Operation/Reference Guide

RMS CodeCrafter

v3.2 or higher

Enterprise Applications

Last Revised: 9/22/2010

Table of Contents

Welcome to RMS CodeCrafter

Overview

RMS CodeCrafter is a Microsoft Windows application that offers accelerated development of the NetLinx® include (.axi) file required to provide code for monitoring devices using one of the applications in the AMX Resource Management Suite®. RMS CodeCrafter provides a wizard interface that allows the user to enter data about the devices to be monitored and desired parameters and options.

Features

- Generates NetLinx code for NetLinx systems
- Persists input data to an RMS CodeCrafter project, so that code files can be regenerated without additional user input
- Ability to create reusable device template files for frequently used devices
- Ability to import an existing RMS Device Monitoring Worksheet into a new RMS CodeCrafter project
- Ability to transfer a SERVERINFO.TXT file containing the RMS Server IP (or hostname) to a NetLinx Master Controller via FTP
- Offers Office XP(R) and Office 2003(R) visual styles
- Integrated with AMX WebUpdate

System Requirements

RMS CodeCrafter supports the following platforms:

- Windows 2000 (Service Pack 2 or greater)

NOTE

If you are installing on a Windows 2000 machine, you must have Administrator rights to install and run all required System files.

- Windows XP Professional (Service Pack 1 or greater)

Other PC requirements:

- Windows-compatible CD-ROM drive.
- Windows-compatible mouse (or other pointing device).
- At least 250 MB of free disk space.
- RMS CodeCrafter will not work properly if your display settings is set to *Large Font*.

The RMS CodeCrafter Work Area

File Menu

The File menu serves as a location for file management, project information and template management.

New:	• New Project - Opens a new RMS CodeCrafter project file (.CGP) and progresses to the Project Start Page. • New Template - Opens a new RMS CodeCrafter template file (.CGT) and progresses to the Device Template Main Page.
Open:	• Open Project - After selecting the desired file in the open dialog window, it opens an existing RMS CodeCrafter project file (.CGP) and progresses to the Project Start Page. • Open Template - After selecting the desired file in the open dialog window, it opens an existing RMS CodeCrafter template file (.CGT) and progresses to the Device Template Main Page. • Import Spreadsheet - Earlier releases of RMS SDK utilized an Excel spreadsheet that RMS CodeCrafter now replaces. Selecting this option opens the Import spreadsheet window.
Close:	Closes the project or template file currently open. If the file has been edited, or is "dirty" as indicated at the top of the window with an asterisks next to the file name, you are prompted to save changes before closing.
Save:	Saves changes made to either project or template file currently open. If the file has not been previously been saved you are prompted to name and designate where to save the file.
Save As:	Opens the save file dialog, allowing you to specify the file name and location which to save.
Transfer Server Address File:	Opens the SERVERINFO.TXT window.
Most Recently Used List:	A list of most recently opened file names.
Exit:	Closes the RMS CodeCrafter Wizard.

Edit Menu

The Edit Menu applies to text changes made within the RMS CodeCrafter wizard.

Undo:	Undoes action taken in text areas.
Cut:	Cut highlighted text from fields.
Copy:	Copy highlighted text within fields.
Paste:	Paste cut and copied text into selected fields.
Delete:	Delete selected text within fields.
Preferences:	Launches the Preferences window.

View Menu

The View menu allows you to toggle the toolbar and status bar of the RMS CodeCrafter wizard on and off. A check next to the title indicates it is visible. Additionally you can change the appearance of the application itself.

Visual Style:	A list of visual appearances for the wizard. Select one of the following: • Default • Office XP® • Office 2003® • Windows XP® Theme

Help Menu

The Help menu contains features for updating the application and provides a resource for understanding the RMS CodeCrafter wizard.

Contents:	Launches the RMS CodeCrafter help, Contents section.
Web Update:	Launches the AMX Web Update utility (requires Internet connection).
About RMS CodeCrafter:	Displays the about RMS CodeCrafter dialog containing version and copyright information.

Toolbar

The toolbar contains icons for the commonly used functions:

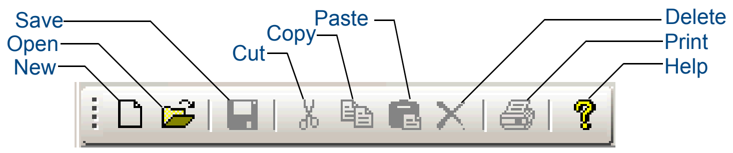


FIG. 1 RMS CodeCrafter Wizard Toolbar

New:	Launches the Project Start page.
Open:	Launches the open project dialog window.
Save:	Saves changes made to either project or template file currently open. If the file has not been previously been saved you are prompted to name and designate where to save the file.
Cut:	Cut highlighted text from fields.
Copy:	Copy highlighted text within fields.
Paste:	Paste cut and copied text into selected fields.
Delete:	Delete selected text within fields.
Print:	Prints the active generated code.
Help:	Launches the RMS CodeCrafter help file.

Text Field Context Menu

Right click within any text field to access the text field context menu.

Undo:	Undoes last text action.
Cut:	Cuts selected text to the clipboard.
Copy:	Copies selected text to the clipboard.
Paste:	Pastes text from clipboard into text field.
Delete:	Deletes selected text.
Select All:	Selects all text within the field.

RMS CodeCrafter Device Template Pages

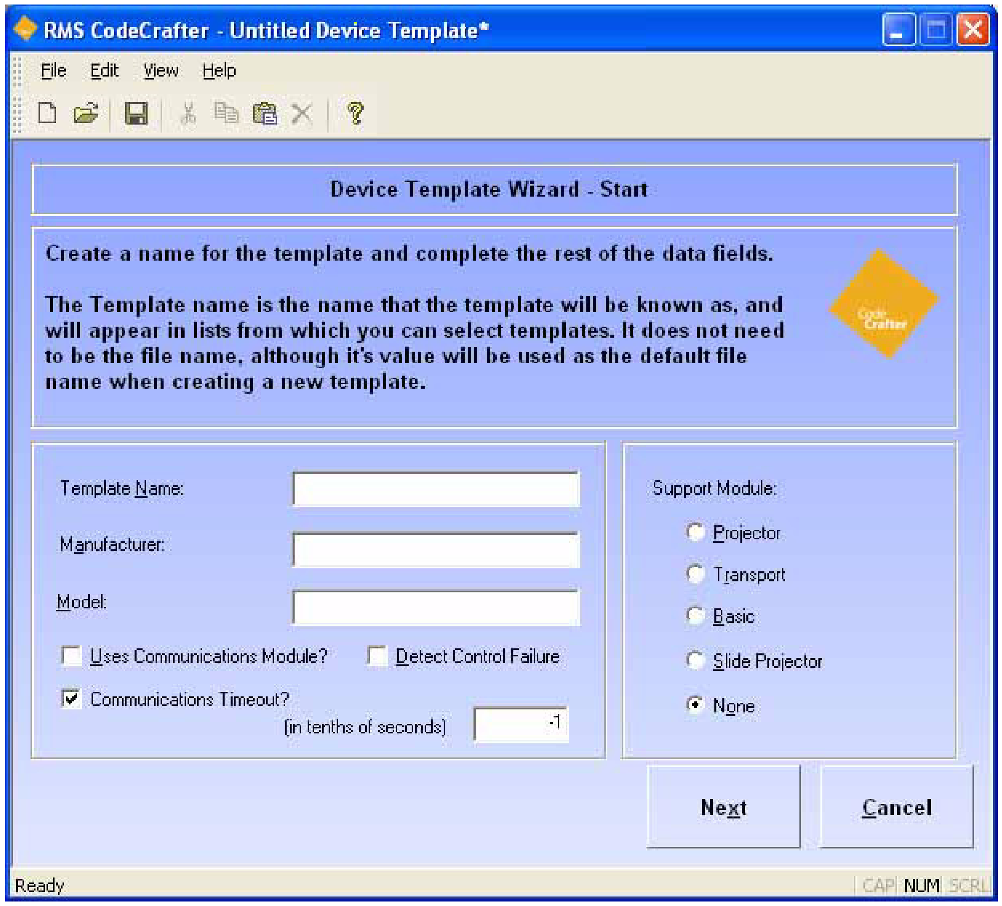
Device Templates

You can create persistent templates for monitored devices, where you provide the settings and parameters. RMS CodeCrafter can then use the device templates for similar or the same devices in other projects.

When you opt to use device templates RMS CodeCrafter pulls all available templates from the template folder destination, which you can set in the Preferences window. You can create a new template or open an existing template from within the File menu, but either way you are brought to the Device Template Main page to edit and set parameters.

Device Template Main Page

The Device Template Main page is the first page of the template creation process.


RMS CodeCrafter - Untitled Device Template*
File
Edit
View
Help
Device Template Wizard - Start
Next
Cancel
Ready
CAP
NUM
SCRL

Create a name for the template and complete the rest of the data fields.

The Template name is the name that the template will be known as, and will appear in lists from which you can select templates. It does not need to be the file name, although it's value will be used as the default file name when creating a new template.

Template Name:

Manufacturer:

Model:

☐ Uses Communications Module? ☐ Detect Control Failure

☑ Communications Timeout?
(in tenths of seconds) -1

Support Module:
- Projector
- Transport
- Basic
- Slide Projector
- [✓ None]

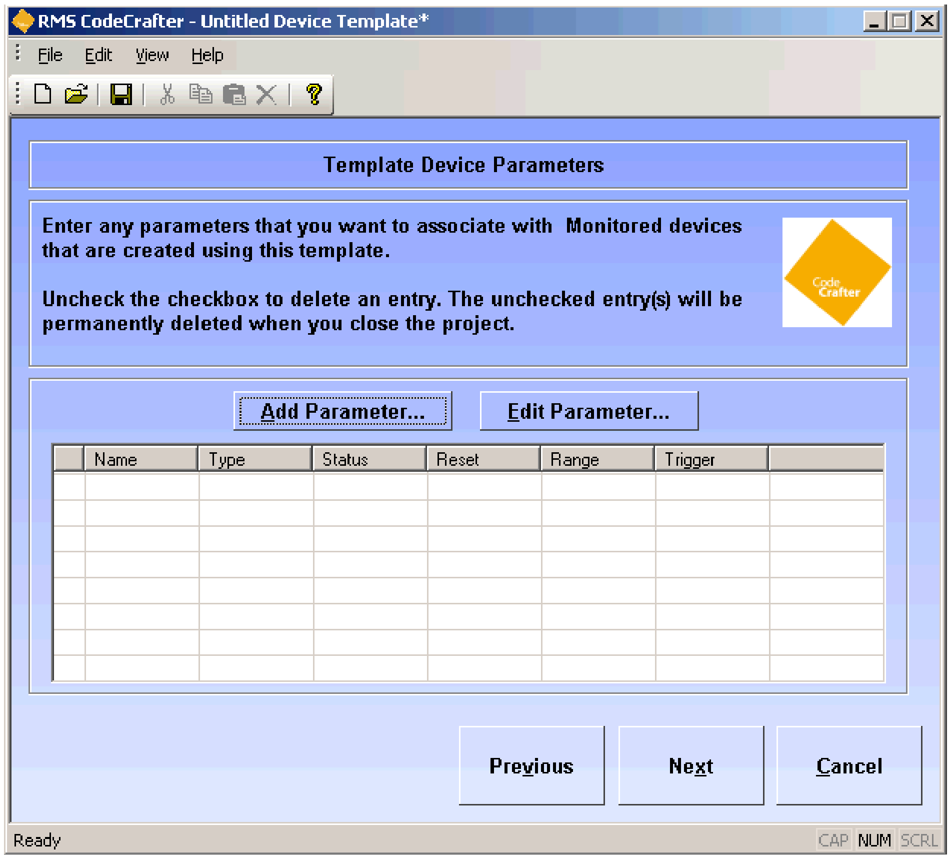
FIG. 1 Device Template Main Page

Device Template Main Page	
Template Name:	This is the template name provided by you. This is not the file name but by default it is used as such when saving. Template name is displayed as a selection when you are associating new monitored devices with an existing template.
Manufacturer:	This is a text field where you can provide the manufacturer information for the device.
Model:	This is a text field where you can provide the model information for the device.
Uses Communications Module?:	Place a check in the box to enable the function. The template will use a communication module.

Device Template Main Page (Cont.)	
Communication Timeout:	Place a check in the box to enable the function. • Measured in 1/10ths of a second, where 0 is a valid value. • Default is 300.
Control Failure Detect:	Placing a check in the box adds a call to RMSEnablePowerFailure() in the generated code file.
Support Module:	Click the radio button that best represents the device for the template: • Projector • Transport • Basic • Slide Projector None
Next:	Progresses the wizard to the *Device Template Parameters* page.

Device Template Parameters Page

The Device Parameters page is a list of available parameters that can be included with the template. There is a grid displaying all device parameters currently within the template. In the left-hand column is a checkbox which, when selected, indicates that the entry is to be deleted.



Enter any parameters that you want to associate with Monitored devices that are created using this template.

Uncheck the checkbox to delete an entry. The unchecked entry(s) will be permanently deleted when you close the project.

FIG. 2 Device Template Parameters Page

Device Template Parameters Page	
Add Parameter:	Add a parameters entry.
Edit Parameter:	Edit a selected parameters entry.
Next:	Progresses to the *Device Template Final* page.
Previous:	Regresses to the *Device Template Main* page.

Add/Edit Device Template Parameters Window

The Add/Edit Device Template Parameters window allows you to add device parameters or edit existing ones.

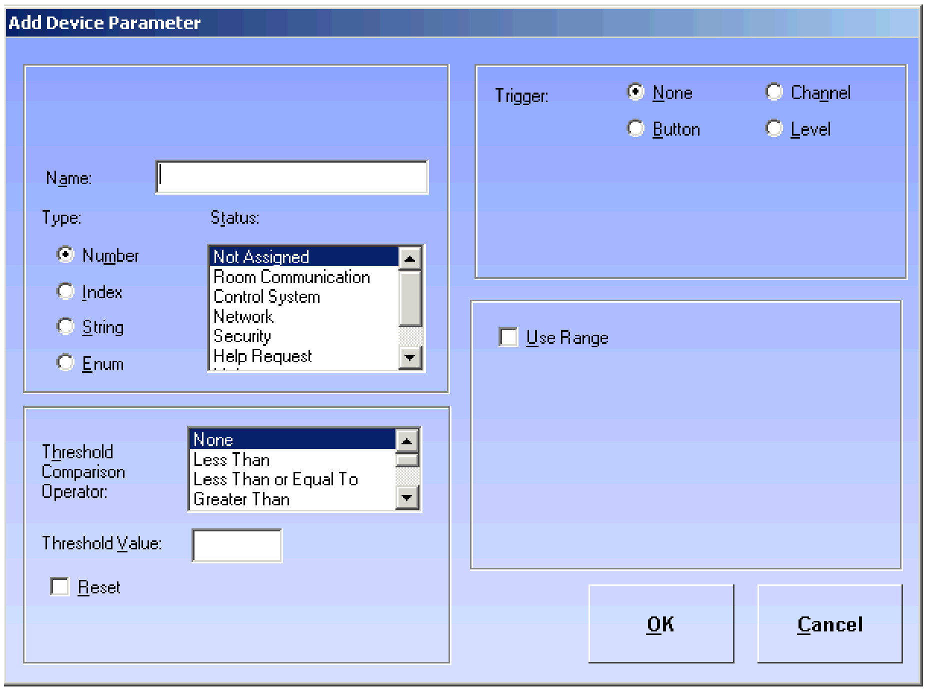


FIG. 3 Add/Edit Device Template Parameters Window

Add/Edit Device Template Parameters Window	
Name:	A text field you can edit to name the parameter.
Type:	Select one of the parameter types: • Number • Index • String • Enum
Status:	select one from the scroll list: • Not Assigned • Room Communication • Control System • Network • Security • Help Request • Maintenance • Equipment Usage
Threshold Comparison Operator:	select from the following: • None • Less Than • Less Than or Equal To • Greater Than • Greater Than or Equal To • Equal To • Not Equal To • Contains (String or Enum only) • Does Not Contain (String or Enum only)
Threshold Value:	A text field you can edit to set the threshold value. 32-bit signed number for Number, 16-bit unsigned number for Index and any string up to 100 characters for String and Enum. • Reset - Place a mark in the box to enable reset of threshold.

Add/Edit Device Template Parameters Window (Cont.)	
Reset Value:	A text field you can edit to set reset value. 32-bit signed number for Number, 16-bit unsigned number for Index and any string up to 100 characters for String and Enum.
Trigger:	select one of the following: • None • Button • Channel • Level
Use Range (Number):	Sets the minimum and maximum for number; this value is optional.
Use Units (Number or String):	Defines the unit type for the parameter, e.g., minutes, hours, degrees, or %. This value is optional.
Set Range (Enum or Index):	A drop down list of enumeration strings. You can elect to: • Add - adds the sting to the parameter use. • Remove - removes the string from parameter use. • Move Up - moves the string up the list of parameter use. • Move Down - moves the string down the list of parameter use.
OK:	Closes the window and saves changes made.
Cancel:	Closes window without implementing changes.

Set Range for Index Parameter

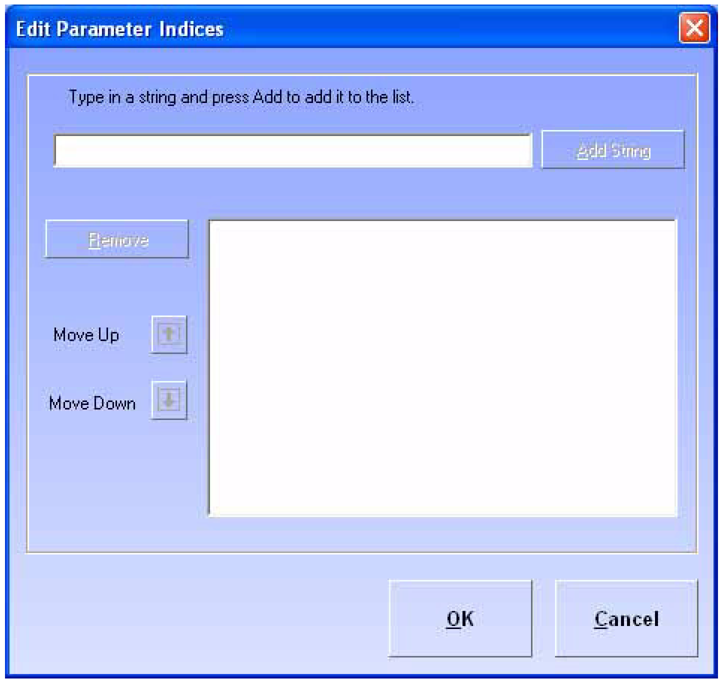


FIG. 4 Edit Parameter Indices dialog

Edit Parameter Indices dialog	
Add string:	adds the sting to the parameter use.
Remove:	removes the string from parameter use.
Move Up:	moves the string up the list of parameter use.
Move Down:	moves the string down the list of parameter use.
OK:	Closes the window and saves changes made.
Cancel:	Closes window without implementing changes.

Set Range for Enum Parameter

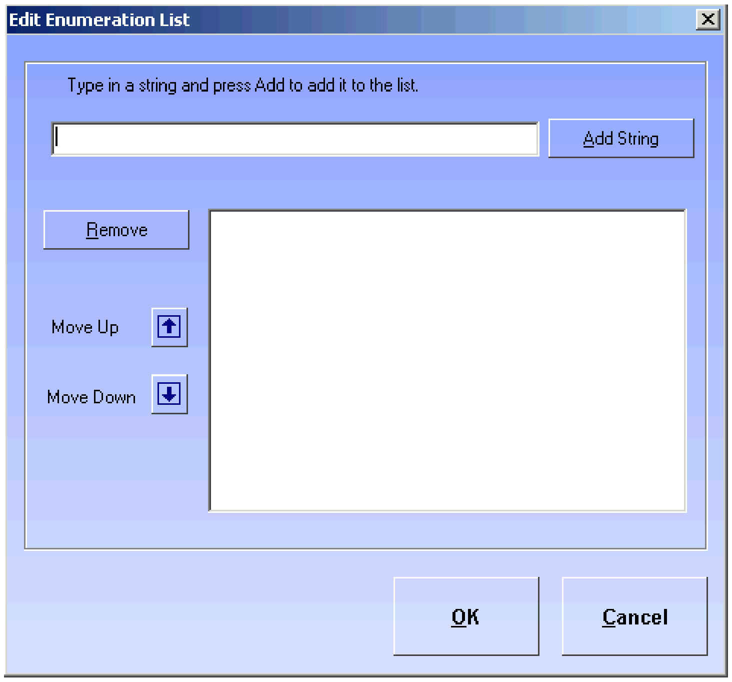
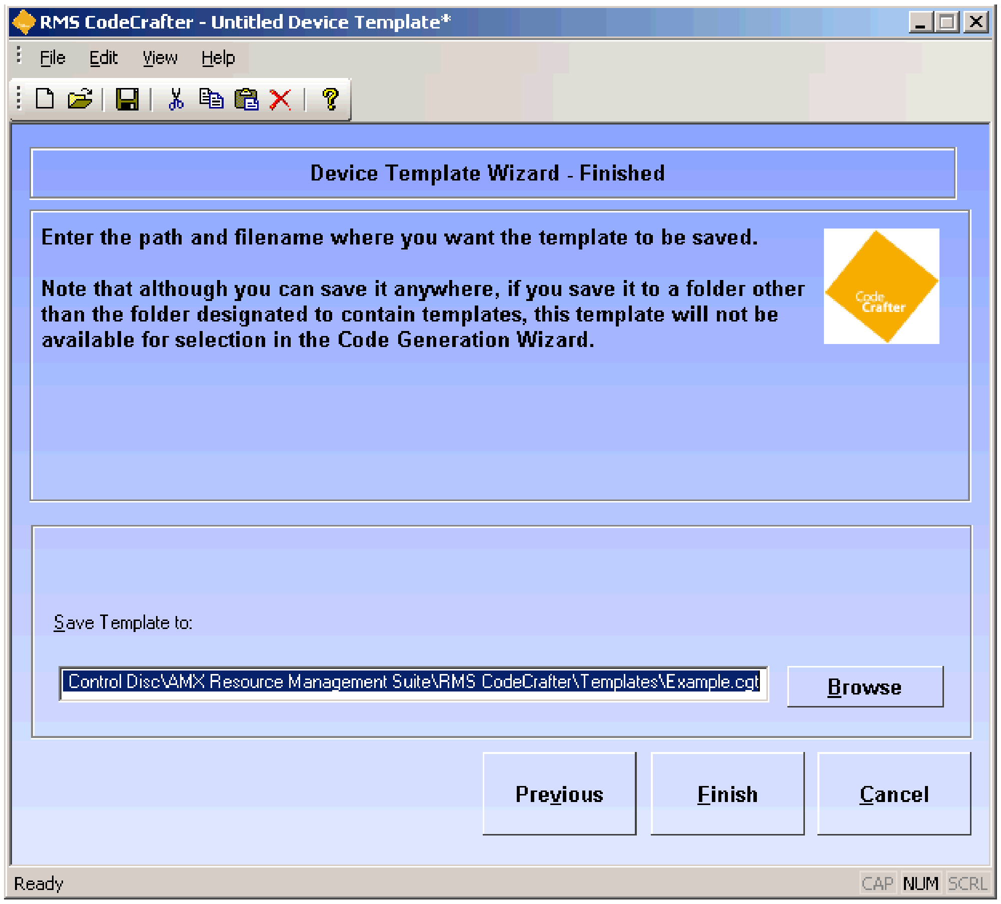


FIG. 5 Edit Numeration List dialog

Edit Enumeration List dialog	
Add string:	adds the sting to the parameter use.
Remove:	removes the string from parameter use.
Move Up:	moves the string up the list of parameter use.
Move Down:	moves the string down the list of parameter use.
OK:	Closes the window and saves changes made.
Cancel:	Closes window without implementing changes.

Device Template Wizard - Finished Page

The Device Template Wizard Finished page is the last step in making a device template.



Enter the path and filename where you want the template to be saved.

Note that although you can save it anywhere, if you save it to a folder other than the folder designated to contain templates, this template will not be available for selection in the Code Generation Wizard.

Save Template to: Control Disc\AMX Resource Management Suite\RMS CodeCrafter\Templates\Example.cgt

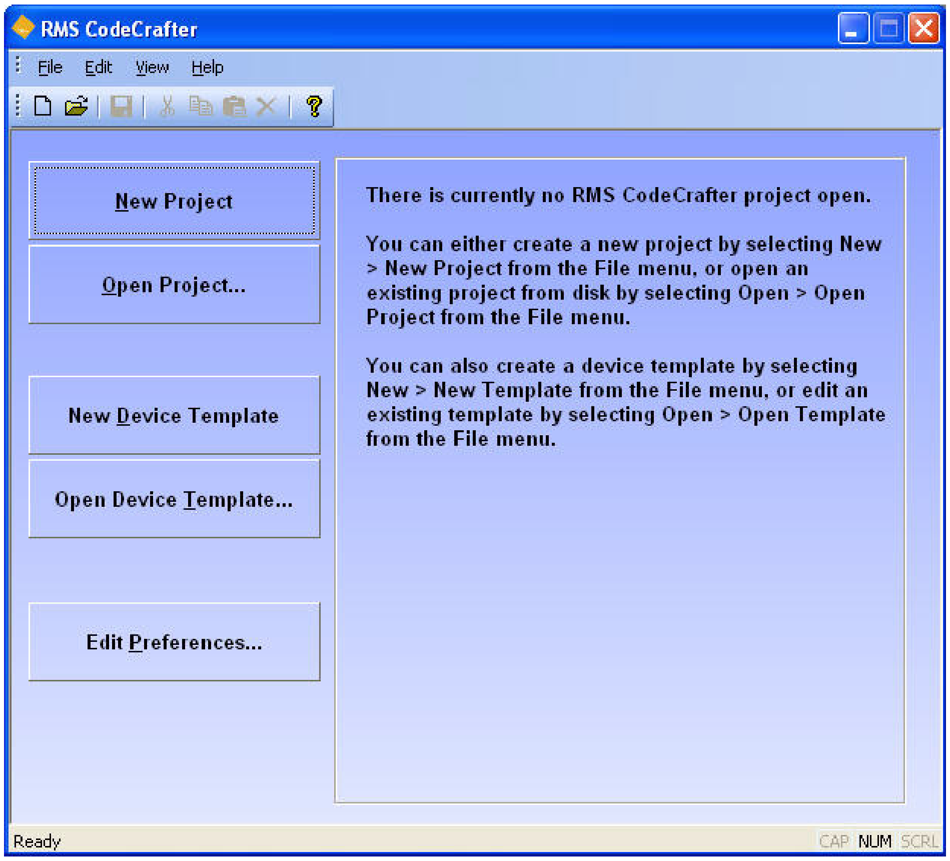
FIG. 6 Device Template Wizard - Finished Page

Edit Enumeration List dialog	
Save Template to:	A text field where you can either type or click the Browse button to designate where RMS CodeCrafter wizard is to save the template file (.CGT).
Previous:	Regresses the RMS CodeCrafter wizard to the Device Template Parameters Page.
Finish:	Generates the template file and stores in the directory specified by the Save Template to field.

RMS CodeCrafter Project Pages

Introductory Page

The *Introductory* page is the initial page displayed upon launching the RMS CodeCrafter wizard.



There is currently no RMS CodeCrafter project open.

You can either create a new project by selecting New > New Project from the File menu, or open an existing project from disk by selecting Open > Open Project from the File menu.

You can also create a device template by selecting New > New Template from the File menu, or edit an existing template by selecting Open > Open Template from the File menu.

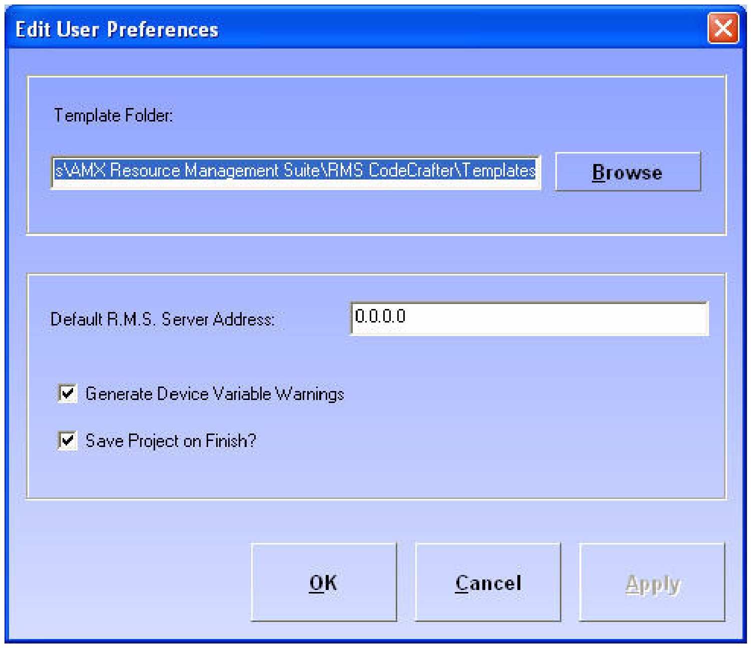
FIG. 1 Introductory Page

The button you select will determine what file is generated.

Introductory Page	
New Project:	Opens a new RMS CodeCrafter project file (.CGP) and progresses to the *Project Start Page*.
Open Project:	After selecting the desired file in the open dialog window, opens an existing RMS CodeCrafter project file (.CGP) and progresses to the *Project Start Page*.
New Device Template:	Opens a new RMS CodeCrafter template file (.CGT) and progresses to the *Device Template Main Page*.
Open Device Template:	After selecting the desired file in the open dialog window, opens an existing RMS CodeCrafter template file (.CGT) and progresses to the *Device Template Main Page*.
Edit Preferences:	Launches the *Edit User Preferences* dialog, where you can set default features of the wizard.

Edit User Preferences dialog

The options in this dialog allow you to set user preferences.



Template Folder: s\AMX Resource Management Suite\RMS CodeCrafter\Templates

Default R.M.S. Server Address: 0.0.0.0

☑ Generate Device Variable Warnings

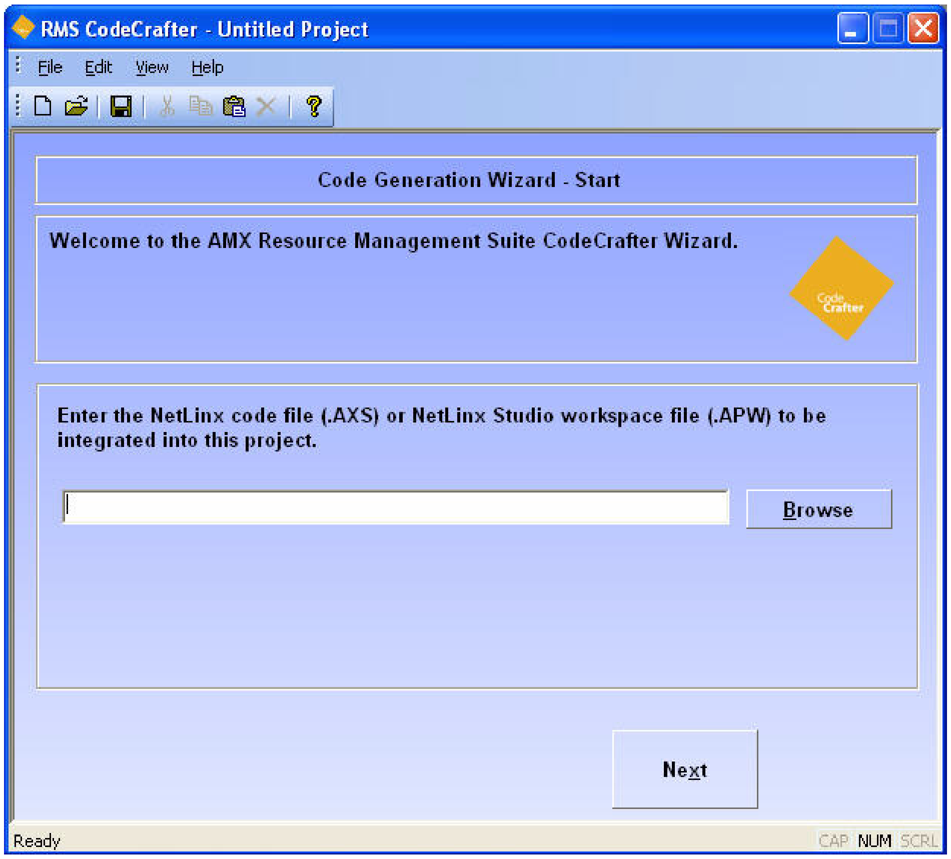
☑ Save Project on Finish?

FIG. 2 Edit User Preferences dialog

Edit User Preferences dialog	
Template Folder:	A text field where you can either type or click Browse to designate the default template folder destination.
Default RMS Server Address:	A text field where you can edit the server address for the RMS server. The default value is 0.0.0.0.
Generate Device Variable Warnings:	A checkbox, when enabled adds variable warnings in the generated code.
Save Project on Finish?	A checkbox, when enabled automatically saves your project when you click the final *Finish* button.
OK:	Accepts changes and closes this dialog.
Cancel:	Closes this dialog without incorporating changes.

Code Generation Wizard - Start Page

The *Code Generation Wizard - Start* page is the first page you see when you open a new or existing project file.


RMS CodeCrafter - Untitled Project
File Edit View Help
Code Generation Wizard - Start
Browse
Next
Ready
CAP NUM SCRL

Welcome to the AMX Resource Management Suite CodeCrafter Wizard.

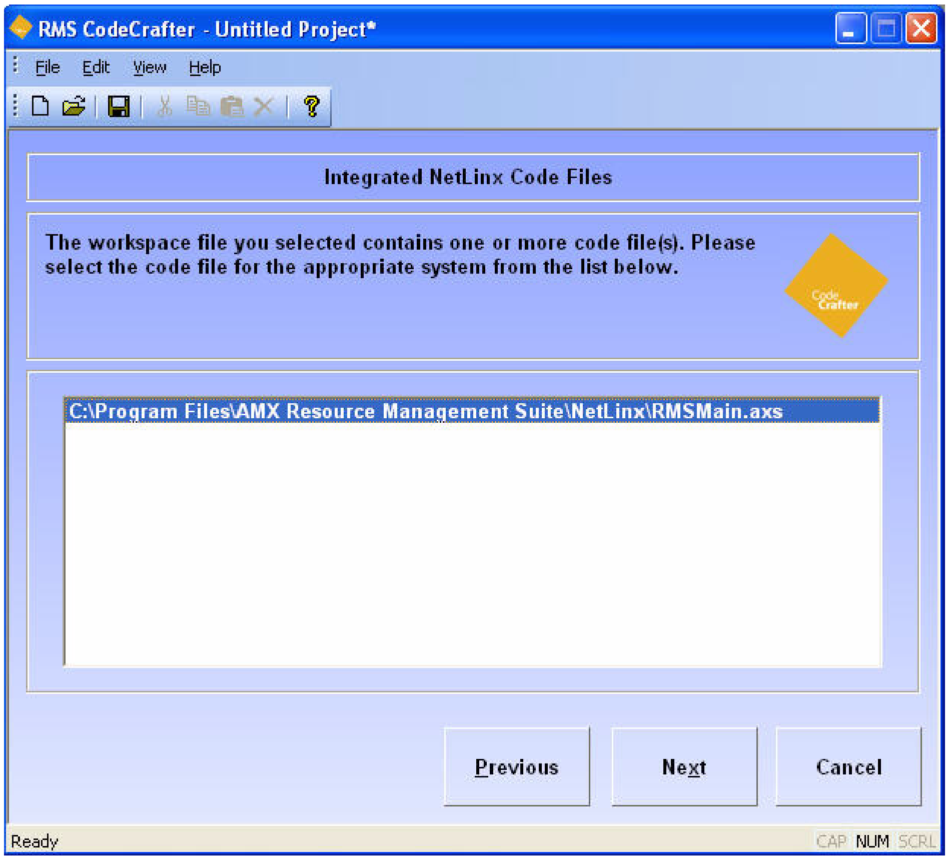
Enter the NetLinx code file (.AXS) or NetLinx Studio workspace file (.APW) to be integrated into this project.

FIG. 3 Code Generation Wizard - Start Page

Code Generation Wizard - Start Page	
Enter the NetLinx code file (.AXS) or NetLinx Studio workspace file (.APW) to be integrated into this project.	Either type or click the Browse button for the location of the file to be integrated into the project. • The file can be either a NetLinx code (.AXS) or NetLinx Studio workspace (.APW) file. • The incorporation of either file into a RMS CodeCrafter project is optional.
Next:	Progress to the next page.

Integrated NetLinx Code Files Page

You will see the Code File Selection page if you selected a NetLinx Studio workspace file (.APW) in the Project Start page. A list of all available NetLinx code files within the .APW file is displayed.


RMS CodeCrafter - Untitled Project*
File
Edit
View
Help
Integrated NetLinx Code Files

The workspace file you selected contains one or more code file(s). Please select the code file for the appropriate system from the list below.

C:\Program Files\AMX Resource Management Suite\NetLinx\RMSMain.axs

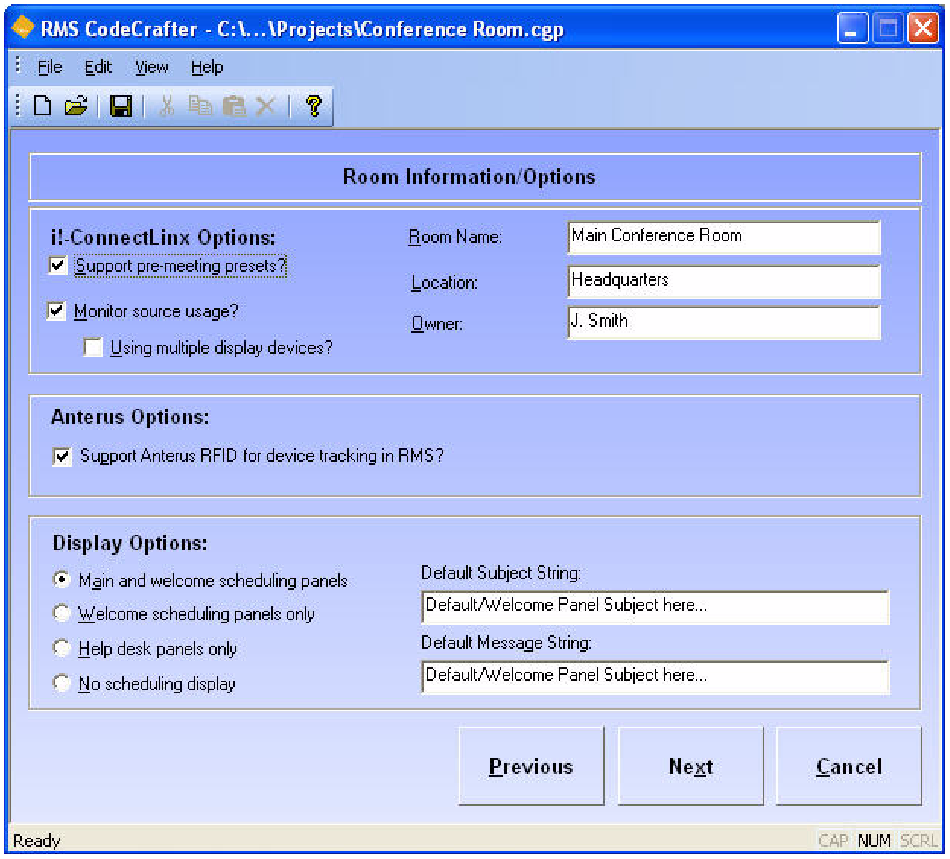
Previous
Next
Cancel
Ready
CAP NUM SCRL

FIG. 4 Integrated NetLinx Code Files Page

Integrated NetLinx Code Files Page	
Previous:	Returns the wizard back to the *Project Start* page.
Next:	Progresses to the *Room Information/Options* page.

Room Information/Options Page

The options in this age allow you to add scheduling and i!-ConnectLinx options to a project file.


RMS CodeCrafter - C:\...\Projects\Conference Room.cgp

Room Information/Options

i!-ConnectLinx Options:

- [✓] Support pre-meeting presets?
- [✓] Monitor source usage?
  - [ ] Using multiple display devices?

Room Name: Main Conference Room

Location: Headquarters

Owner: J. Smith

Anterus Options:

- [✓] Support Anterus RFID for device tracking in RMS?

Display Options:

- ● Main and welcome scheduling panels
- ○ Welcome scheduling panels only
- ○ Help desk panels only
- ○ No scheduling display

Default Subject String: Default/Welcome Panel Subject here...

Default Message String: Default/Welcome Panel Subject here...

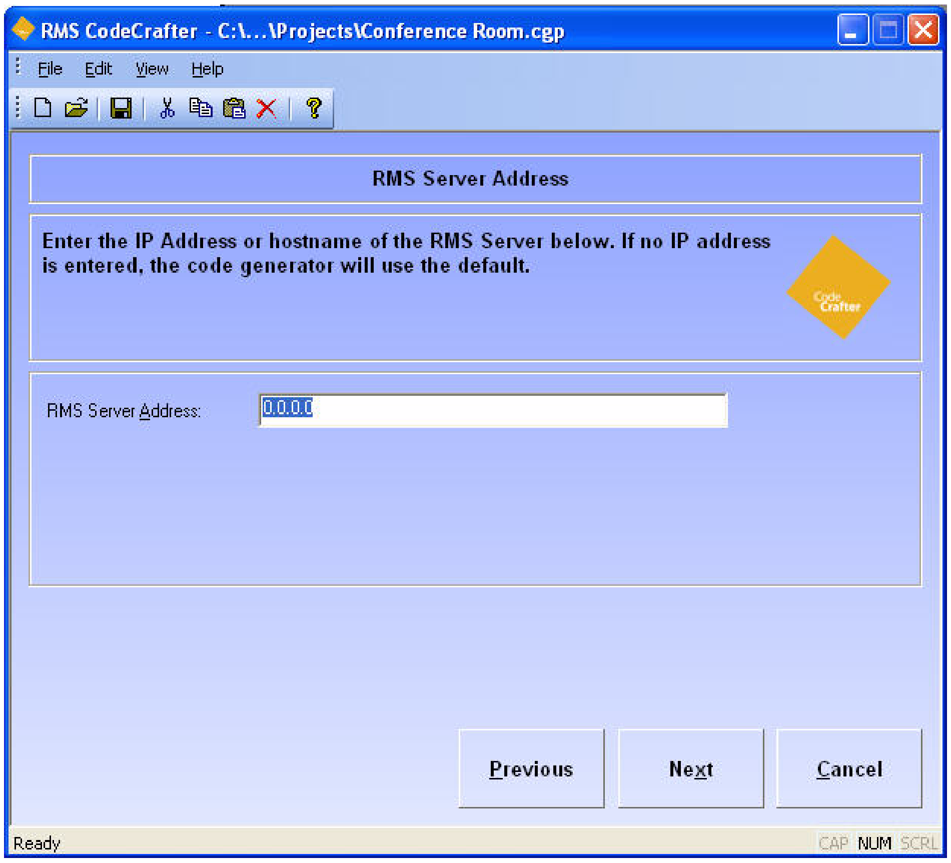
FIG. 5 Room Information/Options Page

Room Information/Options Page	
i!-ConnectLinx Options:	
Support pre-meeting presets?	Select this option if the i!-ConnectLinx module is defined in the generated code file.
Monitor source usage?	Select this option if the RMSSourceUsageMod is defined in the generated code file.
Using multiple display devices?	Select this option if a call to RMSSetMultiSource(TRUE) is included in the DEFINE_EVENT section of the generated code. This option is only available if *Monitor source usage?* has been selected.
Room Name:	Enter the name of the room in this text field (*required*).
Location:	Enter the name of the room's location in this text field (*required*).
Owner:	Enter the name of the room's owner in this text field (*required*).
Anterus Options:	
Support Anterus RFID for device tracking in RMS?	Select this option if AMX Anterus RFID Solution support will be defined in the generated code. RMS SDK version 3.2.x (or greater) is required to support the AMX Anterus RFID Solution. CodeCrafter will check against the detected installed version of the RMS SDK first to determine if version RMS 3.2.x (or greater) is available. If an older version of the SDK is installed, CodeCrafter displays a message indicating that Anterus RFID tracking is only available in RMS v3.2 or later. To enable RFID tracking, you must first install RMS SDK v3.2 or later.

Room Information/Options Page (Cont.)	
Display Options:	
Main and welcome scheduling panels:	If selected, the RMSUIMod is defined in the generated code file. Additionally, the array of this module is dependant upon an input argument defined.
Welcome scheduling panels only:	If selected, only the RMSWelcomeOnlyUIMod is defined in the generated code. Additionally, the array of Welcome Panels is defined.
Help desk panels only:	If selected, only the RMSHelpUIMod is defined in the generated code.
No scheduling display:	If selected, no scheduling mods are included in the generated code.
Default Subject String:	Sets a default subject for the welcome panel. This option is only available if either *Main and welcome scheduling panels* or *Welcome scheduling panels only* is selected in the *Display Options*.
Default Message String:	Sets a default message for the welcome panel. This option is only available if either *Main and welcome scheduling panels* or *Welcome scheduling panels only* is selected in the *Display Options*.
Previous:	Regresses to the *Code File Selection* page.
Next:	Progresses to the *RMS Server Address* page.

RMS Server Address Page

The RMS Server Address page allows you the opportunity to specify the IP address of the RMS server. This is optional and if left unchanged, the default specified in the Preferences window is used.



Enter the IP Address or hostname of the RMS Server below. If no IP address is entered, the code generator will use the default.

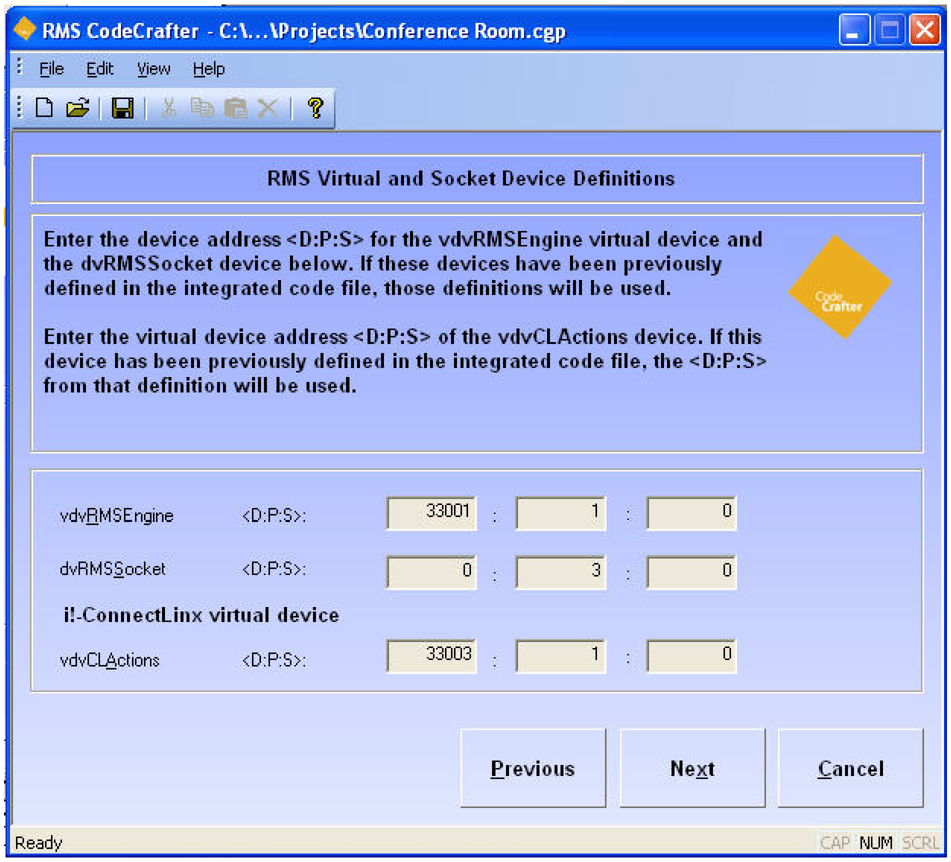
RMS Server Address: 0.0.0.0

FIG. 6 RMS Server Address Page

RMS Server Address Page	
RMS Server Address:	Enter the IP address of the RMS server if necessary in this text field.
Previous:	Regresses to the *Room Information/Options* page.
Next:	Progresses to the *RMS Virtual and Socket Device Definitions* page.

RMS Virtual and Socket Device Definitions Page

The *RMS Virtual and Socket Device Definitions* Page allows you to determine options related to i!-ConnectLinx and Scheduling on the *Scheduling and i!-ConnectLinx Options* page.


RMS CodeCrafter - C:\...\Projects\Conference Room.cgp
File Edit View Help
RMS Virtual and Socket Device Definitions

Enter the device address <D:P:S> for the vdvRMSEngine virtual device and the dvRMSSocket device below. If these devices have been previously defined in the integrated code file, those definitions will be used.

Enter the virtual device address <D:P:S> of the vdvCLActions device. If this device has been previously defined in the integrated code file, the <D:P:S> from that definition will be used.

vdvRMSEngine <D:P:S>: 33001 : 1 : 0

dvRMSSocket <D:P:S>: 0 : 3 : 0

i!-ConnectLinx virtual device

vdvCLActions <D:P:S>: 33003 : 1 : 0

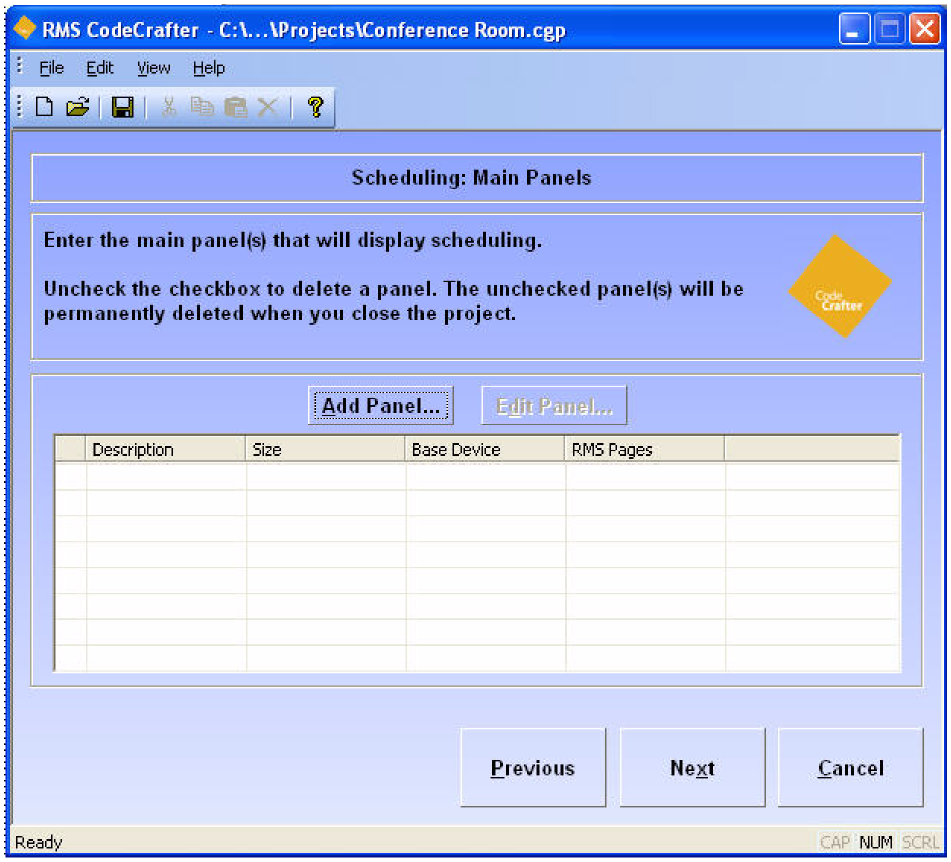
FIG. 7 RMS Virtual and Socket Device Definitions Page

RMS Virtual and Socket Device Definitions Page	
vdvRMSEngine:	Specify the device, port and system information of the virtual device and RMS Engine in these text fields.
dvRMSSocket:	Specify the device, port and system information of the device and RMS Socket in these text fields. ***Note**: If these devices are defined in the integrated file, the fields are automatically populated and protected.*

RMS Virtual and Socket Device Definitions Page (Cont.)	
i!-ConnectLinx Virtual Device:	
vdvCLActions:	Enter the device, port and system information of the virtual device and ConnectLinx in these text fields. If this device is defined in the integrated file, the fields are automatically populated and protected. In the event the devices are not defined by the integrated file or there is no integrated file for the project, the fields are populated by the following defaults: • vdvRMSEngine - 33001:1:0 • dvRMSSocket - 0:3:0 • vdvCLActions - 33002:1:0 Device definitions are added to the generated code file; otherwise, a warning statement is added in the generated code file.
Previous:	Regresses to the *Server Address* page.
Next:	Progresses to the *Main Panels* page.

Scheduling: Main Panels Page

The options on the *Scheduling: Main Panels* page allow you to add, edit and delete main panels used to display scheduling. This page is only available if you selected *Main and welcome panels* on the *Room Information/ Options* page.



Enter the main panel(s) that will display scheduling.

Uncheck the checkbox to delete a panel. The unchecked panel(s) will be permanently deleted when you close the project.

	Description	Size	Base Device	RMS Pages

FIG. 8 Scheduling: Main Panels Page

The main panel grid lists all panels available for the project. Any entry not selected with a check will be removed from the project upon closing the project.

Scheduling: Main Panels Page	
Add Panel:	Launches the Add/Edit Main Panel window. Adds a main panel to the project.
Edit Panel:	Launches the Add/Edit Main Panel window. When a main panel is selected you can edit its properties.
Previous:	Regresses the RMS CodeCrafter wizard to the RMS Virtual and Socket Device Definitions page.
Next:	Progresses the RMS CodeCrafter wizard to the Scheduling: Welcome Panels page.

Add/Edit Main Panel dialog

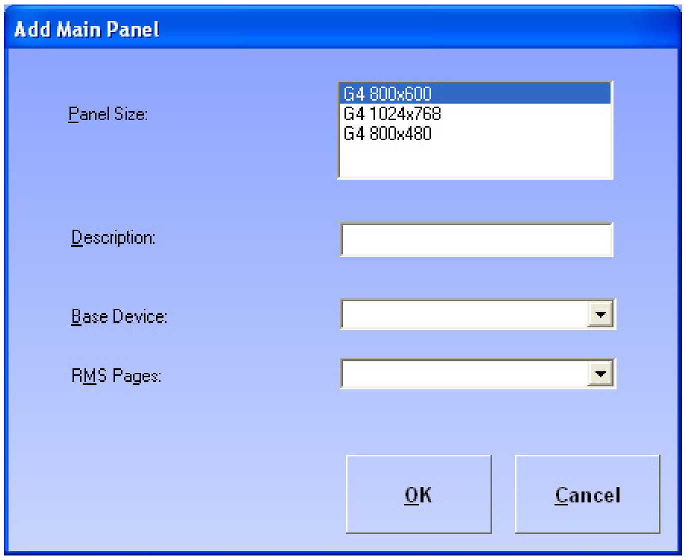


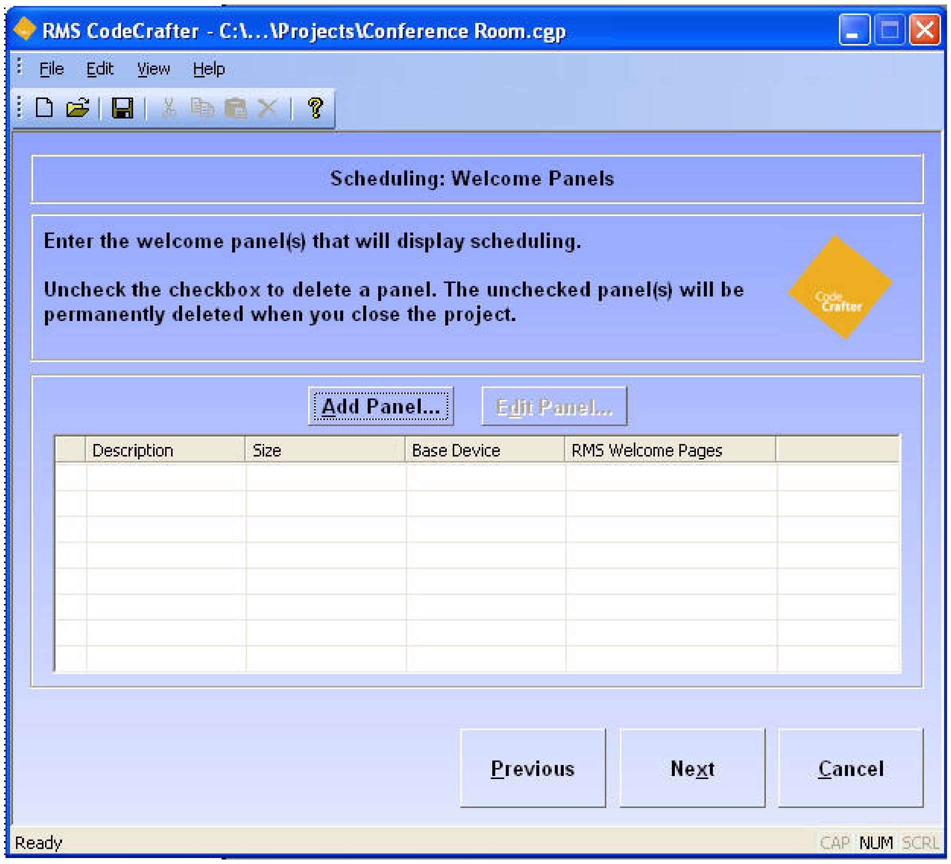
FIG. 9 Add/Edit Main Panel dialog

The *Add/Edit Main Panel* dialog is available to add and edit panel information:

Add/Edit Main Panel dialog	
Panel Size:	A list of supported panel sizes supported by CodeCrafter, click on your desired panel size to select.
Description:	A text field you can edit to describe the panel.
Base Device:	Drop down lists of devices available. This field is populated by devices extracted from the integrated NetLinx file. You can select an existing device, enter a new device, or enter a D:P:S address. If you select an existing device or enter a new device, a warning for the device is added to the generated code file.
RMS Pages:	Drop down lists of devices available. This field is populated by devices extracted from the integrated NetLinx file. You can select an existing device, enter a new device, or enter a D:P:S address. If you select an existing device or enter a new device, a warning for the device is added to the generated code file.
OK:	Accepts the changes and closes this dialog.
Cancel:	Closes this dialog without incorporating the changes.

Scheduling: Welcome Panels Page

The Welcome Panels page allows you to add, edit and delete welcome panels used to display scheduling. This page is only available if you selected *Main and Welcome panels* or *Welcome panels* only on the *Room Information/Options* page.



Scheduling: Welcome Panels

Enter the welcome panel(s) that will display scheduling.

Uncheck the checkbox to delete a panel. The unchecked panel(s) will be permanently deleted when you close the project.

	Description	Size	Base Device	RMS Welcome Pages

FIG. 10 Scheduling: Welcome Panels Page

Scheduling: Welcome Panels Page	
Add Panel:	Launches the *Add/Edit Welcome Panel* dialog, to add a welcome panel to the project.
Edit Panel:	Launches the *Add/Edit Welcome Panel* dialog. When a welcome panel is selected you can edit its properties. The welcome panel grid lists all panels available for the project. Any entry not selected with a check will be removed from the project upon closing the project.
Previous:	Regresses to the *Scheduling: Main Panels* page.
Next:	Progresses to the *Named NetLinx Devices* page.

Add/Edit Welcome Panel dialog

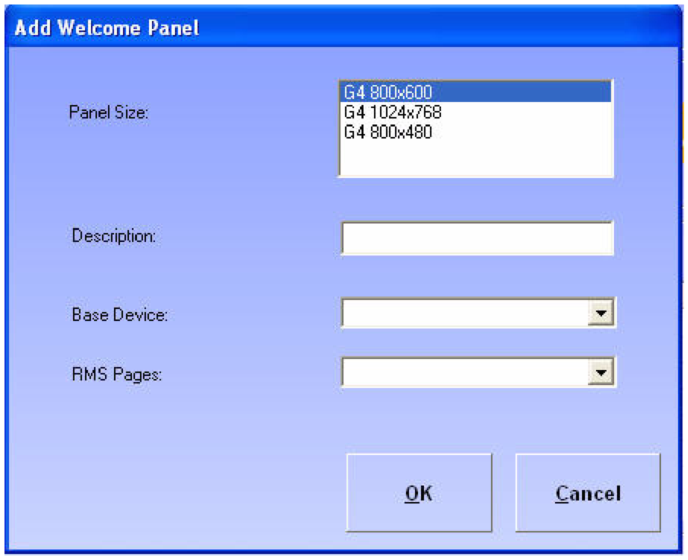


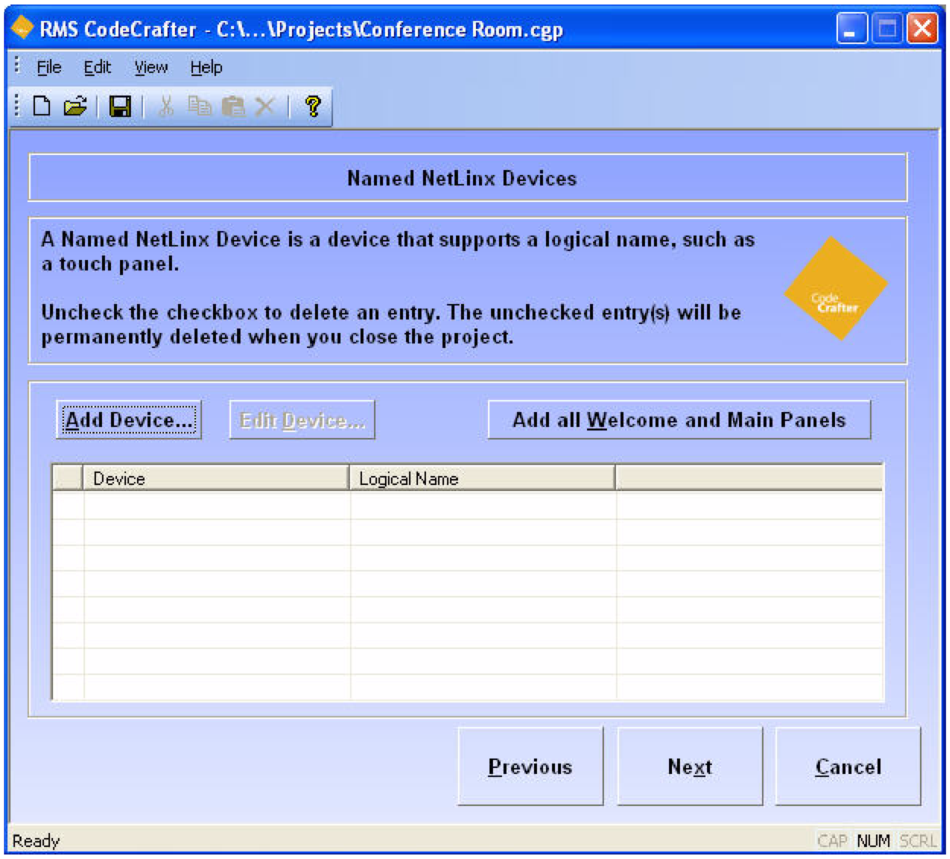
FIG. 11 Add/Edit Welcome Panel dialog

Use the options in the *Add/Edit Welcome Panel* dialog to add and edit panel information:

Add/Edit Welcome Panel dialog	
Panel Size:	Select the panel size for the Welcome panel from a list of panel sizes supported by CodeCrafter.
Description:	Enter a description of the Welcome panel in this text field.
Base Device:	Select a Base Device from a drop down list of devices extracted from the integrated NetLinx file. You can select an existing device, enter a new device, or enter a D:P:S address. If you select an existing device or enter a new device, a warning for the device is added to the generated code file.
RMS Pages:	Drop down lists of devices available. This field is populated by devices extracted from the integrated NetLinx file. You can select an existing device, enter a new device, or enter a D:P:S address. If you select an existing device or enter a new device, a warning for the device is added to the generated code file.
OK:	Saves changes and closes this dialog.
Cancel:	Closes this dialog without saving changes.

Named NetLinx Devices Page

The *Named NetLinx Devices* page allows you to add, edit and delete devices monitored as Named Devices. A Named Device is a NetLinx device that supports a logical name (i.e., a touch panel)..



A Named NetLinx Device is a device that supports a logical name, such as a touch panel.

Uncheck the checkbox to delete an entry. The unchecked entry(s) will be permanently deleted when you close the project.

FIG. 12 Named NetLinx Devices Page

Named NetLinx Devices Page	
Add Device:	Launches the *Add/Edit Named Device* dialog to add a named device to the project.
Edit Device:	Launches the *Add/Edit Named Device* dialog. When a named device is selected you can edit its properties.
Add all Welcome and Main Panels:	Launches the *Add All Welcome and Main Panels* window. Doing this adds all defined devices to the named device grid list. • The named device grid lists all devices available for the project. • Any entry not selected with a check will be removed from the project upon closing the project.
Previous:	Regresses to the *Scheduling: Welcome Panels* page.
Next:	Progresses to the *Monitored Third-Party Devices* page.

Add/Edit Named Devices dialog

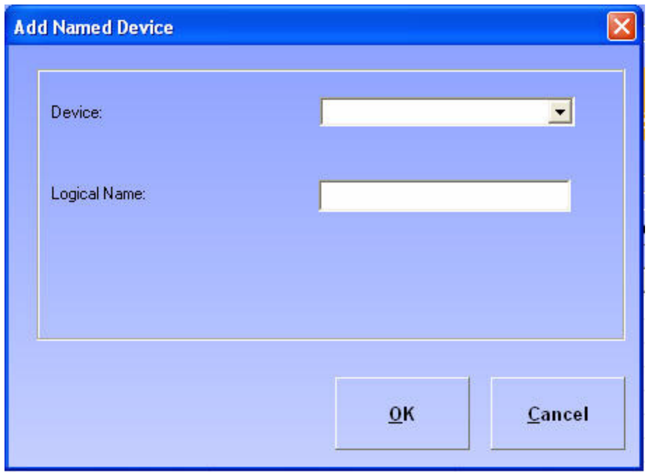


FIG. 13 Add/Edit Named Devices dialog

The *Add/Edit Named Devices* dialog is available to add and edit named devices:

Add/Edit Named Devices dialog	
Device:	Select a Base Device from a drop down list of devices extracted from the integrated NetLinx file. You can select an existing device, enter a new device, or enter a D:P:S address. If you select an existing device or enter a new device, a warning for the device is added to the generated code file.
Logical Name:	Enter a name for the device in this text field.
OK:	Accepts changes and closes this dialog.
Cancel:	Closes this dialog and does not save changes.

Add All Welcome and Main Panels dialog

When you click **Add all Welcome and Main Panels,** CodeCrafter searches the existing Main and Welcome panels for defined devices that are not already listed on the Named Devices list. All devices found are compiled into a list and appear in the *Add All Welcome and Main Panels* dialog. The options in this dialog allow you to select any, all or none for your use.

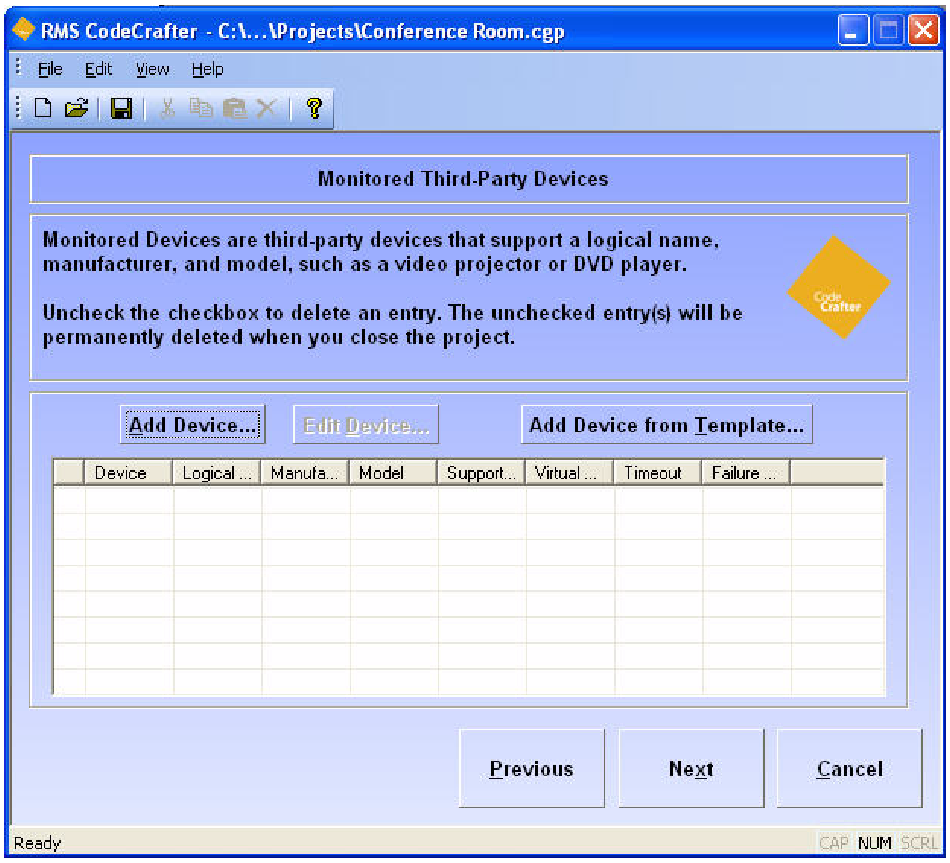
In the event no devices are found, this window will inform you none were found.

The grid list is all defined devices found in the Main and Welcome panels. Any entry not selected with a check is not included in the project. The Device column is the device name and the Logical Name column is the associated name for that device. CodeCrafter uses the panel description for the logical name, it can be changed after you add it to the list (*Add/Edit Named Devices*).

- **OK** - Accepts changes and closes window.
- **Cancel** - Closes window and does not keep changes.

Monitored Third-Party Devices Page

The *Monitored Third-Party Devices* page allows you to add, edit and delete devices intended to be monitored as Monitored Devices. A Monitored Device is a device that supports a logical name, manufacturer, and model, i.e., a projector.



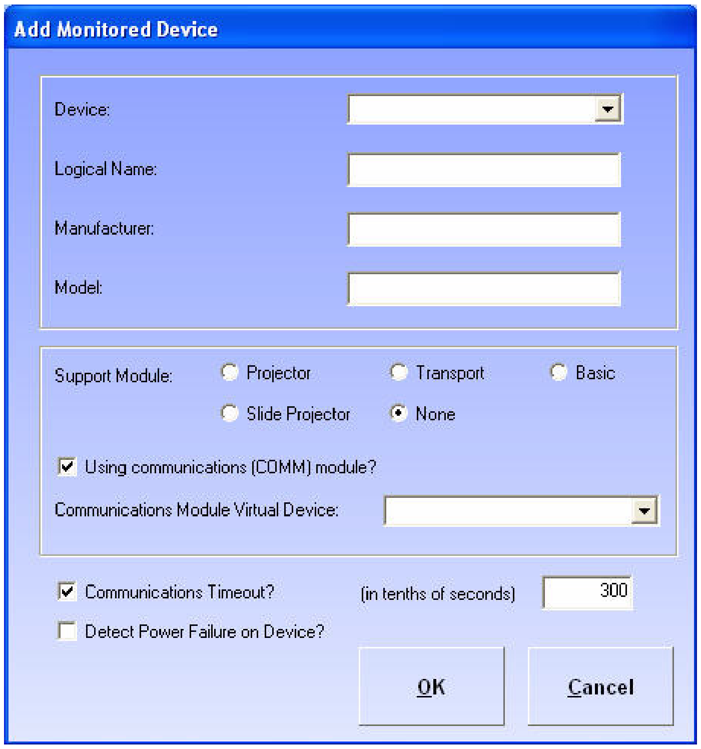
Monitored Devices are third-party devices that support a logical name, manufacturer, and model, such as a video projector or DVD player.

Uncheck the checkbox to delete an entry. The unchecked entry(s) will be permanently deleted when you close the project.

FIG. 14 Monitored Third-Party Devices Page

Monitored Third-Party Devices Page	
Add Device:	Launches the *Add/Edit Monitored Device* window, to add a monitored device to the project.
Edit Device:	Launches the *Add/Edit Monitored Device* window. When a monitored device is selected you can edit its properties.
Add Device from Template:	Launches the *Add Device from Template* dialog, to add a device using a device template. • The monitored device grid lists all devices available for the project. • Any entry not selected with a check will be removed from the project upon closing the project.
Previous:	Regresses to the *Named NetLinx Devices* page.
Next:	Progresses to the *Device Parameters* page.

Add/Edit Monitored Devices dialog



Support Module: ( ) Projector ( ) Transport ( ) Basic ( ) Slide Projector [✓ None]

[✓] Using communications (COMM) module?

[✓] Communications Timeout? (in tenths of seconds) 300

[ ] Detect Power Failure on Device?

FIG. 15 Add/Edit Monitored Devices dialog

The Add/Edit Monitored Devices dialog is available to add and edit named devices. The fields you can edit are as follows:

Add/Edit Monitored Devices dialog	
Device:	Select a device from a drop-down list of devices extracted from the integrated NetLinx file. You can select an existing device, enter a new device, or enter a D:P:S address. If you select an existing device or enter a new device, a warning for the device is added to the generated code file.
Logical Name:	Enter a name for the device in this text field.
Manufacturer:	Enter the manufacturer name for the device in this text field.
Model:	This is a text field where you can provide the model information for the device.
Support Module:	Click the radio button that best represents the device for the template: • Projector • Transport • Basic • Slide Projector • None
Using communications (COMM) module?	When selected, CodeCrafter includes code for a communications module.
Communications Module Virtual Device	Select a Module Virtual Device from a drop-down list of devices extracted from the integrated NetLinx file. You can select an existing device, enter a new device, or enter a D:P:S address. If you select *None in Support Module*, this field is disabled.
Communication Timeout:	Place a check in the box to enable the function. • Measured in 1/10ths of a second, where 0 is a valid value. • Default is 300.

Add/Edit Monitored Devices dialog (Cont.)	
Control Failure Detect:	Placing a check in the box adds a call to RMSEnablePowerFailure() in the generated code file.
OK:	Accepts changes and closes this dialog.
Cancel:	Closes this dialog and does not keep changes.

Select Device Template dialog

If you select *Add Device from Template*, this dialog gives you the selection of all templates found in the default folder and populates them into the grid.

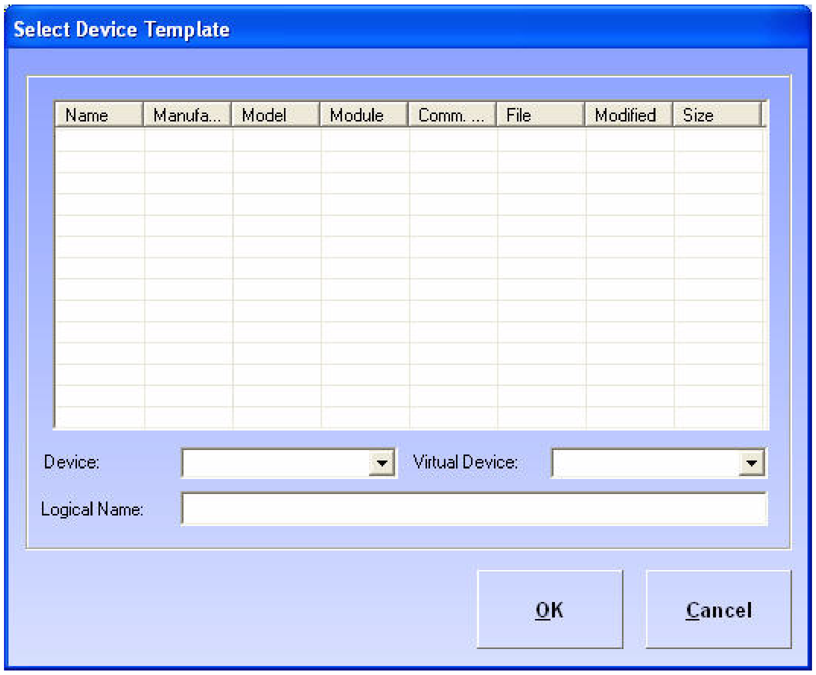
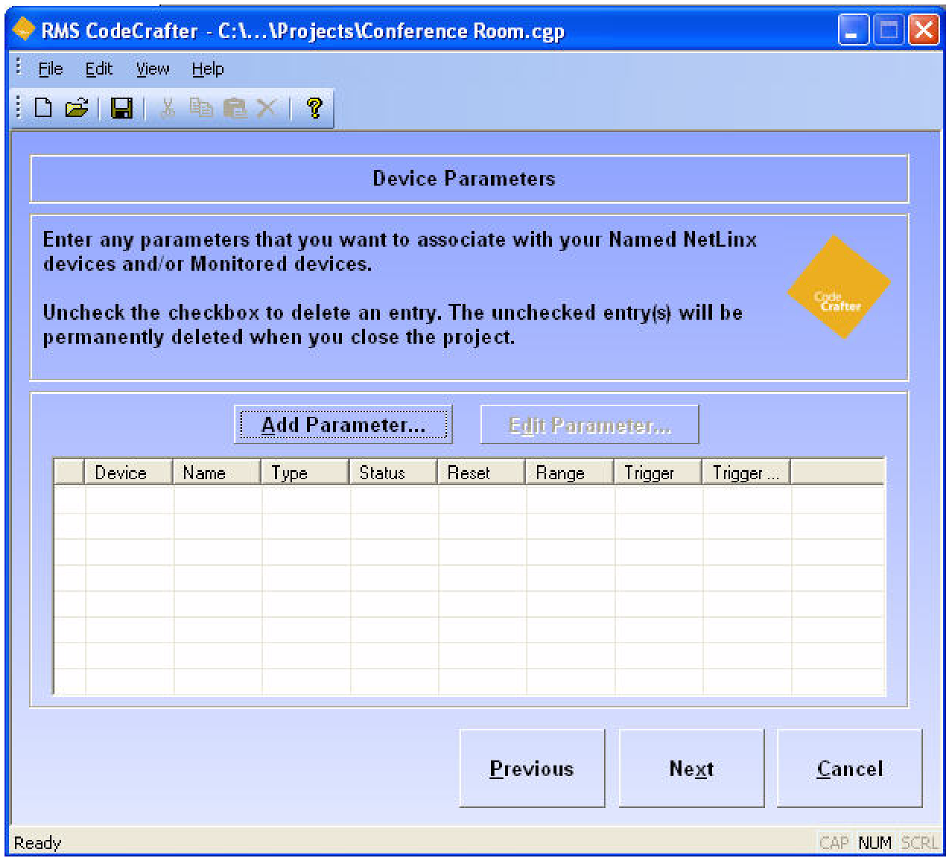


FIG. 16 Select Device Template dialog

Select Device Template dialog	
Device:	Select a device from a drop-down list, or type it in manually.
Virtual Device:	Select a communications module virtual device from a drop-down list, or type it in manually. This field is required only when the selected device template uses a communications module.
Logical Name:	Enter the logical name of the device in this text field.
OK:	Accepts the changes and closes this dialog.
Cancel:	Closes this dialog without keeping changes.

Device Parameters Page

The *Device Parameters* page allows you to add, edit and delete Device Parameters. This page is only available if your project has Named NetLinx, Monitored devices or both.



Enter any parameters that you want to associate with your Named NetLinx devices and/or Monitored devices.

Uncheck the checkbox to delete an entry. The unchecked entry(s) will be permanently deleted when you close the project.

FIG. 17 Device Parameters Page

Device Parameters Page	
Add Parameter:	Launches the *Add/Edit Device Parameters* dialog, to add a device parameter to the project.
Edit Parameter:	Launches the *Add/Edit Device Parameters* dialog. When a device parameter is selected you can edit its properties. The device parameters grid lists all devices available for the project. Any entry not selected with a check will be removed from the project upon closing the project.
Previous:	Regresses to the *Monitored Third-Party Devices* page.
Next:	Progresses to the *i!-ConnectLinx Touch Panel Button Mappings* page.

Add/Edit Device Parameter dialog

The *Add/Edit Device Parameter* dialog allows you to add device parameters or edit existing ones.

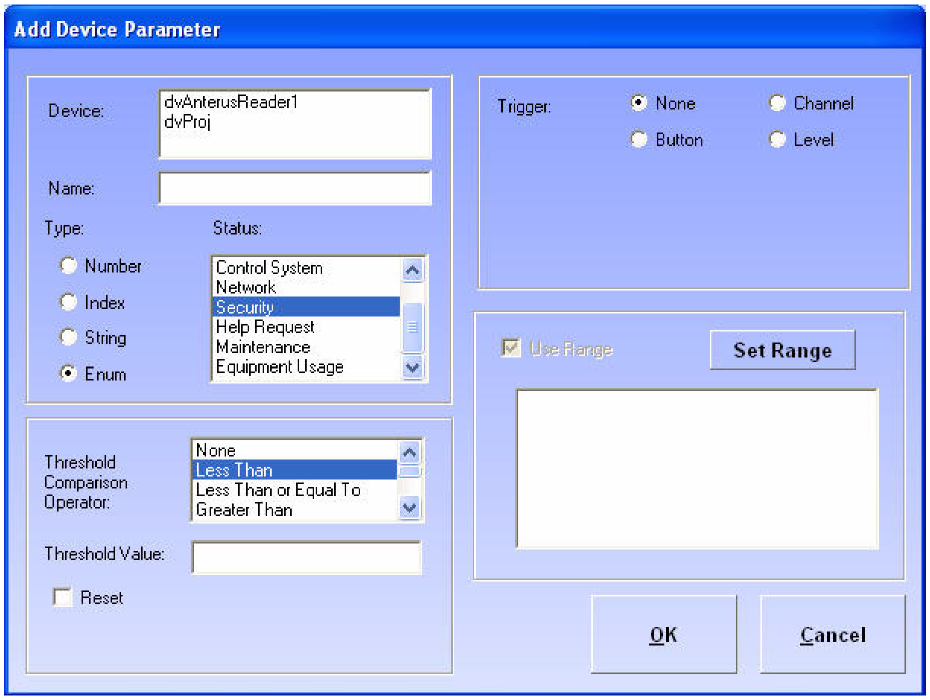
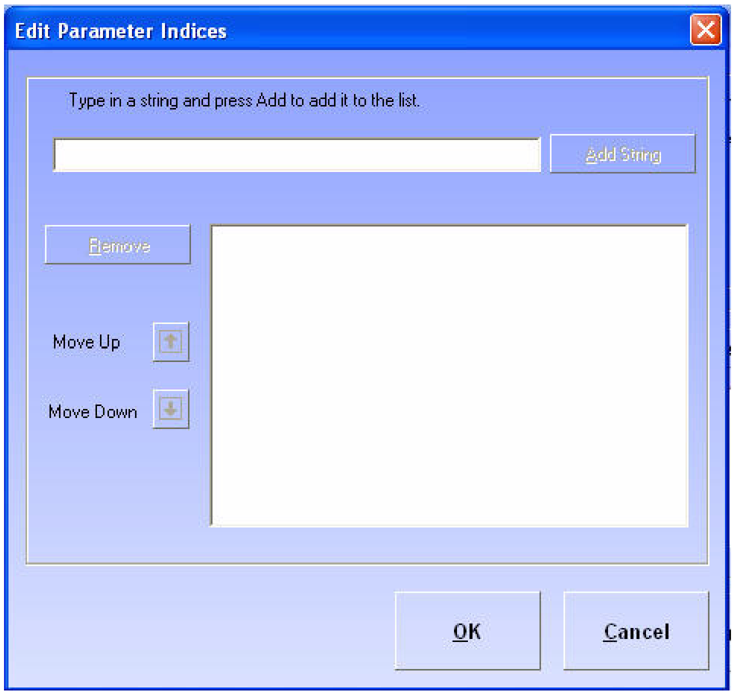


FIG. 18 Add/Edit Device Parameter dialog

Add/Edit Device Parameter dialog	
Name:	A text field you can edit to name the parameter.
Type:	Select one of the parameter types: • Number • Index • String • Enum
Status:	Select one from the scroll list: • Not Assigned • Room Communication • Control System • Network • Security • Help Request • Maintenance • Equipment Usage
Threshold Comparison Operator:	Select from the following: • None • Less Than • Less Than or Equal To • Greater Than • Greater Than or Equal To • Equal To • Not Equal To • Contains (String or Enum only) • Does Not Contain (String or Enum only)
Threshold Value:	A text field you can edit to set the threshold value. 32-bit signed number for Number, 16-bit unsigned number for Index and any string up to 100 characters for String and Enum.
Reset:	Place a mark in the box to enable reset of threshold.

Add/Edit Device Parameter dialog (Cont.)	
Reset Value:	A text field you can edit to set reset value. 32-bit signed number for Number, 16-bit unsigned number for Index and any string up to 100 characters for String and Enum.
Trigger:	Select one of the following: • None • Button • Channel • Level
Use Range (Number):	Sets the minimum and maximum for number; this value is optional.
Use Units (Number or String):	Defines the unit type for the parameter, e.g., minutes, hours, degrees, or %. This value is optional.
Set Range (Enum or Index):	A drop down list of enumeration strings. You can elect to: • Add - adds the sting to the parameter use. • Remove - removes the string from parameter use. • Move Up - moves the string up the list of parameter use. • Move Down - moves the string down the list of parameter use.
OK:	Closes this dialog and saves changes made.
Cancel:	Closes this dialog without implementing changes.

Edit Parameter Indices dialog



Type in a string and press Add to add it to the list.

FIG. 19 Edit Parameter Indices dialog

Edit Parameter Indices dialog	
Add string:	Adds the sting to the parameter use.
Remove:	Removes the string from parameter use.
Move Up:	Moves the string up the list of parameter use.
Move Down:	Moves the string down the list of parameter use.
OK:	Closes this dialog and saves changes.
Cancel:	Closes this dialog without implementing changes.

Edit Enumeration List dialog

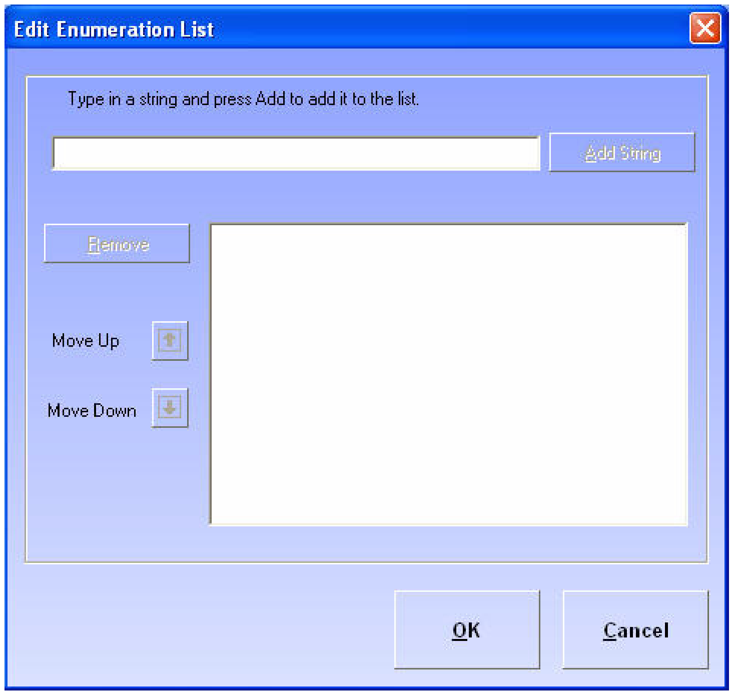


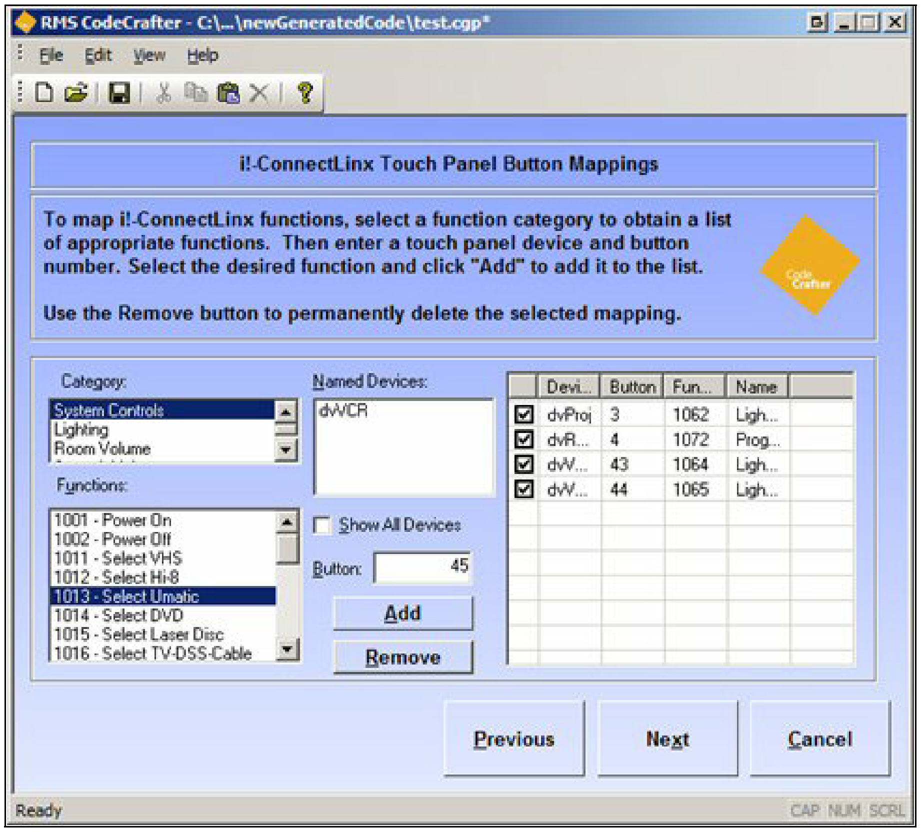
FIG. 20 Edit Enumeration List dialog

Edit Parameter List dialog	
Add string:	Adds the sting to the parameter use.
Remove:	Removes the string from parameter use.
Move Up:	Moves the string up the list of parameter use.
Move Down:	Moves the string down the list of parameter use.
OK:	Closes the window and saves changes made.
Cancel:	Closes window without implementing changes.

i!-ConnectLinx Touch Panel Button Mappings Page

The i!-ConnectLinx Touch Panel Button Mappings page is available if you selected *Support pre-meeting presets?* or *Monitor source usage?* on the *Room Information/Options* page. You can map touch panel buttons to i!-ConnectLinx. Otherwise, only power on/off and source selects are available.

The i!-ConnectLinx functions grid list displays all i!-ConnectLinx functions, with touch panel button mappings, currently available in the project. The column on the left contains a checkbox where the check indicates the function is mapped. Removing the check clears the mapping.


RMS CodeCrafter - C:\...\newGeneratedCode\test.cgp*
File
Edit
View
Help
i!-ConnectLinx Touch Panel Button Mappings

To map i!-ConnectLinx functions, select a function category to obtain a list of appropriate functions. Then enter a touch panel device and button number. Select the desired function and click "Add" to add it to the list.

Use the Remove button to permanently delete the selected mapping.

	Devi...	Button	Fun...	Name
☑	dvProj	3	1062	Ligh...
☑	dvR...	4	1072	Prog...
☑	dvV...	43	1064	Ligh...
☑	dvV...	44	1065	Ligh...

FIG. 21 i!-ConnectLinx Touch Panel Button Mappings page

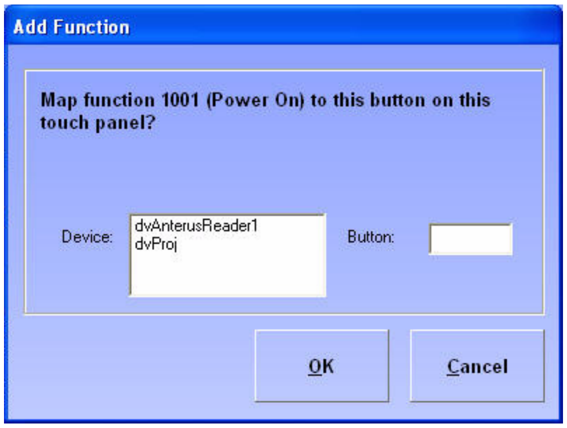
i!-ConnectLinx Touch Panel Button Mappings Page	
Category:	A list of i!-ConnectLinx function categories. The Functions list populates corresponding to your selection in this list.
Functions:	A list of i!-ConnectLinx functions for the selected category.
Named Devices:	A list of devices populated by what is defined in the *Add/Edit Named Devices dialog* section on page 25. This field only contains devices that have been previously defined.
Show All Devices:	A check box that displays all devices in the Named Devices list when checked. If unchecked, only the Named Devices display in the list.
Button:	A numeric field for the button number.
Add:	Adds an entry to the device list. Once you click add you are prompted with a confirmation window. The window contains the current function number and name. The window provides a box to select the touch panel device and a numeric field for the button number, but if the device and button fields are populated, RMS CodeCrafter will attempt to add the mapping without offering the dialog. In the event the mapping already exists, the mapping will fail and you receive a notification. To re-map, the original must be removed from the device list.
Remove:	Removes the device currently selected in the device list.

i!-ConnectLinx Touch Panel Button Mappings Page (Cont.)	
Previous:	Regresses the RMS CodeCrafter wizard to the Device Parameters Page.
Next:	Progresses the RMS CodeCrafter wizard to the Generate NetLinx Code File Page.

Edit Touch Panel Button-Mappings

The *Add Function* dialog allows you to edit the button-mappings.

Listed is the i!ConnectLinx function name and number. This field is static.


Add Function

Map function 1001 (Power On) to this button on this touch panel?

Device: dvAnterusReader1, dvProj

Button:

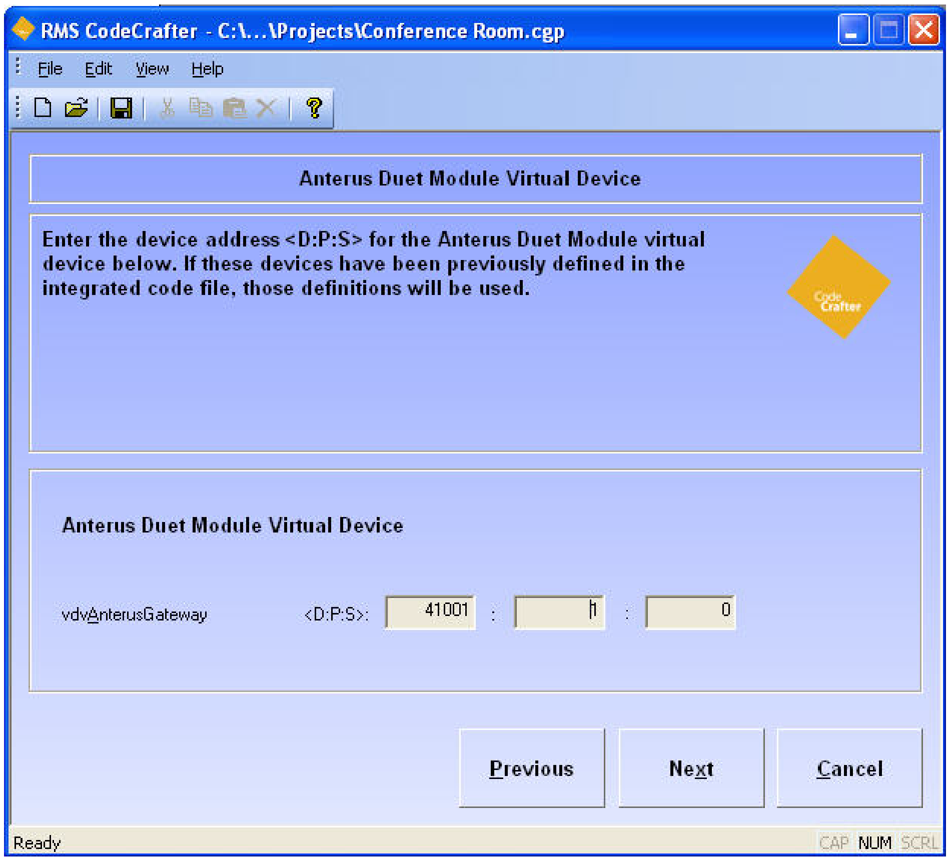
FIG. 22 Add Function dialog

Add Function dialog	
Device:	A list of devices available. This field is populated by devices extracted from the integrated NetLinx file. You can only select an defined device.
Button:	This is a text field you can edit to enter the panel button number.
OK:	Accepts changes and closes window.
Cancel:	Closes window and does not keep changes.

Anterus Duet Module Virtual Device Page

The Anterus Duet Module Virtual Device page is available if you selected *Support Anterus RFID for device tracking in RMS?* on the *Room Information/Options* page.

- Use this page to specify a device address (<D:P:S>) for the Anterus Duet Module Virtual Device.
- If device addresses specified here have been previously defined in the integrated code file, those definitions will be lost.



Anterus Duet Module Virtual Device

Enter the device address <D:P:S> for the Anterus Duet Module virtual device below. If these devices have been previously defined in the integrated code file, those definitions will be used.

Anterus Duet Module Virtual Device

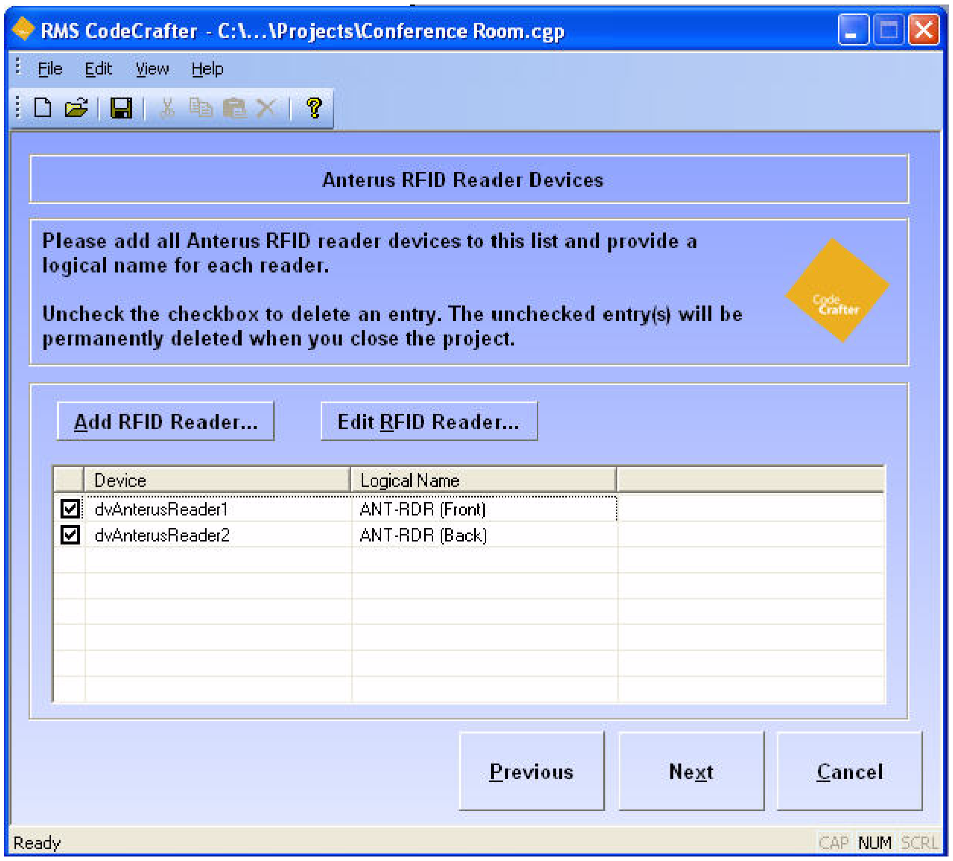
vdvAnterusGateway <D:P:S>: 41001 : 1 : 0

FIG. 23 Anterus Duet Module Virtual Device Page

- **Previous** - Regresses to the *Anterus RFID Reader Devices* page.
- **Next** - Progresses to the *i!-ConnectLinx Touch Panel Button Mappings* page.

Anterus RFID Reader Devices Page

The Anterus RFID Reader Devices page is available if you selected *Support Anterus RFID for device tracking in RMS?* on the *Room Information/Options* page. Use this page to add all Anterus RFID Readers to the system.



Anterus RFID Reader Devices

Please add all Anterus RFID reader devices to this list and provide a logical name for each reader.

Uncheck the checkbox to delete an entry. The unchecked entry(s) will be permanently deleted when you close the project.

	Device	Logical Name
☑	dvAnterusReader1	ANT-RDR (Front)
☑	dvAnterusReader2	ANT-RDR (Back)

FIG. 24 Anterus RFID Reader Devices Page

Anterus RFID Reader Devices Page	
Add RFID Reader:	This option invokes the *Add Anterus RFID Reader* dialog, to add an RFID Reader to the list. The *Add Anterus RFID Reader* dialog provides a drop-down menu to select the RFID Reader device and a text field for the Logical Name.
Edit RFID Reader:	Edits the selected RFID Reader. This option invokes the *Edit Anterus RFID Device* dialog, where you can select a different RFID Reader and/or change the selected Reader's Logical Name assignment.
Previous:	Regresses to the *Anterus Duet Module Virtual Device* page.
Next:	Progresses to the *Generate NetLinx Code File* page.

Add Anterus RFID Reader dialog

The *Add Anterus RFID Reader* dialog allows you to add RFID Readers to the system.

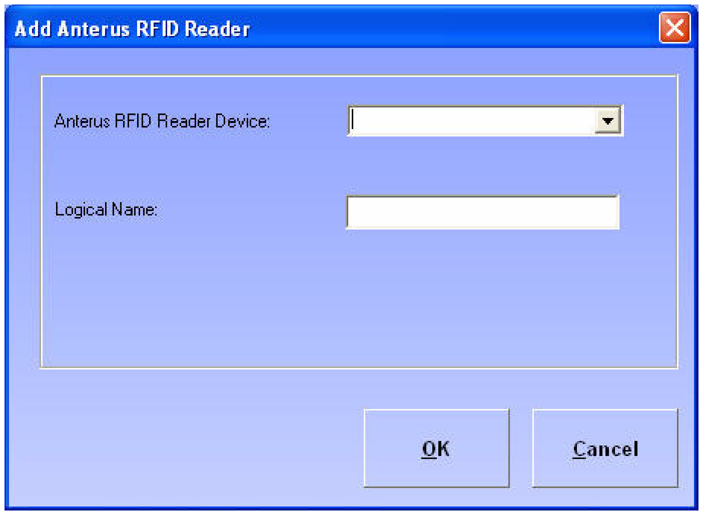
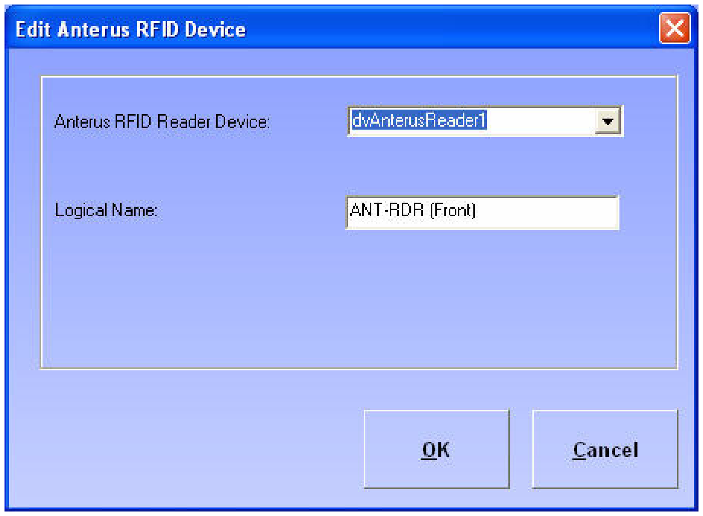


FIG. 25 Add Anterus RFID Reader dialog

Add Anterus RFID Reader dialog	
Anterus RFID Reader Device:	A list of RFID Readers available. This field is populated by devices extracted from the integrated NetLinx file. You can only select an defined device.
Logical Name:	Use this is a text field to enter the logical name for the RFID Reader.
OK:	Accepts changes and closes window.
Cancel:	Closes window and does not keep changes.

Edit Anterus RFID Device dialog

The Edit Anterus RFID Device window allows you to edit the selected RFID Readers.



Anterus RFID Reader Device: dvAnterusReader1

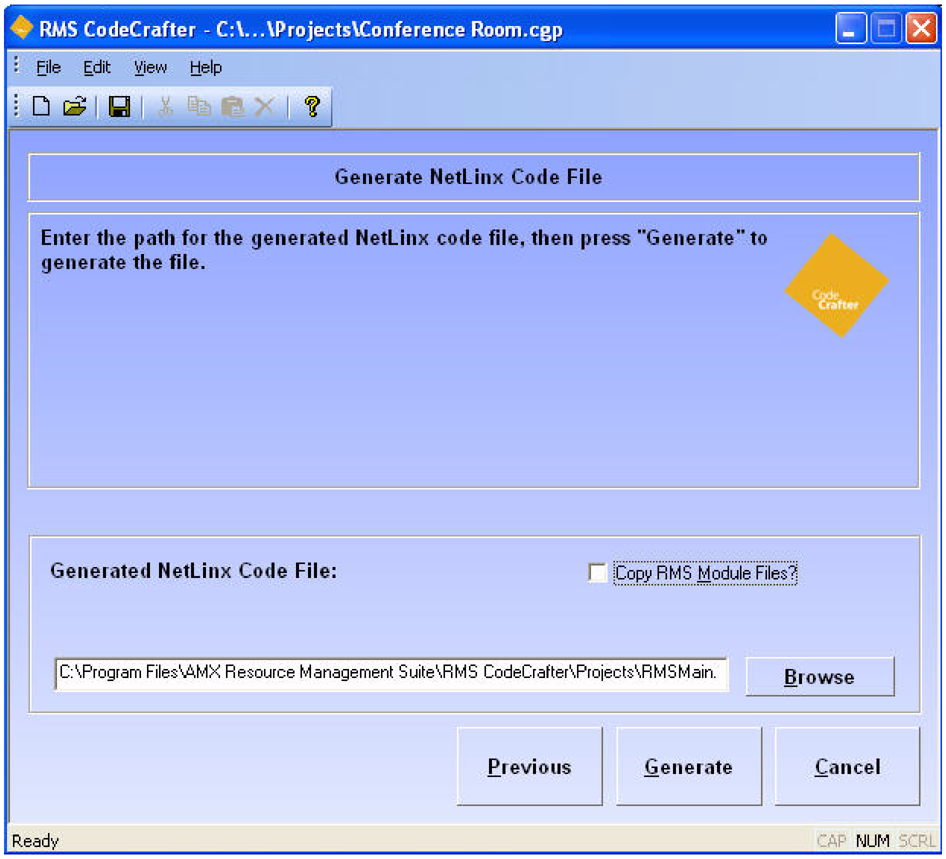
Logical Name: ANT-RDR (Front)

FIG. 26 Edit Anterus RFID Device dialog

The fields in this dialog are the same as those in the *Add Anterus RFID Reader* dialog (see above).

Generate NetLinx Code File Page

The Generate NetLinx Code File page prompts CodeCrafter to generate the NetLinx include code file.



RMS CodeCrafter - C:\...\Projects\Conference Room.cgp

Generate NetLinx Code File

Enter the path for the generated NetLinx code file, then press "Generate" to generate the file.

Generated NetLinx Code File:

Copy RMS Module Files?

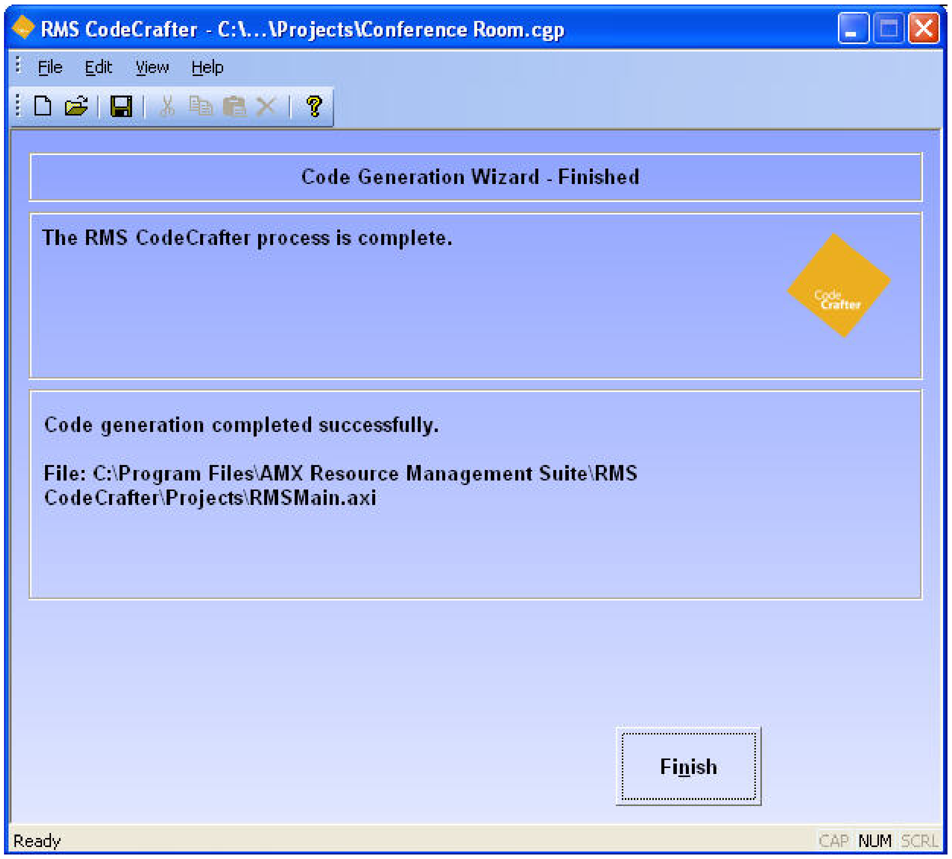
C:\Program Files\AMX Resource Management Suite\RMS CodeCrafter\Projects\RMSMain.

FIG. 27 Generate NetLinx Code File Page

Generate NetLinx Code File Page	
Generated NetLinx Code File:	This is a text field you can either type the path or click Browse and specify the file name and target to generate.
Copy RMS Module Files?	When the box is checked, CodeCrafter will copy all corresponding .TKO files from the SDK install location and paste them in the location you designated in the *Generated NetLinx Code File* path.
Previous:	Regresses the CodeCrafter wizard to the i!-ConnectLinx Touch Panel Button Mappings page.
Generate:	Prompts CodeCrafter to generate the file and advance to the final page.

Code Generation Wizard - Finished Page

The *Code Generation Wizard - Finished* page is the final step in the process of creating your own RMS code. The results of the code generation operation and confirmation of the target location is listed.


RMS CodeCrafter - C:\...\Projects\Conference Room.cgp
File Edit View Help
Code Generation Wizard - Finished
Code Crafter
Finish
Ready
CAP NUM SCRL

The RMS CodeCrafter process is complete.

Code generation completed successfully.

File: C:\Program Files\AMX Resource Management Suite\RMS CodeCrafter\Projects\RMSMain.axi

FIG. 28 Code Generation Wizard - Finished Page

- **Finish** - Clicking this button returns you to the Introductory page.

If you selected *Save Project on Finish?* in User Preferences you are prompted to save your project now.

Basic Operations

Creating A New Device Template

You can create persistent templates for monitored devices, where you provide the settings and parameters. RMS CodeCrafter can then use the device templates for similar or the same devices in other projects.

1. With the RMS CodeCrafter wizard open to the Introductory page, click **New Device Template**. The RMS CodeCrafter wizard opens to the Device Template Main page.
2. Type the name of your Template in the field *Template Name*.
3. Type the manufacturer name in the field *Manufacturer.*
4. Type the model in the field *Model*.
5. Place a check in the box *Uses Communications Module?* if you would like to enable it for this template.
6. If needed, select and set a value for the *Communications Timeout*, measured in 1/10ths of a second, where 0 is a valid value. If no timeout is required, use -1.
7. If you want to enable *Control Failure Detect*, place a check in the box next to it.
8. Click the radio button for the Support Module that fits your device template.
    - Projector
    - Transport
    - Basic
    - Slide Projector
    - None
9. Click **Next**. The next page is *Template Device Parameters* page.
10. Click **Add Parameter** to launch the *Add/Edit Device Template Parameters* window.
11. Type the name of the parameter in the field provided.
12. Click the radio button for the *Type* of parameter and select the *Status* in the menu.
13. In the menu, select the *Threshold Comparison* and enter the *Threshold Value*. Place a check next to *Reset* to enable.
14. Click the radio button for the *Trigger* type.
15. Based on the *Type* of parameter you selected in step 12 you may need to provide a range for the parameter.
16. Click **Next**. The next page is *Device Template Wizard Finished* page.
17. Within the *Save Template to* field, either type or click Browse to select the target location to save your template file (.CGT).
18. Click **Finish** to create the template file and save it.

Creating A New RMS Project

The RMS CodeCrafter wizard creates project files that it uses in generating the NetLinx code you need. To create a .CGP file:

1. From the Introductory page click **New Project**.
2. The next page is the Project Start page. Either type or click **Browse** to select the location of your integrated file.

NOTE

While it is not necessary to have either an .AXS or .AWP file to use the RMS CodeCrafter wizard, it is advisable. The integrated file will populate several fields throughout the wizard that you will need to otherwise enter manually.

3. Click **Next**.

    If NetLinx Studio workspace file (.APW) is selected as the integrated file:

a. From the list of NetLinx code files listed in the Code File Selection page, click on the desired file and click **Next**.

   If NetLinx Studio workspace file (.APW) was not selected as the integrated file:

b. The next page is Scheduling and i!-ConnectLinx options page. Check all i!-ConnectLinx Options that apply.

4. Select your i!-ConnectLinx Options:
   - Support pre-meeting presets?
   - Monitor source usage?
   - Using multiple display devices?
5. Define the *Room Name, Location, Owner.*
6. Click the radio button to select one of the following Scheduling Displays:
   - Main and welcome scheduling panels
   - Welcome scheduling panels only
   - Help Desk panels only
   - No scheduling display
7. If you selected either *Main and welcome scheduling panels* or *Welcome scheduling panels only*, you can set the default welcome panel subject and message at this time. If you do not enter anything in these fields the stings will not be created.

   Type a welcome subject and message in the text fields provided.
8. Click **Next**. The next page is *RMS Server Address* page.
9. Type the RMS Server Address in the field provided or leave blank to use the default. Click **Next**.
10. The next page is *RMS Virtual and Socket Device Definitions* page. Type all D:P:S fields that apply.
11. Click **Next**. The next page is *Scheduling: Main Panels* page.
    - To add a panel, click **Add Panel** and open the *Add/Edit Main Panel* dialog.
    - To edit a panel, select the panel from the list and click **Edit Panel** to open the *Add/Edit Main Panel* dialog.
12. Once you have added all of your main panels, click **Next**. If you selected either *Main and Welcome Panels* or *Welcome panels only* in step 6 the next page is the *Scheduling: Welcome Panels* page. Otherwise it is the *Named NetLinx Devices* page.
13. To add a panel, click **Add Panel** and open the Add/Edit Welcome Panel window. To edit a panel, select the panel from the list and click **Edit Panel** to open the Add/Edit Welcome Panel window.
14. Once you have added all of your main panels, click Next. The next page is Named NetLinx Devices.
15. To add a device, click **Add Device** and open the Add/Edit Named Device window. To edit a device, select the device from the list and click **Edit Device** to open the Add/Edit Named Device window. If you would like to find and add all MeetingManager devices not already in the Named Devices list, click **Add all Welcome and Main Panels** to launch the *Add All Welcome and Main Panels* Window.
16. Once you have included all of your named devices for your project, click **Next**. The next page is *Monitored Third-Party Devices* page.
17. To add a device, click **Add Device** and open the *Add/Edit Monitored Device* dialog. To edit a device, select the device from the list and click **Edit Device** to open the Add/Edit Monitored Device window. You can also add a device from a template, click **Add Device from Template** to open the *Add Device from Template* dialog.
    - If you selected **Add Device** from Template, select the device from the list.
    - Either type or enter the *Device, Virtual Device*, and *Logical Name*.
    - Click **OK**.
18. Once you have included all of your monitored devices for your project, click **Next**. The next page is *Device Parameters*.
19. To add a device parameter, click **Add Device** and open the Add/Edit Device Parameters window. To edit a device parameter, select the device from the list and click **Edit Device** to open the *Add/Edit Device Parameters* dialog.

20. Once you have included all of your device parameters for your project, click **Next**. The next page is *i!-ConnectLinx Touch Panel Button Mappings*.
21. To edit an i!-ConnectLinx entry, select the function from the list and click **Edit Function** to open the *Edit Touch Panel Button-mappings*.
22. Once you have included all of your button-mappings for your project, click **Next**. The next page is *Generate NetLinx Code File*.
23. Either type the file name and target destination or click **Browse** and specify the location. Once you have entered the information, click **Generate**.
24. The next and final page is *Code Generation Wizard - Finished*. You will see this page if the NetLinx include file was successfully created. Displayed within the page is the file name and target destination.
25. Click **Finish**.

Saving A Device Template File

Once you have created a Device Template it is necessary to save it order to keep changes made.

1. With the Device Template open, select **File > Save**.
2. Select the target location for the file and click **Save**.

Saving A Project File

Once you have created a RMS CodeCrafter file, it is necessary to save the file. You can then return to the file at a later date.

1. With the project file open, select **File > Save**.
2. Select the target location to save the file.
3. Click **Save**.

Opening An Existing Device Template

You can create persistent templates for monitored devices, where you provide the settings and parameters. RMS CodeCrafter can then use the device templates for similar or the same devices in other projects.

If you already created a template, you can open and use it following these steps:

1. With RMS CodeCrafter wizard open to the Introductory page, click **Open Device Template** to launch the Open RMS CodeCrafter Device Template window.
2. Select the template file (.CGT) you would like to use and click **Open**.
3. The wizard opens to Device Template Main page.

Follow these steps to edit your template:

1. Type the name of your Template in the field *Template Name*.
2. Type the manufacturer name in the field *Manufacturer.*
3. Type the model in the field *Model*.

4. Set a value for the *Communications Timeout*, measured in 1/10ths of a second, where 0 is a valid value. If no timeout is required, use -1.
5. If you want to enable *Control Failure Detect*, place a check in the box next to it.
6. Click the radio button for the Support Module that fits your device template.
   - Projector
   - Transport
   - Basic
   - Slide Projector
   - None
7. Click **Next**. The next page is Device Template Final page.
8. Within the *Save Template to* field, either type or click **Browse** to select the target location to save your template file (.CGT).
9. Click **Finish** to create the template file and save it.

Opening An Existing Project

If you have already created a project (.CGP) and would like to use it as a starting point, follow these steps:

1. With the RMS CodeCrafter wizard open to the Introductory page, click **Open Project**.
2. In the Open RMS CodeCrafter Project window, navigate to your project and click **Open**.

You can now begin working from your saved project.

Closing A Project Or Template File

Once you are done with a project or template file, but would like to keep RMS CodeCrafter open, you can close the file. To close a file:

- Select **File > Close**. If you have not already saved your file you are prompted to do so at this time.

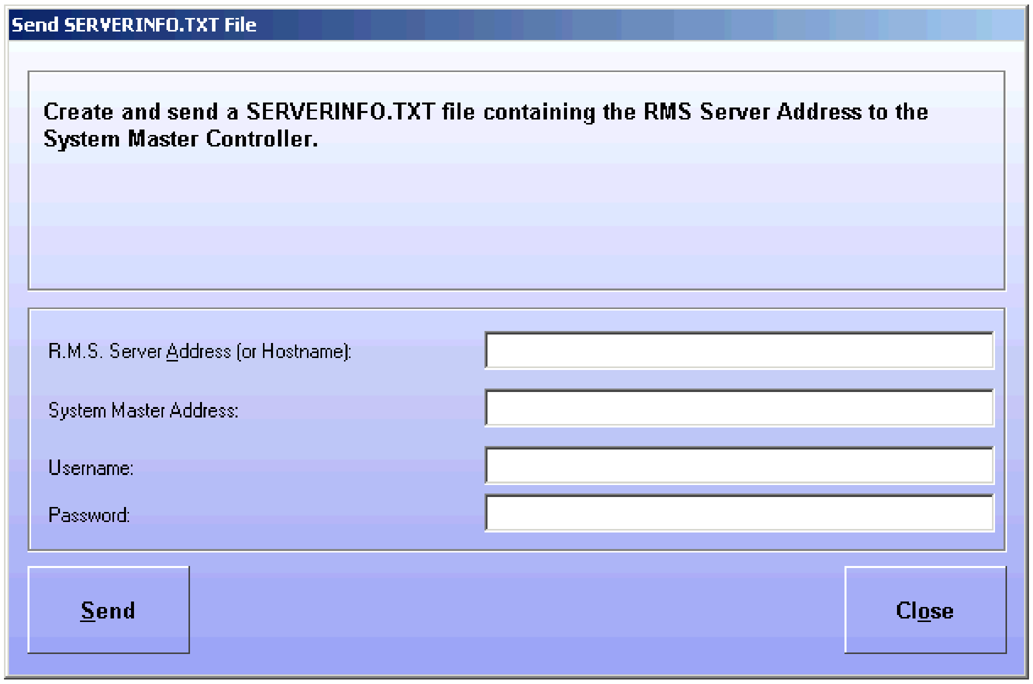
Importing RMS SDK Spreadsheets

Earlier releases of RMS SDK utilized an Excel spreadsheet that RMS CodeCrafter now replaces. You can import your device and parameter information from the Excel spreadsheet into RMS CodeCrafter.

1. Select **File > Open > Import Spreadsheet**.
2. Browse to your desired .XLS file and click Open.

SERVERINFO.TXT Window

RMS CodeCrafter allows you to generate a text file, SERVERINFO.TXT, that contains the IP address of the RMS server, and then transfer the file to the RMS directory on the system master for the room.


Send SERVERINFO.TXT File

Create and send a SERVERINFO.TXT file containing the RMS Server Address to the System Master Controller.

R.M.S. Server Address (or Hostname):	
System Master Address:	
Username:	
Password:	

FIG. 1 SERVERINFO.TXT Window

RMS Server Address (or Hostname) - A text field for entering either the IP address or hostname of the RMS server.

System Master Address - A text field for entering either the IP address or hostname of the master controller for the room.

Username - A text field for entering the FTP username on the master controller for the room.

Password - A text field for entering the FTP password on the master controller for the room.

Send - Creates the text file, places the file in the Temp directory, attempts file transfer and then deletes the file upon successful transfer.

Close - Closes the window without saving and no action taken.

Exiting RMS CodeCrafter

You've created a device template, implemented said template into a project file and then used RMS CodeCrafter to generate the inclusion file code. Now you would like to close the project file and exit RMS CodeCrafter.

- Select **File > Exit**. If you have not already saved you are prompted to do so at this time.

Adjusting RMS CodeCrafter

Changing Visual Style

To change the appearance of RMS CodeCrafter:

Select **View > Visual Styles** and select from the following

- Default
- Office XP®
- Office 2003®

Web Update

Newer versions of RMS CodeCrafter can now be acquired via the internet. The Web Update utility accesses the AMX web site with the application ID and version number and searches for updates.

1. Go to **Help > Web Update** to check for updates.
2. If updates are available for the installed version of RMS CodeCrafter they are downloaded at this time.
3. After the download is complete Web Update closes RMS CodeCrafter and the updates are added.

If you do not have Web Update installed, you receive a window indicating where it can be downloaded.

Setting Template Folder Destination

You can create persistent templates for monitored devices, where you provide the settings and parameters. RMS CodeCrafter can then use the device templates for similar or the same devices in other projects.

RMS CodeCrafter will only look in this folder destination for devices, and templates are saved in this target location by default. If you save your templates in a location other than what is listed here, RMS CodeCrafter will not find them.

1. With RMS CodeCrafter open to the Introductory page, click **Edit Preferences** to open the Preferences window.
2. In the Template Folder field, either type or click **Browse** to set the target location.
3. Click **OK**.

Setting Default RMS Server Address

As part of the code generated by RMS CodeCrafter, the IP address of the RMS server can be included. The IP address is necessary when using Scheduling and i!-ConnectLinx options.

To set the location RMS CodeCrafter uses:

1. With RMS CodeCrafter open to the Introductory page, click **Edit Preferences** to open the Preferences window.
2. Type the IP address of the RMS server in the field, *Default RMS Server Address*.
3. Click **OK**.

Setting Wizard to Generate Device Variable Warnings

RMS CodeCrafter generates an inclusion code file. You can designate if you want Device Variable Warnings listed in the code file. By default the warnings are enabled, to disable:

1. With RMS CodeCrafter open to Introductory page, click **Edit Preferences** to open the Preferences window.
2. Click the checkbox next to *Generate Device Variable Warnings* to remove the check.
3. Click **OK**.

Code Generation

RMS CodeCrafter will output a single, all-inclusive NetLinx include (.axi) file, with the following features:

File Header

RMS CodeCrafter should include a header including an AMX copyright notice, an RMS SDK version, and information about the RMS CodeCrafter program such as name and version.

```
(**********************************************************************)
(*                                                                    *)
(*              AMX Resource Management Suite  (3.2.25)               *)
(*                                                                    *)
(**********************************************************************)
/*
 *  Legal Notice:
 *
 *     Copyright, AMX LLC, 2008
 *
 *     Private, proprietary information, the sole property of AMX LLC.  The
 *     contents, ideas, and concepts expressed herein are not to be disclosed
 *     except within the confines of a confidential relationship and only
 *     then on a need to know basis.
 *
 *     Any entity in possession of this AMX Software shall not, and shall not
 *     permit any other person to, disclose, display, loan, publish, transfer
 *     (whether by sale, assignment, exchange, gift, operation of law or
 *     otherwise), license, sublicense, copy, or otherwise disseminate this
 *     AMX Software.
 *
 *     This AMX Software is owned by AMX and is protected by United States
 *     copyright laws, patent laws, international treaty provisions, and/or
 *     state of Texas trade secret laws.
 *
 *     Portions of this AMX Software may, from time to time, include
 *     pre-release code and such code may not be at the level of performance,
 *     compatibility and functionality of the final code. The pre-release code
 *     may not operate correctly and may be substantially modified prior to
 *     final release or certain features may not be generally released. AMX is
 *     not obligated to make or support any pre-release code. All pre-release
 *     code is provided "as is" with no warranties.
 *
 *     This AMX Software is provided with restricted rights. Use, duplication,
 *     or disclosure by the Government is subject to restrictions as set forth
 *     in subparagraph (1)(ii) of The Rights in Technical Data and Computer
 *     Software clause at DFARS 252.227-7013 or subparagraphs (1) and (2) of
 *     the Commercial Computer Software Restricted Rights at 48 CFR 52.227-19,
 *     as applicable.
*/
```

RMS CodeCrafter obtains the RMS SDK version by reading a registry key set by the SDK installer, HKEY_LOCAL_MACHINE\SOFTWARE\AMX Corp.\RMS SDK\Version.

DEFINE_DEVICES Section

Device Definitions

RMS CodeCrafter will generate device definitions for the following devices in the DEFINE_DEVICE section of the generated file, if they cannot be located in the integrated source file:

- vdvRMSEngine – the RMS Engine virtual device
- dvRMSEngine – the socket device for communication with the RMS server
- vdvCLActions – the i!-ConnectLinx virtual device.

The values used in these definitions are those entered on the RMS Devices wizard page.

Device Definition Warnings

For all other devices referenced in the code-generation process, RMS CodeCrafter will optionally create warnings that notify the user that the device has not been defined. Below is one such warning:

```
// dvTP3a
#IF_NOT_DEFINED dvTP3a
#WARN 'RMS: This Device Needs to be Defined in your Main Program: dvTP3a'
#END_IF
```

An option is provided to enable/disable these warnings. By default they are enabled.

DEFINE_CONSTANT Section

Server Address Definition

RMS CodeCrafter will create a definition for the server address, using the value provided in the RMS Server Address Page.

```
// RMS Server IP
CHAR RMS_SERVER_IP[100]    = '192.168.222.51'
```

Maximum String/Enum Param Length

RMS CodeCrafter will create a definition for RMS_MAX_PARAM_LEN constant, with a value of 100.

```
// Max String/Enum Param Length
RMS_MAX_PARAM_LEN          = 100
```

RMSCommon.axi

RMS CodeCrafter will create an include statement for the RMSCommon.axi include file. This is inserted after the DEFINE_CONSTANT section and prior to the DEFINE_VARIABLE section.

```
(***********************************************************)
(*                 INCLUDE FILES GO BELOW                  *)
(***********************************************************)
#INCLUDE 'RMSCommon.axi'
```

DEFINE_VARIABLE Section

Device Arrays

- **Main Panels** - CodeCrafter creates a device array for all Main Panels. This array is used as a parameter to the RMSUIMod and RMSHelpUIMod modules.

  This array is created only if the user selects *Main and Welcome Panels* under Scheduling Options, or selects the *Include Help Desk Code* option.
- **Welcome Panels** - CodeCrafter creates a device array for all Welcome Panels.

  This array is used as a parameter to the RMSUIMod and/ or RMSWelcomeOnlyUIMod module. This array is created only if the user selects either Main and Welcome Panels or Welcome Panels Only under Scheduling Options.

i!-ConnectLinx variables

- **String and Level parameters** - CodeCrafter creates variables to hold string and level parameters from i!-ConnectLinx if the i!-ConnectLinx module is included. These variables are arrays and have a length of three (3) to support standard i!-ConnectLinx functions.

  These arrays will only be included if Pre-Meeting Presets and/or Monitor Source are selected in the i!-ConnectLinx Options page.

```
// i!-ConnectLinx parameter storage
VOLATILE SLONG   asnNumberLevelArgValues[3]
VOLATILE CHAR    acStringEnumArgValues[3][50]
```

- **Channel Arrays** - CodeCrafter creates two channel arrays for i!-ConnectLinx if i!-ConnectLinx/ touch panel button mapping has been defined.

```
 // i!-ConnectLinx Standard Actions
VOLATILE INTEGER nchCLButtons[] =
{
  1002  // Power Off
// i!-ConnectLinx Touch Panel Buttons
VOLATILE INTEGER nchCLPanelButtons[] =
{
  1     // Power Off
}
```

Device Parameters

CodeCrafter creates a variable for each device parameter specified, of the following types:

- Number – Signed Long (SLONG)
- Index – Integer (INTEGER)s
- String – String (CHAR Array), length is RMS_MAX_PARAM_LEN
- Enum– String (CHAR Array), length is RMS_MAX_PARAM_LEN

Each variable name will contain one of the following Hungarian prefixes, based on type:

- Signed Long (SLONG) – sl
- Integer (INTEGER) – n
- CHAR array – c

The name itself is constructed as follows:

```
<prefix> + "RMS" + <device logical name> + <parameter name>
```

Each variable definition is defined as VOLATILE.

Each variable definition is preceded on the line above it by a comment containing its corresponding device's name.

Each definition is immediately followed on the same line with a comment containing the parameter name, like so:

```
// dvLProj
VOLATILE SLONG   slRMSLeftProjectorLampHours  // Lamp Hours
```

Function Definitions

RMSCommon Callbacks

RMS CodeCrafter creates definitions for the following callback functions:

RMSDevMonRegisterCallBack

This function has the following signature:

```
 (***************************************)
 (* Call Name: RMSDevMonRegisterCallback*)
 (* Function:  time to register devices *)
 (* Param:     None                     *)
 (* Return:    None                     *)
 (* Note:      Caller must define this  *)
 (***************************************)
DEFINE_FUNCTION RMSDevMonRegisterCallback()
{
}
```

Each monitored device and parameter not monitored with a support module will add several lines of code within this function.

- A comment containing the devices logical name
- A call to RMSRegisterDevice()
- A call to RMSRegisterDeviceNumberParam()

RMSDevMonSetParamCallBack

This function has the following signature:

```
 (***************************************)
 (* Call Name: RMSDevMonSetParamCallback*)
 (* Function:  Reset parameters         *)
 (* Param:     DPS, Name, Value         *)
 (* Return:    None                     *)
 (* Note:      Caller must define this  *)
 (***************************************)
DEFINE_FUNCTION RMSDevMonSetParamCallback(DEV dvDPS, CHAR cName[], CHAR cValue[])
{
}
```

Each device parameter with Reset selected should have an entry in an ACTIVE/CASE statement. Further, all parameters of the same device is grouped in the same ACTIVE/CASE statement. As in the following example:

```
SELECT
{
  // Left Projector
  ACTIVE (dvLProj == dvDPS):
  {
    IF ('Lamp Hours' == cName)
      slLeftProjectorLampHours = ATOI(cValue)
  }
}
```

RMS Device Parameters

A custom function is generated for each parameter to set the value of the parameter and report it to the RMS server. This function will follow the following structure:

```
(*******************************************)
(* Call Name: RMSSetLeftProjectorLampHours *)
(* Function:  Set Lamp Hours               *)
(* Param:     Value                        *)
(* Return:    None                         *)
(*******************************************)
DEFINE_FUNCTION RMSSetLeftProjectorLampHours(SLONG slValue)
LOCAL_VAR
CHAR bInit
{
  IF (slRMSLeftProjectorLampHours <> slValue || bInit = FALSE)
    RMSChangeNumberParam(dvLProj,'Lamp Hours',RMS_PARAM_SET,slValue)
  slRMSLeftProjectorLampHours = slValue
  bInit = TRUE
}
```

Module Definitions

RMS CodeCrafter generates the following module definitions in the DEFINE_START section.

Monitored Device/Support Modules

RMS CodeCrafter generates a module definition for each Monitored device that uses a Support Module. Each module declaration is created using the Module name, the monitored devices virtual and actual device names, and the RMS Engine’s virtual device name (vdvRMSEngine).

Each module definition is preceded by a comment containing the device name of the monitored device.

Source Usage

This module definition is generated if the user selects *Monitor Source Usage* in the i!-ConnectLinx Options page.

```
// RMSSrcUsageMod - Tracks equipment usage.
DEFINE_MODULE 'RMSSrcUsageMod' mdlSrcUsage(vdvRMSEngine,
                                           vdvCLActions)
```

RMS Engine

This module definition is always generated boilerplate.

```
// RMSEngineMod - The RMS engine.  Requires i!-ConnectLinxEngineMod.
DEFINE_MODULE 'RMSEngineMod' mdlRMSEng(vdvRMSEngine,
                                       dvRMSSocket,
                                       vdvCLActions)
```

RMSUIMod

This module definition is generated if the user selects *Main and Welcome Panels* in the Scheduling Options.

```
// RMSUIMod - The RMS User Interface.
// Channel And Variable Text Code Defined Inside The Module
DEFINE_MODULE 'RMSUIMod' mdlRMSUI(vdvRMSEngine,
                                  dvRMSTP,
                                  dvRMSTPWelcome)
```

RMSWelcomeOnlyUIMod

This module definition is generated if the user selects *Welcome Panels Only* in the Scheduling Options.

```
// RMSWelcomeOnlyUIMod - The RMS Welcome Panel User Interface.
// Channel And Variable Text Code Defined Inside The Module
DEFINE_MODULE 'RMSWelcomeOnlyUIMod' mdlRMSUI(vdvRMSEngine,
                                             dvRMSTPWelcome)
```

i!-ConnectLinxEngineMod

This module definition is generated if the user selects *Pre-Meeting Presets* and/or *Monitor Source Usage* in the i!-ConnectLinx Options page.

```
// i!-ConnectLinxEngineMod
DEFINE_MODULE 'i!-ConnectLinxEngineMod' mdlCL(vdvCLActions)
```

RMSHelpUIMod

This module definition is enabled with the Include Help Desk Code option.

```
DEFINE_MODULE 'RMSHelpUIMod' mdlRMSHelpUI(vdvRMSEngine,
                                          dvRMSTP)
```

Event Definitions

RMS CodeCrafter generates the following DATA_EVENT blocks the DEFINE_EVENT section.

RMS Engine Device

RMS CodeCrafter generates a DATA_EVENT block for the RMS Engine device. This block will always be generated, boilerplate, with these exceptions:

- If the Use Multiple Displays option is selected, RMS CodeCrafter will generate a call to RMSSetMultiSource(TRUE)
- A call to RMSSetDeviceInfo() is generated for each monitored device using an RMS Support Module; If the device has a non-negative timeout value, a call to RMSSetCommunicationTimeout() is generated; Finally, if the device has Detect Control failure enabled, a call to RMSEnablePowerFailure is generated.

See the example block below:

```
(********************************************)
(* DATA: RMS Engine                         *)
(********************************************)
DATA_EVENT[vdvRMSEngine]
{
  ONLINE :
  {
    // Configure RMS Server Address
    RMSSetServer(RMS_SERVER_IP)
    // Track Multiple Sources
    RMSSetMultiSource(TRUE)
    // Display
    RMSSetDeviceInfo(dvProj,'Display','Display Manufacturer','Display Model')
    RMSSetCommunicationTimeout(dvProj,200)
    RMSEnablePowerFailure(dvProj)
  }
}
```

Named Devices

RMS CodeCrafter generates a DATA_EVENT block for each Named Device. This block will always be generated as follows:

```
(********************************************)
(* DATA: Main Touch Panel                   *)
(********************************************)
DATA_EVENT [dvTP1a]
{
  ONLINE:
    RMSNetLinxDeviceOnline(dvTP1a,'Main Touch Panel')
  OFFLINE:
    RMSNetLinxDeviceOffline(dvTP1a)
}
```

Monitored Devices

RMS CodeCrafter generates a DATA_EVENT block for each Monitored Device not using a support module. This block will always be generated as follows:

```
(*******************************************)
(* DATA: Left Projector                    *)
(*******************************************)
DATA_EVENT [dvLProj]
{
  ONLINE:
  {
    // Register Device and Parameters
    RMSRegisterDevice(dvLProj,'Left Projector','Manufacturer','Model')
    RMSRegisterDeviceNumberParam(dvLProj,'Lamp
Hours',1000,RMS_COMP_GREATER_THAN,RMS_STAT_MAINTENANCE,TRUE,0,RMS_PARAM_SET,slRMS
LeftProjectorLampHours,0,0)
  }
  OFFLINE:
    RMSNetLinxDeviceOffline(dvLProj)
}
```

Device Parameters

RMS CodeCrafter generates an event handler for each device parameter of BUTTON. LEVEL, or CHANNEL type. See the RMS CodeCrafter Requirements specification for an example.

i!-ConnectLinx

RMS CodeCrafter generates an DATA_EVENT block for the i!-ConnectLinx virtual device. If Pre-Meeting Presets is selected, a LEVEL_EVENT block will also be created.

Additionally, a BUTTON_EVENT and a CHANNEL_EVENT is included to connect i!-ConnectLinx buttons to touch panel buttons.

LEVEL_EVENT Block

This block is of fixed format and is generated as follows. It does not make use of the i!-ConnectLinx variables defined in the variables section.

```
(*******************************************)
(* LEVEL: i!-ConnectLinx Engine            *)
(*******************************************)
LEVEL_EVENT[vdvCLActions,0]
{
  // Store it if we have room
  IF (MAX_LENGTH_ARRAY(asnNumberLevelArgValues) >= LEVEL.INPUT.LEVEL)
    asnNumberLevelArgValues [LEVEL.INPUT.LEVEL] = LEVEL.Value
}
```

DATA_EVENT Block

RMS CodeCrafter generates a DATA_EVENT block for the vdvCLActions virtual device. This block will create the following:

A fixed format STRING parsing block

A SEND_COMMAND statement to register the room name, owner, and location.

Registration commands for each function-touch panel button mapping. These registration commands can be combined using the AND and THROUGH syntax.

An example block is provided below:

```
(*******************************************)
(* DATA: i!-ConnectLinx Engine             *)
(*******************************************)
DATA_EVENT[vdvCLActions]
{
  STRING:
  {
    STACK_VAR
    CHAR    cTemp[1000]
    CHAR    cTrash[10]
    INTEGER nId
    // Look for arguments
    IF (LEFT_STRING(DATA.TEXT,3) = 'ARG')
    {
      // Get arg ID
      cTemp = DATA.Text
      cTrash = REMOVE_STRING(cTemp,'ARG',1)
      nId = ATOI(cTemp)
      cTrash = REMOVE_STRING(cTemp,'-',1)
      // Store it if we have room
      IF (MAX_LENGTH_ARRAY(acStringEnumArgValues) >= nId)
        acStringEnumArgValues [nId] = cTemp
    }
  }
  ONLINE:
  {
    // Set Room Info
    SEND_COMMAND DATA.DEVICE,'SET ROOM INFO- Sample,Richardson TX,AMX Corp.'
    // Configure by Macros
    SEND_COMMAND DATA.DEVICE,'ADD MACRO-power'
    SEND_COMMAND DATA.DEVICE,'ADD MACRO-dvd'
    SEND_COMMAND DATA.DEVICE,'ADD MACRO-cd'
    SEND_COMMAND DATA.DEVICE,'ADD MACRO-lights'
    SEND_COMMAND DATA.DEVICE,'ADD MACRO-blinds'
    // System Controls
    //   Power Off (1002)
    //   Select VHS (1011)
    //   Select DVD (1014)
    //   Select Video Conference (1017)
    //   Select Rack Computer (1021)
    //   Select Aux PC Input (1022)
    //   Select Slide (slide to video) (1024)
    //   Select CD Player (1042)
    SEND_COMMAND DATA.DEVICE,'ADD STD-1002&1011&1014&1017&1021-1022&1024&1042'
    // Lighting
    //   Lights All Off (1061)
    //   Lights All On (1062)
    //   Lights Meeting Mode (1063)
    //   Lights Presentation Mode (1064)
    //   Lights Conference Mode (1065)
    SEND_COMMAND DATA.DEVICE,'ADD STD-1061-1065'
    // Blinds
    //   Blinds Open (1095)
    //   Blinds Close (1096)
    SEND_COMMAND DATA.DEVICE,'ADD STD-1095-1096'
    // VCR - VHS
    //   VCR Play (1131)
    //   VCR Stop (1132)
    //   VCR Pause (1133)
    //   VCR Fast Forward (1134)
    //   VCR Rewind (1135)
    //   VCR Search Fwd (1136)
    //   VCR Search Rev (1137)
    SEND_COMMAND DATA.DEVICE,'ADD STD-1131-1137'
    // DVD
    //   DVD Play (1221)
    //   DVD Stop (1222)
    //   DVD Pause (1223)
    //   DVD Skip Forward (1224)
    //   DVD Skip Back (1225)
    //   DVD Search Fwd (1226)
    //   DVD Search Rev (1227)
    SEND_COMMAND DATA.DEVICE,'ADD STD-1221-1227'
    // CD Player
    //   CD Play (1731)
    //   CD Stop (1732)
    //   CD Pause (1733)
    //   CD Skip Forward (1734)
    //   CD Skip Back (1735)
    //   CD Search Fwd (1736)
    //   CD Search Rev (1737)
    SEND_COMMAND DATA.DEVICE,'ADD STD-1731-1737'
  }
}
```

BUTTON_EVENT Block

RMS CodeCrafter generates a BUTTON_EVENT block for each device with button mappings. It does not make use of the i!-ConnectLinx variables defined in the variables section. An example block is provided below.

```
(*********************************************)
(* BUTTON: i!-ConnectLinx Engine            *)
(*********************************************)
BUTTON_EVENT[vdvCLActions,nchCLButtons]
{
  PUSH:
    DO_PUSH(dvTP1a,nchCLPanelButtons[GET_LAST(nchCLButtons)])
}
```

CHANNEL_EVENT Block

RMS CodeCrafter generates a CHANNEL_EVENT block for each device with button mappings. It does not make use of the i!-ConnectLinx variables defined in the variables section. An example block is provided below.

```
 (*******************************************)
 (* CHANNEL: i!-ConnectLinx Feedback        *)
 (*******************************************)
CHANNEL_EVENT[dvTP1a,nchCLPanelButtons]
{
  ON:
    ON[vdvCLActions,nchCLButtons[GET_LAST(nchCLPanelButtons)]]
  OFF:
    OFF[vdvCLActions,nchCLButtons[GET_LAST(nchCLPanelButtons)]]
}
```

Code Generation for Anterus RFID Support

Anterus Duet Virtual Device

If support is enabled in the CodeCrafter project for Anterus, then the code generator will need to check the user's AXS file for the defined virtual device:

```
vdvAnterusGateway = 41001:1:0      (* Duet RFID Virtual Device *)
```

If the device is not found, then the code generator will need to add a device definition in the generated AXI file for this device. This works exactly like the existing CodeCrafter behavior for the RMS Engine virtual device and the i!-ConnectLinx virtual device.

Anterus RFID Readers Device Array

If support is enabled in the CodeCrafter project for Anterus, then the code generator will need generate a DEVICE ARRAY that includes all the user defined RFID reader devices:

```
// Anterus RFID Readers Device Array
//  (all RFID readers must be included in this array!)
VOLATILE DEV dvAnterusReaders[] =
{
  dvAnterusReader1,    // RFID Reader (Front)
  dvAnterusReader2     // RFID Reader (Back)
}
```

The reader's logical name is placed in a comment to the right of each device array element. This works similar to the device arrays the code generates creates for the touch panel devices.

Anterus & RMS RFID Modules

If support is enabled in the CodeCrafter project for Anterus, then the code generator will need include the following modules:

```
// Anterus RFID Duet Module
DEFINE_MODULE 'AMX_Anterus_Comm_dr1_0_0'
mdlAnterusDuetMod(vdvAnterusGateway,dvAnterusReaders[1])

// RMS RFID Tag Tracking Module
//   (This module is used to synchronize RFID tag changes with the RMS engine.)
DEFINE_MODULE  'RMSRFIDTrackingMod'
mdlRMSRFIDTracking(vdvAnterusGateway,vdvRMSEngine)
```

This works similar to the other modules that the code generator already includes.

Anterus RFID Device Registration & Monitoring

If support is enabled in the CodeCrafter project for Anterus, then the code generator will need include the following code.

In the "RMSDevMonRegisterCallback()" function, each RFID reader device will need to call into the "RMSNetLinxDeviceOnline" function, passing the RFID reader device name and its associated logical name.

```
DEFINE_FUNCTION RMSDevMonRegisterCallback()
{
  // Anterus RFID Reader Devices
  RMSNetLinxDeviceOnline(dvAnterusReader1,'RFID Reader (Front)')
  RMSNetLinxDeviceOnline(dvAnterusReader2,'RFID Reader (Back)')
}
```

In addition to the registration function calls above, the code generator must also create a DATA_EVENT for each reader device as shown below:

```
(********************************************)
(* DATA:  RFID Reader (Front)              *)
(********************************************)
DATA_EVENT [dvAnterusReader1]
{
  ONLINE:
    RMSNetLinxDeviceOnline(dvAnterusReader1,'RFID Reader (Front)')
  OFFLINE:
    RMSNetLinxDeviceOffline(dvAnterusReader1)
}
(********************************************)
(* DATA:  RFID Reader (Back)               *)
(********************************************)
DATA_EVENT [dvAnterusReader2]
{
  ONLINE:
    RMSNetLinxDeviceOnline(dvAnterusReader2,'RFID Reader (Back)')
  OFFLINE:
    RMSNetLinxDeviceOffline(dvAnterusReader2)
}
```

This works very similar to how to code generator deals with NetLinx Named Devices.

Anterus Module Identifiers

If support is enabled in the CodeCrafter project for Anterus then the code generator will need to generate the following code:

A single copy of the following function will need to be included in the AXI file.

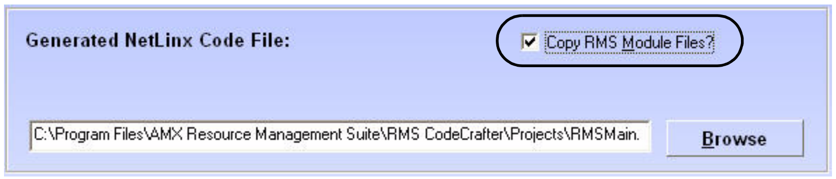
```
(***************************************************************)
(* Call Name: RMSSendRFIDModuleIdentifiers                    *)
(* Function:  Identify the reader instances to the Duet mod.  *)
(* Param:     ReadersDeviceArray, AnterusDuetVirtualDevice    *)
(* Return:    None                                            *)
(***************************************************************)
DEFINE_FUNCTION RMSSendRFIDModuleIdentifiers(DEV dvRFIDReaders[], DEV vdvRFIDGtwy)
{
  STACK_VAR
  INTEGER nLoop
  CHAR cRFIDReaderDeviceIdentifiers[200]

  // loop over the RFID reader device array and construct an identifiers string
     to send to the Duet module
  FOR(nLoop = 1; nLoop <= LENGTH_ARRAY(dvRFIDReaders); nLoop++)
  {
    // append device number to string
    cRFIDReaderDeviceIdentifiers =
"cRFIDReaderDeviceIdentifiers,ITOA(dvRFIDReaders[nLoop].Number)"

    // if this is not the last device in the array, then append a semicolon
       delimiter character
    IF(nLoop != LENGTH_ARRAY(dvRFIDReaders))
      cRFIDReaderDeviceIdentifiers = "cRFIDReaderDeviceIdentifiers,';'"
  }

  // send the IDENTIFIERS property to the Anterus Duet module to identify
     additional reader devices
  SEND_COMMAND vdvRFIDGtwy,"'PROPERTY-Identifiers,',cRFIDReaderDeviceIdentifiers"
  SEND_COMMAND vdvRFIDGtwy,"'REINIT'"
}
```

Anterus DATA_EVENT

If support is enabled in the CodeCrafter project for Anterus, then the code generator will include a single copy of the following DATA_EVENT in the AXI file:

```
(*******************************************)
(* DATA: Anterus RFID Duet Module          *)
(*******************************************)
//
// We need to identify additional Anterus RFID reader devices to the Anterus RFID
// Duet module. We do this by sending the 'PROPERTY-Identifiers,' command to the
// Anterus Duet module virtual device with a semicolon delimited listing of
// additional device numbers when the Anterus Duet module comes online.
// This command is followed by a 'REINIT' command to ensure the Duet module is
// initialized properly.
DATA_EVENT[vdvAnterusGateway]
{
  ONLINE:
  {
    // send the IDENTIFIERS property to the Anterus Duet module to identify
       additional reader devices
    RMSSendRFIDModuleIdentifiers(dvAnterusReaders, DATA.DEVICE)
  }
}
```

Anterus Copy Files



Generated NetLinx Code File:

[✓] Copy RMS Module Files?

C:\Program Files\AMX Resource Management Suite\RMS CodeCrafter\Projects\RMSMain.

FIG. 1 Generate NetLinx Code File page - Copy RMS Module Files option

If support is enabled in the CodeCrafter project for Anterus AND the "Copy RMS Module Files" option is checked on the *Generate NetLinx Code File* page, then the code generator will also include the following files:

- AMX_Anterus_Comm_dr1_0_0.jar (*only included if this file exists in the project directory*)
- RMSRFIDTrackingMod.axs

RMS Concepts

Device Monitoring Framework

RMS provides device monitoring through a user extensible framework. This framework allows you to customize what devices are monitored, the conditions that indicate a problem or fault and what type of problem or fault this condition represents. RMS generates notifications when a fault condition occurs, as determined by the notification configuration.

Each room has one or more monitored devices. Each device can be a physical device, such as a video projector, or a logical device, like the RMS software. However, each monitored device must be associated with a NetLinx-connected device. In the case of a video projector, this device would be the IR card, Serial Card or IP Socket used to communicate with the projector. The RMS software is associated with the NetLinx master itself.

Each monitored device has one or more device parameters that represent monitored items. For instance, monitoring lamp hours of a video projector is accomplished through a "Lamp Hours" parameter that belongs to the "Video Projector" device. All parameters must be associated with a device.

In order to monitor a device, the NetLinx system must register the device and one or more parameters with RMS. For instance, monitoring of Lamp Hours of the Video Projector is only available if the NetLinx system has added the appropriate code. In many cases, this is as simple as adding a RMS support module.

Device Values

Each monitored device has a set of values used in its description. These values are supplied when the device is registered and consist of the following:

Device Values	
Device Number	This is the device number of the device, as defined in the NetLinx program. Devices are tracked by Device ID so this value must be unique within the devices of a given room. For instance, you can have multiple "1:1:0" devices as long as there is only 1:1:0" device per room.
Name	This is the name of device. This name is displayed on the administrators console and readily identifies the device.
Manufacturer	This is the manufacturer of the device. If this value is not supplied during registration, the manufacturer of the NetLinx-connected device will be used.
Model	This is the model number of the device. If this value is not supplied during registration, the model name of the NetLinx-connected device will be used.
Device Type	This is the device type of the NetLinx-connected device. This might be "NI-2000" or "NXP-TPI/4 Touch Panel". This is available for Axcess and NetLinx devices. This information is registered automatically by the RMS server.
Serial Number	This is the serial number of the NetLinx-connected Device. This is only available for NetLinx devices. This information is registered automatically by the RMS server.
Firmware Version	This is the firmware version of the NetLinx-connected device. This is only available for NetLinx devices. This information is registered automatically by the RMS server.
Address and Address Type	This is the physical address and address type for the NetLinx-connected device. This information describes how the device is connected to the NetLinx master. A device connected via ICSNet will display "ICSNet" for the address type and the hardware's network address for the address. A device connected via IP will display "TCP/IP" for the address type and the IP address for the address. Axcess devices will display "AxLink" for both values. This information may be useful for diagnosing device connectivity problems. This information is registered automatically by the RMS server.

Parameter Values

Each parameter has a set of values used to determine what conditions indicate a problem and what type of problem this condition represents. These values are supplied when the parameter is registered and consist of the following:

Parameter Values	
Name	This is the name of the parameter. This name is displayed on the RMS server console and readily identifies the parameter. Parameters are tracked by name so this name must be unique within the parameters of a given device. For instance, you can have multiple "Lamp Hours" parameters as long as there is only one "Lamp Hours" parameter per monitored device.
Parameter Type	This value indicates if this value is a number or a string. This information is used to determine how to perform certain operation, such as addition and comparisons between the new and threshold values. For instance, comparing "10" and '2" as strings results in "10" less than "2" but comparing them as numbers results in "2" less than "10".
Value and Units	This is the current value of the parameter. Units are appended to the value when displayed in the web console.
Threshold Value and Comparison Operator	The threshold value is the value for which this parameter is considered to indicate a problem or fault. The comparison operator is used to detect when the value changes from the un-faulted to the faulted condition. The comparison operators "Less Than", "Less Than or Equal To", "Greater Than", "Greater Than or Equal To", "Equal To" and "Not Equal To" can be used for strings and number parameters. The comparison operators "Contains" and "Does Not Contain" are primarily used for string parameters. For example, "Lamp Hours" might have a threshold value of 1000 and any value over this would require maintenance. The comparison operator would then be "Greater Than". When this parameter changes from a value that is not greater than 1000 to a value that is greater than 1000, the fault status is set. When the value changes from a value greater than 1000 to a value not greater than 1000, the fault status is cleared. These value are supplied during registration but can be modified by the administrator from the RMS server console.
Status Type	The status represents the type of problem a faulted condition represents. Status Types include "Help Request", "Maintenance Request", "Room Communication Error", "Control System Error", "Network Error", "Security" and "Equipment Usage." For example, when "Lamp Hours" changes from an un-faulted (not greater than 1000) to a faulted (greater than 1000), this change represents a "Maintenance Request" status that requires an AV technician to repair the equipment. If the "Device Online" parameter changes from "Online" to "Offline", this change could represent a "Security" or "Control System Error" status. These value are supplied during registration but can be modified by the administrator from the RMS server console.
Reset Flag and Reset Value	These values determine if and how the parameter can be reset from the RMS server console. If the Reset Flag is set, then the administrator can reset the value remotely. When the administrator selects "Reset" from the console, the Reset Value is copied to the Value and the faulted condition is cleared. These values are useful for parameters such as VCR "Run Time" which would be manually reset when the VCR is cleaned.
Minimum and Maximum Values	These values are used to restrict the range of the threshold and reset values that the administrator can enter on the RMS server console. These values would be used when the parameter represents a value with a bounded range, such as a Volume Level.
Enumeration List	This value is used to restrict the range of the threshold and reset values that the administrator can enter on the RMS server console. This value would be used when the parameter represents a value with a bounded list, Power On and Power Off.

All parameters must be registered by the NetLinx system. The administrator cannot add parameters from the RMS console. The administrator can modify Threshold Value, Comparison Operator and Status Type for any parameter. This provides the administrator with the ability to set their own threshold and re-classify messages based on their facility. For instance, an administrator can set the Video projector's "Lamp Hours" threshold to the expected lamp life of a newly replaced lamp or change the "Device Communicating" parameter from a "Control System Error" to a "Security" status if the projector is in danger of being stolen.

Status Types

RMS supports the following status types for device monitoring: "Help Request", "Maintenance Request", "Room Communication Error", "Control System Error", "Network Error", "Security" and "Equipment Usage."

While there are no firm rules for what these status types mean and how they are used, AMX provides the following description of each status type and recommends that your usage is consistent with these descriptions.

Status Types	
Help Request	A user generated help request such as a help button on the touch panel.
Maintenance Request	A user or monitored equipment generated maintenance request. Maintenance issues would include items that require a technician to visit the room.
Room Communication Error	A loss of communication between the room and the RMS server.
Control System Error	Any error that represents a control system error, such as an offline device or loss of communication with a device.
Network Error	Any network related error. These would most commonly be associated with loss of communication with devices that communicate via IP.
Security	Any security related issue. It might be appropriate to classify issues that might normally be classified as Control System or Network errors as Security issues instead. This might include a touch panel going offline or loss of communication with a projector depending on the physical security of these devices.
Equipment Usage	Any issue that does not require repair or maintenance and that is mainly used for status.

RMS SDK And RMS CodeCrafter

The RMS SDK consists of a series of modules to simplify device monitoring programming. Device monitoring module handles the registration of devices and parameters and keeping track of lamp hours and transport run time. In most cases, adding device monitoring is achieved by selecting the appropriate device monitoring module and adding code to inform the module of important device changes. The RMS support modules register and monitor the following parameters:

Basic Device (RMSBasicDeviceMod):

Device Online/Offline, Power, Communication Status for Serial devices, Control Failure (Optional), IP Address of Socket-based devices.

Projector (RMSProjectorMod):

Device Online/Offline, Power, Lamp Hours, Communication Status for Serial devices, Control Failure (Optional), IP Address of Socket-based devices.

Transport (RMSTransportMod):

Device Online/Offline, Power, Run Time, Communication Status for Serial devices, Control Failure (Optional), IP Address of Socket-based devices.

Slide Projector (RMSSldProjMod):

Device Online/Offline, Power, Lamp Hours.

i!-ConnectLinx

Overview

i!-ConnectLinx™ is an application that allows you to expose NetLinx™ actions to the outside world. i!-ConnectLinx allows a programmer to define and program actions that can be utilized by other user interfaces or processes outside the NetLinx Control System. For instance, i!-ConnectLinx can be programmed to expose source select functions and i!-ConnectLinx compatible technologies, such as i!-PCLink/PresentationControl™, can use this information to allow the source selects to be executed from a Microsoft PowerPoint® presentation.

i!-ConnectLinx also provides a mechanism to request actions to be executed on the NetLinx Control System. Once a process outside the NetLinx Control System has obtained the action list, the process can then make a request to i!-ConnectLinx to execute that action. i!-ConnectLinx handles this request and makes this request available to the NetLinx program for execution.

i!-ConnectLinxEngineMod, is the main i!-ConnectLinx module that handles exposing and executing action requests. To support i!-ConnectLinx, you simply include this module in your program, define your actions and write programming to support those actions. The i!-ConnectLinxEngineMod module makes the list of actions available to other processes, executes their requests and provides your program with a push when an action needs to be executed.

Using i!-ConnectLinx

Little work is required to add i!-ConnectLinx to your existing NetLinx code. i!-ConnectLinx is implemented as a NetLinx module. Adding the module definition and all it's parameters to your code is all that is required.

In order to use i!-ConnectLinx, you need to program and define a series of actions in the NetLinx Control System. The key to the i!-ConnectLinx engine is the virtual device, `vdvCLActions`. Support the actions you want executed remotely using this virtual device.

Think of the virtual device, `vdvCLActions`, as a touch panel. Normally, you write your NetLinx program to respond to certain push channel from a touch panel; i!-ConnectLinx is exactly the same. Let's say you want the user to be able to allow i!-PCLink/PresentationControl the ability to play and stop a VCR. Imagine you have two touch panel buttons that do these functions; write code that responds to the pushes:

```
BUTTON_EVENT[TP,1]                      (* VCR Play *)
{
  PUSH:
  {
    PULSE[VCR,1]
  }
}
BUTTON_EVENT[TP,2]                      (* VCR Stop *)
{
  PUSH:
  {
    PULSE{VCR,2]
  }
}
```

To expose these actions using i!-ConnectLinx, write the same code substituting the touch panel device for your i!-ConnectLinx virtual device:

```
BUTTON_EVENT[vdvCLActions,1]          (* VCR Play *)
{
  PUSH:
  {
    PULSE[VCR,1]
  }
}
BUTTON_EVENT[vdvCLActions,2]          (* VCR Stop *)
{
  PUSH:
  {
    PULSE{VCR,2]
  }
}
```

When the i!-ConnectLinx engine gets a request to play the VCR, i!-ConnectLinx will "push" the button of the virtual device just like a user pushes a button on a touch panel. There is now only one thing left to do: Tell the user which actions are which.

In order to expose an action for execution via i!-ConnectLinx, you need to support the programming for the action, as we have just seen, and you need to tell i!-ConnectLinx what that action is.

To specify the name of an action, send a command to the i!-ConnectLinx virtual device describing the name of a given channel code. To specify the names of the actions in the above example, you would add some code like this:

```
DATA_EVENT[vdvCLActions]
{
  ONLINE:
  {
    (* Setup actions *)
    (* VCR Play *)
    SEND_COMMAND vdvCLActions," 'ADD ACTION-1,VCR Play' "

    (* VCR Stop *)
    SEND_COMMAND vdvCLActions," 'ADD ACTION-2,VCR Stop' "
}
```

Once i!-ConnectLinx receives these commands, it stores this information in an XML file that can be used by i!-ConnectLinx compatible technologies to browse available actions.

In addition to specifying the name of an action, you can also supply a help string and a folder name. The help string helps a user understand the intent of the action more clearly. The folder name allows you to organize the actions in a tree view so that actions are more easily browsed.

It’s Your World - Take Control™