MICROSAR Post-Build Loadable XML Generator

Technical Reference

Version 1.3.0

Authors	Cornelius Reuß
Status	Released

Document Information

History

Author	Date	Version	Remarks
Cornelius Reuß	2013-04-12	1.0.0	Creation
Cornelius Reuß	2013-09-27	1.1.0	ESCAN00068146 AR-420: Postbuild Loadable Support for Diagnostic Modules
Cornelius Reuß	2013-11-05	1.2.0	ESCAN00071566 AR4-611: Flash-Download Verification Pattern
Cornelius Reuß	2014-09-26	1.3.0	ESCAN00076801, AR4-786: PB-L Diagnostic License

Reference Documents

No.	Source	Title	Version
[1]	Vector	TechnicalReference_PostBuildLoadable.pdf	as delivered
[2]	Vector	TechnicalReference_IdentityManager.pdf	as delivered

Caution
We have configured the programs in accordance with your specifications in the questionnaire. Whereas the programs do support other configurations than the one specified in your questionnaire, Vector´s release of the programs delivered to your company is expressly restricted to the configuration you have specified in the questionnaire.

Caution
This symbol calls your attention to warnings.

Contents

1 Component History

The component history in Table 1-1 gives an overview over the important milestones that are supported in the different versions of the component.

Component Version	New Feature
1.0.0	> Global post-build update for several MICROSAR BSW modules. > Conversion of post-build configuration data into an XML description. > Generation of a Map file with updated configuration data address information.
1.1.0	> Module individual post-build update.
1.2.0	> Support of a Flash-Download Verification Pattern at the end of post-build ROM block. > Support of Post-Build Selectable
1.3.0	> Support Diagnostic License

Table 1-1 Component History

2 Introduction

This document describes the usage of the Post-Build XML Generator which is part of the MICROSAR Post-Build loadable tool chain.

The Post-Build XML Generator is used to extract post-build configuration data from C source file(s) (naming scheme *_PBcfg.c) and to convert this information into an XML description. In a second step the generated XML file(s) can then be transformed into a flashable HEX file by the Post-Build tool chain.

Two post-build update strategies can be distinguished:

- Global update where several BSW modules share a common ROM and RAM post-build memory block.
- Module individual update to allow the update of a single BSW with a non shared ROM and RAM post-build memory block.

Caution
Before continuing with this documentation it is essential that the general concept of the MICROSAR Post-Build Loadable concept is understood. This important information is provided along with a step by step guidance in [1].

3 Functional Description

3.1 Features

Table 3-1 covers the supported features of the Post-Build XML Generator.

Supported Features
Conversion of post-build configuration data into an XML description.
Generation of a Map file with updated configuration data address information.
Global post-build update for several MICROSAR BSW modules.
Module individual post-build update.
Support of a Flash-Download Verification Pattern at the end of post-build ROM block (Note: the flash verification pattern reduces the available ROM block size that can be used for the post-build configuration data by its size).
Support of Post-Build Selectable.

Table 3-1 Supported Features

3.2 Limitations

Table 3-2 covers limitations of the Post-Build XML Generator.

Limitations
Bit-fields are not supported as a part of post-build configuration data.
Pointer to array elements in post-build data: Only pointer to the first element of an array can be resolved by the Post-Build Xml Generator. Pointer to other array elements must not be used in post-build data.
The Post-Build Xml Generator is not able to resolve function addresses.

Table 3-2 Limitations

4 Post-Build XML Generator Interface

4.1 Scope of Delivery

The delivery of the Post-Build Xml Generator has the following content:

Files	Description
PostBuildXmlGen.exe (including several DLLs)	Command line Microsar Post-Build XML Generator (front end)
EcuC_PbXmlGenerator_pre.arxml	Microsar Post-Build XML Generator Preconfig file

Table 4-1 Scope of Delivery

4.2 Command Line API

4.2.1 Invocation

The executable of the Post-Build XML Generator is named PostBuildXmlGen.exe. It can be invoked with the following syntax:

```
PostBuildXmlGen.exe --source=<file|dir> --project=<file>
[--include=<dir>] [--include-recursive=<dir>] [--output=<file|dir>]
[--define=<name>[=<value>]]
```

All options are given in a long-option format. For some options there is also a short-option format available (see chapter 4.2.2). Options which are enclosed in brackets are non-mandatory, the others without brackets are mandatory.

Files <file> are given as relative or absolute path names to files, directories <dir> as relative or absolute paths.

4.2.2 Options

PostBuildXmlGen.exe accepts the command line options listed in Table 4-2.

Long format	Short format	Description
--source=<file|dir>	-s <file|dir>	Specifies a single *_PBcfg.c file or the directory name <dir> which contains the *_PBcfg.c files
--project=<file>	-p <file>	Specifies the file name <file> of the DaVinci Project (*.dpa)
--include=<dir>	-I <dir>	Adds directory name <dir> to include file search path
--include-recursive=<dir>		Adds directory name <dir> and all subfolders which contain header files to include file search path
--output=<file|dir>	-o <file|dir>	Specifies the file name <file> of the output XML file or the name of the output directory
--define=<name>[=<value>]	-D <name>[=<value>]	Defines macro with name <name>
--end-pattern=<value>		Specifies a flash download verification pattern which will be appended at the end of the post-build ROM block (by reducing the available ROM block size for post-build configuration data). <value> must consist of one or several bytes in hexadecimal representation (2 hexadecimal digits per byte). Adjacent bytes have to be separated by colons, e.g. --end-pattern="1a:2b:3c:4d"
--version	-V	Version information
--help	-h, -?	Help

Table 4-2 Command Line Options

4.2.3 Return Codes

PostBuildXmlGen.exe returns the codes specified in Table 4-3 on program termination.

Return Code	Description
0	Success
1	Failure

Table 4-3 Return Code

4.3 Required Header Files

The Post-Build XML Generator requires access to all header files included in the *_PBcfg.c files in order to resolve the type definitions of the post-build data.

Please provide at least the SIP BSW folder and your GenData folder as include directory to the Post-Build XML Generator. Additional include paths, e.g. paths to application specific header files may be required additionally.

Note

If header files from the application are required to resolve all include paths, it is sufficient for the post-build loadable tool chain to provide empty header files.

Type definitions required to resolve the types used in the *_PBcfg.c files should be provided in the header files provided with the BSW module (SIP) or should be generated in the code generation output folder of DaVinci Configurator.

4.4 Platform Settings

The Post-Build XML Generator evaluates the parameters specified in Table 4-4 from the EcuC module as the layout of post-build data is platform and compiler specific.

Parameter	Description
EcuC/EcucGeneral/StructAlignment	Alignment of struct elements.
EcuC/EcucGeneral/ArrayAlignment	Alignment of arrays.
EcuC/EcucGeneral/StructInArrayAlignment	Alignment of structs in an array.
EcuC/EcucGeneral/CPUType	Register width of the CPU in bit.
EcuC/EcucGeneral/SizeOfEnum	Size of type enum in bit.
EcuC/EcucGeneral/SizeOfInt	Size of type int in bit.
EcuC/EcucGeneral/ByteOrder	Byte order on memory level.
EcuC/EcucGeneral/BitOrder	Bit order on register level.
EcuC/EcucGeneral/SizeOfROMPointer	Size of ROM pointer in bit.
EcuC/EcucGeneral/SizeOfRAMPointer	Size of RAM pointer in bit.
EcuC/EcucGeneral/PostbuildLoadable/AlignmentFillPattern	Byte pattern which is used for the inserted padding bytes

Table 4-4 Platform Settings

Configuration instructions are given in [1] and must be consulted before editing these parameters.

4.5 Configuration of post-build update strategy

The Post-Build Xml Generator evaluates the EcuC module to determine which post-build update strategies shall be applied. It is possible to process several updates in one step. Additional configuration instructions (e.g. choice of parameter values) are given in [1].

4.5.1 Global update strategy

The start and stop addresses for the post-build RAM and ROM block of the global update have to be configured in the container “EcuC/EcucGeneral/PostbuildLoadable” (see Figure 4-1). The RAM and ROM memory blocks must not overlay with any other memory blocks.

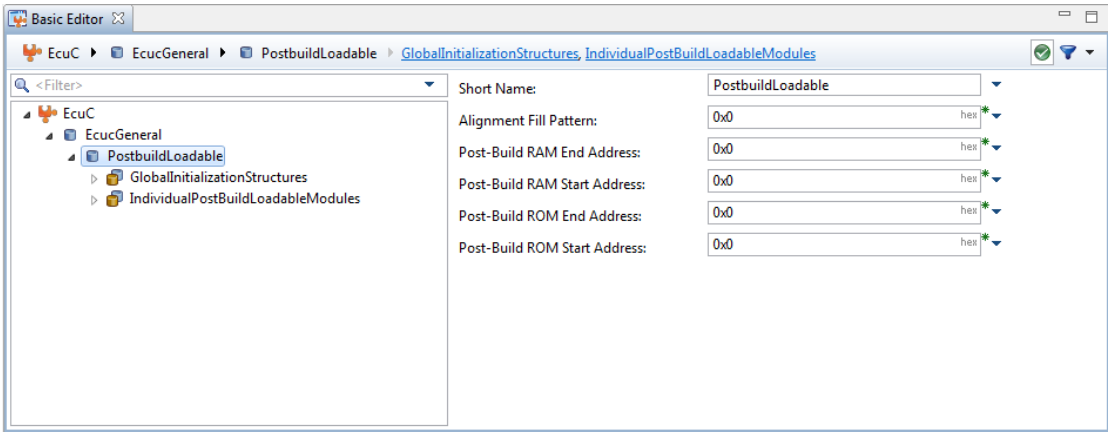

Figure 4-1 Global update: Configuration of memory block

To configure the global root structure name a new container “GlobalInitializationStructure” (see Figure 4-2, e.g. “EcuM_GlobalConfigRoot”) has to be created. The Post-Build Xml Generator evaluates the Short Name (e.g. “EcuM_GlobalConfigRoot”) of this container to determine the name of the global root structure. If this container does not exist, the name “EcuM_GlobalConfigRoot” is used as default. The configuration of the global root structure can therefore be omitted if the name does not differ from “EcuM_GlobalConfigRoot”!

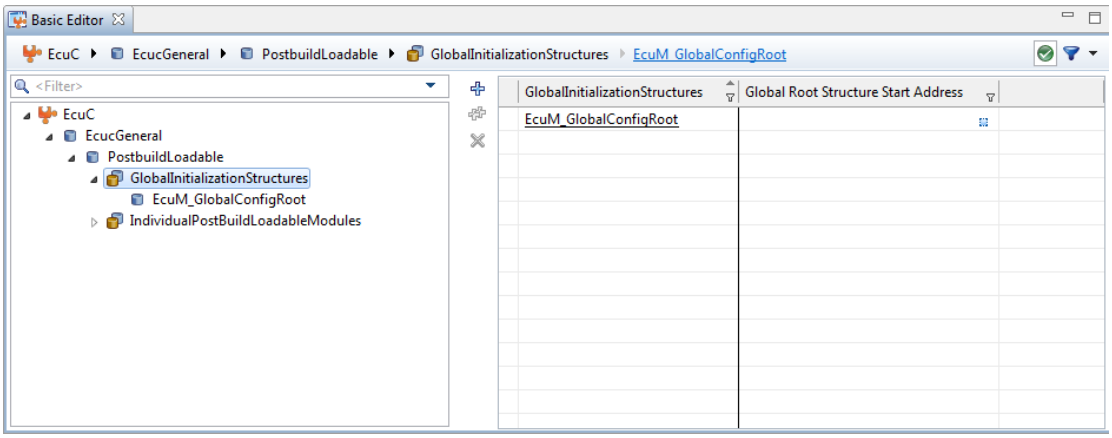

Figure 4-2 Global update: Configuration of global root structure

4.5.2 Module individual update strategy

The start and stop addresses for the post-build RAM and ROM block have to be configured for each module specific update. To configure these parameters a new “IndividualPostBuildLoadableModule” container (see Figure 4-3, e.g. “Dem”) has to be created. The RAM and ROM memory blocks must not overlay with any other memory blocks.

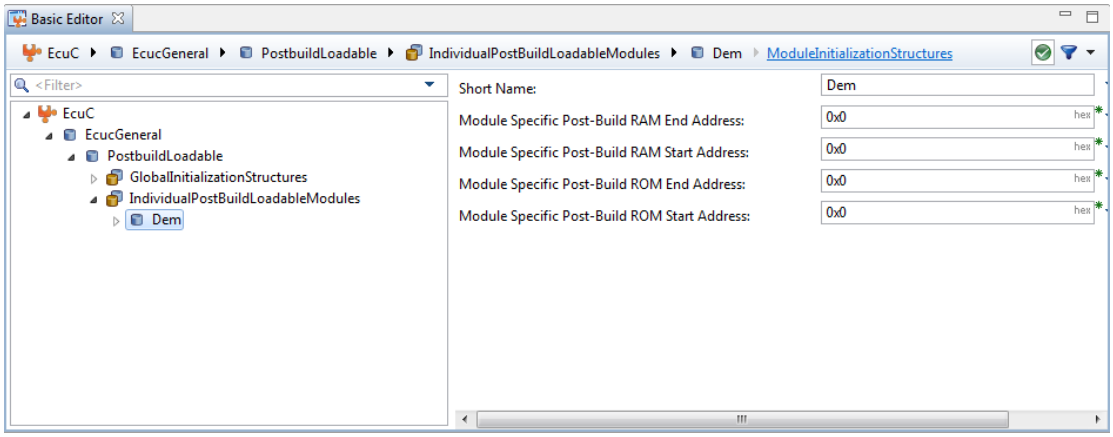

Figure 4-3 Module specific update: Configuration of memory block

To configure the module specific root structure name a “ModuleInitializationStructure” container (see Figure 4-4, e.g. “Dem_Config”) has to be created. The Post-Build Xml Generator evaluates the Short Name (e.g. “Dem_Config”) of this container to determine the name of the module specific root structure.

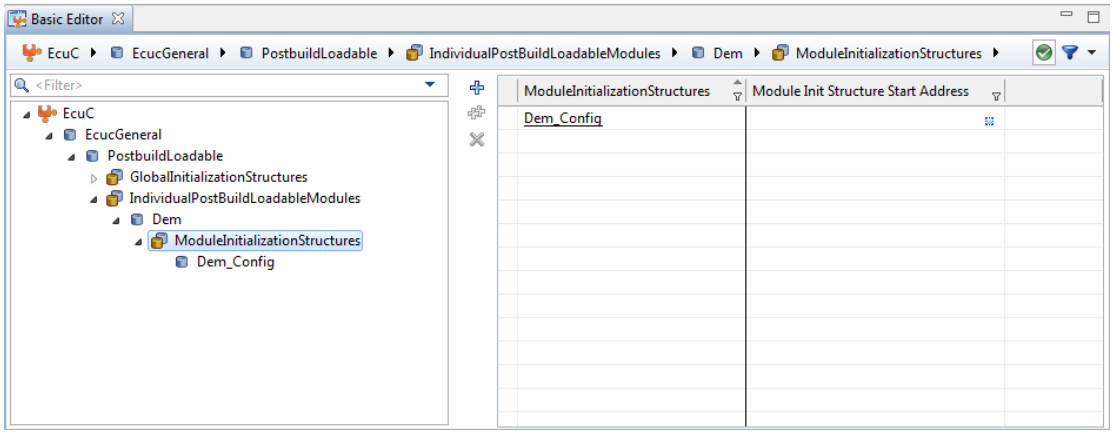

Figure 4-4 Module specific update: Configuration of root structure

4.6 Post-Build Selectable

The Post-Build Xml Generator supports post-build loadable in combination with post-build selectable. Both previously mentioned post-build update strategies can be applied for variant ECU projects. Detailed information about post-build selectable is provided in [2].

With post-build selectable variant specific global root structures are used. Therefore additional root structures have to be configured which is shown in the following chapters.

4.6.1 Global update strategy

For each variant specific global root structure a new “GlobalInitializationStructure” container has to be created (see Figure 4-5, e.g. “EcuM_GlobalConfigRoot_Variant1” and “EcuM_GlobalConfigRoot_Variant2”). A ROM start address has to be configured for each global root structure. These start addresses are typically fixed at link time and have to be consistent with the configured ROM memory block (see ch. 4.5.1).

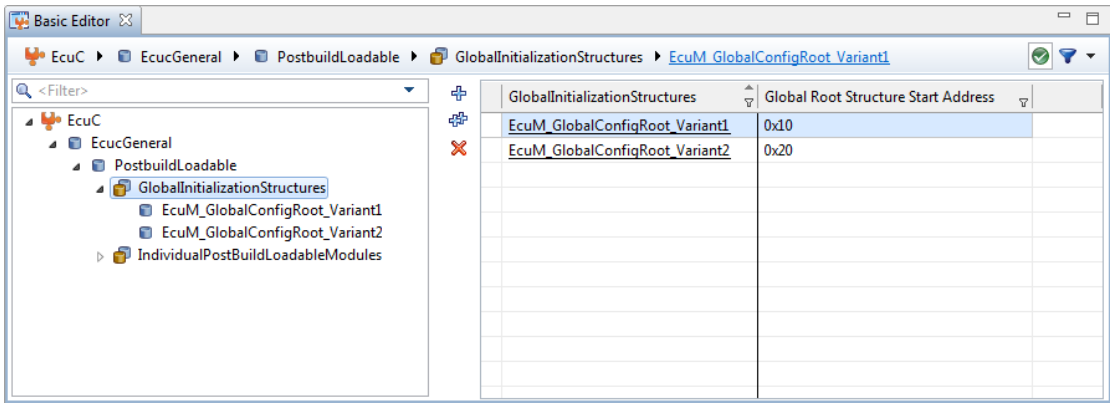

Figure 4-5 Global update: Configuration of variant global root structures

4.6.2 Module individual update strategy

For each variant module specific root structure a new “ModuleInitializationStructure” container has to be created (see Figure 4-6, e.g. “Dem_Config_Variant1” and “Dem_Config_Variant2”). A ROM start address has to be configured for each variant module specific root structure. These start addresses are typically fixed at link time and have to be consistent with the configured ROM memory block (see ch. 4.5.2).

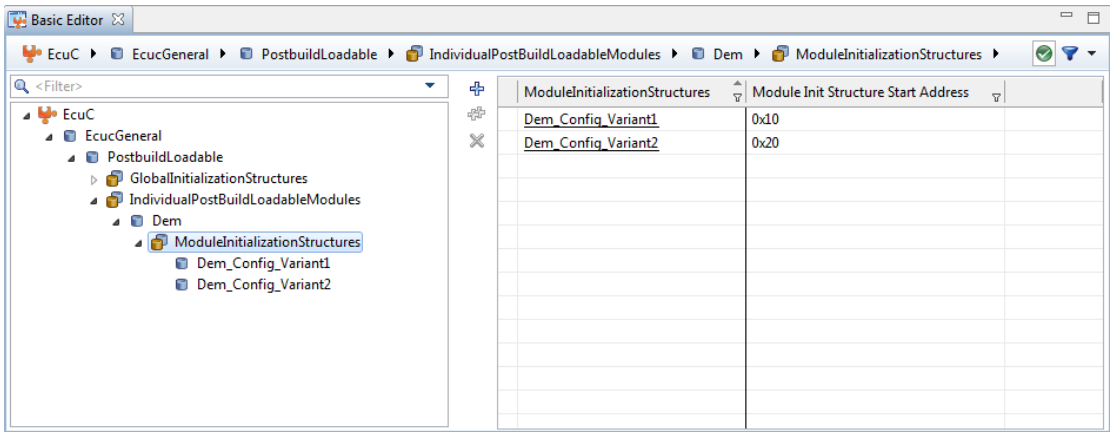

Figure 4-6 Module specific update: Configuration of variant root structures

4.7 Output Files

The Post-Build XML Generator will output the files listed in Table 4-5.

File	Description
XML File (e.g. <Project>.xml)	Post-build information used for HEX file creation. For each update, a single XML File will be created.
Map File (e.g. <Project>.map)	Updated configuration data address information for each update.
Log File (e.g. <Project>.log)	Contains information as memory usage (Post-build RAM block and Post-build ROM block), configuration parameters, processed files, errors and warnings.

Table 4-5 Output Files

All files are located in the same directory. The output XML file can be specified with the command line option --output=<file> or -o <file> only if a single XML file has to be created. In all other cases only the output directory can be specified with --output=<dir> or -o <dir>.

If no output file or directory is specified on the command line, the Post-Build XML Generator sets the output directory to <GenData>\PostBuild.

If no output file is specified, the name of the output XML and Map files will be derived from the DaVinci Project name <Project>.dpa for the global update. For module individual post-build updates, the name of the Map and XML file will be derived from the module name configured in the EcuC module (see [1]). The name of the Log file will be derived from the DaVinci Project name.

5 Glossary and Abbreviations

5.1 Glossary

Term	Description
Post-build RAM block	Linear RAM area reserved for post-build loadable MICROSAR modules. This memory area is used for data structures that can change their size at post-build time (e.g. by adding a new message). The memory area need to be assigned by the integrator at link time and must be of sufficient size to support increased RAM consumption at post-build time.
Post-build ROM block	Linear memory area in FLASH reserved for post-build loadable MICROSAR modules. This memory area is used for configuration data that can change their size at post-build time (e.g. by adding a new message). The memory area is assigned by the integrator at link time and must be of sufficient size to support increased configuration data size at post-build time.
DaVinci Configurator	Configuration and generation tool for MICROSAR BSW
Post-build	In this document post-build refers to the Post-Build loadable feature which allows updating the configuration at post-build time by downloading an updated configuration as a hex file.

Table 5-1 Glossary

5.2 Abbreviations

Abbreviation	Description
API	Application Programming Interface
AUTOSAR	Automotive Open System Architecture
BSW	Basis Software
ECU	Electronic Control Unit
MICROSAR	Microcontroller Open System Architecture (the Vector AUTOSAR solution)
SWS	Software Specification

Table 5-2 Abbreviations

6 Contact

Visit our website for more information on

- News
- Products
- Demo software
- Support
- Training data
- Addresses

www.vector.com