *CAPS**Projects* location. Enter your project name and click **Open** to create the project. We enter the project name *blink_led*.

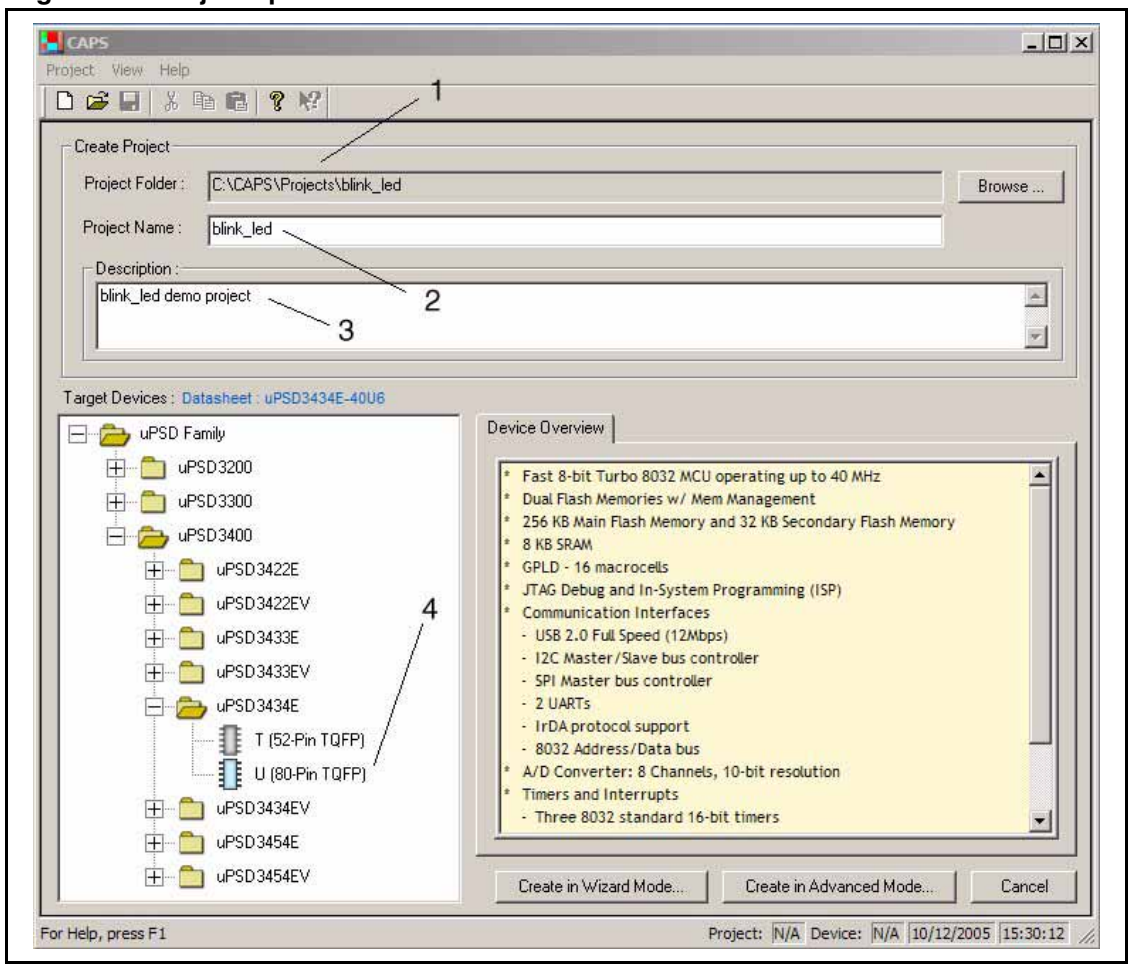
Figure 14. Project blink_led assignment



Click **Open** to open a new project filename.

The dialog displays the default or new *Project Folder* you selected [1] and the default or new *Project Name* [2], as shown in *Figure 15*.

Figure 15. Project specification



You may also enter an optional project *Description* [3].

The next step is to select the target device, using the *Target Devices* window device folder tree. We select the µPSD3434E, 80-pin TQFP [4]. Notice that when you select the target device, the tabbed *Device Overview* window displays device-specific information. (Click on other devices to see the contents of this window change). Select the target device corresponding to your hardware.

This completes the steps used to open a CAPS project. Remember that the project is not saved until you reach the final step of the wizard mode of operation. Click the **Create in Wizard Mode...** button to continue, and configure the firmware to the target device.

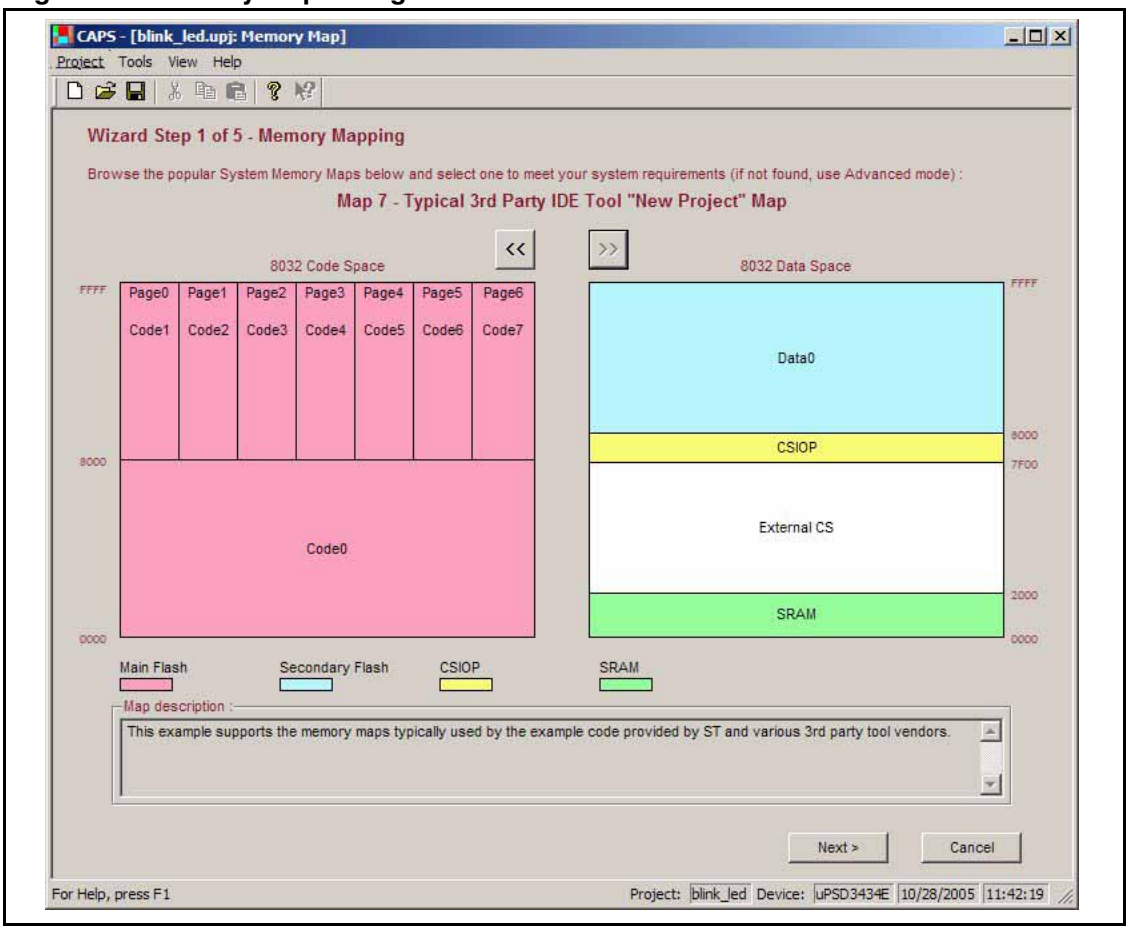
2.4.2 Designing your project

The wizard mode of operation lets you configure your project by choosing from common configuration options.

The first wizard design step, *Figure 16*, displays the memory map configuration dialog.

Note: If you selected an 80-pin device, there are five wizard design steps. If you selected a 52-pin device, there are four wizard design steps. The external chip selects step is omitted for 52-pin devices because of GPIO limitations.

Figure 16. Memory map configuration



Use the double-arrow scroll buttons to traverse the available memory map options. Choose the memory map corresponding to your hardware and firmware.

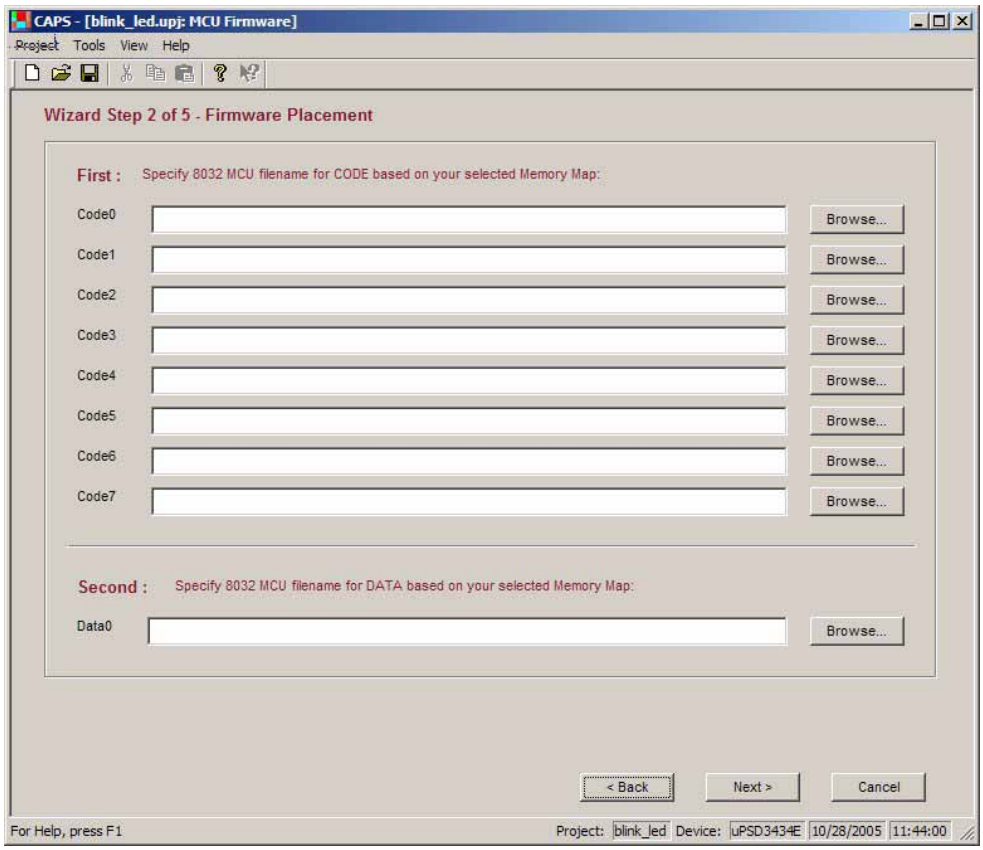
Each memory map option window displays a description of the memory configuration and Code, Data and Paging attributes. The dialog also shows a graphic visualization of the memory layout. (The legend describes the color-coded memory segments and descriptive text is provided in the *Map description* window).

If there are no special memory requirements for code or data space and no chip select requirements, use the default memory configuration: *Large Code Flash, Small Data Flash, No Paging*. However, because this example uses Port D for the LED, we select Map 7 (*Typical 3rd Party IDE Tool "New Project" Map)* for the correct CSIOP settings.

Once you have selected the desired memory configuration for your target application using the memory map navigation buttons, click the **Next >** button at the bottom of the screen to go to the *Firmware Placement* design step.

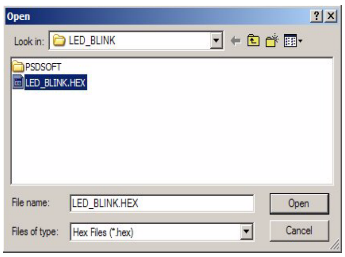
Use the *Firmware Placement* dialog, shown in *Figure 17*, to associate a firmware *.hex* file with your project.

Figure 17. Firmware placement



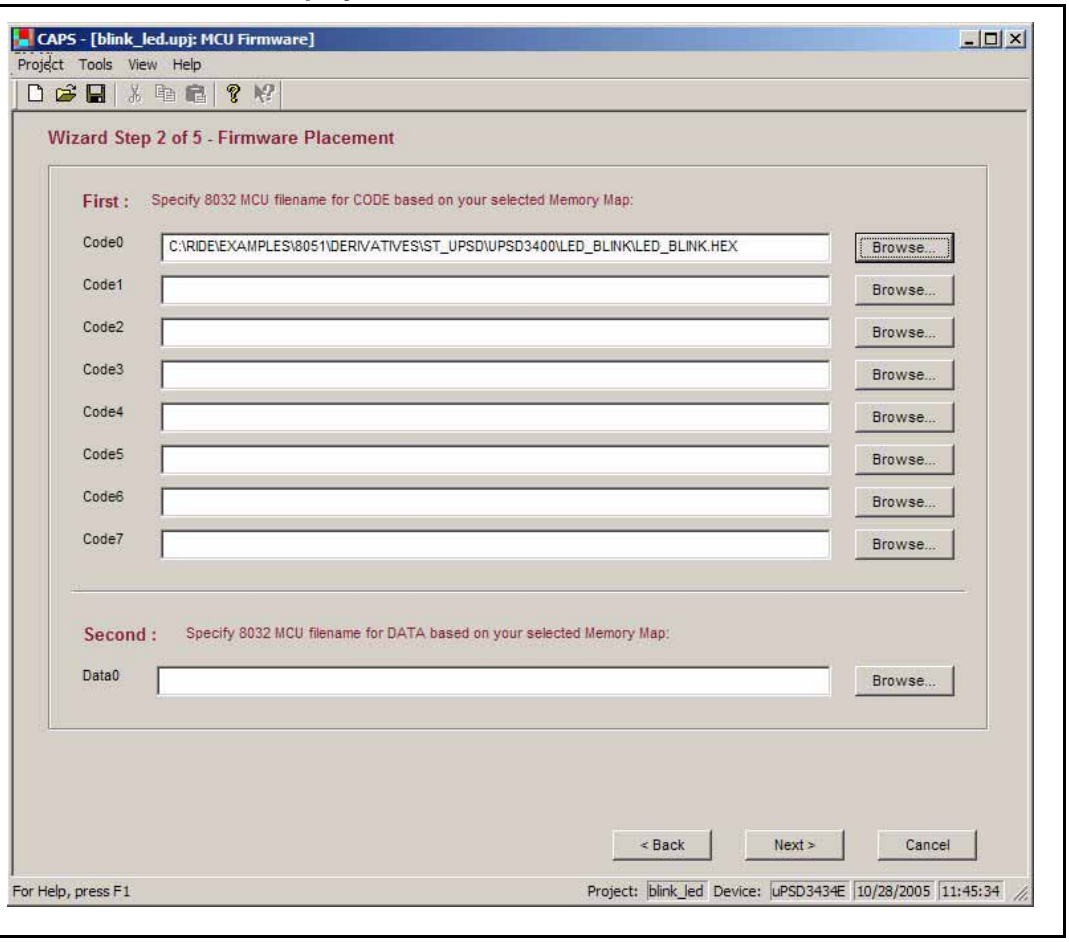
Click the **Browse ...** button to use the file browser to select your firmware file, or enter a fully qualified path and filename. Because this example uses the RIDE *led_blink* example, we select file *LED_BLINK.HEX*, located at *C:\RIDE\EXAMPLES\8051\DERIVATIVES\ST_UPSD\UPSD3400\LED_BLINK*. (*Figure 18*).

Figure 18. Firmware file browser



Click **Open** to select the firmware file, which is then displayed in the *Code0* text box shown in *Figure 19*.

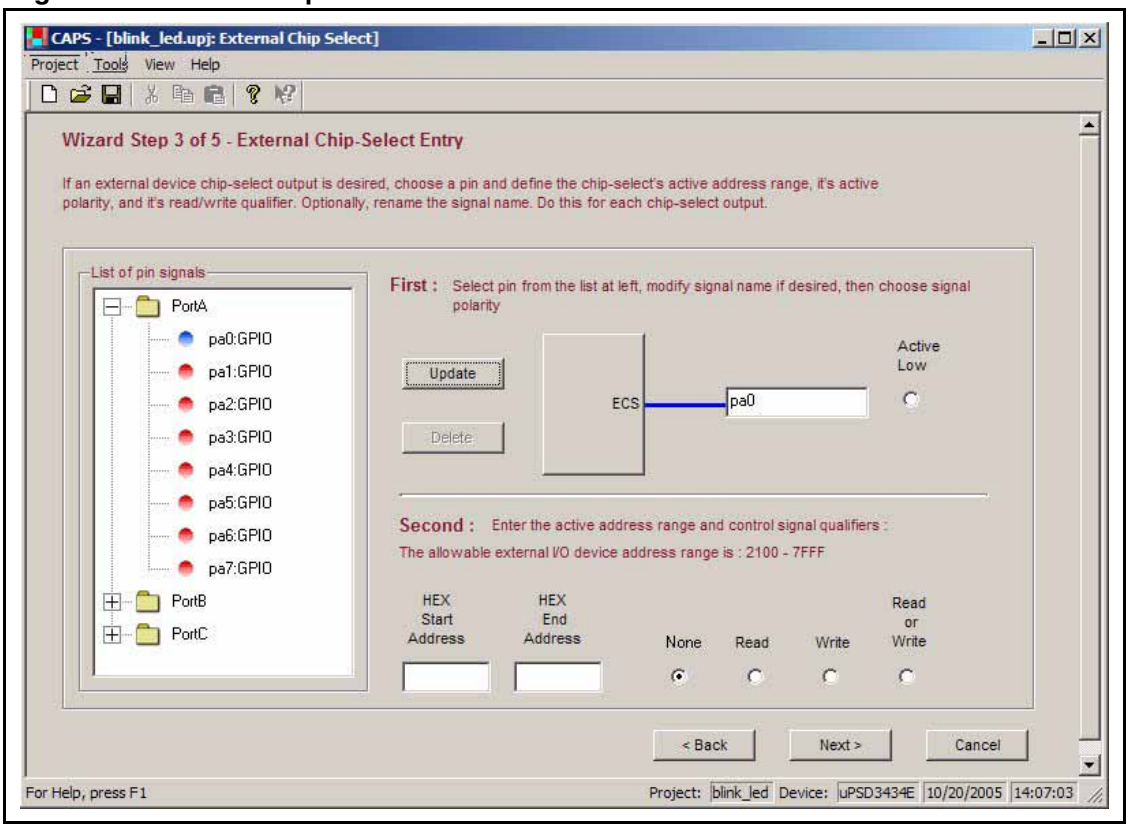
Figure 19. Firmware file display



Click **Next>** to continue to the next step, which will be either the *External Chip-Select Entry*, *Figure 20*, or *Configuration Settings* dialog, *Figure 21*.

Note: This step, 3 of 5, is only applicable to 80-pin devices. If your project uses the 52-pin device, you will not see Figure 20 but will continue with the Configuration Settings step, shown in Figure 21.

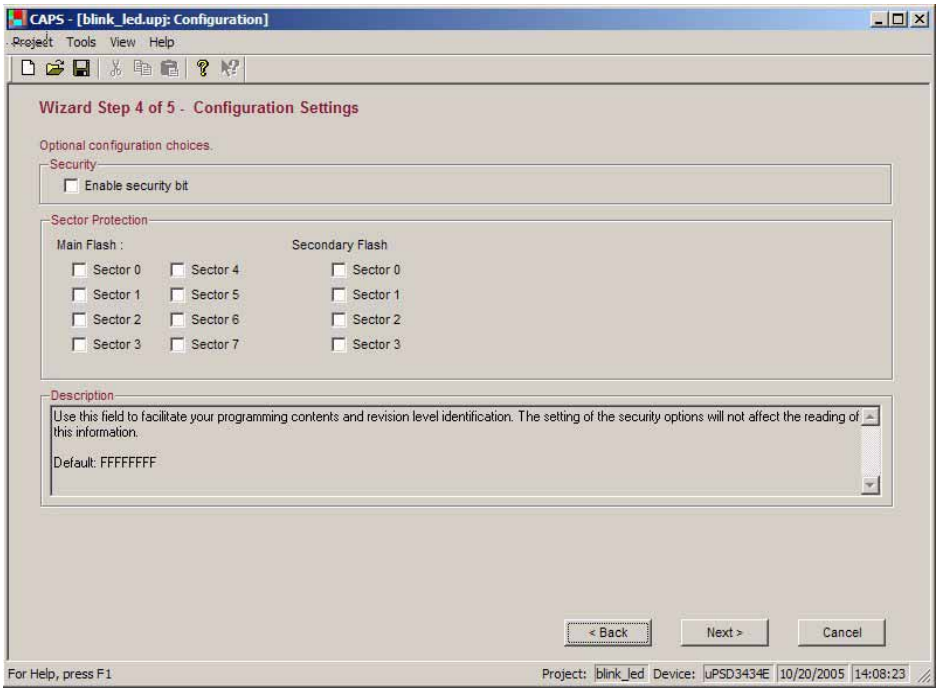
Figure 20. External chip selects



Because our example does not need any special chip-selects declared, we continue to the next step without making any entries on this form.

*Click **Next>** to continue to the Configuration Settings dialog, Figure 21.*

Figure 21. Configuration settings



We accept the default settings for *Security* and *Sector Protection*.

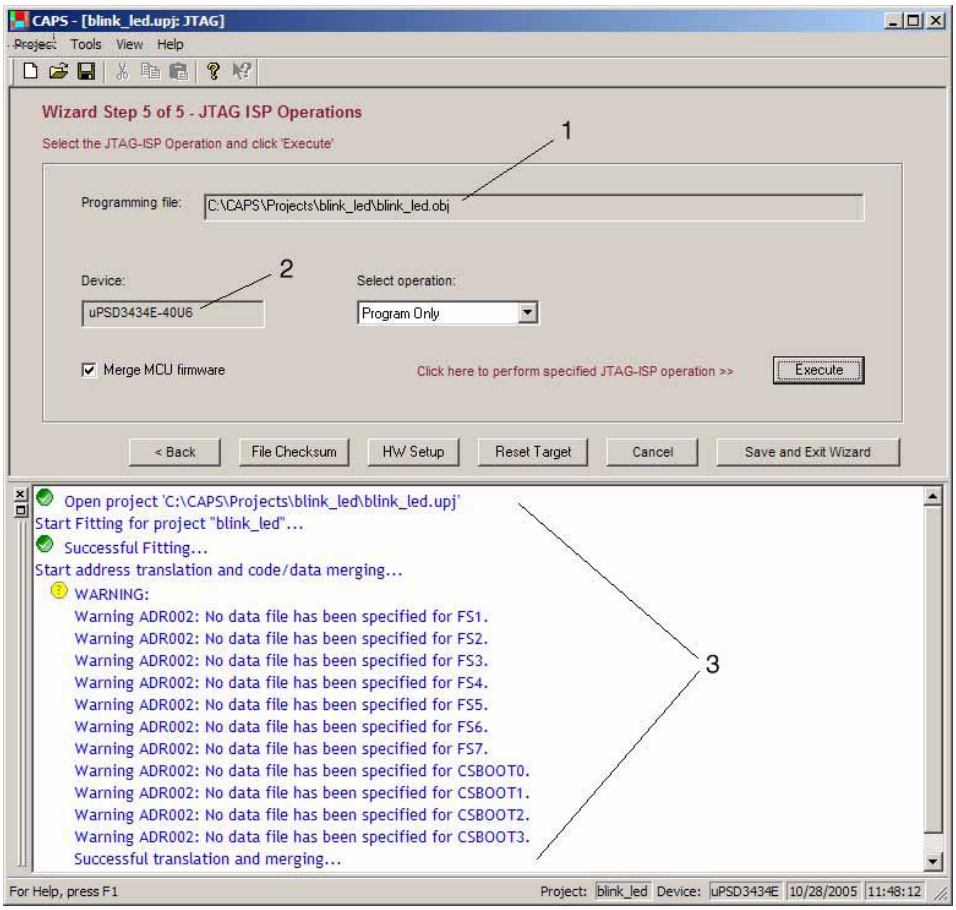
Click **Next >** to continue to the *JTAG ISP Operations* dialog.

At this point, the CAPS software is able to “fit” the design to the silicon architecture. You may notice a small delay while CAPS “fits” the design. This creates a programming data file in your project directory, named *blink_led.obj* in our example.

2.4.3 Programming your target device

Figure 22 shows the device programming dialog.

Figure 22. JTAG ISP operations



The name of the programming data file created in the previous step is displayed in the *Programming file* window [1] and the selected device is displayed in the *Device* window [2]. The *Output Log* displays the fitting results [3], provided logging is enabled using the *View* menu.

Before programming the device, it is necessary to specify the type of connection to the target device and to verify connectivity. Click **HW Setup** to begin the connectivity dialog (*Figure 23*). Because we use RLINK in this example, we accept the default *Hardware Selection* (*RLink*) setting and click the **RLink Connect Test** button to verify connectivity.

Figure 23. Hardware setting

Note: If you are using FlashLINK, use the drop-down menu to select the FlashLINK connection.

After a short delay during which a loopback test is performed, a successful test message is displayed, provided the loopback test passed (*Figure 24*). If the test failed, check the hardware power and cable connections.

Figure 24. RLINK connect test status

Click **OK** to acknowledge the test results and continue.

In the same way, test connectivity to the target board by clicking the **Target Connect Test** button (*Figure 23*).

The *Target connect test passed* message (*Figure 25*) indicates there is no open circuit along the JTAG chain on the target board.

Figure 25. Target connect test status

Click **OK** to acknowledge the test results message.

Click **OK** or **Cancel** to exit the *Hardware Setting* dialog window (*Figure 23*).

Use the *Select operation* drop-down menu, *Figure 22*, to choose the desired JTAG-ISP operation. In our example, we choose the *Program Only* option. As we shall see, a blank test will be done before programming the device.

Click **Execute** to begin the selected operation.

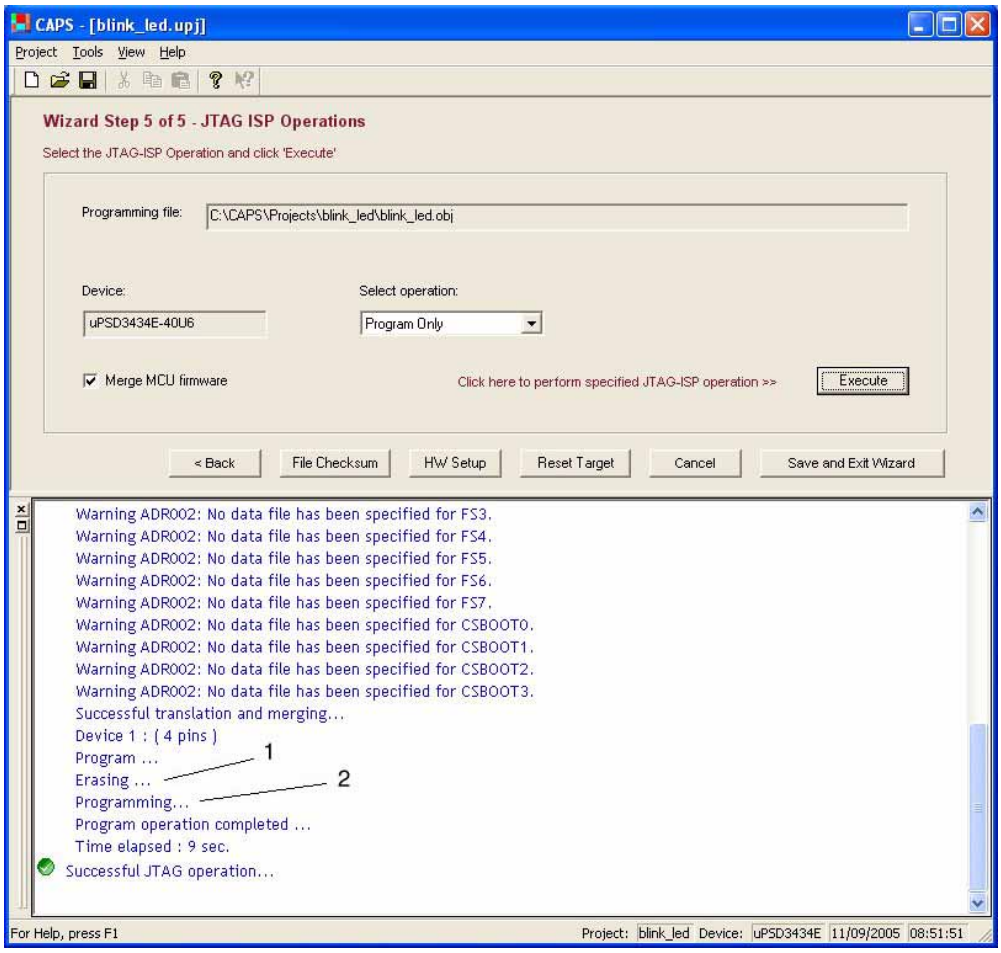
CAPS always verifies that the device is blank before attempting to program the device. If the device is not blank, CAPS prompts to erase the chip (*Figure 26*).

Click **Yes** to erase the chip and to continue with programming flash.

Figure 26. Device blank test

The Output Log shows the programming progress (*Figure 27*). In this example, the log shows the progress of the erasing [1] and programming [2] steps. When device programming is complete, the log displays “Successful JTAG operation.”

Figure 27. Programming output log



2.4.4 Ending your CAPS session

After successfully configuring and programming your device, click **Save and Exit Wizard** to save the project. This ends your CAPS session, saving the project in the directory you specified when you created the project. You may then work on the project at a later time by opening an existing project.

3 Using the wizard

The wizard mode allows you to use a simple step-by-step approach to design, encapsulating some of the most common design and programming configurations. The wizard hides many of the details found in the advanced mode, and is especially helpful to new users with relatively simple design requirements.

From the window used to create or open a project, click the **Open Wizard** button to use the wizard design mode.

The wizard provides five design entry forms:

Wizard Step 1 of 5	Memory Mapping (see *Specifying the memory map*)
Wizard Step 2 of 5	Firmware Placement (see *Firmware placement*)
Wizard Step 3 of 5	External Chip-Select Entry (Due to I/O pin limitations, this feature is not available for the 52-pin package) (see *Specifying external chip selects*)
Wizard Step 4 of 5	Configuration Settings (see *Setting security and sector protection, and fitting the design*)
Wizard Step 5 of 5	JTAG ISP Operations (see *Validating and programming the target device*)

Navigate, sequentially, through each design form using the **Next** and **Back** buttons at the bottom of each form.

Each of the forms contains descriptive text to help guide you through the wizard, and to describe the selection and data entry fields.

*Note: Click the **Cancel** button on a wizard design form to exit the wizard mode and close the project, without saving the data entered during the current session.*

This section provides a detailed description of the project development steps for the wizard mode of operation, including:

- Specifying the memory map.
- Firmware placement.
- Specifying external chip selects.
- Setting security.
- Setting JTAG/ISP parameters.

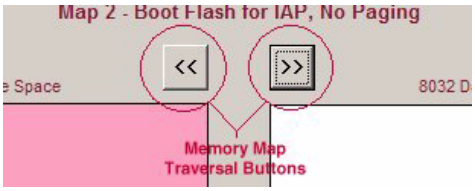
- Setting sector protection.
- Validating and programming a target device.

3.1 Specifying the memory map

Use the memory map design form, *Wizard Step 1 of 5*, to select one of the pre-defined memory maps that meets your system requirements.

Use the memory map traversal buttons to navigate through the available maps (*Figure 28.: Memory Map Traversal Buttons*).

Figure 28. Memory Map Traversal Buttons



Each memory map option graphically shows the memory map, using color-coded sectors, and also provides a *Map description*, describing the typical usage for the particular map.

Addresses specified in subsequent steps, including external chip select definition, will be validated against the memory map selected.

To select the currently displayed memory map, click the **Next** button.

Once the memory map is specified, proceed to the next step to specify the firmware assignment to the particular memory segments.

3.2 Firmware placement

Firmware placement locates your firmware in code- or data-space. Use the firmware placement form, *Wizard Step 2 of 5*, to locate (place) your firmware file(s).

Given the currently selected memory map, click the **Browse** button to use file browser dialog to select your firmware files, Intel hex files (*Appendix A: Intel hex-32 record format*), associated with the memory map CODE and DATA spaces. You are presented with an entry for each memory segment of your memory map.

Click the **Next** button to accept your firmware placement, and to continue with external chip select entry.

3.3 Specifying external chip selects

Use the external chip selects form, *Wizard Step 3 of 5*, to define the external chip select address range, active polarity and read/write qualifier, if an external device chip-select output is desired.

Note: This step is skipped on 52-pin packages, due to limited PLD I/O pins,

Port A, port B and port C I/O pins are listed in the *List of pin signals* expandable folders. Use the following sequence for *each* external chip select in your design.

1. Expand the appropriate folder and click the desired port I/O pin to select it.
2. In the text edit box to the right, do nothing to accept the default signal name or edit the signal name according to match your design.
3. If the signal is active low, click the *Active Low* radio button.
4. Enter the active start and end address for the signal, in hexadecimal notation (0000 through FFFF). The address range must be with the range specified for *External CS* in the memory map, which is shown in descriptive text on this form.
5. Click the radio button corresponding to the desired signal qualifier: *None*, *Read*, *Write*, *Read/Write*.
6. Click the **Update** button to add the signal to your design. The CAPS program validates the form data and the new signal name is appended to the signal in the *List of pin signals*.

To remove an external chip select definition,

1. Click the port I/O pin in the *List of pin signals* to select it.
2. Click the **Delete** button.

When your external chip select definitions are complete, click **Next** to accept the definitions and continue to specify various configuration settings.

3.4 Setting security and sector protection, and fitting the design

Use the configuration settings form, *Wizard Step 4 of 5*, to enable or disable the security bit and to set sector protection.

Setting the security bit blocks all access to the contents of the device from a JTAG or conventional programming tool.

Click the *Enable security bit* checkbox to enable the security bit. Uncheck the box to disable the security bit.

Click the checkbox for each of the NVM segments in main and secondary flash that you want protected.

Click **Next** to accept and fit the design, as specified in wizard steps one through four. This causes CAPS to regenerate a new programming data file for the design.

3.5 Validating and programming the target device

Following the previous steps, you are now ready to program the target device. Use the JTAG/ ISP operations form, *Wizard Step 5 of 5*, to validate the target device hardware and communications, and to program the device.

Before programming the device, the wizard interface allows you to compute a checksum, and to reset and validate the target board.

3.5.1 File checksum

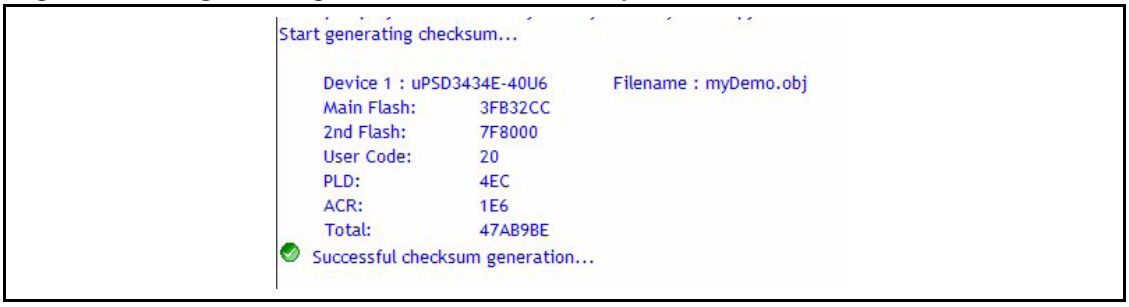
Click the **File Checksum** button to calculate NVM checksum values for,

- Main flash
- Secondary flash
- User code

- PLD
- Configuration bits
- The entire programming data file

The results of the checksum operation are displayed in the *Output Log* window (enable *Output Log* in the *View* menu). The device type and programming data file name are also displayed.

Figure 29. Programming data file checksum output



3.5.2 Target device setup and validation

Specify your JTAG programming hardware; either FlashLINK cable (parallel port communication) or RLINK-ST (USB port communication).

Click the appropriate **FlashLINK Connect Test** button or **RLink Connect Test** button, which performs a loopback test to verify connectivity between your PC port and the cable or dongle.

It is recommended that you run connectivity tests before attempting any JTAG operations.

Click **Target Connect Test** button to verify connectivity between your PC port and the JTAG chain on the target board, itself.

3.5.3 Reset hardware

Click **Reset Target** to reset the target board.

3.5.4 Program target device

Click the **Execute** button to program the target device.

3.6 Ending the wizard session

The wizard session can be terminated at any step by clicking the **Cancel** button. You will be prompted with the option to save or exit without saving any changes made during the current session.

Exit and save changes made during the current wizard session, immediately, by pressing the **Save and Exit Wizard** button.

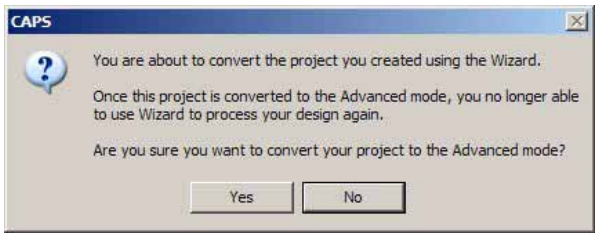
4 Working in advanced mode

From the window used to create a new project, click the **Create in Advanced Mode** button to use the advanced design mode.

When opening an existing wizard mode project, click the **Convert to Advanced** button to switch to advanced design mode. Since you may not revert to the wizard mode once you have switched to advanced mode, a dialog box appears asking you to confirm the switch to advanced mode, *Figure 30.: Convert to Advanced Mode Confirmation Dialog*. To convert to advanced mode, click **Yes**. Click **No**, if you are uncertain about using the advanced mode features or if your project can most likely be designed using the wizard.

*Note: It is recommended that you rename your project, using **Project / Save As Project**, prior to converting to advanced mode. This allow you to retain the wizard version, if you decide not to continue using advanced mode.*

Figure 30. Convert to Advanced Mode Confirmation Dialog



This section provides a description of how to access and use the functionality provided by the advanced design mode, including:

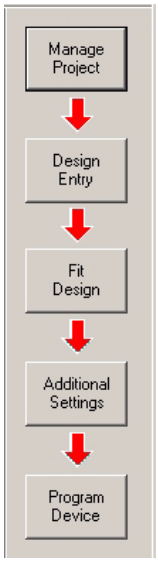
- The design flow model that guides you through the advanced mode design process.
- Using the module library to implement a design using the general purpose PLD.
- Specifying the memory map.
- Firmware placement.
- Specifying external chip selects.
- Editing PSDabel-HDL code, for PLD logic requirements beyond those provided by the module library.
- Specifying I/O pin assignment.
- Fitting the design.
- Setting security.
- Setting JTAG/ISP parameters.
- Setting sector protection.
- Validating and programming the target device.

4.1 Design flow

The CAPS design flow interface models the typical steps used in the design process. These steps have corresponding navigation buttons, shown in the left frame of the main window (*Figure 31: Design flow interface buttons*).

*Note: Depending on the device chosen and the **Preference** settings, some choices shown in the figure may be masked.*

Figure 31. Design flow interface buttons



The CAPS design flow shown in the figure, above, changes dynamically. When a box is gray with a gray shadow, it indicates that the process within the box cannot be invoked until you have completed a previous step. The red shadow indicates the next action to be performed and can be used as a guide for the next action to take in the design process.

Click on the button for the desired design and programming operation.

4.1.1 Manage project dialog

Click the **Manage Project** button to access the project lifecycle management functions while in advanced mode. These include the functions listed below. Invoke the function by clicking the radio button associated with the function, then click **OK**.

Create a new project	Select this option to create a new project for your design.
Open an existing project	Select this option to open a project that you have previously created.
Save current opened project to a different name	Select this option to rename the current project. The project with the new name becomes the currently active project.
Delete an existing project	Select this option to delete a project and all of its associated files.

Close current opened project	Select this option to close the currently active project.

4.1.2 Design entry forms

Use one or more of the design entry forms for CAPS design. These forms are accessed by clicking the **Design Entry** button:

Module Library	Use this form to include popular logic modules. See *Using the module library* for a detailed discussion on using this form.
Memory Map Editor	Use this form to describe the system memory map requirements. See *Specifying the memory map* for a detailed discussion on using this form.
PSDabel-HDL Editor	Use this form to describe your design with boolean equations, truth tables, state diagrams or combination of these. See *Editing PSDabel-HDL code* for a detailed discussion on using this form.
I/O Pin Editor	Use this form to define or change pin assignments. See *Specifying I/O pin assignment* for a detailed discussion on using this form.

Each of the available forms is accessible as a main window tab; see *Figure 32: Module library tab* as an example.

4.1.3 Fit design command

The fitting process allocates resources required for your design based on the target device architecture constraint. The fitting information generated in this process also determines the configuration information, which is included in the programming data file.

4.1.4 Additional settings form

This design entry form, accessed by clicking on the design flow **Additional Settings** button presents a single, tabbed window.

Configuration	Use this form for additional design settings, including setting the security bit (*Setting security*), specifying a user-defined code (*Setting a JTAG/ ISP user code*) and setting the internal memory sector protection (*Setting sector protection*).

4.1.5 Program device form

Use this form to program your design into the target device. Options are also available to calculate the file checksum, generate a SVF or JAM file, and validate the target device. See *Validating and programming the target device* for a detailed description on using this form. Chose the **Program Device** design flow button to select the form.

4.2 Using the module library

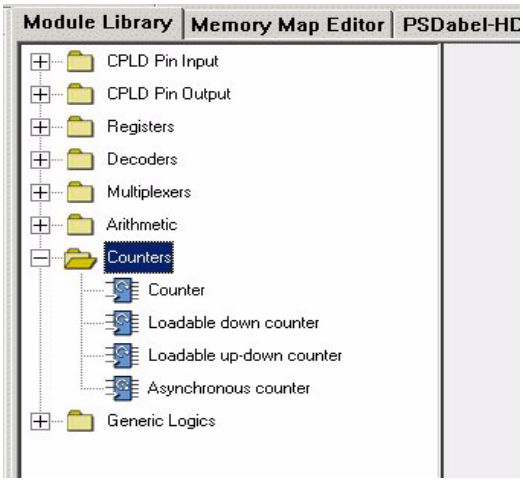
CAPS includes a module library containing popular logic modules for use in your design.

Access the CAPS module library by clicking the **Module Library** tab of the Design Entry form (*Figure 32.: Module library tab*).

Figure 32. Module library tab

The Module Library form lists the library modules, organized in folders in the left frame of the main window (*Figure 33.: Module library folder contents example*). Click on a folder to expand it and view the logic modules. The blank window to the right is the library frame workspace. Drag-and-drop a module from the list to the workspace to add the module to your project and define module signals.

Figure 33. Module library folder contents example



Note: When a module is added to a project, CAPS automatically assigns a unique module name. The user may use the editor window to change the module name but the name must remain unique for the project.

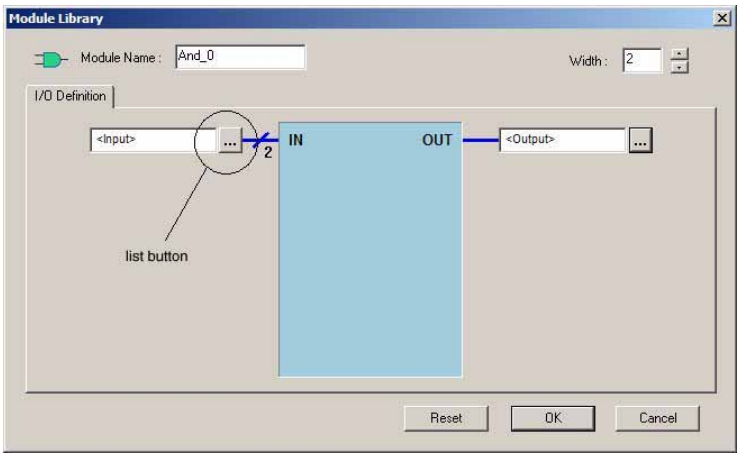
During module definition, you may also connect modules within your design (*4.2.3: Connecting modules*).

4.2.1 Adding and editing a logic module

Use these steps to add a module to your project and edit the module.

1. To add a module to your project, drag-and-drop the desired module from the module list to the workspace.
2. Click on the module icon in the workspace to open it for editing. A module edit dialog window is displayed similar to the one shown in *Figure 34.: Library module edit dialog window*.

Figure 34. Library module edit dialog window



3. You may modify the module name and the available signals, using the following procedures.

- To modify the module name, change the name in the *Module Name* edit box. A valid name,
  - can be up to 25 characters.
  - must begin with an alphanumeric character, underscore or tilde.
  - cannot contain spaces, symbols or punctuation.
- To edit signal names,
  - enter a name in the signal name edit box (The signal name must follow the same naming conventions as the module name, described above).

Note: Multiple names may be entered for a signal, separated by a comma.

  - optionally, choose a name from a list using the list button; see *Figure 34.: Library module edit dialog window*.
  - click the radio button next to a signal name box to change the signal polarity, if applicable. A blue dot on the signal line indicates inverse signal polarity.
  - use the *Width* field scroll buttons to change the number of lines associated with a signal whose width is variable.

4. Click the **Reset**, **Apply** or **Cancel** button to end the edit session for a module.

- Reset: Reset all entries to the default state.
- Apply: Save all changes to the module and exit the module edit session.
- Cancel: Exit the edit session without saving changes made to the module.

4.2.2 Deleting a logic module

To delete a logic module from your project design file,

1. Right-click on the module icon in the library frame workspace.
2. In the resulting dialog, click the **Delete module** button.
3. When asked to confirm deletion, choose **Yes** or **No** as appropriate. Clicking **Yes** removes the module from your project. Clicking **No** has no affect.

4.2.3 Connecting modules

This feature allows you to connect signals between modules within your project.

1. In the workspace, click on a module icon for the module you wish to connect to another module for editing.
2. Click the list button (*Figure 34.: Library module edit dialog window*) for the signal to be connected to another module. This lists the other modules and signal groups in your project.
3. Expand the folder for the module to connect to and expand the folder for the signal to connect to in that module. This displays the available signal names.

Note: A signal name associated with a blue circle indicates a connectable signal. A signal name associated with a gray circle indicates the signal cannot logically be connected to your module.

4. If the signal can be connected to your module, double-click on the signal name, then click **OK**. The signal name appears in your module opened for editing.

4.3 Specifying the memory map

Use the memory map editor to enter the target system memory requirements, including external chip selects and peripheral I/O definitions.

Figure 35. Memory map editor tab

You may choose one of these methods to specify the memory map:

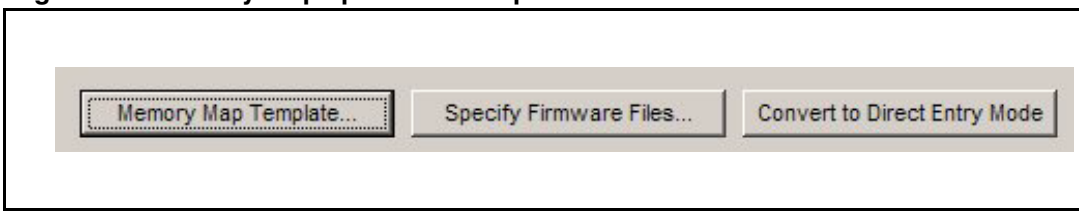
- Graphical entry mode using a memory map template (this is the default mode).
- Direct entry mode.

The graphical entry mode allows you to work with common memory layout templates, while direct entry mode allows you to directly enter additional logic qualifiers for individual memory chip select signals. Direct entry mode is typically needed only for advanced or complex memory mapping schemes that allow the re-mapping of physical memory segments to multiple logical address ranges, or to no address range, during run-time.

In the Memory Map Editor dialog, click either the **Memory Map Template** button or the **Convert to Direct Entry Mode** button to select the memory specification method, as shown in

Figure 36.: Memory map specification option buttons. (The dialog also includes the facility for firmware file placement in memory, **Specify Firmware Files**, which is discussed in more detail in *4.4: Firmware placement*).

Figure 36. Memory map specification option buttons



The remainder of this section describes memory map specification in more detail.

4.3.1 Using the memory map template

The memory map template presents a graphical interface that allows you to choose from common, pre-defined memory layouts, to drag-and-drop individual memory chip-select signals to either code or data space, and to define peripheral I/O pin assignments. This is useful for simple memory map schemes.

Note: If the original memory map was defined using wizard mode, some of these chip-selects are already defined for advanced mode. You may choose to make additional modifications in advanced mode.

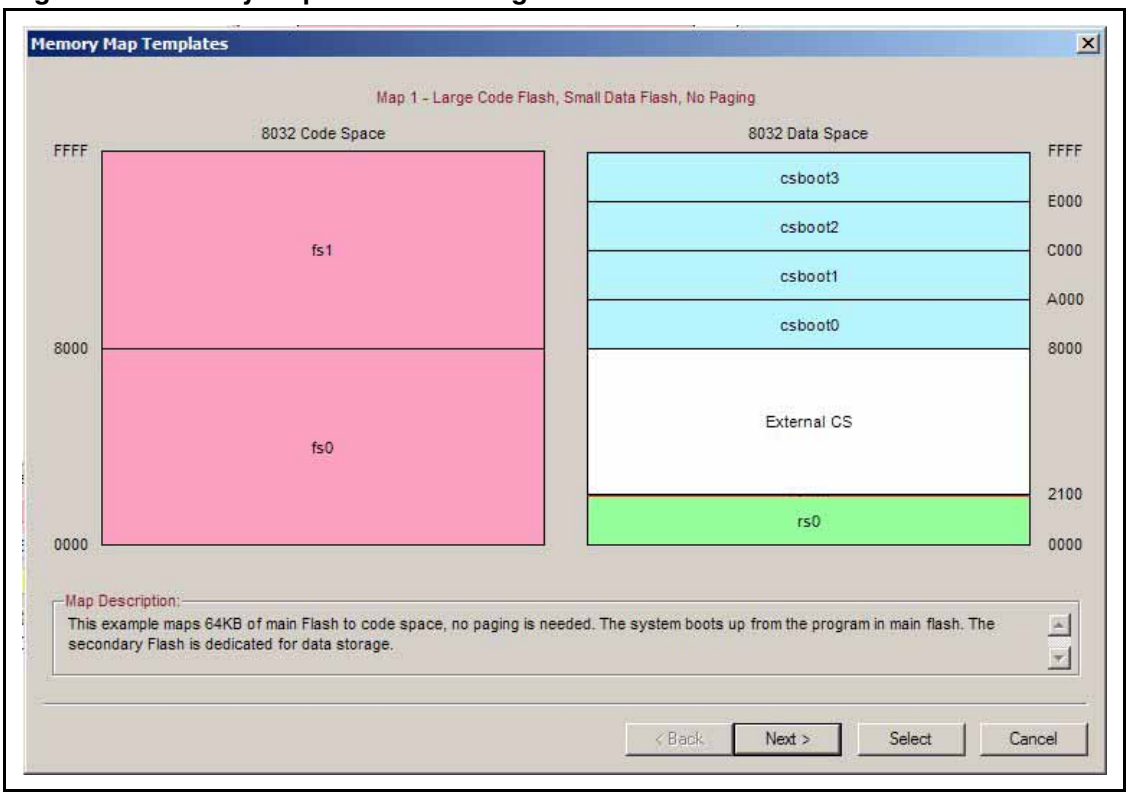
Click the **Memory Map Template** button to use the template method for specifying the memory map. Refer to *Figure 36.: Memory map specification option buttons*.

Each memory map template dialog window includes a title and description that describes the selected memory configuration option.

1. Select the desired memory map template.

   A number of common memory configurations are available, such as the example shown in *Figure 37.: Memory map selection dialog*.

Figure 37. Memory map selection dialog



Use the **Next** and **Back** buttons to browse a list of available memory map templates.

2. Click **Select** to choose a memory configuration option. This returns you to the Memory Map Editor form, displaying the selected map.

   Click **Cancel** to exit the memory map selection dialog, without choosing a different memory configuration.

In conjunction with the memory map templates, you may also use the memory segment list to the left of the workspace to assign memory. Simply, expand the folder (*Main Memory*, *External Chip Select*, etc.) and drag the label to the appropriate code- or data-space of the workspace. See *4.3.2: Other memory map editor functions* for the description of how to edit the memory segment starting and ending addresses.

4.3.2 Other memory map editor functions

Once a memory segment is defined and displayed in the memory map workspace, you may continue to edit the memory map as needed.

Placing the cursor over a memory segment, *right-click* to use the following editing options:

- Delete

  Click **Delete** to delete the selected memory segment. This visually removes the segment from the memory map workspace as well as from your design.

- Edit

  Click **Edit** to view the memory definition dialog, which allows you to manually enter a *Page Number*, *HEX Start Address* and *HEX End Address*. Click **Apply** to accept your changes.

Firmware placement is discussed separately in *4.4: Firmware placement*.

Chip select definition is discussed separately in *4.5: Specifying chip selects*.

4.4 Firmware placement

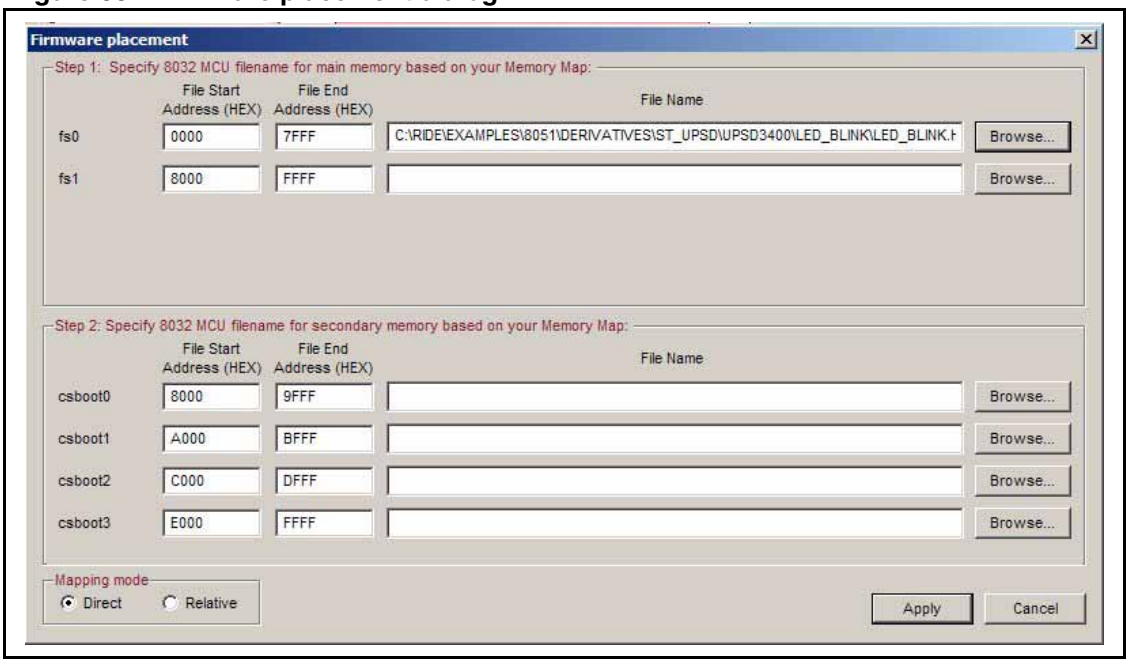
Use the firmware placement feature to specify firmware file placement in code- or data-space, assigning a firmware file to its respective location defined in the memory map.

Note: The firmware file must be in Intel hexadecimal format (Appendix A: Intel hex-32 record format).

Clicking the **Specify Firmware Files** button (*Figure 36.: Memory map specification option buttons*) displays the firmware placement dialog window, below.

Note: The Specify Firmware Files button is context-sensitive, only active when code or data memory segments are defined, by either dragging a memory segment label (fs1, fs2, fs3 ...) to the memory map workspace or by choosing one of the memory map templates.

Figure 38. Firmware placement dialog



For each memory segment shown in the dialog,

- If you are using paging or other memory manipulation, verify that the **File Start** and **File End** addresses are consistent with your system design. Edit or add the start and end addresses as needed.
- Click the **Browse** button to locate the firmware file to be loaded into the device non-volatile memories. The firmware filename may also be entered manually, and if a path is not included the entered firmware file is expected to be in the project file folder.

You must also use the radio button to select the desired mapping mode, direct or relative. Most applications use direct mapping mode.

- Direct mapping implies a one-to-one correspondence with the address output by the MCU and the address that selects the internal memory segments of the device.

- Relative mapping enables you to specify different physical addresses (output by the MCU based on the firmware file) than the address that selects the memory segments within the device.

Click **Apply** when you have finished entering the mapping information.

Click **Cancel** to exit firmware file mapping, without making changes.

4.5 Specifying chip selects

Chip selects may be defined using,

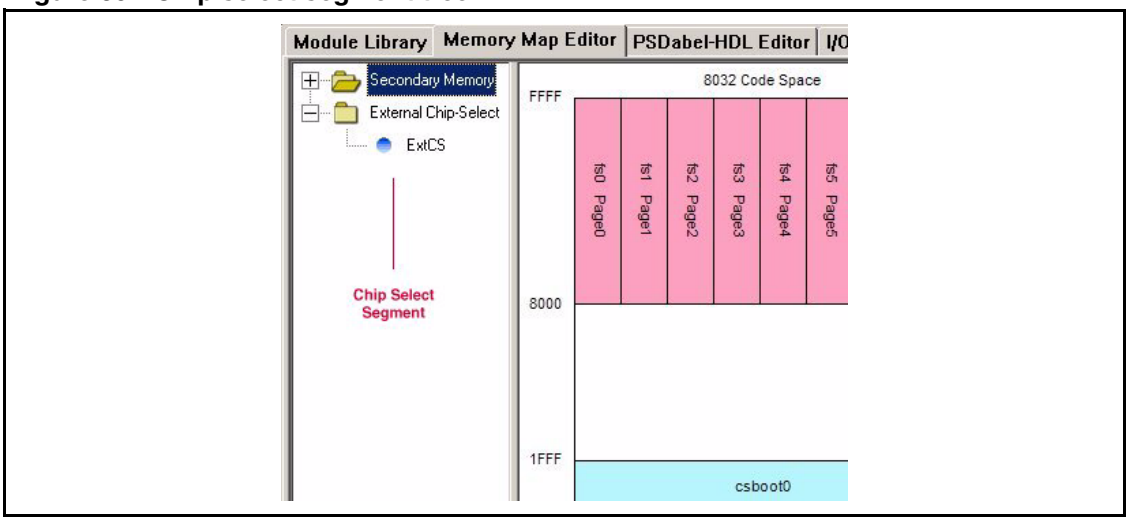
- Graphical mode.
- Direct mode.

Note: Sometimes it can be beneficial to draw a graphical, system-level memory map before beginning this process. Also, see the appropriate datasheet to determine the size of each of the internal segments and the rules (priority scheme) for overlapping memory segments.

4.5.1 Defining chip selects in graphical mode

Define chip selects using the memory map editor form. The chip select segment is displayed in the left frame, as shown in *Figure 39.: Chip select segment tree*. When the memory map template is used, the available chip select segments differ according to the memory map template selected.

Figure 39. Chip select segment tree



Define each chip select using these steps.

1. Expand the tree and drag-and-drop a desired chip-select segment to data space.
2. If you wish to make changes to the address range, double-click the space in the chip-select segment and make the change accordingly. You can also set the polarity of your chip select to active-low by clicking the *active-low* radio button.

Note: 1 Page number can only be entered for the type of space, code or data, for which paging is specified.

2 (µPSD-only) The active-low internal power-down signal (!pdn) is automatically included to each internal memory segment definition for devices that have the Automatic Power Down feature.

3 *(µPSD-only) If you choose to use Port D (pd2) as the chip-select input (_csi), it is also automatically included to each internal memory segment definition.*

4.5.2 Using direct entry mode to define chip selects

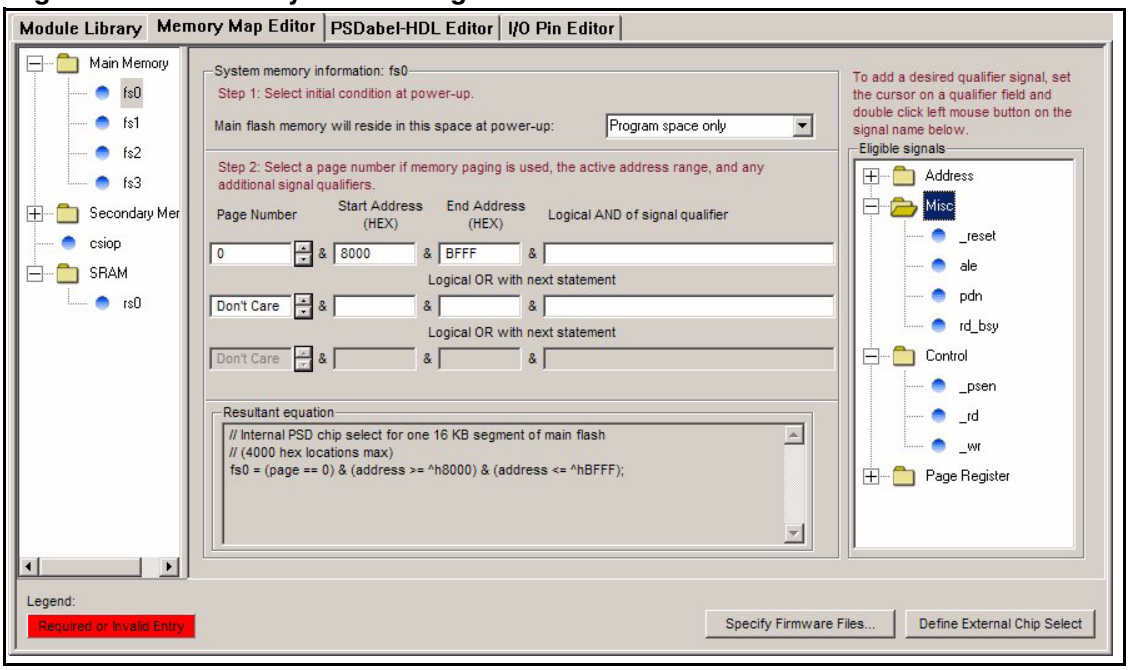
If you have a complex memory mapping scheme that requires additional qualifying information for the memory segments, use the direct entry mode. (Such additional qualifying information is entered in the *Logical AND of signal qualifier* column, shown in *Figure 40.: Direct entry mode dialog*).

Warning: Once you convert the CAPS project to direct entry mode, you cannot return to graphical entry mode. You may wish to save a copy of the project before converting to direct entry mode, as a way to recover from potential errors. The dialog displays a warning with the option to cancel before entering direct mode.

Click the **Convert to Direct Entry Mode** button to convert the memory map to text-based entry method. Refer to *Figure 36.: Memory map specification option buttons*.

A memory map specification dialog appears, similar to the example shown in *Figure 40.: Direct entry mode dialog*.

Figure 40. Direct entry mode dialog



Repeat these steps for each chip select definition:

1. Highlight a chip select segment from the tree panel you wish to define.
2. For memory chip select segments, select the initial configuration of how the flash memory output enable signal is activated. The resulting configuration takes effect upon power-up or system reset and remains in effect until the MCU optionally overrides the settings at run-time using the control register named VM.
   – Code Space: Choose this option to allow the flash memory array to drive the MCU data bus while the $\overline{\text{PSEN}}$ signal is active. This places all of the flash sectors into program space. That is, it is only considered code memory.

– Data space: Choose this option to allow the flash memory array to drive the MCU data bus while the $\overline{RD}$ signal is active. This places all of the flash sectors into data space.

3. To use paging for the chip select highlighted, type in a page number in its column or select it using the arrows. If no paging is being used for a particular select signal, no action is needed.
4. To make changes to the address range, enter the start and stop addresses in hexadecimal format in the next two columns for which the chip select highlighted should be valid.
5. In the “Logical AND of signal qualifiers” column, you can enter a signal from the list of eligible signals in the far right column to be included in the end equation.

Note: To add a desired qualifier signal, set the cursor on a qualifier field and double-click the left mouse button on the signal name. This causes the ampersand (&) symbol to be added to the signals, thus ANDing the signals.

6. To logically AND additional signals, enter *ampersand (&)* followed by the signal name. Use the exclamation (!) symbol to negate the signal.
7. If the chip select signal also requires more OR logic (logical ORs), repeat steps 3 through 6 for the next row.

4.6 Editing PSDabel-HDL code

Use the PSDabel-HDL editor for logic requirements beyond that provided by the module library.

The PSDabel-HDL code declares pin and node definitions, and signal group logic equations. Equations define I/O and node states based on the behavior of other signals. For a complete guide to the ABEL language, see the PSDabel-HDL Language Reference, which may be downloaded from **Support Files** menu at http://mcu.st.com.

Click the **PSDabel-HDL Editor** tab of the Design Entry form (*Figure 41.: PSDabel-HDL editor tab*) to access the CAPS PSDabel-HDL editor.

Figure 41. PSDabel-HDL editor tab

The main window displays a frame on the left containing a list of logic modules used in the project and a list of default reserved device signal names, organized in expandable folders. The frame to the right contains your design description in ABEL notation.

As noted on the form, the ABEL code statements color-coded in blue describe the logic modules shown in the left frame, including all entries entered in the memory map editor. These statements are automatically generated by CAPS and may not be edited.

You may add additional declarations and equations definition to the ABEL design file between the respective *//Start Declaration* and *//End Declaration*, and *//Start Equations* and *//End Equations* statements, color-coded in green.

Click **View Equation** to display the output of the optimized compiled equations.

Click **Apply** to save, compile and optimize your design file.

If you have the output log enabled, the output frame displays the viewing or compilation progress.

4.7 Specifying I/O pin assignment

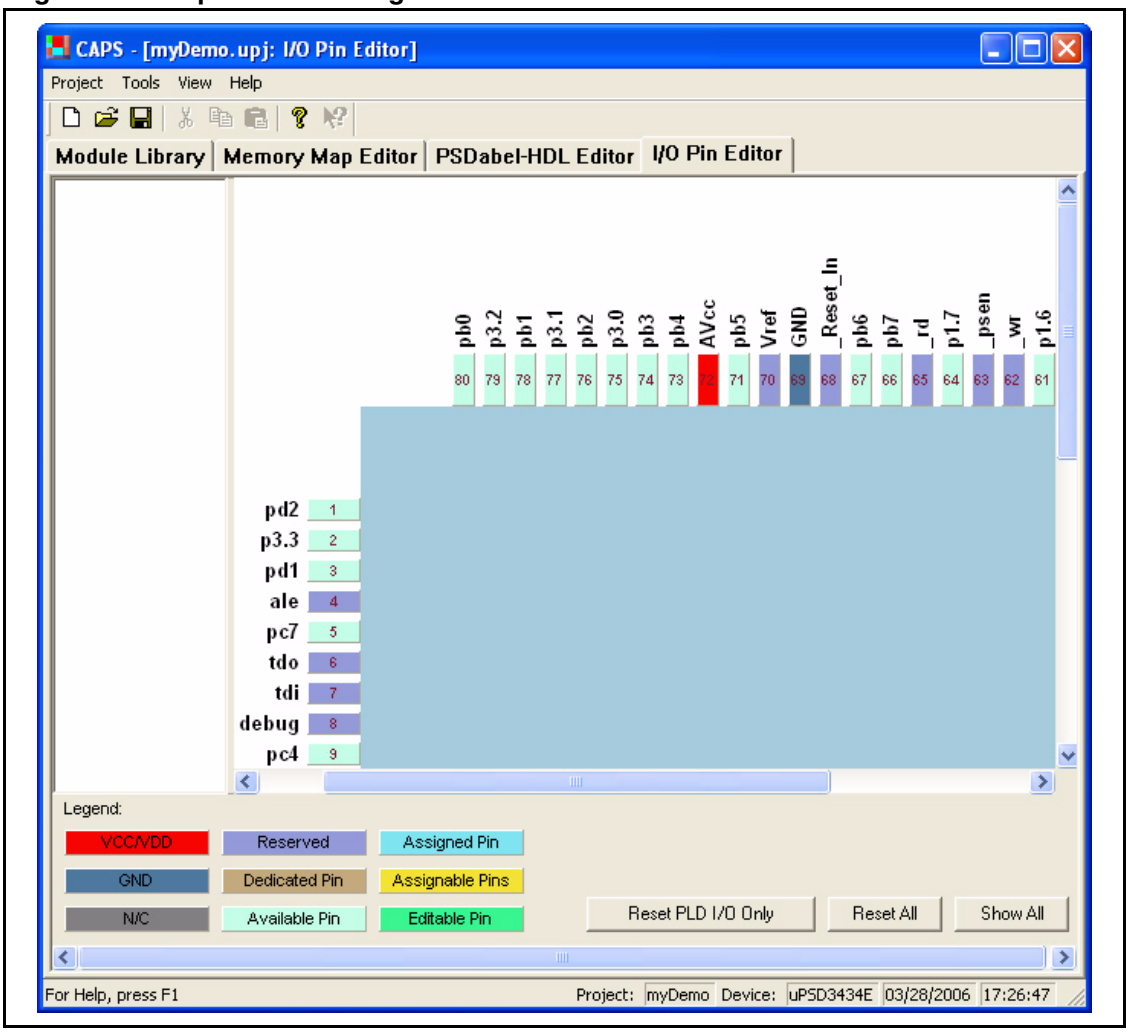
The I/O pin assignment dialog allows you to define each pin function for a device, using a point-and-click interface.

Click the **I/O Pin Editor** tab to begin I/O pin assignment.

Figure 42. I/O pin editor tab

The I/O pin editor dialog is displayed (*Figure 43: I/O pin editor dialog*).

Figure 43. I/O pin editor dialog



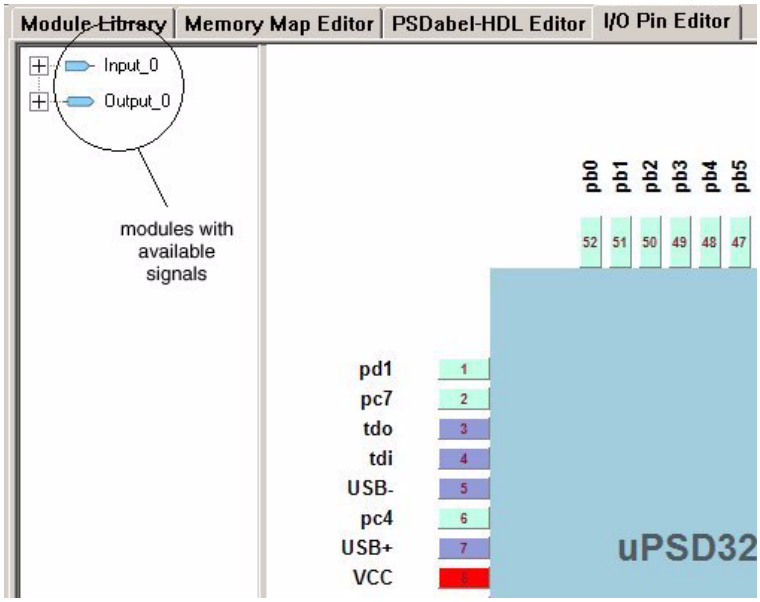
CAPS displays a graphic representation of the device selected when you created the project, showing the current pin assignments. The legend at the bottom of the display explains the meaning of the color-coded pin assignment state.

Notice that pins with dedicated functions are predefined. Only those pins shown in shades of green are either available or editable. Signals that were defined in either the module library or using the PSDabel-HDL editor may be assigned to these pins.

4.7.1 Assign a signal to a pin

The I/O Pin Editor dialog shows, in the left frame, modules which have signals defined in the design but not yet associated with a pin. See *Figure 44.: I/O pin editor available signals*.

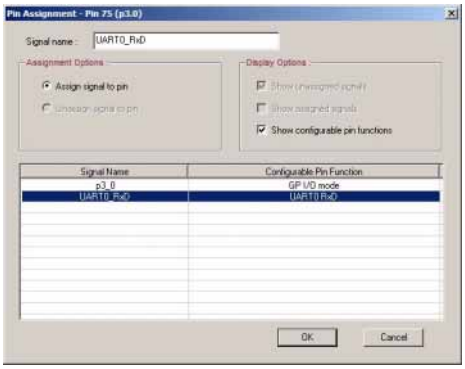
Figure 44. I/O pin editor available signals



Associate a signal with a pin on the target device by dragging the signal to the pin. While dragging the signal, assignable pins are highlighted in the dialog, and when the signal is dropped to the pin, the signal name is displayed with the pin.

If a special function is needed, click the *available* or *editable* device pin. A dialog window, similar to the one shown in *Figure 45.: I/O pin assignment dialog* displays the names and source(s) of the all unassigned signals as default (provided the *Show unassigned signals* checkbox is checked and the *Assign signal to pin* radio button is selected).

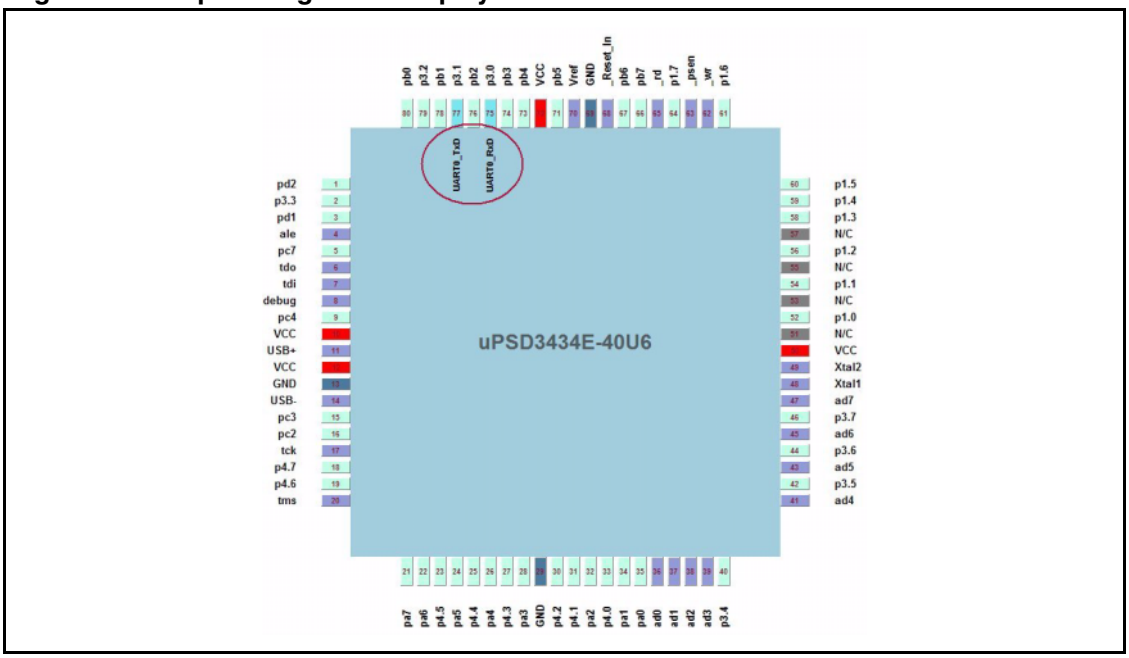
Figure 45. I/O pin assignment dialog



Check the *Show assigned signals* checkbox to view the signals that are already assigned and the target device pin to which they are assigned.

Highlight the desired signal, as shown in the above figure, and click **OK** to assign the signal to the device pin. The name of the signal appears next to the pin in the *I/O Pin Editor* workspace, as shown in *Figure 46.: I/O pin assignment display*.

Figure 46. I/O pin assignment display



4.7.2 Unassign a signal to a pin

To unassign a signal to a pin, again, click the device pin to display the *Figure 45.: I/O pin assignment dialog* dialog window. Now, select the *Unassign signal to pin* radio button and check the *Show assigned signals* checkbox. Highlight the signal to be unassigned and click **OK**. The signal name is removed from the device pin in the workspace.

4.7.3 Assign a configurable function to a pin

Some device pins have configurable functions. Clicking a pin to show *Figure 45.: I/O pin assignment dialog*, Check the *Show configurable pin functions* checkbox to view the options for the pin. Highlight the desired option and click **OK** to assign the option to the pin. This limits the availability of the pin for assigning other signals.

Assigning alternate functions to configurable pins (e.g., UARTs, ADC, PCA, Timers, SPI, I2C) allows you to see which pins are already allocated for alternate functions, and thus which pins are still available to be assigned to other functions in CAPS. Please, note that the MCU firmware still needs to correctly configure the alternative function at reset using the applicable SFR. Please, see the datasheet for the µPSD you are using for detailed information.

4.8 Fitting the design

Click the **Fit Design** button to start the fitting process that resolves design resources to the target device architecture. During the fitting process, you can view error and warning messages in the output log window.

Warning: Errors reported by the fitter must be fixed before proceeding with your design.

Note: Occasionally, the fitter seems to run longer than expected. The most likely cause is that the fitter is attempting to fit the design used on the pin assignment constraint and device resources imposed by your design. In this situation, it is likely that your design does not fit to the target

device. A simple check is to un-assign pin numbers to the signals used in your design and then refit your design. To do this,

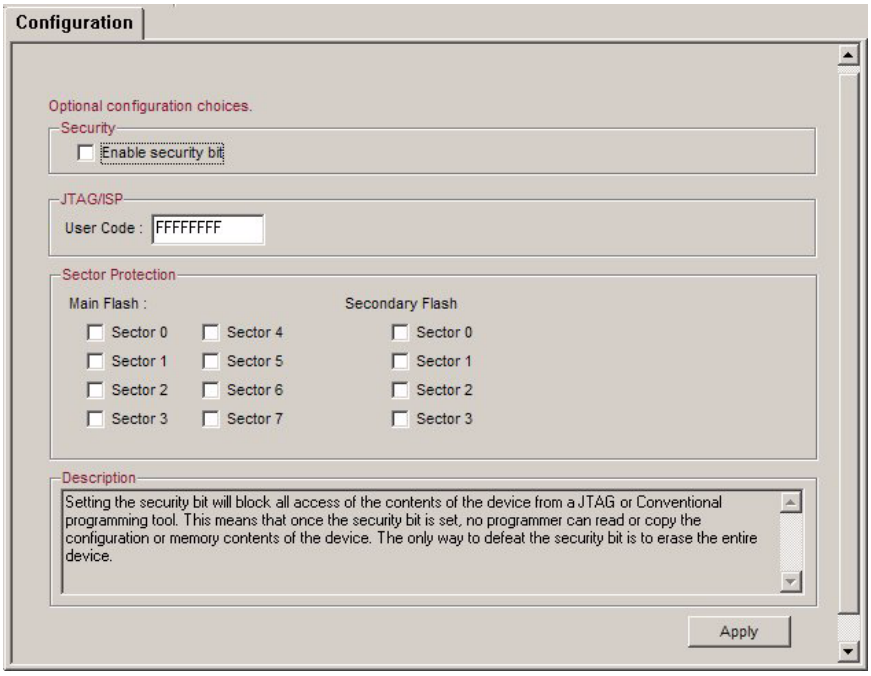
- *In PSDabel-HDL editor, remove all pin number assignments in the design.*
- *In I/O Pin Editor, click the* ***Reset PLD I/O Only*** *button to remove PLD input/output pin assignments.*

4.9 Setting security

Setting the security bit blocks all access to the content of the device by means of JTAG or a conventional programmer. This means that once the security bit is set, no programmer can read or copy the configuration or memory contents of the device. The only way to erase the security bit is to completely erase the device.

Click the **Additional Settings** design flow button to use the *Configuration* form for setting security (*Figure 47.: Configuration form*). This form is also used to set the JTAG/ISP user-defined code and to set NVM write protection.

Figure 47. Configuration form



Click the *Security* checkbox to enable security. Uncheck the box do disable security.

Note: The only way to override the security bit is to erase the entire device.

After completing all *Configuration* form entries, click the **Apply** button. This causes CAPS to regenerate a new programming data file for the design.

4.10 Setting a JTAG/ISP user code

This option allows you to enter a 32-bit code, which can be used for various functions, such as to facilitate programming contents and revision level identification.

Click the **Additional Settings** design flow button to use the *Configuration* form for setting the user code (*Figure 47.: Configuration form*). This form is also used to set the security bit and to set NVM sector protection.

Enter the *JTAG/ISP User Code* on the *Configuration* form, entering any 32-bit hexadecimal value. The default value is FFFFFFFF.

After completing all *Configuration* form entries, click the **Apply** button. This causes CAPS to regenerate a new programming data file for the design.

4.11 Setting sector protection

Individual NVM segments within the device may be statically write-protected to prevent accidental data loss.

Click the **Additional Settings** design flow button to use the *Configuration* form for setting security (*Figure 47.: Configuration form*). This form is also used to set the security bit and to set the JTAG/ISP user code.

Click the checkbox for the sector you wish to write-protect. Uncheck the box to disable protection. Protection granularity is eight sectors in *Main Flash* and four sectors in *Secondary Flash*.

After completing all *Configuration* form entries, click the **Apply** button. This causes CAPS to generate a new programming data file with the entered options included.

4.12 Validating and programming the target device

After completing the design and fitting of your project, you are ready to program the firmware into your target device.

Before you begin programming the device,

1. Set your default programming to use either *Single device view* or *Multiple device view*. Use single device view if there is only one device on your board. Use multiple device view if multiple devices are daisy-chained.

   A JTAG chain is defined to be two or more JTAG-compliant, IEEE 1149.1 standard, devices connected together in a chain. Each device in the chain must support the four basic JTAG signals: TDI, TDO, TCK and TMS.

   The following JTAG chaining rules apply.

   – Different brands of chips must be set in “ByPass” operation and will require different programming software.
   – Only one device in a chain will be programmed at a time, not concurrently.
   – Before any JTAG operation can begin, the chain order must be defined and the instruction register length for each device must be known by the software.

Note: The order of the chain is important when programming the devices. You must set up the chain in your programming software such that it matches physical ordering of the devices on your board.

Click **Project** | **Preferences** and select the desired option, as shown in the figure:

Figure 48. Project preferences dialog

2. Setup and power your target device as described in *1.2: Setting up the target hardware*.

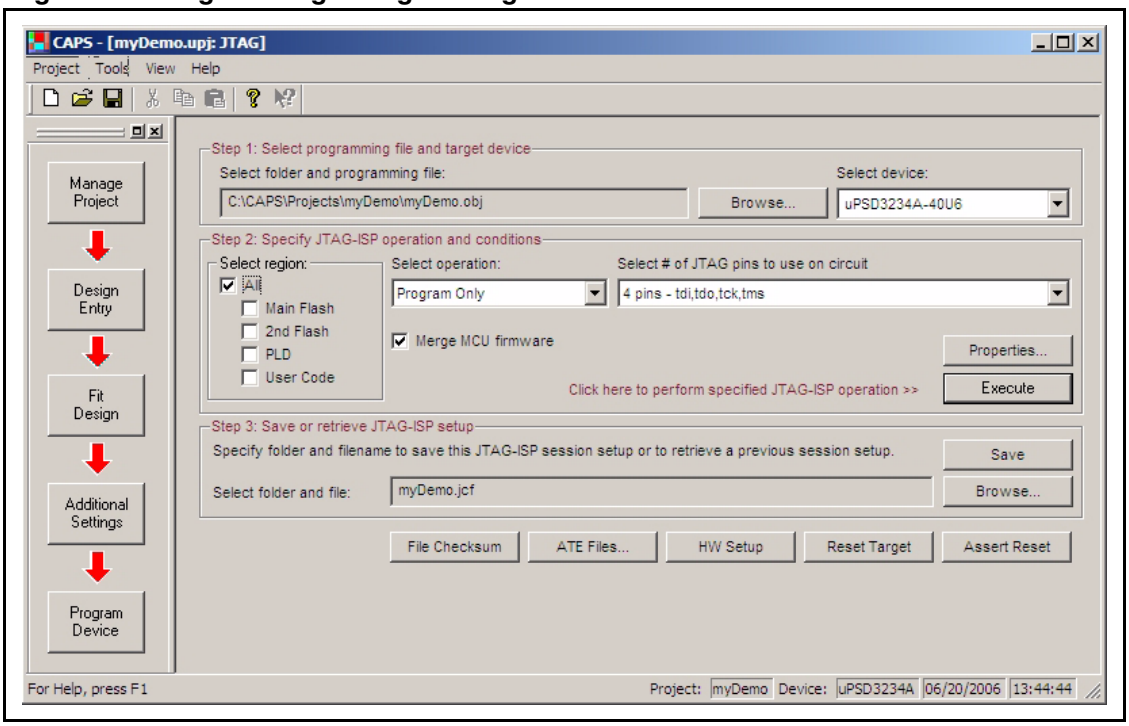
Click the **Program Device** design flow button to access the device verification and programming dialog. If *Single device view* is selected in the JTAG/ISP *Preferences*, *Figure 49.: Programming dialog for single device* is displayed. If multiple device view is selected, *Figure 50.: Programming dialog for multiple devices* is displayed.

The programming dialog presents the following functions, discussed in more detail, below.

Note: The single device view is discussed initially, since it provides an easier introduction to device programming. The multiple device view is discussed, separately, in 4.12.2: JTAG-ISP operations for multiple device view.

- JTAG/ISP operations
  - Blank test
  - Erase/Program/Verify
  - Program only
  - Verify
  - Upload
  - ByPass
- File generation
  - Calculate NVM checksum values
  - Generate SVF or STAPL format files
- Target hardware configuration
  - Define and validate communication link to the target board
  - Reset target board

Figure 49. Programming dialog for single device



The dialog guides you through the following programming steps.

1. *Select programming file and target device*

   By default, the programming data file for the currently active session is displayed. In *Figure 49: Programming dialog for single device*, the name of the file is *myDemo.obj*, including the pathname. You may use the **Browse** button to open another, previously saved programming data file, using the file browser.

   The type of target device associated with the programming data file is shown in the *Select device* box. (This field may not be edited).

Note: All other file-related operations within this dialog window operate on the programming data file opened in step 1.

2. *Specify JTAG-ISP operation and conditions*

   If the target device and communication links are in a known state, you may choose from one of the JTAG-ISP operations (*Select operation*). Refer to *4.12.1: JTAG-ISP operations for single device view* for a more detailed discussion of the programming options.

   Use the *Select region* checkboxes to select the device memory region to which to apply the JTAG-ISP operation. One or more regions may be selected simultaneously by checking the desired checkbox. As a convenience, check *All* to apply the operation to all regions.

   You may also choose to merge the MCU firmware by clicking the *Merge MCU firmware* checkbox.

3. *Save or retrieve JTAG-ISP setup*

   To save the JTAG/ISP settings entered in the preceding steps, click the **Save** button. A browser dialog window appears, allowing you to enter the file name and directory location to save the file.

This saves a JTAG Chain File with extension *.jcf*. Later, you may use the **Browse** button to open the file to restore your current settings.

4. Additional functions

   If this is the first time programming the device, use these operations to set the target device to a known state, select and validate the communication link, and generate NVM checksums. See *4.12.3: Compute checksum* and *4.12.5: Target hardware operations* for a more detailed discussion of these operations.

   For a discussion on generating ATE files, see *4.12.4: Generate ATE file*.

4.12.1 JTAG-ISP operations for single device view

Choose *Single device view* in the **Project** | **Preferences** menu if JTAG chaining is not used,

Follow these steps to program in single device view, referring to *Figure 49: Programming dialog for single device*.

Note: The "Select # of JTAG pins to use on circuit board" pull-down menu and "Properties" button are set to default values for single device view mode and no change to these options is needed.

1. Select the desired programming operation from the *Select operation* drop-down menu:

Blank Test	Determine if any region of the device has been programmed.
Erase	Erase one or more regions of the device.
Program/Verify	Write code, configuration or data to the device based on the contents of the programming data (*.obj*) file. Following programming, compare the actual contents of the device with the contents of the current programming data (*.obj*) file.
Program Only	Write code, configuration or data to the device based on the contents of the programming data (*.obj*) file.
Verify	Compare the actual contents of the device with the contents of the current programming data (*.obj*) file.
Upload	Read the contents of the device and save the contents to a programming data (*.obj*) file.
ByPass	Place the device in bypass mode. This option has limited use in single device view mode, other than to verify that the JTAG interface works correctly.

2. Use the *Select region* checkboxes to select one or more device memory region(s) to which to apply the programming operation.

All	The entire device
Main Flash	The main flash region only
2nd Flash	The secondary flash region only
PLD	The PLD region only
User Code	The user code region only

3. Configure the port pins.

Note: Port pin configuration is only available when FlashLINK hardware is used.

– *Set Pins*

This configures the device I/O pins during JTAG operations. The default direction, except for the JTAG pins, is input, which is usually acceptable for most pins. However, sometimes it may be desirable to set a pin or pins to output during JTAG to avoid pins that could potentially drive signals on the JTAG lines.

Note: The device ignores any non-JTAG I/O.

– *JTAG-ISP Attributes*

This shows the device name and its instruction register length. This information may be useful to setup the JTAG chain for 3rd party programming tools that do not have an auto-detect function.

4. Click the **Execute** button to perform the programming operation selected, above.

If you attempt to program a device that is not blank, a warning message verifies that you "want to do the full chip erase" before continuing. Click **Yes** to continue.

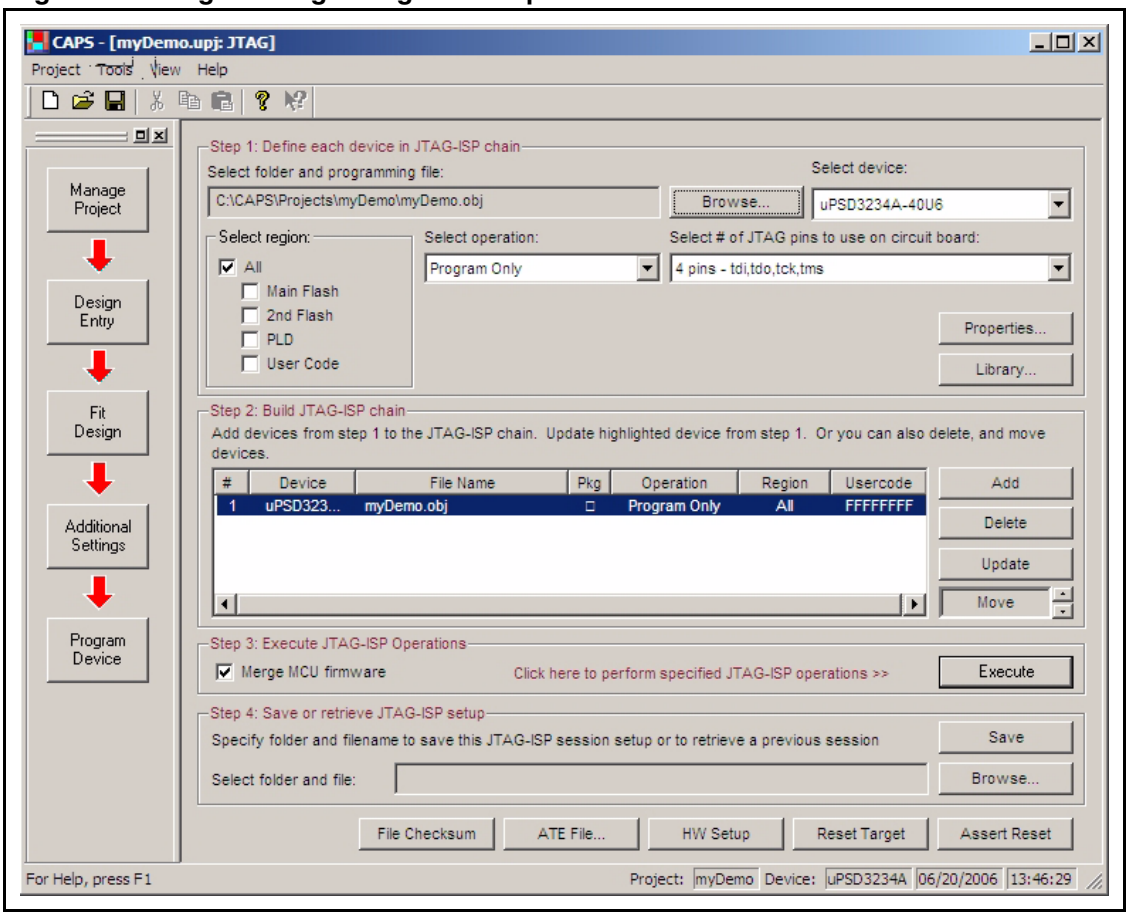
A delay may be observed while the CAPS software performs the operation.

4.12.2 JTAG-ISP operations for multiple device view

If *Multiple device view* is specified in the **Project** | **Preferences** menu, the dialog in *Figure 50.: Programming dialog for multiple devices* is displayed. In the device list window, the current programming data file and properties are automatically displayed. Other programming data files may be added and any programming data file may be deleted.

The "Select # of JTAG pins to use on circuit board" pull-down menu and "Properties" button are set to default values for multiple device view mode and no change to these options is needed.

Figure 50. Programming dialog for multiple devices



Follow these steps **for each device** you wish to add to the device programming chain.

1. Click the **Browse** button to use the file browser to locate a saved programming data (*.obj*) file you wish to add to your JTAG chain.
2. After opening the programming data file, click the **Add** button to add the file to the JTAG chain list window. Click **Delete** to remove any of the files listed. Highlight the device and click the **Move** scroll buttons to reorder the chain list.

Note: 1 The device order must match the order in which the devices are physically in the chain.

2 Device #1 should be the first device on the board that has its TDI pin connected to the JTAG/ ISP programmer.

*3 To change the attributes of any device, highlight the device with the cursor, make the desired changes and click **Update** to apply the change.*

3. For the highlighted device in the list, select the desired programming operation using the *Select region* checkboxes. One or more regions may be selected, or check *All* to select all regions. Available regions include,

All	The entire device
Main Flash	The main flash region only
2nd Flash	The secondary flash region only
PLD	The PLD region only
User Code	The user code region only

Different regions can be specified for each device.

4. Select the desired programming operation from the *Select operation* drop-down menu. Available regions include,

Blank Test	Determine if any region of the device has been programmed.
Erase	Erase one or more regions of the device.
Program/Verify	Write code, configuration or data to the device based on the contents of the programming data (*.obj*) file. Following programming, compare the actual contents of the device with the contents of the current programming data (*.obj*) file.
Program Only	Write code, configuration or data to the device based on the contents of the programming data (*.obj*) file.
Verify	Compare the actual contents of the device with the contents of the current programming data (*.obj*) file.
Upload	Read the contents of the device and save the contents to a programming data (*.obj*) file.
ByPass	Place the device in bypass mode.

Note: The programming option applies to all devices in the device list window.

5. Configure the port pins.

Note: Port pin configuration is only available when FlashLINK hardware is used.

Highlight a device in the device list window and click the **Properties** button to configure the device I/O during JTAG operations. Two tabbed windows are displayed.

– *Set Pins*

This configures the device I/O pins during JTAG operations. The default direction, except for the JTAG pins, is input, which is usually acceptable for most pins. However, sometimes it may be desirable to set a pin or pins to output during JTAG to avoid pins that could potentially drive signals on the JTAG lines.

Note: The device ignores any non-JTAG I/O.

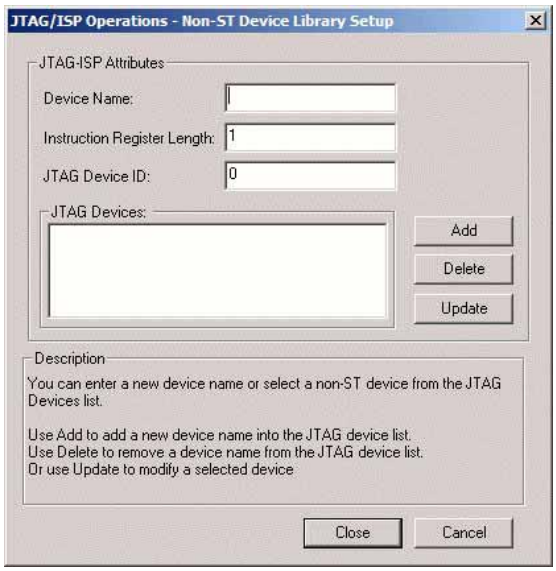
– *JTAG-ISP Attributes*

This shows the device name and its instruction register length. This information may be useful to setup the JTAG chain for 3rd party programming tools that do not have an auto-detect function.

Follow these steps to add non-ST devices to the device list. Repeat the steps for each device to be added.

1. Click the **LIbrary** button (*Figure 50.: Programming dialog for multiple devices*) to add non-ST devices to the JTAG device list library (*Figure 51: Add non-ST device dialog*). The figure shows an example of a previously added device named *libDevice*.

Figure 51. Add non-ST device dialog



2. In the edit boxes,
   – Enter the device name.
   – Enter the instruction register length. This is a hexadecimal number and can be obtained from the device manufacturer.
   – Enter the JTAG device ID.
3. Click **Add** to add the device to the *JTAG Devices* list. The device is also added to the *Select device* drop-down list.

To add the non-ST device to the JTAG chain, select the device in the *Select device* drop-down list, then click the **Add** button.

4. Click **Delete** to remove any previously entered devices. To change any of the attributes of a device, highlight the device in the list, modified the desired attribute(s) and click **Update**.
5. Click **Close** to return to the main *Multiple device view* window.

Once the JTAG chain and device properties are defined, click the **Execute** button to perform the specified programming operation, above.

If you attempt to program a device that is not blank, a warning message verifies that you “want to do the full chip erase” before continuing. Click **Yes** to continue.

A delay may be observed while the CAPS software performs the operation.

4.12.3 Compute checksum

Referring to *Figure 49: Programming dialog for single device*, click the **File Checksum** button to compute the checksum for the various non-volatile memory areas within the device. These areas include,

- Main Flash
- 2nd Flash
- User code
- PLD
- Configuration bits

A total file checksum is also computed.

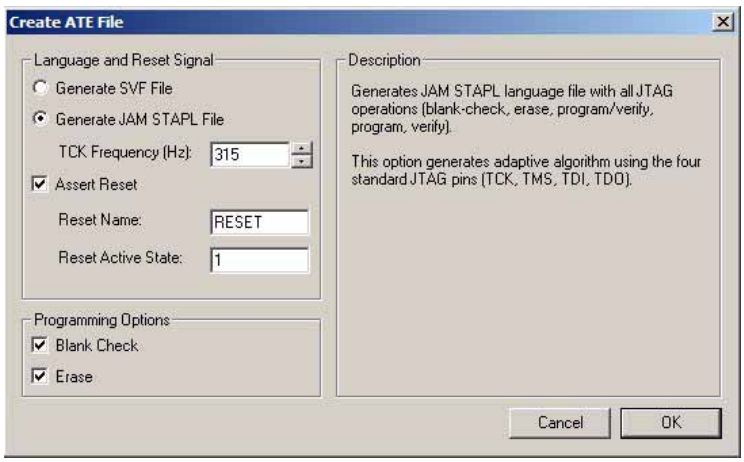
The checksums are displayed in the output log.

4.12.4 Generate ATE file

To generate either a Serial Vector Format (SVF) or JEDEC Standard programming Language (STAPL) format file, click the **ATE File** button (*Figure 49: Programming dialog for single device*). The dialog shown in *Figure 52: Create ATE File dialog* guides you through the file generation options.

1. The description frame on the right of the dialog provides a context-sensitive help for the various selection options.

Figure 52. Create ATE File dialog



2. Click the **OK** button to accept the specified file format. A browser dialog window appears, allowing you to enter the file name and directory location to save the file.
3. Click the **Save** button. The file is saved to disk and “Successful ATE file creation ...” displays in the output log.

4.12.5 Target hardware operations

Use the programming dialog to reset the target hardware to an initial state, to specify the type of communication link and perform loopback tests on the link, and to validate the target board connectivity.

Refer to *Figure 49: Programming dialog for single device*.

The target device and communication links must first be connected and powered, as described in *1.2: Setting up the target hardware*.

1. Click the **Reset Target** button to reset the target board.
2. Use the drop-down menu to choose the type of communication link (*RLink* or *FlashLINK*) connected between the PC and target board.

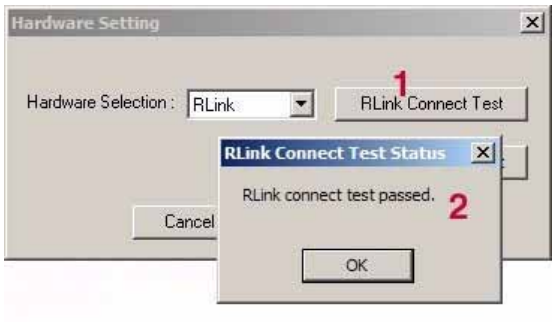
RLink	Communication using a USB port.
FlashLINK	Communication using a parallel port.

   If *FlashLINK* is selected, also select the *Parallel Port*: *Auto Select* or the LPT port.
3. Click the **Hardware Connect Test** button to confirm connectivity between your PC and the target board. (*Figure 53.: Link connectivity test dialog*, note 1).

   A successful connectivity test message appears, if the test passed. Click **OK** to continue. (*Figure 53.: Link connectivity test dialog*, note 2).

   If the test fails, check the physical connections between your PC and target board by performing a loopback test.

Figure 53. Link connectivity test dialog

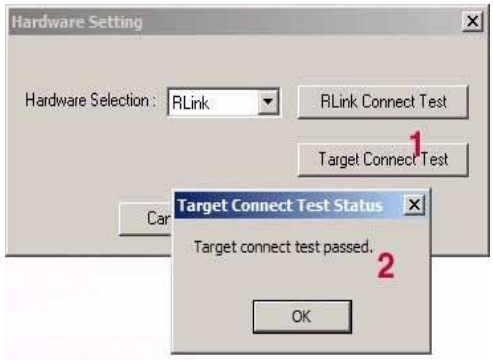


4. Click the **Target Connect Test** button to test communication from your PC through the JTAG chain on the target board. (*Figure 54.: Target connectivity test dialog*, note 1).

   If the JTAG chain on the target board successfully responded, a message appears indicating the test passed. (*Figure 54.: Target connectivity test dialog*, note 2).

   If the test fails, check the JTAG chain connectivity on the target board and that the target board is powered-on.

Figure 54. Target connectivity test dialog



Note: It is recommended that link and connectivity tests be run before attempting any JTAG operations.

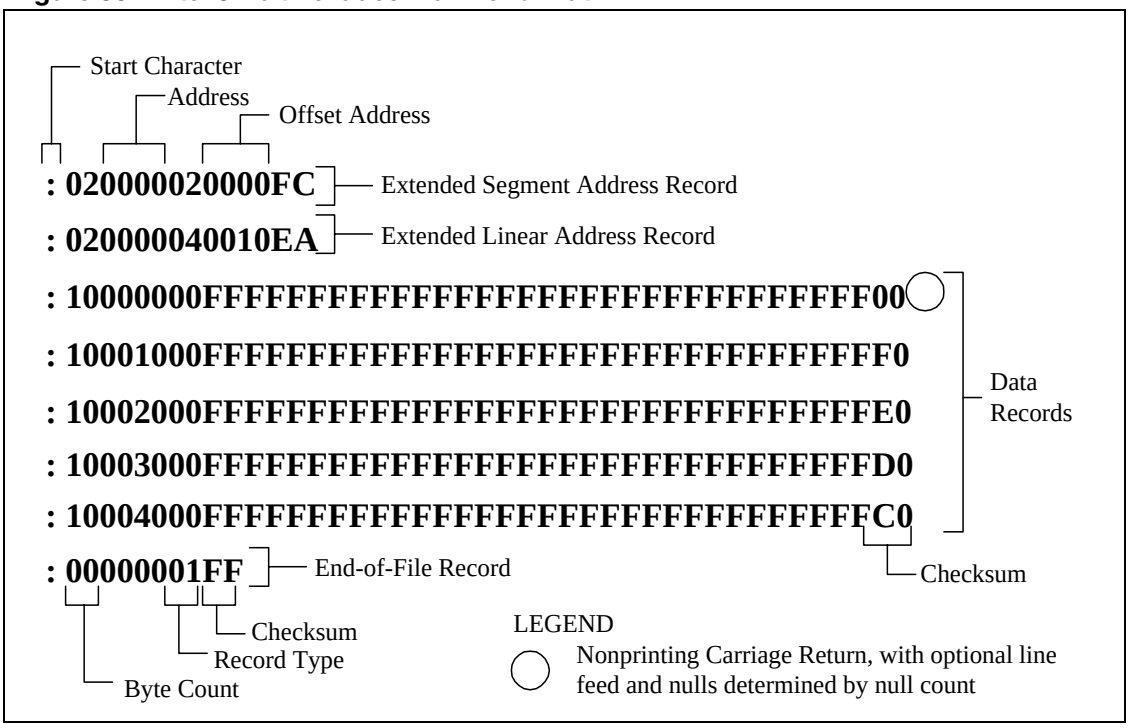
4.12.6 Assert/de-assert reset

Click **Assert Reset** to hold reset low. This toggle option provides a way to hold the reset signal low, at the user's discretion. When the option is selected, the reset signal is continuously asserted. Click **De-Assert Reset** to release the reset signal.

Appendix A Intel hex-32 record format

The Intel 32-bit hexadecimal file record format has a 9-character (4-field) prefix that defines the start of the record, byte count, load address, record type, and a 2-character checksum suffix. The *.hex* file (*Figure 55.* illustrates the sample records of this format).

Figure 55. Intel 32-bit hexadecimal file format



Four record types are defined:

- Data record
- End record
- Extended segment address record
- Extended linear address record

A.1 Data record

This record begins with the colon (:) start character, which is followed by the byte count (in hexadecimal notation), the address of the first data byte, and the record type (equal to “00”). Following these are the data bytes. The checksum follows the data bytes and is the two’s complement (in binary) of the preceding bytes in the record, including the byte count, address, record type, and data bytes.

A.2 End record

This end-of-file record also begins with the colon (:) start character and is followed by the byte count (equal to “00”), the address (equal to “0000”), the record type (equal to “01”), and the checksum, “FF.”

A.3 Extended segment address record

This is added to the offset to determine the absolute destination address. The address field for this record must contain ASCII zeros (hexadecimal 30s). This record type defines Bits 4 to 19 of the segment base address; it can appear randomly, anywhere within the file and affects the absolute memory address of subsequent data records in the file. The following example illustrates how the extended segment address is used to determine a byte address.

Problem:

Find the address for the first data byte for the following file:

```
:02 0000 04 0010 EA
:02 0000 02 1230 BA
:10 0045 00 55AA FF ..... BC
```

Solution:

1. Find the extended linear address offset for the data record (0010 in the example).
2. Find the extended segment address offset for the data record (1230 in the example).
3. Find the address offset for the data from the data record (0045 in the example).
4. Calculate the absolute address for the first byte of the data record as follows:

```
00100000  Linear address offset, shifted left 16 bits
+  12300  Segment address offset, shifted left 4 bits
+   0045  Address offset from data record

________
00112345  32-bit address for first data byte
```

The address for the first data byte is 112345.

Note: Always specify the address offset when using this format, even when the offset is zero.

During output translation, the firmware forces the record size to 16 (decimal) if the record size is specified greater than 16. There is no such limitation for record sizes specified less than 16.

A.4 Extended linear address record

This record specifies bits 16–31 of the destination address for the data records that follow. This address is added to the offset to determine the absolute destination address, and can appear anywhere within the file. The address field for this record must contain ASCII zeros (hexadecimal 30s).

Appendix B Project Report

This is an example of a project report. The report includes:

- CAPS version, project name, location, description, target device and entry mode
- Logic modules
- Memory map
- External chip select equations
- Logic equations
- User code and security settings
- I/O pin assignments
- Fitting results

Sample report:

```
********************************************************************************
* Project file generated by CAPS Version 1.00 - 3/20/2006 15:41:11
* Project Name           : project
* Project Folder         : C:\uPSDsoft\UPSD3434E-40\CAPS
* Project Description    :
* Target Device          : uPSD3434E-40U6
* Design Entry Mode      : Wizard mode
********************************************************************************
--------------------------------------------------------------------------------
System Memory Map
=================
  Main Flash memory will reside in this space at power-up      : Program Space Only
  Secondary Flash memory will reside in this space at power-up : Data Space Only

  fs0 = (address >= ^h0000) & (address <= ^h7FFF);
  fs1 = (page == 0) & (address >= ^h8000) & (address <= ^hFFFF);
  fs2 = (page == 1) & (address >= ^h8000) & (address <= ^hFFFF);
  fs3 = (page == 2) & (address >= ^h8000) & (address <= ^hFFFF);
  fs4 = (page == 3) & (address >= ^h8000) & (address <= ^hFFFF);
  fs5 = (page == 4) & (address >= ^h8000) & (address <= ^hFFFF);
  fs6 = (page == 5) & (address >= ^h8000) & (address <= ^hFFFF);
  fs7 = (page == 6) & (address >= ^h8000) & (address <= ^hFFFF);
  csboot0 = (address >= ^h8000) & (address <= ^h9FFF);
  csboot1 = (address >= ^hA000) & (address <= ^hBFFF);
  csboot2 = (address >= ^hC000) & (address <= ^hDFFF);
```

```
  csboot3 = (address >= ^hE000) & (address <= ^hFFFF);
  rs0 = (address >= ^h0000) & (address <= ^h1FFF);
  csiop = (address >= ^h7F00) & (address <= ^h7FFF);

  Assigned Firmware Files
  -----------------------
  Mapping mode : Direct

                File        File
  Memory Block  Start Address   End Address   Firmware File
  ------------  -------------   -----------   -------------
  fs0       0000        7FFF
  fs1       8000        FFFF
  fs2       8000        FFFF
  fs3       8000        FFFF
  fs4       8000        FFFF
  fs5       8000        FFFF
  fs6       8000        FFFF
  fs7       8000        FFFF
  csboot0   8000        9FFF
  csboot1   A000        BFFF
  csboot2   C000        DFFF
  csboot3   E000        FFFF
-------------------------------------------------------------------------------
External Chip-Select Equations
==============================
  N/A
-------------------------------------------------------------------------------
Additional Setting
==================

  Device Security Protection     : Off

  Sector Protection :
        Main Flash          Protection Status
        ----------          -----------------
        Sector 0            unprotected
```

```
        Sector 1            unprotected
        Sector 2            unprotected
        Sector 3            unprotected
        Sector 4            unprotected
        Sector 5            unprotected
        Sector 6            unprotected
        Sector 7            unprotected

        2nd  Flash          Protection Status
        ----------          -----------------
        Sector 0            unprotected
        Sector 1            unprotected
        Sector 2            unprotected
        Sector 3            unprotected
--------------------------------------------------------------------------------
I/O Pin Assignment
==================
  Pin Function                  Signal Name           Pin Number
  ------------                  -----------           ----------
  ALE output                    ale                   4
  Dedicated JTAG - TDO          tdo                   6
  Dedicated JTAG - TDI          tdi                   7
  JTAG debug pin                JTAG_debug_pin        8
  USB+ bus                      USB_plus              11
  USB- bus                      USB_minus             14
  Dedicated JTAG - TCK          tck                   17
  Dedicated JTAG - TMS          tms                   20
  Data/Address line             a0                    36
  Data/Address line             a1                    37
  Data/Address line             a2                    38
  Data/Address line             a3                    39
  Data/Address line             a4                    41
  Data/Address line             a5                    43
  Data/Address line             a6                    45
  Data/Address line             a7                    47
  Xtal1                         Xtal1                 48
  Xtal2                         Xtal2                 49
```

```
  Bus control output            _wr                 62
  Bus control output            _psen               63
  Bus control output            _rd                 65
  Reset In                      _Reset_In           68
  VREF input                    VREF                70


--------------------------------------------------------------------------------
Fitting Result
==============

                                          ----------------------------
                                         |                            |
                                         |1 ] pd2             adio4 [41| ad4
                                         |2 ] p3_3             p3_5 [42|
                                         |3 ] pd1             adio5 [43| ad5
                                    ale  |4 ] pd0              p3_6 [44| ad6
                                tdo, TDO |6 ] pc6/TDO          p3_7 [46| ad7
                           JTAG_debug_pin |8 ] debug           Xtal1 [48| Xtal1
                                         |9 ] pc4/TERR        Xtal2 [49| Xtal2
                                         |10] 3.3V VCC     5.0V VCC [50|
                             USB_plus    |11] USBp              N/C [51|
                                         |12] 5.0V VCC         p1_0 [52|
                                         |13] GND               N/C [53|
                            USB_minus    |14] USBm             p1_1 [54|
                                         |15] pc3/TSTAT         N/C [55|
                                         |16] pc2              p1_2 [56|
                              tck, TCK   |17] pc1/TCK           N/C [57|
                                         |18] p4_7             p1_3 [58|
                                         |19] p4_6             p1_4 [59|
                              tms, TMS   |20] pc0/TMS          p1_5 [60|
                                         |21] pa7              p1_6 [61|
                                         |22] pa6             cntl0 [62| _wr
                                         |23] p4_5             cntl2 [63| _psen
                                         |24] pa5              p1_7 [64|
                                         |25] p4_4             cntl1 [65| _rd
                                         |26] pa4               pb7 [66|
                                         |27] p4_3               pb6 [67|
```

```
                                          |28] pa3          Reset_In [68| _Reset_In
                                          |29] GND               GND [69|
                                          |30] p4_2             Vref [70| VREF
                                          |31] p4_1              pb5 [71|
                                          |32] pa2              AVcc [72|
                                          |33] p4_0              pb4 [73|
                                          |34] pa1               pb3 [74|
                                          |35] pa0              p3_0 [75|
              ad0, Address Bus a0/Data Port d0 |36] adio0            pb2 [76|
              ad1, Address Bus a1/Data Port d1 |37] adio1            p3_1 [77|
              ad2, Address Bus a2/Data Port d2 |38] adio2             pb1 [78|
              ad3, Address Bus a3/Data Port d3 |39] adio3            p3_2 [79|
                                          |40] p3_4              pb0 [80|
                                          |                           |
                                           -----------------------------

     ===== Resource Usage Summary =====

Total Product Terms Used:  15

Device Resources                    used / total
-----------------------------------------------
Port A: (pins 35 34 32 28 26 24 22 21)
I/O Pins :                            0   /  8
   GP I/O or Address Out        :    0
   Peripheral I/O               :    0
   Logic Inputs                 :    0
   Address Latch Inputs         :    0
   PT Dependent Latch Inputs    :    0
   PT Dependent Register Inputs :    0
   Combinatorial Outputs        :    0
   Registered Outputs           :    0
Other Information
   Microcells                   :    0   /  8
     Micro-Cells AB :
      Buried Microcells         :    0
      Output Microcells         :    0
   Product Terms                :    0   /  24
```

```
   Control Product Terms        :    0   /  34

Port B: (pins 80 78 76 74 73 71 67 66)
I/O Pins :                           0   /  8
   GP I/O or Address Out        :    0
   Logic Inputs                 :    0
   Address Latch Inputs         :    0
   PT Dependent Latch Inputs    :    0
   PT Dependent Register Inputs :    0
   Combinatorial Outputs        :    0
   Registered Outputs           :    0
Other Information
   Microcells                   :    0   /  8
     Micro-Cells AB :
      Buried Microcells         :    0
      Output Microcells         :    0
     Micro-Cells BC :
      Buried Microcells         :    0
      Output Microcells         :    0
   Product Terms                :    0   /  24
   Control Product Terms        :    0   /  34

Port C: (pins 20 17 16 15 9 7 6 5)
I/O Pins :                           4   /  8
   GP I/O or Address Out        :    0
   Logic Inputs                 :    0
   Address Latch Inputs         :    0
   PT Dependent Latch Inputs    :    0
   PT Dependent Register Inputs :    0
   JTAG signals                 :    4
   Standby Voltage Input        :    0
   Rdy/Bsy signal               :    0
   Standby On Indicator         :    0
   Combinatorial Outputs        :    0
   Registered Outputs           :    0
Other Information
   Microcells                   :    0   /  8
```

```
     Micro-Cells BC :
      Buried Microcells         :    0
      Output Microcells         :    0
   Product Terms                :    0   /  32
   Control Product Terms        :    0   /  34

Port D: (pins 4 3 1)
I/O Pins :                           1   /  3
   GP I/O or Address Out        :    0
   Logic Inputs                 :    0
   Chip-Select Input            :    0
   Clock Input                  :    0
   Control Signal Input         :    1
   Fast Decoding Outputs        :    0
Other Information
   Product Terms                :    0   /  3
   Control Product Terms        :    0   /  3


     ==== OMC Resource Assignment ====

  Resources           PT             User
  Used                Allocation     Name
 ----------------------------------------------------------
Micro-Cell AB :

Micro-Cell BC :

External Chip Select :


     ========= Equations =========

DPLD          EQUATIONS :
=======================
     fs0 = !pdn & !a15;
```

```
fs1 = !pdn & !pgr2 & !pgr1 & !pgr0 & a15;

fs2 = !pdn & !pgr2 & !pgr1 & pgr0 & a15;

fs3 = !pdn & !pgr2 & pgr1 & !pgr0 & a15;

fs4 = !pdn & !pgr2 & pgr1 & pgr0 & a15;

fs5 = !pdn & pgr2 & !pgr1 & !pgr0 & a15;

fs6 = !pdn & pgr2 & !pgr1 & pgr0 & a15;

fs7 = !pdn & pgr2 & pgr1 & !pgr0 & a15;

csboot0 = !pdn & a15 & !a14 & !a13;

csboot1 = !pdn & a15 & !a14 & a13;

csboot2 = !pdn & a15 & a14 & !a13;

csboot3 = !pdn & a15 & a14 & a13;

csiop = !pdn & !a15 & a14 & a13 & a12 & a11 & a10 & a9 & a8;

rs0 = !pdn & !a15 & !a14 & !a13;

jtagsel = !_reset;

PORTA         EQUATIONS :
=======================
PORTB         EQUATIONS :
=======================
PORTC         EQUATIONS :
=======================
PORTD         EQUATIONS :
=======================
                                --- End ---
```

Appendix C FlashLINK Cable – Install fast JTAG driver

Warning: Dual-Processor System or HyperThreading Enabled System: Please do NOT install the fast JTAG driver (JTD) for FlashLINK cable. The reason is that the JTD driver is not designed to handle code reentrancy as such it cannot support two processes at the same time. Refer to the workaround solution, described below.

C.1 Driver installation

To install the FlashLINK Cable fast JTAG driver (JTD),

1. Locate the “Drivers\FLink" folder under your CAPS folder.
2. At command prompt,
   a) To install the JTD driver
      `>>JTDINSTALL`
   b) To uninstall the JTD driver
      `>>JTDCFGW uninstall`
3. You must reboot the system after installing or uninstalling the driver.

Note: To achieve better programming performance in conjunction with the JTD driver, we recommend that you use the PCI parallel port card from SIIG: Cyber Parallel PCI Single Parallel Port

C.2 Workaround solutions

There are two workaround options for dual-processor systems, which are described below.

1. Boot into single processor mode and use the fast JTD driver.
2. Use the standard Parallel Port driver (PEP).

There are two workaround options for systems with hyper-thread enabled, which are described below.

1. Disable hyper-thread and use the fast JTD driver.
2. Leave hyper-thread as enabled and use the standard parallel port driver (PEP).

From JTAG/ISP standpoint, the tradeoff between the two solutions is the programming performance.

C.2.1 Dual-processor System: using solution 1

1. Boot your system in single processor mode and install JTD driver:
   If you have not already installed the JTD driver, go to the CAPS subfolder “Drivers\Flink” and invoke *JTDINSTALL.BAT*.
2. Reboot your system.

C.2.2 Dual-processor System: using solution 2

Follow the steps below to uninstall the JTD driver and install the PEP driver.

1. To uninstall the JTD, go to the command line prompt and type,

   `>> JTDCFGW uninstall`

   This command uninstalls the fast JTAG driver from your system.

   If you are unable to boot into Windows,

   a) press and hold F8 immediately during boot up.

   b) Select Safe Mode to skip loading the JTD driver.

      On systems that continue loading JTD driver, boot into Safe Mode with command prompt. Rename *JTD.sys* and *JTDMSG.dll* in the “WINDOWS\system32\drivers” folder to prevent loading the JTD driver upon bootup.

   c) Reboot into Normal mode and restore the above two file names before executing “`JTDCFGW uninstall`.”

2. If you have not already installed PEP driver, go to CAPS subfolder “Drivers\Needhams” and invoke *INSTALL.BAT*. If *INSTALL.BAT* does not exist, then go to your prompt command line prompt and follow the steps below:

   `>> Run regpep`

   `>> Copy pepnt.sys %windir%\system32\drivers\*.*`

3. Go to the CAPS folder and update *uPSDsoft.INI* as follows:

   `[JTAG]`

   `Driver=PEP` (indicates the PEP driver is used. OD indicates the JTD driver is used.)

4. Reboot your system.

C.2.3 Hyper-thread enabled system: using solution 1

Use the following steps to disable hyper-thread and use the fast JTD driver.

1. During power-on – Press F2 to enter CMOS setup.
2. Move to CPU Information.
3. Choose hyper-thread and toggle to disabled.
4. Save the configuration and continue with the boot process.

Follow the steps below to install JTD driver:

1. If you have not already intalled JTD driver, go to CAPS subfolder “Drivers\Flink” and invoke *JTDINSTALL.BAT*.
2. Reboot your system.

C.2.4 Hyper-thread enabled system: using solution 2

Use the following steps to leave hyper-thread enabled and use the standard parallel port driver (PEP). (Uninstall the JTD driver and install the PEP driver).

1. To uninstall the JTD, enter the following at the command line prompt:

   `>> JTDCFGW uninstall`

   If you are unable to boot into Windows,

   a) press and hold F8 immediately during boot up.

   b) Select Safe Mode to skip loading the JTD driver.

      On systems that continue loading JTD driver, boot into Safe Mode with command prompt. Rename *JTD.sys* and *JTDMSG.dll* in the “WINDOWS\system32\drivers” folder to prevent loading the JTD driver upon bootup.

c) Reboot into Normal mode and restore the above two file names before executing "`JTDCFGW uninstall`."

2. If you have not already installed the PEP driver, go to the CAPS subfolder "Drivers\Needhams" and invoke *INSTALL.BAT*. If *INSTALL.BAT* does not exist, then enter the following commands at the command prompt.

   ```
   >> Run regpep
   >> Copy pepnt.sys %windir%\system32\drivers\*.*
   ```

3. Go to the CAPS folder and update *uPSDsoft.INI* as follows,

   ```
   [JTAG]
   Driver=PEP <<< this is to indicate PEP driver is used. (OD indicates JTD driver is used)
   ```

4. Reboot your system with hyper-thread enabled.

5 Revision history

Date	Revision	Changes
5-Jun-2006	1	Initial release
9-Aug-2006	2	Installation procedure updated, *Section 1.1.2 on page 4* Procedure updated for validating and programming target device, *Section 4.12 on page 50*

Please Read Carefully:

Information in this document is provided solely in connection with ST products. STMicroelectronics NV and its subsidiaries (“ST”) reserve the right to make changes, corrections, modifications or improvements, to this document, and the products and services described herein at any time, without notice.

All ST products are sold pursuant to ST’s terms and conditions of sale.

Purchasers are solely responsible for the choice, selection and use of the ST products and services described herein, and ST assumes no liability whatsoever relating to the choice, selection or use of the ST products and services described herein.

No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted under this document. If any part of this document refers to any third party products or services it shall not be deemed a license grant by ST for the use of such third party products or services, or any intellectual property contained therein or considered as a warranty covering the use in any manner whatsoever of such third party products or services or any intellectual property contained therein.

UNLESS OTHERWISE SET FORTH IN ST’S TERMS AND CONDITIONS OF SALE ST DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY WITH RESPECT TO THE USE AND/OR SALE OF ST PRODUCTS INCLUDING WITHOUT LIMITATION IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE (AND THEIR EQUIVALENTS UNDER THE LAWS OF ANY JURISDICTION), OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

UNLESS EXPRESSLY APPROVED IN WRITING BY AN AUTHORIZED ST REPRESENTATIVE, ST PRODUCTS ARE NOT RECOMMENDED, AUTHORIZED OR WARRANTED FOR USE IN MILITARY, AIR CRAFT, SPACE, LIFE SAVING, OR LIFE SUSTAINING APPLICATIONS, NOR IN PRODUCTS OR SYSTEMS WHERE FAILURE OR MALFUNCTION MAY RESULT IN PERSONAL INJURY, DEATH, OR SEVERE PROPERTY OR ENVIRONMENTAL DAMAGE. ST PRODUCTS WHICH ARE NOT SPECIFIED AS "AUTOMOTIVE GRADE" MAY ONLY BE USED IN AUTOMOTIVE APPLICATIONS AT USER’S OWN RISK.

Resale of ST products with provisions different from the statements and/or technical features set forth in this document shall immediately void any warranty granted by ST for the ST product or service described herein and shall not create or extend in any manner whatsoever, any liability of ST.

ST and the ST logo are trademarks or registered trademarks of ST in various countries.

Information in this document supersedes and replaces all information previously supplied.

The ST logo is a registered trademark of STMicroelectronics. All other names are the property of their respective owners.

STMicroelectronics group of companies

Australia - Belgium - Brazil - Canada - China - Czech Republic - Finland - France - Germany - Hong Kong - India - Israel - Italy - Japan - Malaysia - Malta - Morocco - Singapore - Spain - Sweden - Switzerland - United Kingdom - United States of America

www.st.com