This document is released with Libero SoC Design Suite v11.8 SP4 without changes from v11.8 SP3.

SmartGen Cores Reference Guide

Libero SoC v11.8, v11.8 SP1, SP2, and SP3

NOTE: PDF files are intended to be viewed on the printed page; links and cross-references in this PDF file may point to external files and generate an error when clicked. **View the online help included with software to enable all linked content.**

Microsemi
Power Matters.™

Microsemi makes no warranty, representation, or guarantee regarding the information contained herein or the suitability of its products and services for any particular purpose, nor does Microsemi assume any liability whatsoever arising out of the application or use of any product or circuit. The products sold hereunder and any other products sold by Microsemi have been subject to limited testing and should not be used in conjunction with mission-critical equipment or applications. Any performance specifications are believed to be reliable but are not verified, and Buyer must conduct and complete all performance and other testing of the products, alone and together with, or installed in, any end-products. Buyer shall not rely on any data and performance specifications or parameters provided by Microsemi. It is the Buyer's responsibility to independently determine suitability of any products and to test and verify the same. The information provided by Microsemi hereunder is provided "as is, where is" and with all faults, and the entire risk associated with such information is entirely with the Buyer. Microsemi does not grant, explicitly or implicitly, to any party any patent rights, licenses, or any other IP rights, whether with regard to such information itself or anything described by such information. Information provided in this document is proprietary to Microsemi, and Microsemi reserves the right to make any changes to the information in this document or to any products and services at any time without notice.

Microsemi Corporate Headquarters
One Enterprise, Aliso Viejo,
CA 92656 USA
Within the USA: +1 (800) 713-4113
Outside the USA: +1 (949) 380-6100
Fax: +1 (949) 215-4996
Email:
sales.support@microsemi.com
www.microsemi.com

©2017 Microsemi Corporation. All rights reserved. Microsemi and the Microsemi logo are registered trademarks of Microsemi Corporation. All other trademarks and service marks are the property of their respective owners.

About Microsemi

Microsemi Corporation (Nasdaq: MSCC) offers a comprehensive portfolio of semiconductor and system solutions for aerospace & defense, communications, data center and industrial markets. Products include high-performance and radiation-hardened analog mixed-signal integrated circuits, FPGAs, SoCs and ASICs; power management products; timing and synchronization devices and precise time solutions, setting the world's standard for time; voice processing devices; RF solutions; discrete components; enterprise storage and communication solutions; security technologies and scalable anti-tamper products; Ethernet solutions; Power-over-Ethernet ICs and midspans; as well as custom design capabilities and services. Microsemi is headquartered in Aliso Viejo, California, and has approximately 4,800 employees globally. Learn more at **www.microsemi.com.**

5-02-9108-3401.17

Table of Contents

Introduction to SmartGen 7
SmartGen user interface 8
Create a workspace 9
Open a workspace 10
Import a SmartGen core 11
Import a legacy core 12
Remove or Delete a core 12
Save the workspace 13
SmartGen Preferences 13
Workspace settings 14
Generating Reports for Design Blocks 15
Create a new core in SmartGen 16
Reconfigure an existing core in SmartGen 16

FlashROM 17
Welcome to FlashROM 17
Create/Configure FlashROM 19
Modify an existing FlashROM configuration 21
Simulate Pre/Post Synthesis 21
Place-and-Route and FlashROM 21

Analog System Builder 23
Analog System Builder Reference 23
ASB Port List 24
Analog System Builder Main Window 25
Modify Sampling Sequence 26
External Trigger Signals in Analog System Builder 29
Analog System Builder Output Files 31
ASB Advanced Options 32
ASB Advanced Options - Calibration 34
ASB Connectivity and Calibration 35
Analog System Builder Calibration Output Files 37
ASB Advanced Options - ASSC RAM 38
ASB Advanced Options - SMEV RAM 39
ASB Advanced Options - SMEV Status 40
ASB Advanced Options - ADC results 41
ASB Advanced Options - ADC status 41
ASB Advanced Options - ACM Bus 41
ASB Advanced Options - ACM Clock 43
ASB - Calculating a Threshold 44
Prescaler Range 45
Acquisition Time 45
ASB Channel Mapping 45

Designing with the Analog System 46
Analog System Clocks 46
Analog System Builder and Flash Memory Block Basic Configuration 47
Analog System Builder with Real Time Counter (RTC) 48
Analog System Builder with ACM Access 49
Current Monitor 51
Differential Voltage Monitor 52
Direct Digital Input 54
Gate Driver 54
Internal Temperature Monitor 55
Internal Voltage Monitor 55
Real Time Counter 56
Temperature Monitor 59
Voltage Monitor 59
Configuring Current, Differential Voltage, Temperature, and Voltage peripherals 60
Sampling Rate in Analog System Builder 63
Sequencing in Fusion 67

Flash Memory System Builder 69
Welcome to the Flash Memory System Builder 69
Analog System Client 70
Data Storage Client 70
CFI Data 71
Initialization Client 72
RAM Initialization client 73
Flash Memory Block with Data Storage Client 74
Flash Memory System Output Files 74
Memory File Formats in Flash Memory System Builder 75

Introduction to SmartGen

The SmartGen software generates a large variety of commonly used functions. You can create a workspace and generate structural netlists in EDIF, VHDL, and Verilog. Furthermore, you can generate VHDL and Verilog behavioral models for most parameterized functions (the behavioral models may be used in a simulation environment).

This help provides descriptions of cores that you can generate using the SmartGen software. For more information about instantiating specific cores refer to the HDL Coding Style Guide.

The SmartGen environment enables you to create a workspace, and then import existing cores and create new ones.

SmartGen includes several updated features:

- The Analog System Builder (with advanced memory configuration options and updated clock information)
- The Flash Memory System Builder
- Generated state information for your core - Enables you to see if your core has been updated since the last release.

SmartGen Workspace

You must open a workspace in SmartGen before you can create or modify a core. Workspaces enable you to create, modify, and import existing SmartGen cores. The workspace is a logical grouping for your cores; each workspace contains cores for a specific product family.

Cores

When you generate a SmartGen core, SmartGen creates a core file that includes a structural netlist, an optional VHDL or Verilog behavioral netlist, a log file that summarizes your core parameters, and a GEN file that includes all the parameters you selected when you generated the core. Do not manually edit any of your core files. Instead, use SmartGen to edit the parameters for a core and then regenerate it.

Fusion, with the Analog System and Flash Memory System Builders, creates special directories for each core. Your cores can only be saved in the workspace directory. Do not save other files in your workspace directory, or in sub-directories created by SmartGen; the files are over-written when you re-save the core.

Varieties

Varieties list different implementations of similar functions. For example, when you click Arithmetic in the Category tab, SmartGen displays the complete list of varieties available for Arithmetic. Click a column header to sort the list of varieties. Click the Adder function to display the list of core varieties for Adder.

Core Catalog

The top of the Categories tab in the Core Catalog window displays the device family for your workspace at the top. Categories vary according to your device family; select the latest families to see and test the newest cores available. Click a core category to expand the list and display the functions.

The Alphabetic tab displays an alphabetical list of all the functions available for your device family. Click a function to display a list of varieties. Click the column headers to sort your core varieties.

IP Core Catalog

The IP Core Catalog lists IP cores available from Microsemi and our solution partners. You can browse the list, double-click an IP core for more information, and download datasheets and application notes for any

core if you are connected to the web. Contact Microsemi if you are interested in purchasing any of the cores listed in the catalog.

Core Variety View

The Core Variety View window displays the list of cores available for your device family. Click core categories and functions to narrow your list of core varieties. For example, click your device family for a complete list of all the core varieties for your family. Click the Arithmetic category to view a list of only Arithmetic core varieties, and click a specific function to narrow your list even further and view core varieties for the function.

The Generated State column displays information on the state of your core. It notifies you if you need to regenerate your core, if an update is available, or essential files are missing. See the SmartGen user interface for more information. A core in the Regenerate or Not Generated state is liable to cause your design to fail.

Log Window

The log window displays tips and instructions on the SmartGen flow. It may list errors and warnings that occur during core generation, as well as core parameters after you generate a core.

See Also

SmartGen user interface

Create a workspace

Import a legacy core

Create a new core in SmartGen

SmartGen user interface

The SmartGen software generates a large variety of commonly used functions. You can generate structural netlists in EDIF, VHDL, and Verilog. Furthermore, you can generate VHDL and Verilog behavioral models for most parameterized functions (the behavioral models may be used in a simulation environment). SmartGen includes workspace and core-management features.

SmartGen is divided into the Core Catalog / Intellectual Property Catalog (IP Catalog), the Variety View window, the Configured Core View window, and the Log Window (as shown in the figure below).

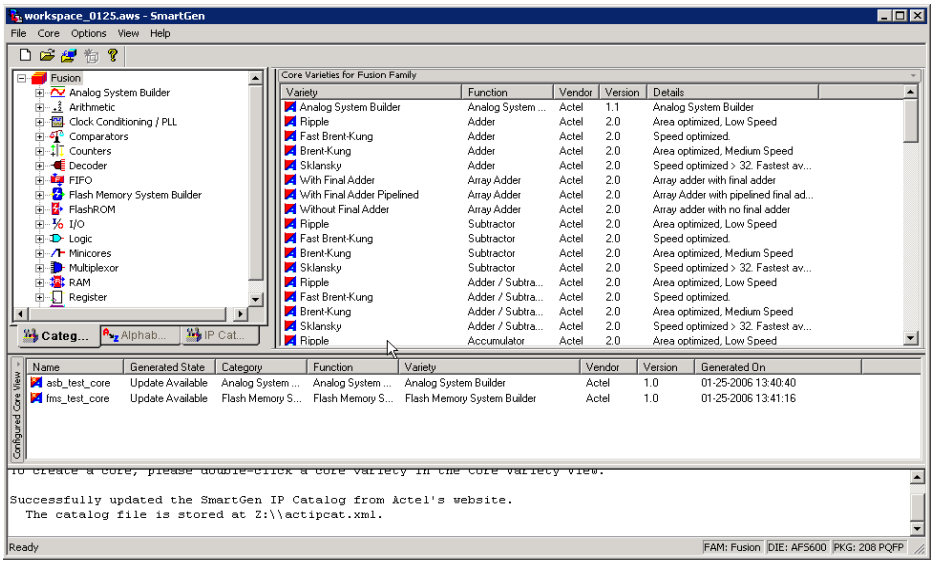


Figure 1 · SmartGen Main Window

Use this interface to browse your cores, review your configured cores, and select cores to create or modify.

The **Variety View** window displays a list of core varieties available for the core you selected in the Categories tab. For example, if you select the Arithmetic core type, and then select the Adder core, the Variety View window displays a list that includes Sklansky, Fast Brent-Kung, Ripple, etc.

Click a heading in the Variety View window to sort your list of cores. Cores are listed in Function order by default (Arithmetic cores, Clock Conditioning cores, Comparators, etc.). If you click a column heading they are sorted alphabetically by that column value (except for Version - then they are sorted numerically).

The **IP Catalog** displays a list of IP cores available for your device. You can access the complete core catalog library from within the Catalog. Cores are separated by type; select a type to display and double-click a core to display specific core information, or to download the core's datasheet. Click the headings in the Variety view to sort your cores alphabetically according to variety, function, vendor, etc.

The **Configured Core View** window displays a list of configured cores in your workspace. The configured cores appear each time you re-open your workspace; by default cores are listed in the order you generated them. Click a column heading to alphabetically by that column value (except for Version - then they are sorted numerically). The Configured Core view also displays information on the state of your core in the **Generated State** column:

- **Regenerate –** The core must be regenerated; the currently generated core is obsolete. This occurs if the core was generated using a previous version of software and the workspace contains a core generated from a newer file set. A core in the Regenerate state is liable to cause your design to fail.
- **Update available –** It is not mandatory that the core be regenerated but you may elect to in order to take advantage of a new feature or defect resolution. This state occurs when all cores of a given type have been generated successfully but the software has access to a newer version of that type.
- **Not Generated –** The core is missing its HDL file and must be regenerated. This may occur through user error or the import of very old cores. The core must be regenerated in order to create the proper HDL and associated files. A core in the Not Generated state is liable to cause your design to fail.

You can delete cores from your Configured Core View (and leave them on the disk), or you may delete them from your disk entirely. Select the core(s) you want to delete and press the **Delete** key, or right-click and choose **Remove from Workspace** or **Remove from Disk and Workspace**.

The **Log Window** displays information as you configure your cores in your workspace.

See Also

Import a legacy core

Create a workspace

You must create a workspace to generate a core in SmartGen.

A workspace defines your family and the default directory in which you save your configured cores. If you wish, you may save your workspace in a different directory. The workspace is a logical grouping for your cores; each workspace contains cores for a specific product family.

To create a workspace:

1. Start SmartGen. When you open SmartGen it asks if you wish to create a workspace or open an existing workspace. You may choose to create a workspace or open an existing workspace. Select the check box to hide this dialog the next time you open SmartGen.
2. From the **File** menu, choose **New Workspace**, or click CTRL + N.

Specify the **Workspace name**, **Workspace location** (click the **Browse** button to navigate to or create a directory), **Family**, **Die**, **Package** and **Defaultoutput format.**

If available, click the **Use a specific die and package for resource reports** checkbox to select a specific die and package. Resource report information is unavailable if you do not select a specific die and package

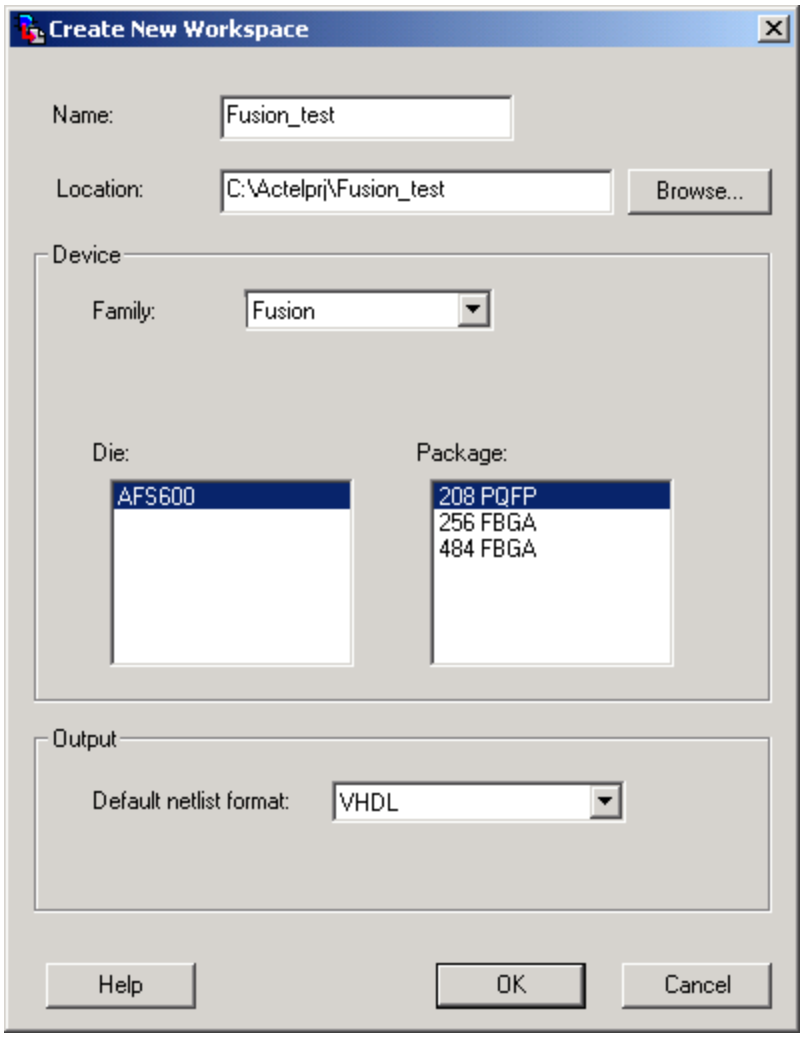


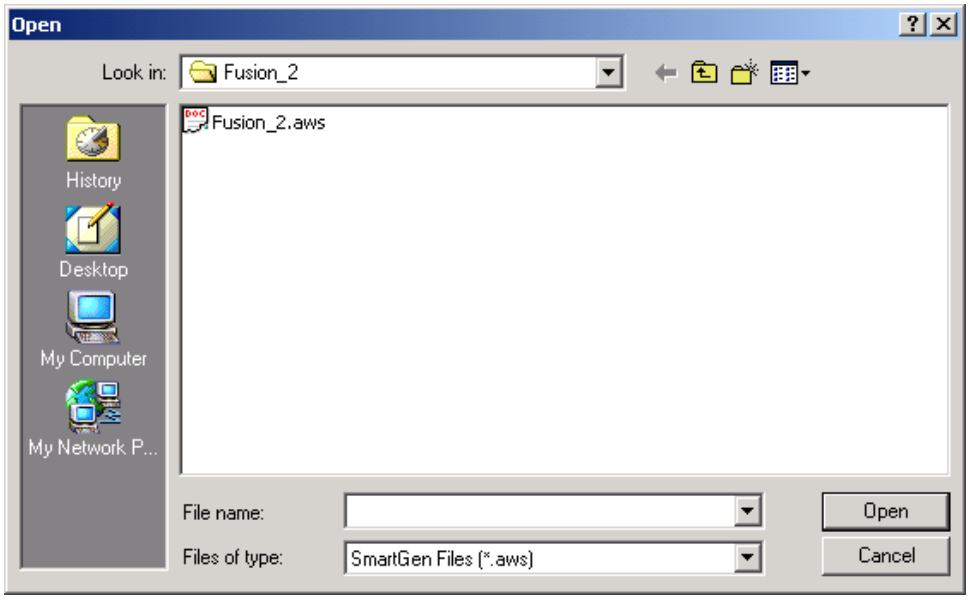
Figure 2 · Create New Workspace Dialog Box

3. Click OK to create the new workspace. The new workspace appears in the SmartGen window. By default, the list of cores available in a new workspace is sorted by **Categories**. Click the **Alphabetic** tab to display the complete list of SmartGen cores.

Open a workspace

You must open a workspace in SmartGen to generate cores. If you have never opened a workspace, you can create one. To open an existing workspace:

1. Start SmartGen.
2. From the **File** menu, choose **Open Workspace**, or click the Open Workspace button in the SmartGen toolbar. This opens the Open Workspace dialog box (as shown in the figure below).



3. Navigate to the workspace you wish to open and click **Open.**

SmartGen audits the workspace for existing cores and their associated files (CXF, LOG, VHD/V, and other auxiliary files) each time you open it. SmartGen reports if any of the files associated with an existing core are missing. If a CXF (core configuration) file for a core is missing, SmartGen loads the core into the workspace with default parameters.

Import a SmartGen core

You can import SmartGen cores into your current workspace. To do so:

From the **File** menu, choose **Import Core**. This displays the Import Core dialog box (as shown in the figure below).

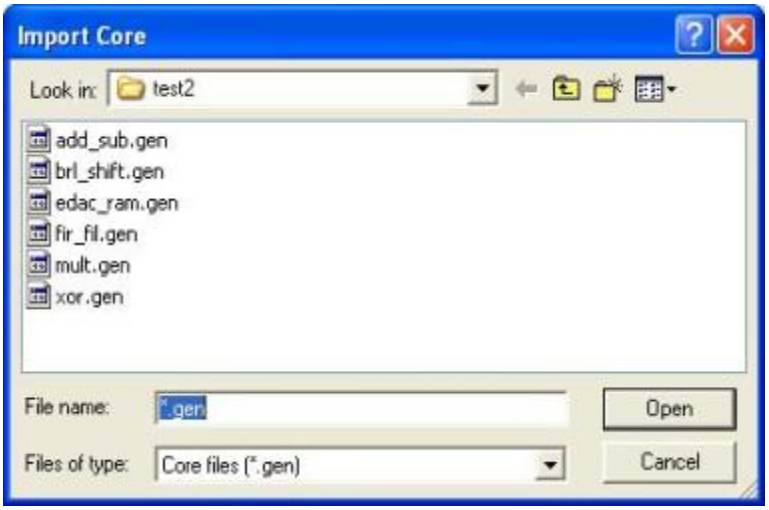


Figure 3 · Import Core Dialog Box

1. Navigate to the core you wish to import and click **Open**. The core is added to your current workspace and remains in your workspace until you **Remove** it.

Note: You cannot import two cores with the same name. SmartGen is case-insensitive; "core_A" is equivalent to "core_a".

Import a legacy core

You can import SmartGen legacy cores into your current workspace. When you import a legacy core the import copies only the genfile into your workspace. You MUST regenerate the core to get a new netlist in SmartGen.

To import a legacy core:

1. From the **File** menu, choose **Import Core**. This displays the Import Core dialog box (as shown in the figure below).

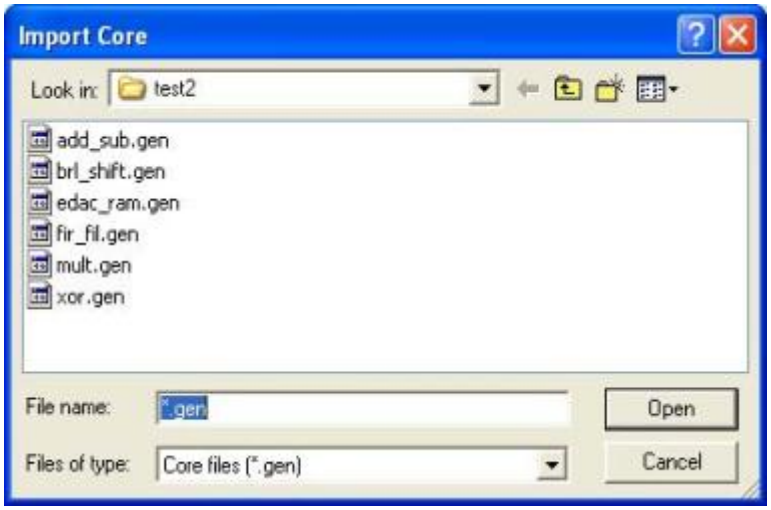


Figure 4 · Import Core Dialog Box

2. Navigate to your core you wish to import and click **Open**. The core is added to your current workspace and remains in your workspace until you **Remove** it.

   You can only import legacy cores that were generated for the family on which the workspace operates. If you import cores from other families, you get an error message, as shown in the figure below.



Figure 5 · Import Core Error Message

Remove or Delete a core

You can delete cores from your Configured Core View (and leave them on the disk), or you may delete them from your disk entirely. Select the core(s) you want to delete and press the **Delete** key, or right-click and choose **Remove from Workspace** or **Remove from Disk and Workspace**.

You do not need to save your workspace.

Note: Removing a core from your workspace does not delete the core from your directory. You can import the core later if you wish. If you choose Remove from Disk and Workspace the core is deleted forever.

Save the workspace

All changes to the workspace are saved automatically. You can save the workspace with a different name (useful if you want to create a copy of your workspace in a different directory).

To save your workspace, from the **File** menu, choose **Save Workspace As** and specify the **Name** and **Location**.

SmartGen Preferences

Use the Preferences dialog box to set the default directory for new workspaces. Whenever you try to Open a GEN file or generate a netlist, SmartGen uses the preferences you set here to generate your file.

Your default netlist output and family type are workspace settings.

To set or modify your General preferences:

From the **File** menu, choose **Preferences**. The Preferences dialog box (**General** tab) appears. To set the default directory for your workspaces, type the pathname, or click the browse button and navigate to your new default directory.

UNIX Only: Specify the location of your PDF reader and web browser.

In At Start Up, you can choose to **Load last loaded workspace (default)**, **Show [an] empty [work] environment**, or **Show the Welcome to SmartGen dialog box**.

To set your IP Catalog preferences:

Click the IP Catalog tab to specify when SmartGen checks for updates. The default is to check each time you open the tool. You can also specify the location of the IP Catalog info on your hard drive. Type the pathname, or click the Browse button and navigate to your new default directory.

The Proxy tab enables you to set your internet proxy settings in SmartGen. This enables SmartGen to use an FTP connection to update some data files. If you use a proxy server, enter the server name in the Proxy field.

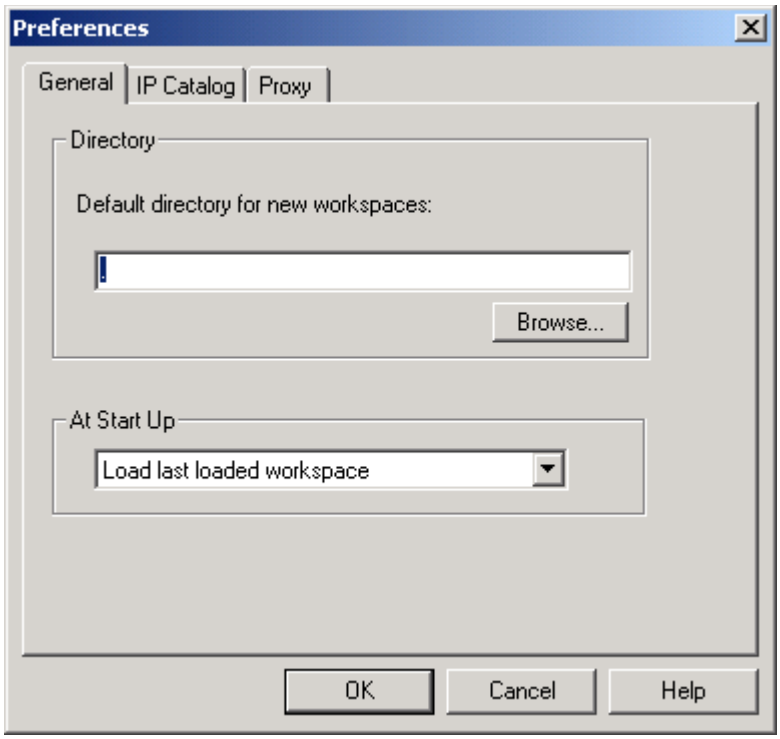


Figure 6 · SmartGen Preferences Dialog Box

Workspace settings

To change your workspace settings, from the **Options** menu, choose **Workspace settings**. If you wish to change your preferences, you can modify them in the File menu.

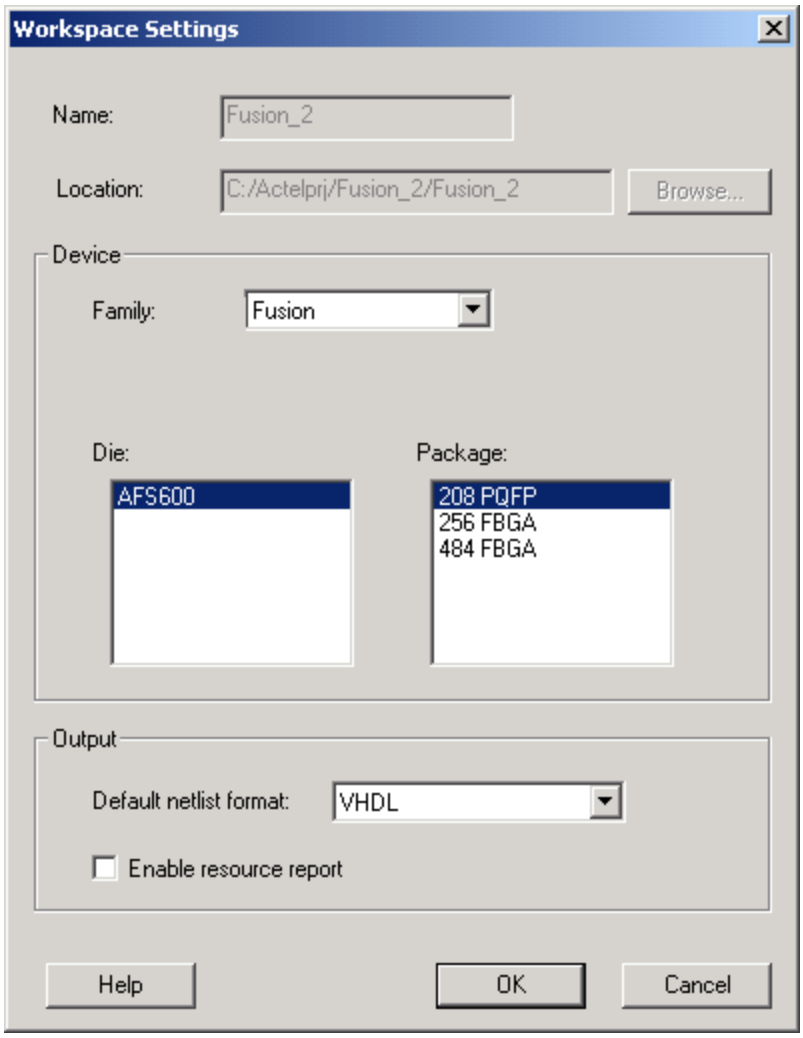


Figure 7 · Workspace Settings Dialog Box

The Workspace Settings dialog box (above) specifies your device family, die, package, default output netlist format, enables or disables the Resource report (antifuse families only), and lists the device family you have selected for your workspace.

You may only change your Device Family if there are no cores in your workspace. If you wish to change families, delete all the cores from your workspace or create a new workspace.

Select the check box to enable the Resource report.

Antifuse families only: The Resource report calculates and displays the sequential, combinatorial, I/O, memory, and PLL resources used in the generated core as a percentage of the largest device in the family.

Generating Reports for Design Blocks

As the software generates a Design Block core it writes information to the Log window. The report contains information defining the core, and is divided into the following sections:

- **Core Parameters -** This section lists the options selected to build the core
- **Fan-in Control information -** This section defines the type of buffering for each control signal and the values used to distribute the total load
- (Optional) **Compile report -** Lists compile information related to the core

Create a new core in SmartGen

You can create a new core in SmartGen only after you have created a workspace.

To create a new core in SmartGen:

1. Select a core type in the **Categories** tab (Arithmetic, Comparator, Converters, etc.). The **Variety View** window displays the list of configurable cores.
2. Double-click the core variety you wish to generate, or right-click and choose **Create Core**. The core configuration dialog box appears.
   The configuration dialog box varies depending on which core you select.
3. Set the parameters for your core and click **Generate**. The **Generate Core** dialog box appears. The Generate Core dialog box lists the names of all the configured cores already in your workspace.
4. Specify the name of your new core and click **OK** to continue. SmartGen saves your core and adds it to your workspace (in the Configured Core View Window). You cannot save your core anywhere but in your workspace.
   Your core remains in the workspace until you choose to **Remove** it.

Note: You cannot create two cores with the same name. SmartGen is case-insensitive; "core_A" is equivalent to "core_a".

Reconfigure an existing core in SmartGen

You can reconfigure your cores in SmartGen. To do so, you must first have added a core to your workspace.

To reconfigure a core in SmartGen:

1. Select the core in the Configured Core View window.
2. From the **Core** menu, choose **Modify Core**. The configuration dialog box opens to display the configuration options specified in the core. Your core configuration options vary depending on the device family you selected when you created your workspace.

FlashROM

Welcome to FlashROM

FlashROM memory provides the security of stored data in addition to a 128-bit AES decryption core. You can read, modify, and write to the FlashROM using the JTAG interface; however, you can only read it from the FPGA core.

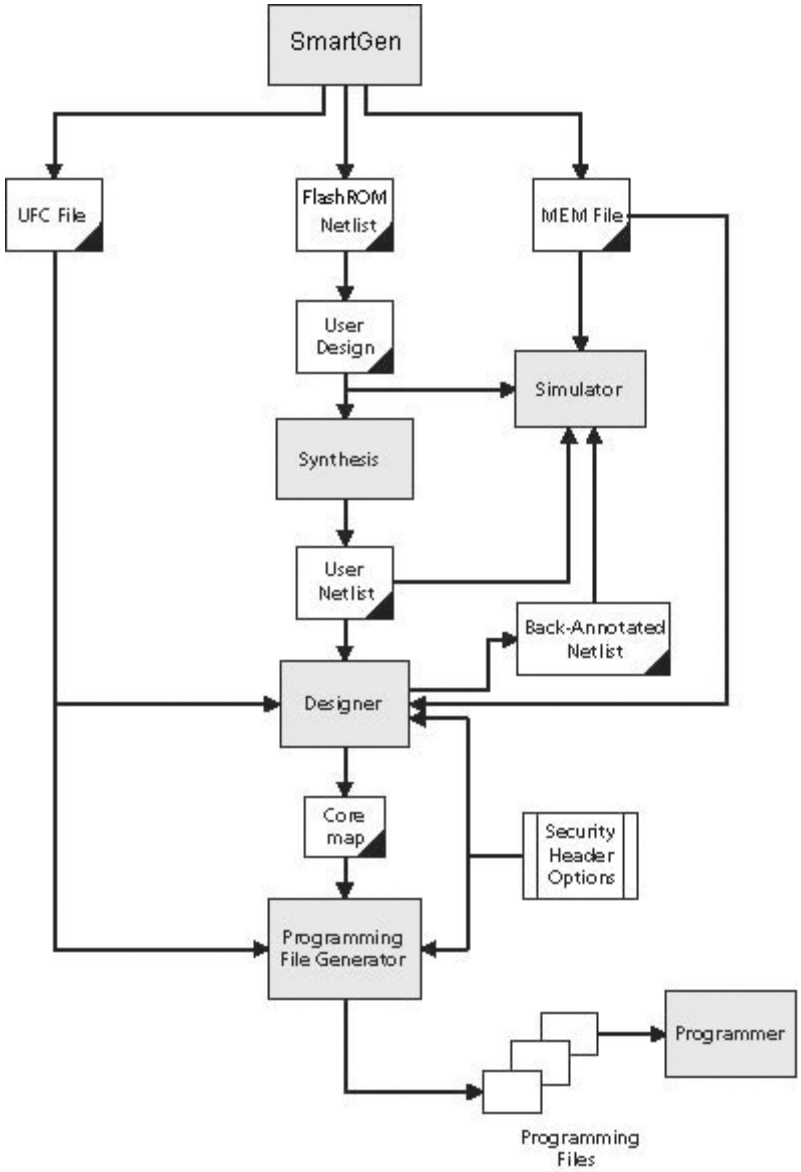


Figure 8 · FlashROM Flow

FlashROM is available from the Catalog the Libero SoC. Add a FlashROM core from the Catalog to your SmartDesign in the Libero SoC Project Manager.

Adding a FlashROM core opens a special FlashROM core generator that enables you to configure the FlashROM functionality.

Note: FlashROM is available only for IGLOO and ProASIC3 devices.

IGLOO and ProASIC3 devices have a flexible programming option. The FlashROM and the FPGA core fabric can be programmed independently of each other, allowing the FlashROM to be updated without changing the FPGA core fabric. The following are just a few examples of possible applications for the FlashROM feature:

- Internet protocol (IP) addressing (wireless or fixed)
- System-calibration settings
- Device serialization and/or inventory control
- Subscription-based business models (e.g. set-top boxes)
- Secure key storage
- Asset management tracking
- Date stamping
- Version management

The FlashROM is programmed using the standard IEEE1532 JTAG programming interface. Pages can be individually programmed (erased and written) and on-chip AES decryption can be used selectively to load data securely into the FlashROM (such as application-based security keys stored in the FlashROM for a design). See the FlashPoint help or user's guide for information on how to program your FlashROM-enabled devices.

The FlashROM can selectively be read back either through the JTAG programming interface or via direct FPGA core addressing. Its contents can only be updated via the JTAG interface. A seven-bit address from the FPGA core defines which of the eight pages (3 MSBs of the addresss) is being read and which of the 16 bytes in the page (4 LSBs) are being read.

The FlashROM is physically organized as 8x128 bit blocks and logically organized as eight pages by 16 bytes. Only Flash FPGAs contain on-chip nonvolatile memory (NVM); Microsemi's SmartFusion, IGLOO, ProASIC3 and Fusion devices are the only FPGAs to support this feature.

You can assign specific regions of the FlashROM for specific purposes by floorplanning the FlashROM and assigning properties. The content of these regions can be modified during programming time if you assign a modifiable content property to a given region. If you do not want the FlashROM content to be modified, you can fix the content in .

When you generate a new FlashROM file, the generator saves the following files for you to use throughout the design cycle:

- **CXF file** - Contains project info for Libero SoC.
- **Netlist file -** Use this file to instantiate your core, just as you would instantiate any other core in your design
- **UFC file -** User Flash configuration file; it contains all the configuration information regarding the FlashROM data content and is used for programming. You can export a core map file that contains the core programming information and use it along with the UFC file to generate programming files. Designer software supports importing the UFC file and launching the programming file generator to merge the FPGA core map file and the FlashROM programming file.
- **MEM file -** FlashROM specific memory initialization file. The MEM file has 128 rows of eight bits, representing the contents of the FlashROM. FlashROM defaults to 0s for any unspecified locations of the FlashROM memory. This file is used exclusively for simulation.

Use the FlashROM help to:

- Configure FlashROM
- Simulate Pre/Post Synthesis
- Synthesize
- Place-and-Route
- Run Back-Annotation and Timing Simulation
- Specify security settings
- Specify FlashROM content
- Generate a programming file

Create/Configure FlashROM

The GUI enables you to create and configure memory regions in FlashROM. The regions you create can be of arbitrary widths (up to sixteen). You can specify whether the content in this region is Fixed (meaning what you enter here in the GUI is fixed), or Modifiable if you expect it to change it the future.

To actually specify the data that goes into the memory region you just created, you must specify the Type of the content you are about to enter: Binary, Hexadecimal, Decimal, or Text (Character String). And finally, you must specify the actual data in the Value field.

Note: If you use STAPL files be sure that your memory REGION name does not contain illegal characters. The FlashROM Configurator Dialog enables you to specify a REGION Name that contains characters which are not legal in the STAPL File Language format. STAPL identifier names are limited to 32 characters and must begin with an alphabetic character - not a number or underscore character (_). Identifier names consist of alphabetic characters, numeric characters and the underscore character - no other characters are allowed. Identifier names are not case sensitive.

The FlashROM can be partitioned into regions and each region can be used for a specific purpose, like serial number storage, version number saving, etc.

Use the FlashROM core generator to create a region within a page, modify the region, and assign properties to that region.

The FlashROM user interface includes the Configuration Grid, a list of existing regions, and the list of Properties. You can assign values to the following properties:

Content

Static - Data entered manually when the core is configured and is not changeable. This option is useful when you have fixed data stored in this region that is required for the operation of the design in the FPGA. Key storage is one example.

Auto Inc - Specify a starting number, a maximum number and the size of each step between. The starting value and maximum value can be modified in FlashPoint.

Read from File - Data is read from a content file into the selected region. A different content file may be selected in FlashPoint from either Designer or FlashPro.

When Content is specified as **Read from File**, the following file formats are supported: Binary, Hex, Dec, and Text.

Note that when configuring FlashROM in SmartGen there is a limit on the decimal value when using read from file and format type DEC. The limit is 32-bits, so the largest number is 4294967295. Even if you have an 128-bit region, the largest number will be 32 bits.

In each format, the file is a simple list of data based on your content size. For example:

Binary format: 0101110011

Hex format: 1112222ffffaaaabbbb

Dec format: 123456789

Text format: myexampletext

State

Fixed - Enables you to fix the data so that it cannot be changed during programming time. This option is useful when you have fixed data stored in this region that is required for the operation of the design in the FPGA. Key storage is one example.

Modifiable - Select this option when the data in a particular region is expected to be static data (such as a.version number, which remains the same for a long duration, but could conceivably change in the future). This option enables you to identify this region so that you need not come back and change the value every time you enter new data.

Type (Format)

Specifies the format that you intend on specifying in the Value/Simulation Value field. For example, if you choose DEC for the type then you can only type a decimal value into the Value/Simulation Value field. Text enables you to enter any character string.

Value/Simulation value

This is the actual content that you want programmed into that FlashROM region.

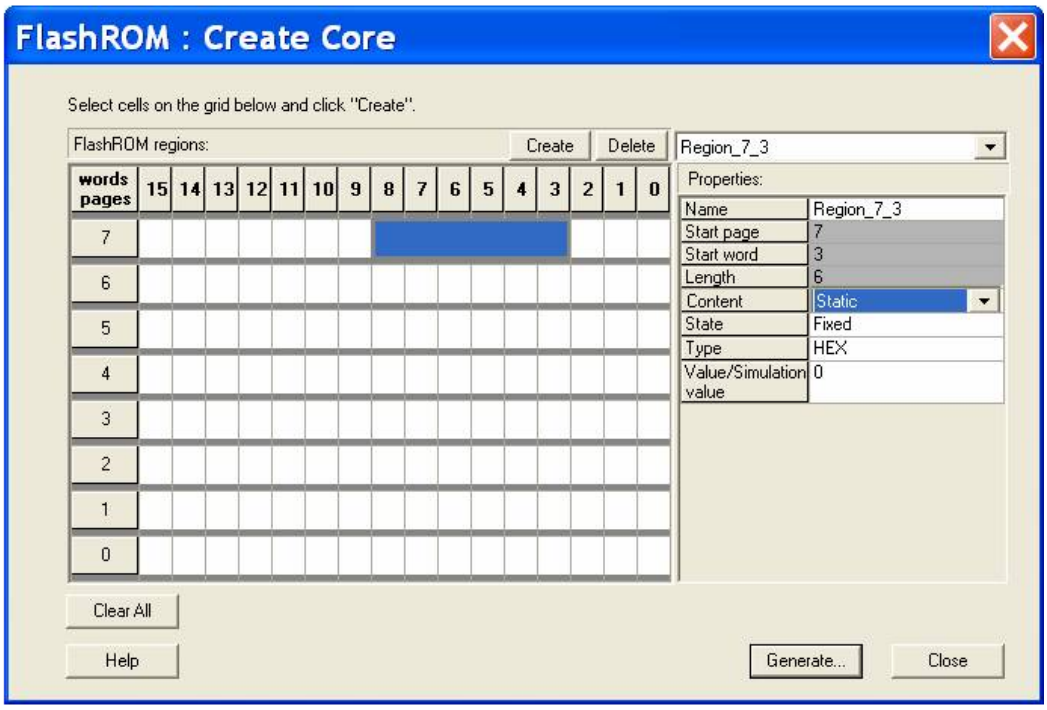


Figure 9 · FlashROM Core Generator

To create a new FlashROM:

1. In the Libero SoC Core Catalog, choose **Memory & Controllers**. Double-click the **FlashROM**core in the Core Catalog to start the core generator.
2. Click and drag the mouse to select words, then click the **Create** button. The core generator displays the new region properties in the **Properties** grid.

   You may also right-click a word and choose **Create** from the shortcut menu, or select a word and press the **Insert** key on your keyboard. You can copy and paste regions in FlashROM; to do so, right-click a word and choose **Copy**, then click an empty word, right-click, and choose **Paste**. If the region is not copied, the page does not have enough room. Try another page with more room.
3. Click in the **Properties** grid to modify a regions properties. **Start page**, **Start word**, and **Length** are read-only. The data you enter is verified and stored in the FlashROM as soon as you leave the Properties grid and select another FlashROM region.
4. Click **Generate** to generate a netlist (output format matches the HDL type you specified when you created your project),GEN, CXF, LOG, UFC, and MEM file. The **Generate** button opens the Generate Core dialog box. Specify a name and click **OK**.

To delete a FlashROM Region:

1. Click to select a region in the **Regions** window.
2. Click the **Delete** button in the core generator, press the **Delete** key on the keyboard, or right-click and choose **Delete** from the shortcut menu.
3. Click **OK**.

See the FlashPoint help or user's guide for information on how to program your FlashROM-enabled devices.

Modify an existing FlashROM configuration

You can modify your existing FlashROM configuration the same way that you modify any configured core. To do so:

1. Open your configured FlashROM core. The FlashROM core opens with all the settings you saved.
2. Modify the values you wish to change and click **Generate** to save your changes. **Generate** opens the **Generate Core** dialog box. Save your new file with the same name if you wish to overwrite the old file.

   You cannot edit the configuration of an existing FlashROM GEN file, only the data. If you wish to change the configuration you must generate a new core.

Simulate Pre/Post Synthesis

FlashROM uses the MEM file for simulation.

The MEM file has 128 rows of eight bits, representing the contents of the FlashROM. FlashROM defaults to 0s for any unspecified locations of the FlashROM memory.

During simulation, employ the MEM file, which contains the memory content, along with the design netlist and testbench. The VITAL and Verilog simulation models accept the generics passed by the netlist, read the MEM file, and perform simulation with the data in the file.

In addition to using the MEM file, you may create a binary file with 128 rows of eight bits and save the file as a MEM file. Microsemi recommends using different names if you plan to generate multiple MEM files. During place-and-route in Designer, the software recognizes the generic property in the netlist and passes the MEM file links through to the output netlist.

Place-and-Route and FlashROM

There are no special instructions for place-and-route for the FlashROM. Run Layout in Designer to place-and-route your design.

Welcome to the Analog System Builder (ASB)

The Analog System Builder enables you to configure an entire analog system. You can:

- Choose the number of Analog Input Channels to monitor
- Choose the type of each Input Channel
- Choose the number of Analog Output Channels
- Specify the placement of each channel
- Set Channel-specific options
- Sequence the channels in the required sampling order
- Define the operations on converted digital output from the ADC
- Specify the RTC settings.

The ASB is available in the Core Catalog in the Project Manager in Libero SoC.

The Analog System Builder enables you to create, configure, and place the following analog blocks (or "peripherals"):

- Voltage Monitor
- Current Monitor
- Temperature Monitor
- Differential Voltage Monitor
- Direct Digital Input
- Gate Driver
- Real Time Counter
- Internal Temperature Monitor
- Internal Voltage Monitor

Analog System Builder

Analog System Builder Reference

The Analog System Builder uses some terminology that may be unfamiliar. Here is a list of terms and acronyms that appear in the software and the help.

Term	Description
ADC	Analog-to-digital converter
ASSC	>Analog sample sequence controller; sets the sample order in the ADC (includes IP + RAM)
Analog System	The complete system, including the analog block (AB) hard IP and one or more of ASSC, SMEV, and SMTR soft IPs.
SMEV	Evaluates the converted analog data (IP + RAM)
SMTR	Processes the evaluated analog data and generates flag signals on certain conditions (includes IP + RAM)
AB	Analog block - The hard macro in the CAE library that includes the analog MUX and the ADC
Analog MUX	The 32 -1 MUX, select signals of which determine the channel being sampled by the analog to digital converter.
ACM	Analog Configuration MUX - stores configuration data related to analog channels (channel type, pre-scalar value, polarity, etc.)
FMSB	Flash Memory System Builder
INIT IP	INIT / CFG Soft IP, responsible for all initialization and Save activity to the NVM
ASB	Analog System Block. Analog System top level, includes the Analog Block (AB) and Analog System Soft IP.
FMB	Flash memory block. Flash Memory top level, includes Flash Memory System Builder and INIT IP
FASTCLK	Intended clock during normal system execution
SLOWCLK	Intended clock during system initialization

ASB Port List

All signals are active high unless explicitly specified.

Name	Type	Description
SYS_CLK	INPUT	Clock input for Analog Block and Soft IP
SYS_RESET	INPUT	Active low asynchronous reset
VAREF	INOUT	Voltage reference; connects to external voltage for external voltage reference. Returns the internal VREF in the case of internal voltage reference. Must be connected to a top-level port without any I/Os.
Analog Input Channels	INPUT	User specified or port names. Analog input channels being used.
Analog Output Channels	OUTPUT	User specified port names; analog output channels being used.
Input Channel Compare Flags	OUTPUT	Threshold flags specified for each peripheral
Flash Memory Block Interface		
INIT_DATA[8:0]	INPUT	Initialization data
INIT_DONE	INPUT	Initialization done signal from the Flash Memory Block
INIT_ACM_WEN	INPUT	Initialization ACM write enable
INIT_ASSC_WEN	INPUT	Initialization ASSC RAM write enable
INIT_EV_WEN	INPUT	Initialization SMEV RAM write enable
INIT_TR_WEN	INPUT	Initialization SMTR RAM write enable
INIT_ACM_RTC_WEN	INPUT	Initialization of ACM during second pass; only exported when you use the RTC Peripheral.
Status Signals		
DATAVALID	OUPUT	Indicates data from the ADC is valid
ASSC_DONE	OUTPUT	Indicates the ASSC is finished processing the current sampling slot
ASSC_WAIT	OUTPUT	Indicates the ASSC is inserting wait states to accommodate SMEV and SMTR processing times.

Name	Type	Description
ASSC_CHSAT	OUTPUT	Sampled channel saturated; indicates that the current channel sampled by the ADC has a value that is saturated (too high for the ADC voltage range). Once this output becomes active, it remains active until the next timeslot is processed. If this output is active, you may need to move up to a higher voltage range (decrease prescaler value for the particular analog input pad).
ASSC_CHLATD	OUTPUT	Channel selector latched; signal indicates that the ADC_CHNR[4:0] (ADC channel selector) value has been latched.

Analog System Builder Main Window

The Analog System Builder main window enables you to create and configure your analog system (as shown in the figure below).

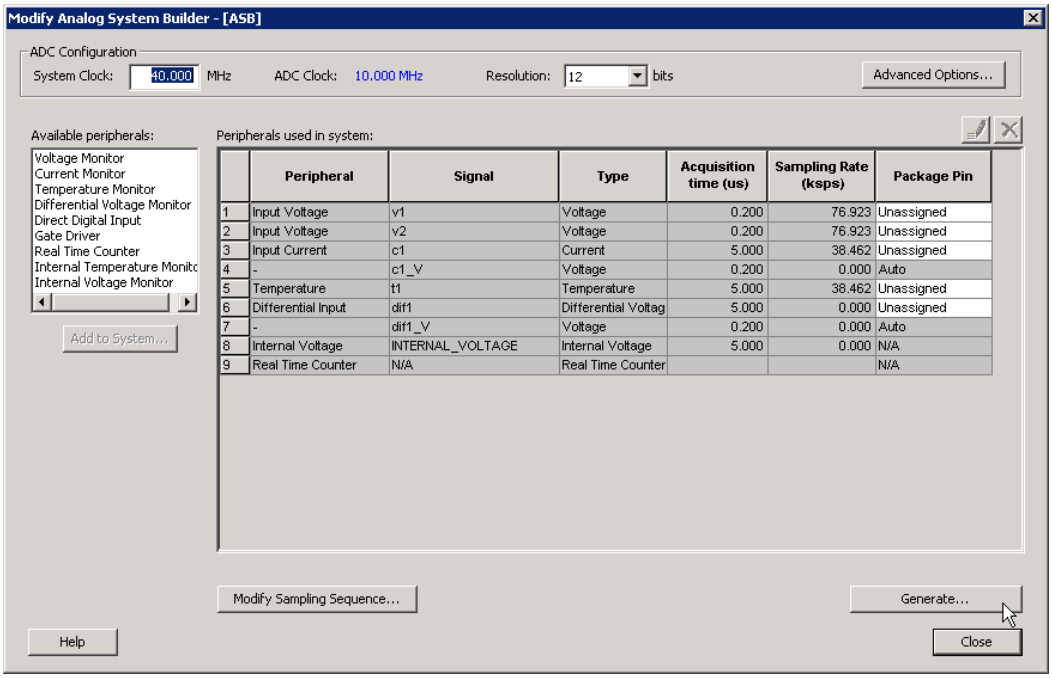


Figure 10 · Analog System Builder Dialog Box

The number of peripherals you can add to the system is limited by the size of your device.

The ADC Clock is the frequency at which analog to digital conversions occur. The ASB evaluates the required acquisition times for all peripherals and the system frequency to compute the maximum possible ADC clock frequency. Only certain divider factors exist to create the ADC Clock; because of this and peripheral acquisition times, certain system frequencies result in a faster ADC Clock. Refer to the Designing with Analog System Builder section for a discussion on Analog System clocks.

You can select the resolution of the ADC (8-, 10-, and 12-bit modes). Selecting the resolution affects the meaningful bits read from the ADCRESULT port, ASSC_RAM, and SMEV_RAM.

- In 12-bit mode, the ADC uses 11:0
- In 10-bit mode, the ADC uses 11:2; 1:0 are grounded
- In 8-bit mode, the ADC uses 11:4; 3:0 are grounded

Click **Advanced Options**to set your Analog System configuration.

Available Peripherals lists all the analog peripherals you can add to your design. As you add peripherals, some resources are exhausted. If you exceed the resource limit, the ASB returns a warning during file generation. If you want to add additional peripherals, select a larger device, or remove some existing peripherals from your design.

The **Peripherals used in system** grid lists specific information about each peripheral, including

- Peripheral - The type of the peripheral (such as Voltage Monitor, Temp. Monitor, etc.).
- Signal - Name you specified for the signal of your service in the service configuration dialog box.
- Type - Identifies channel type for the service.
- Acquisition Time - The required acquisition time for a given input channel. ASB takes the required acquisition times for all peripherals and computes the maximum possible ADC clock frequency and the number of ADC clocks per sample and per peripheral.
- Sampling Rate (in μs) - This field only displays the sampling rate for the channels specified in the “Main” procedure. See the Modify Sampling Sequence topic for more information on setting your sampling sequence. See theSampling rate in Analog System Builder topic for more information on how the sampling rate is calculated.
- Package Pin - ASB automatically assigns a package pin for each channel in each peripheral added to the system. However, if you require a specific channel for a certain package pin (if you have board layout issues), you can choose a specific pin for that channel.
- Real Time Counter - You can configure the Real Time Counter so that it functions as a chronometer, allowing it to generate periodic alarms in conjunction with other peripherals (such as the Voltage monitor, etc.).

Modify Sampling Sequence - Displays the Sample Sequencer. Since there are thirty analog input channels but only one ADC, the channels must be sequenced in the order in which they are to be sampled.

If the Analog System resources you build exceed the total system resources available for your device, ASB issues a warning. You cannot generate a system that exceeds your total system resources. The Analog System Builder also generates a warning if you have a port name conflict between two or more services. You cannot generate a system with port name conflicts.

When you click **Generate the system**, ASB creates HDL source files, memory (MEM) files, configuration files, and log files. They all appear in your project folder under the <core_name> directory. Do not modify any of these generated files or store additional files in this folder. This folder will be recreated every time you overwrite the core.

Modify Sampling Sequence

Since there are 30 input channels (depending on the device) but only one ADC, the channels must be sequenced in their desired order. There are 64 time slots available for sequencing. You can also run non-sampling operations in the sequencer (such as calibration or powerdown).

Your application requirements dictate the sampling sequence.

The sampling sequence specifies the Analog System’s sampling order. For example, the sequence may be specified to sample "voltage channel 1" continuously, or it can be specified to sample "voltage channel 1", "voltage channel 2", "temperature channel 1", and repeat. In either case, the Sample Sequence Controller will drive the ADC signals to sample the channels in the specified sequence.

The sampling sequence has an Automatic Sequence calculation feature. A checkbox to enable or disable this feature is labeled “Allow manual modification of operating sequence”. When unchecked, sampling rate requirements may be entered for channels and software attempts to calculate a sequence that meets the rate requirements and satisfies the ordering rules. When unchecked the operating sequence may be specified manually.

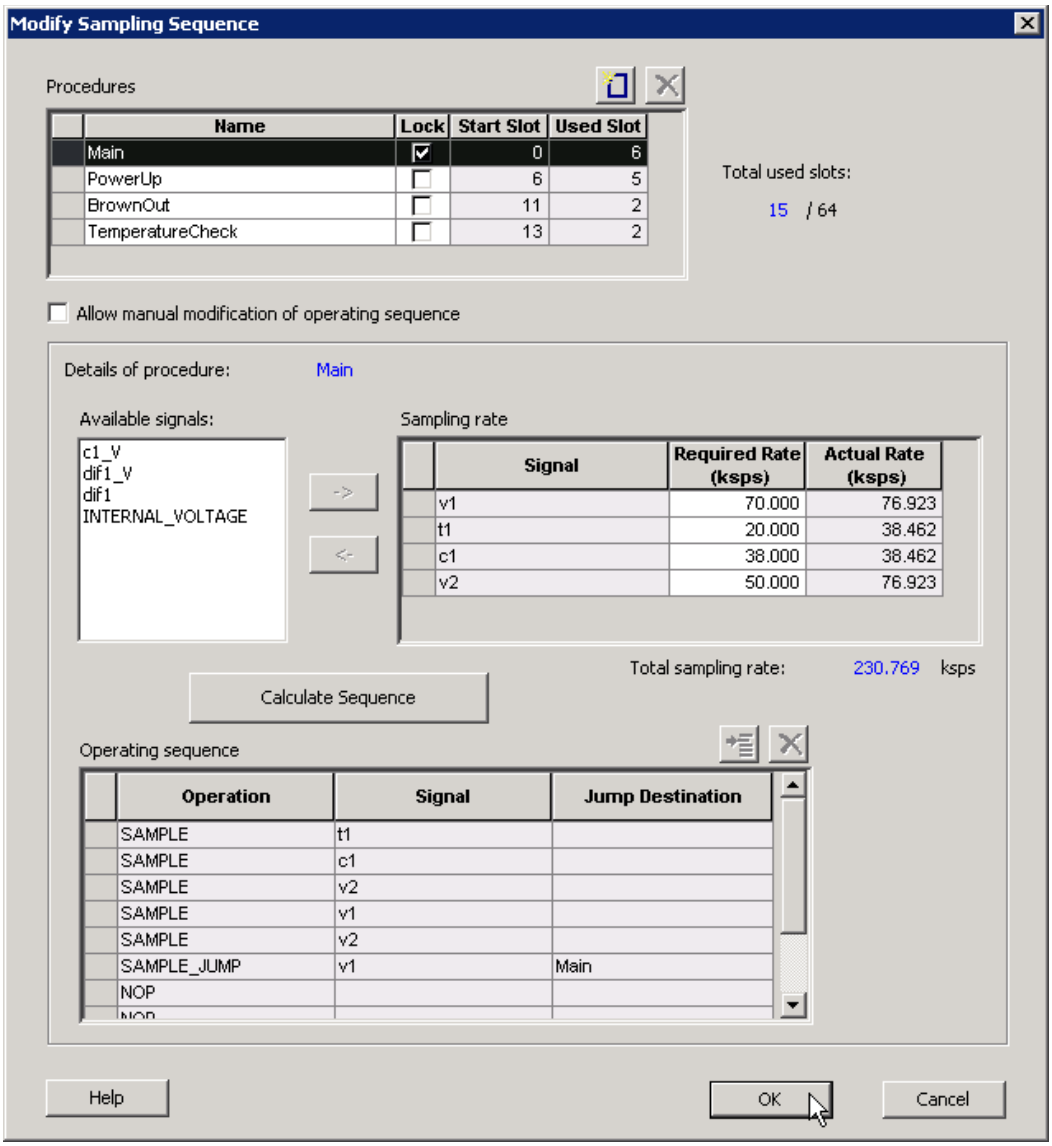


Figure 11 · Modify Sampling Sequence Dialog Box

Procedures

Procedures are a logical composition of sequences. Each procedure is intended to be completely independent of another procedure.

An example use-model for multiple procedures is a system that requires one set of samples during system power-up, and another set after power-up. For example, upon power-up the system needs voltage channels 1 and 2 monitored. Then after a certain event, such as reaching a stable voltage level, a different set of analog channels need to be monitored.

In this case, you would create 2 procedures: A ‘Powerup’ procedure that samples voltage 1, voltage 2, and repeat; and a ‘SteadyState’ procedure that continually monitors the rest of the analog inputs once it has been determined that a steady and stable voltage level has been reached.

The intelligence to determine when to trigger another procedure must be performed by the user through the external sequencer control interface. The External Trigger interface is only exported if there is more than one procedure.

The ability to have multiple procedures that can continually loop upon themselves allows for these types of use-models.

The system defaults to having a single “Main” procedure that can not be deleted or unlocked. It always starts at slot 0 and will always be the procedure that is executed upon reset.

- Name – Enter a name for your procedure
- Lock – Lock the starting slot of the procedure, this is useful if you have an existing design that already has logic to trigger a procedure. If lock is checked, the start slot field is modifiable otherwise it is read-only and software will assign a starting slot for the procedure.
- Start Slot – This is the starting slot number for this procedure. This number is required to trigger this procedure to start executing. Drive this value into the ASSC_SEQIN port.
- Used Slots – Indicates the number of physical slots used up by this procedure. Recall that the sequencer only supports up to a total of 64 slots.
- Total Used Slots – The total number of slots used by all the procedures, if it exceeds 64 it will turn red indicating that a violation has occurred.

Input your procedure values and click the Add Procedure button to create a new procedure. Click the Delete Procedure button to delete a procedure. .

The Operating Sequence and Sampling Rate grids change when you select a procedure.

Sampling Rate

When **Allow manual modification of operating sequence** is unchecked, you can specify your rate requirements per channel here. The channel must selected in the list of available signals and added / removed from the grid. After it has been added to the grid, a required sampling rate (in kilosamples per second) may be specified for that channel. Clicking **Calculate Sequence** initiates software to calculate a sequence that most closely meets the your requirement(s).

When **Allow manual modification of operating sequence** is checked, this section is used only to report the sampling rate of the channels. By adding and removing channels to the operating sequence, this grid automatically updates with the actual sample rate for that channel for this procedure. Recall that procedures are completely independent, so the sample rate calculation is performed per procedure and include any of the other procedures.

The actual sampling rate for each channel is displayed in this grid. The total sampling rate indicates the total sampling rate of all channels for the selected procedure.

Operating Sequence

The operating sequence for a selected procedure. The supported operations are:

- SAMPLE - Sample a channel that is configured in the system and proceed to the next slot
- SAMPLE_JUMP – Sample a channel that is configured in the system and jump to the start of the specified procedure
- CALIBRATE – Perform a full calibration of the ADC (this requires 3840 ADC Clocks to complete) and proceed to the next slot
- CALIBRATE_JUMP – Perform a full calibration and jump to the start of the specified procedure
- JUMP – Jump to the start of the specified procedure
- POWERDOWN – Perform a powerdown operation on the ADC; after a powerdown is initiated, a calibration operation is required to resume sampling
- STOP – Stop the sequencer; an external trigger is required to re-start the sequencer
- NOP – No operation is performed and proceed to the next slot. NOP’s in the middle of a sequence use up a time slot, but NOP’s after the end of the last functional slot do not.

Terminating slots: Each operating sequence must end with a terminating operation. A terminating operation is a SAMPLE_JUMP, CALIBRATE_JUMP, JUMP, POWERDOWN, or STOP.

Slots can be inserted or deleted with the Add Slot button or Delete Slot button, respectively.

Calculate Sequence

The ASB creates a sampling sequence that attempts to meet the sampling rate requirement specified for the procedure.

An additional INTERNAL_VOLTAGE peripheral with an acquisition and hold time of 5µs may be added to your system to satisfy strobe requirements.

The sequence calculation attempts to fairly balance the sampling rate among the signals by reducing the difference between the actual and required rate.

External Trigger Signals in Analog System Builder

The external trigger signals are used to control the sequencer from user logic. These signals are exposed if there is more than one procedure in the sequencer.

There are two external jump modes:

- Manual - The system completes operations in the current slot, then waits for the signal to move to the next slot; ASSC_XMODE = 1
- Auto Forwarding - The system completes the operation in the jump slot and moves to the next slot, until the sequence reaches slot 63, at which point it begins sampling slot 0. If the operation is a STOP or POWERDOWN, the sequencer stops processing until another jump is initiated to a different slot; ASSC_XMODE = 0

All signals are active high.

Name	Type	Description
ASSC_XMODE	Input	External Trigger Mode: If this input is logic 1, the ADC Sample Sequence Controller will use the ASSC_XTRIG signal to transition to and complete the current sequence timeslot. If this input is logic 0 (default operation for automated sequencing), the internal timeslot counter will be used to automatically advance to the next sequence number.
ASSC_XTRIG	Input	External Trigger: If the ASSC_XMODE input is logic 1 and this input is held at logic 1 for exactly 1 clock cycle, the ASSC block will transition to and complete the current sequence. If the ASSC_XMODE input is logic 0 (default operation for automated sequencing), this input is ignored. If this signal is used to control external triggering, monitor the ASSC_DONE signal to know after which point the ASSC_XTRIG will again have effect.
ASSC_SEQJUMP	Input	Sequence Jump Enable: Setting this signal to logic 1will jump to the sequence number indicated in the ASSC_SEQIN input pins after the current sequence timeslot has completed.
ASSC_SEQIN[5:0]	Input	Sequence Number In: These inputs are used in conjunction with the ASSC_SEQJUMP signal to jump to a particular sequence number from the current sequence after the current sequence timeslot has completed. These inputs can come from user logic external to the Analog Interface Soft IP blocks, or can be statically tied off to any combination of logic 0 and logic 1 values
ASSC_SEQOUT[5:0]	Output	Sequence Number Out: These outputs denote the current sequence timeslot.

Name	Type	Description
ASSC_SEQCHANGE	Output	Sequence Change: This output indicates that the ASSC_SEQOUT outputs are about to change after the very next rising edge of CLK.
ADC_CHNUMBER[4:0]	Output	Channel number being sampled. Refer to the ASB log file for the logical number your peripheral was mapped to.
ASSC_SAMPFLAG	Output	ASSC Sample Function Flag. Indicates that the sample function is active for the currently selected ADC channel.

Timing Diagrams

The diagrams below show the ASB External Sequencer in full, and a detail of the assert and de-assert states.

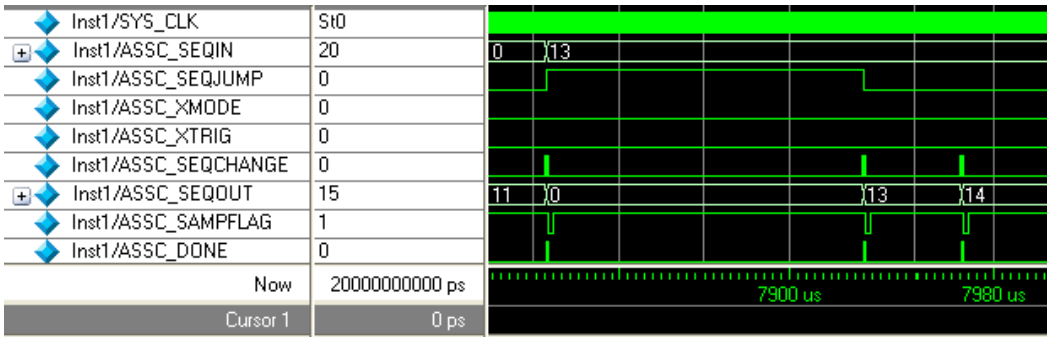


Figure 12 · ASB External Sequencer Control

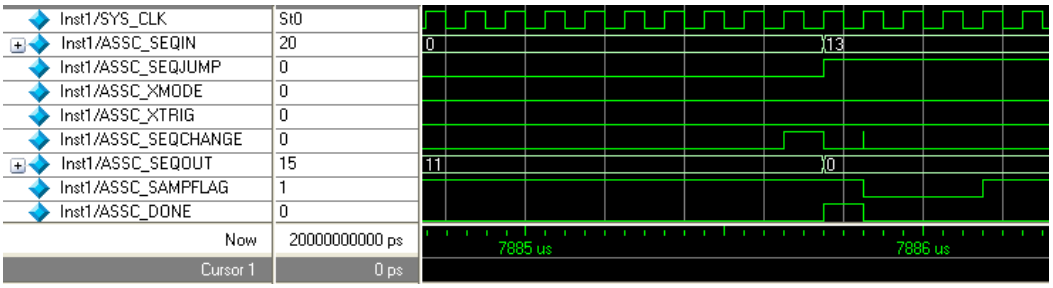


Figure 13 · ASB External Sequencer Control Assert Detail

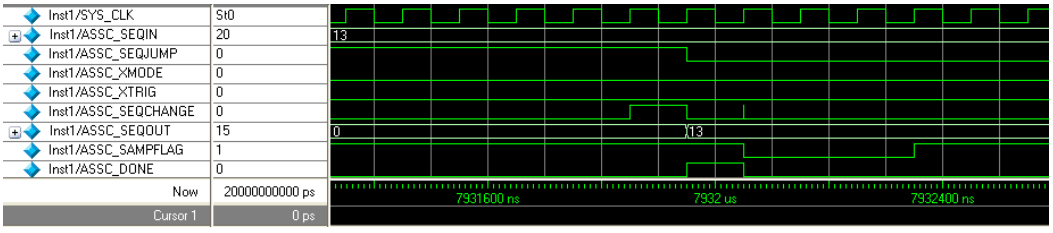


Figure 14 · ASB External Sequencer Control De-Assert Detail

Analog System Builder Output Files

ASB creates the following output files in your project directory when you generate an analog system.

HDL Source Files

<user_name>.vhd/.v – Top-level design that combines all the blocks together

<user_name>_assc_wrapper.vhd/.v – Analog system controller instantiation wrapper for this design

<user_name>_smev_wrapper.vhd/.v – Analog system data processing module instantiation

<user_name>_smtr_wrapper.vhd/.v – Analog system data processing module instantiation

<user_name>_assc_ram.vhd/v – Analog system controller RAM

<user_name>_smev_ram.vhd/v – Analog system data processing module RAM

<user_name>_smtr_ram.vhd/v – Analog system data processing module RAM

<workspace_directory/<common>/<Vhdl>/<Verilog>/assc.vhd/v – Analog sample sequence controller file; common file for all analog system cores.

<workspace_directory/<common>/<Vhdl>/<Verilog>/smev.vhd/v – Analog system data processing module file, common for all analog cores.

<workspace_directory/<common>/<Vhdl>/<Verilog>/smtr.vhd/v – Analog system data processing module file, common for all analog cores.

Memory Files

These memory files are used by the Flash Memory System (or any external microprocessor) to initialize the contents of the RAM and AB.

<user_name>_acm_ram.hex/ .s - Intel-hex or Motorola S-record memory files for AB Hard IP

<user_name>_assc_ram.hex/ .s - Intel-hex or Motorola S-record memory files for ASSC RAM

<user_name>_smev_ram.hex/ .s - Intel-hex or Motorola S-record memory files for SMEV RAM

<user_name>_smtr_ram.hex/ .s - Intel-hex or Motorola S-record memory files for SMTR RAM

The memory files below are used to initialize the RAM contents for simulation only. These files enable simulation of the Analog system in isolation (there is no need to connect the initialization circuitry).

<user_name>_acm_R0_C0.mem - Memory File for simulation for AB

<user_name>_assc_ram_R*_C*.mem - Memory Files for simulation for ASSC RAM

<user_name>_smev_ram_R*_C*.mem - Memory Files for simulation for SMEV RAM

<user_name>_smtr_ram_R*_C*.mem - Memory Files for simulation for SMTR RAM

Configuration Files

<user_name>.ncf – The embedded Flash configuration file used to communicate information from the Analog System to the Flash Memory system regarding the size of the Analog System Client and the location of the memory content.

<user_name>.cfg – This captures information about the settings that were specified for the system.

<user_name>.gen – Enables the software to open the system with your saved specifications.

<user_name>.cxf – The Core Configuration file that contains information required by the Libero SoC for file management.

Log Files

The log file contains all the information used to generate your system, as well as any messages related to conflicts or system resource limitations. The file is called <user_name>.log.

See Also

Analog System Builder Calibration output files

ASB Advanced Options

The Advanced Options in the Analog System Builder (ASB) enable you to set the external reference voltage and generate custom system configurations (as shown in the figure below). Some custom configurations (such as **ADC only**, and **IP cores for ADC sequence control only**) disable some of the functionality in the Analog System.

See the Analog Block Pin Description in the Fusion Datasheet for more information on these ports.



Figure 15 · Analog System Builder Advanced Options Dialog Box

Setting your External Vref (external voltage) enables you to use a specific Analog to Digital Converter (ADC) reference voltage, allowing more accurate ADC conversions. Vref = VAREF.

This Vref value specifies the voltage reference driven into the Vref interface of the analog block. If you do not enter a value then ASB uses an internal Vref of 2.56V; applying an external Vref affects the threshold computations.

This implies that when modifying the External Vref, the legal range of thresholds for current and voltage may be altered.

Note: The legal threshold range for the temperature monitor is not based on Vref.

Therefore, it is possible that you could create a current or voltage peripheral, set up some flags with threshold values, and then decide to use and change the external Vref. This could lead to errors in the existing flag thresholds.

If you invalidate your flag thresholds by setting an external Vref, the ASB main window displays an icon notifying you of errors in the configured peripheral.

Vref Capacitor value - Lists values of external capacitors that can be used when generating Vref internally. Select the value that is equal to or greater than the capacitor value used on the board. For information on selecting the Vref capacitor value, please see the Fusion datasheet and the Fusion handbook.

The chosen capacitor value determines the amount of delay that is inserted before the Analog Block can begin sampling. This delay circuit is automatically inserted by Analog System builder in the form of a counter circuit.

To assist in simulations, a special capacitor value of 0.00 µF is provided that enables faster simulations. This selection should be used only for your simulations, as the actual device requires the proper capacitor values to ensure accurate sampling results.

Enabling calibration changes the internal Analog System connectivity and creates additional output files; please see the ASB - Calibration options for more information on Calibration.

The ASB Advanced Options dialog box enables you to generate the following in your Analog Block:

- IP Cores for ADC data processing and sequence control
- IP Cores for ADC sequence control
- ADC only

Each of these are described below.

IP Cores for ADC data processing and sequence control

Enables all the Analog Block features: sequencing, flag generation, data averaging, and general ADC management. You can enable or disable access to ADC results, ADC Status ASSC RAM, SMEV RAM, SMEV Status, ACM Bus, and ACM Clock.

ASSC is responsible for setting the sample order in the ADC and SMEV evaluates the converted analog data. This option instantiates the Analog Block and the complete Analog System Controller (includes ASSC RAM, SMEV RAM, and SMTR RAM), as shown in the figure below.

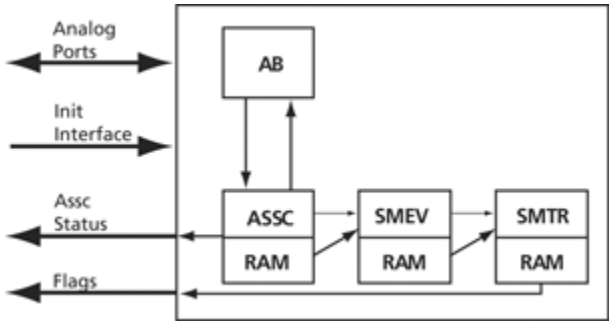


Figure 16 · System Diagram for IP Cores and ADC Data Processing and Sequence Control Options

This option generates the following files:

- ACM MEM files
- ASSC IP, ASSC RAM, ASSC Wrappers, & ASSC MEM files
- SMEV IP, SMEV RAM, SMEV Wrappers & SMEV MEM files
- SMTR IP, SMTR RAM, SMTR Wrappers & SMTR MEM files

Enabling user access to ADC results, ADC status, ASSC RAM, SMEV RAM, and SMEV Status exposes additional interfaces and ports. See the help topics associated with each option for more information.

IP Cores for ADC Sequence control

This configuration instantiates only the analog block model and the ASSC RAM. The data processing portions of the controller (SMEV and SMTR) are omitted from the design (as shown in the figure below). If you select this option, you must process the ADC data directly from the ADC RESULT bus or the ASSC RAM.

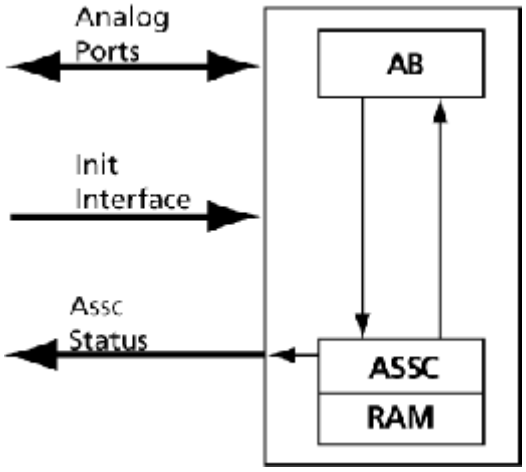


Figure 17 · System Diagram for IP Cores for ADC Sequence Control Only

This configuration disables flag generation for peripherals (the flag grid for peripherals); data averaging (Digital Filtering Factor and Initial Averaging value); SMEV RAM access; and the ability to specify the external resistor in the Current Monitor.

Note: You must explicitly choose to expose the ADC result and/or ASSC RAM data interfaces to gain access to the ADC data.

IP cores for ADC sequence control generates the following files:

- ACM MEM files
- ASSC IP, ASSC RAM, ASSC Wrappers, & ASSC MEM files

ADC only

This configuration instantiates only the Analog Block model (as shown in the figure below). It omits the data processing, sequence controller, and the ADC management features. If you use this configuration you must completely manage the ADC and all related AB functionality.

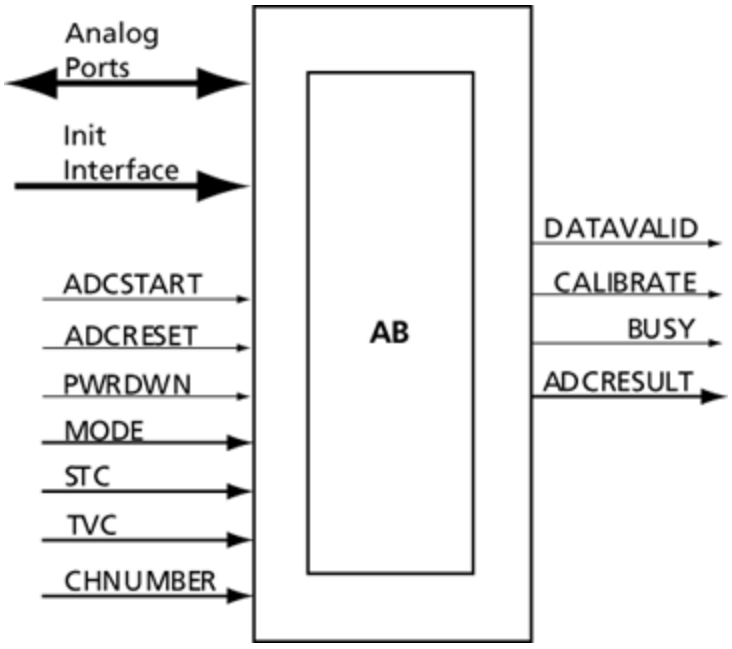


Figure 18 · System Diagram for ADC Only

This option disables sequencing (Sequencer dialog box); flag generation for peripherals (the flag grid for peripherals); data averaging (Digital Filtering Factor and Initial Averaging value); ASSC RAM access, SMEV RAM access; SMEV Status access; and the ability to specify the external resistor in the Current Monitor.

Without the ASSC, you must also manage some general ADC features. They are:

- ADC clock divider (derived from the system clock frequency)
- ADC resolution (resolution combo box on main screen)
- Acquisition time for peripherals (available on each peripheral)

The ADC only option generates ACM MEM files.

See Also

ASB Advanced Options - ASSC RAM
ASB Advanced Options - SMEV RAM
ASB Advanced Options - SMEV Status
ASB Advanced Options - ADC results
ASB Advanced Options - ACM Bus

ASB Advanced Options - Calibration

Analog systems typically require calibration of the analog inputs to achieve more accurate measurements to account for any drift in the manufacturing process. The Analog System Core enables you to include a

Calibration IP that performs this function. The Calibration IP performs a “two point” calibration scheme using the formula Y = M*X + C, where M is the GAIN, X is the Analog converted value, and C is the OFFSET.

Using Calibration changes the default internal Analog System connectivity and creates additional output files. Please see the ASB connectivity and Calibration topic for an explanation.

How does Calibration in the ASB work?

During manufacturing each Fusion part is calibrated with the pertinent M (GAIN) and C (OFFSET) data stored inside the Embedded Flash Memory. At run time, these values are pulled automatically from Flash by the Calibration IP and used to perform the Calibration function on each analog conversion. This calibrated value is then passed to the rest of the system.

Voltage Reference

The M and C data stored into Flash Memory are based on a voltage reference of 2.56V. If you decide to use an external voltage reference that is different then 2.56V then the Calibration option is disabled, and you cannot generate your system with the Calibration IP. Please contact Microsemi technical sales for more information in this case.

Saturation

There are two options related to saturation of the analog input when using Calibration.

- Calibration is always active and a saturation condition on the analog input can be calibrated and adjusted such that it is no longer in saturation.
- If the analog input is saturated, Calibration is bypassed and the saturated value is passed to the rest of the system.

The ASB - Advanced Options dialog box enables you to select the saturation behavior.

How does Calibration affect Simulations?

In simulation, the Analog Block core returns an ideal conversion, which means that the data does not need to be calibrated. To account for this, the Flash Memory CAE model is loaded with M (GAIN) and C (OFFSET) data equaling M=1 and C=0; in other words, the calibration function simply passes thru the same value.

In the real system, this section of Flash Memory is preprogrammed by Microsemi during manufacturing with the proper M and C data for that particular part, and during real system operation, the actual M and C data will be used.

Calibration Side Effects

Using calibration requires an extra 14 system clock latency for each conversion (i.e. reduction in sampling rate) and an increase in tile count.

The actual tile count increase depends on your system clock frequency input in the ASB dialog. If it is greater than 60MHz, then the tile count increase is ~470, otherwise it is ~370.

See Also

ASB Advanced Options

ASB connectivity and calibration

Analog System Builder Calibration output files

ASB Connectivity and Calibration

Enabling calibration changes the internal Analog System connectivity, as shown in red in the diagrams below, and creates additional output files.

IP Cores for ADC data processing and sequence control with calibration

Enables all the Analog Block features: sequencing, flag generation, data averaging, and general ADC management. You can enable or disable access to ADC results, ADC Status ASSC RAM, SMEV RAM, SMEV Status, ACM Bus, and ACM Clock.

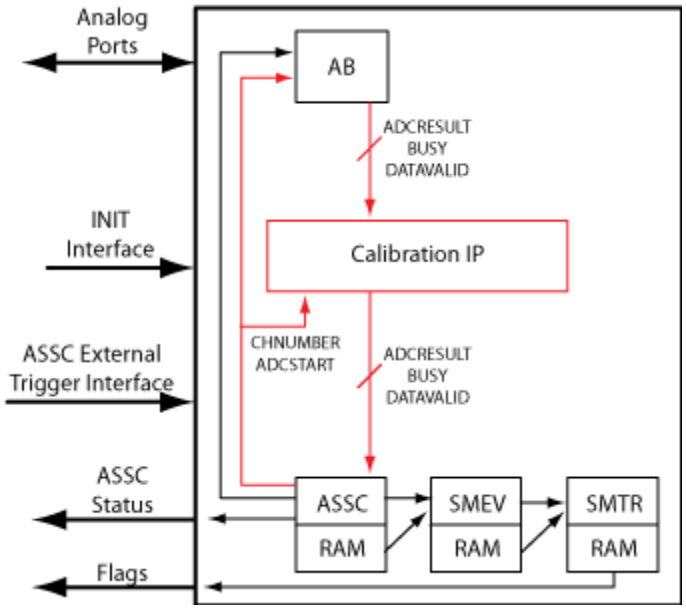


Figure 19 · System Diagram for IP Cores and ADC Data Processing and Sequence Control Options

IP Cores for ADC Sequence control with calibration

This configuration instantiates only the analog block model and the ASSC RAM. The data processing portions of the controller (SMEV and SMTR) are omitted from the design (as shown in the figure below). If you select this option, you must process the ADC data directly from the ADC RESULT bus or the ASSC RAM.

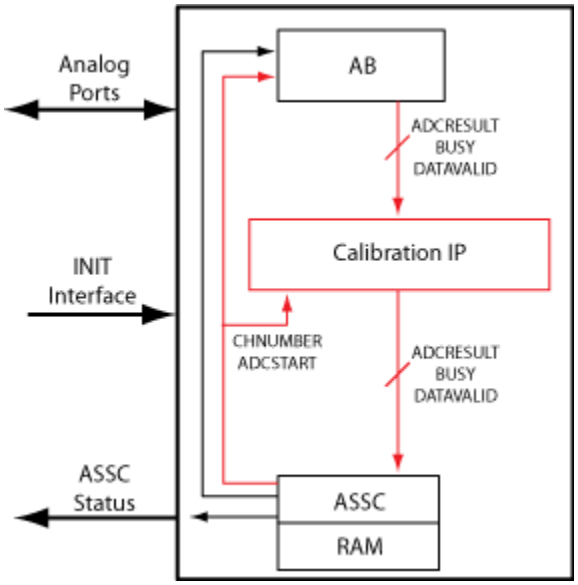


Figure 20 · System Diagram for IP Cores for ADC Sequence Control Only

ADC only with calibration

This configuration instantiates only the Analog Block model (as shown in the figure below). It omits the data processing, sequence controller, and the ADC management features. If you use this configuration you must completely manage the ADC and all related AB functionality. Note that when you export DATAVALID, BUSY, and ADCRESULT signals with calibration, the signals come from the Calibration IP block as shown in the figure below.

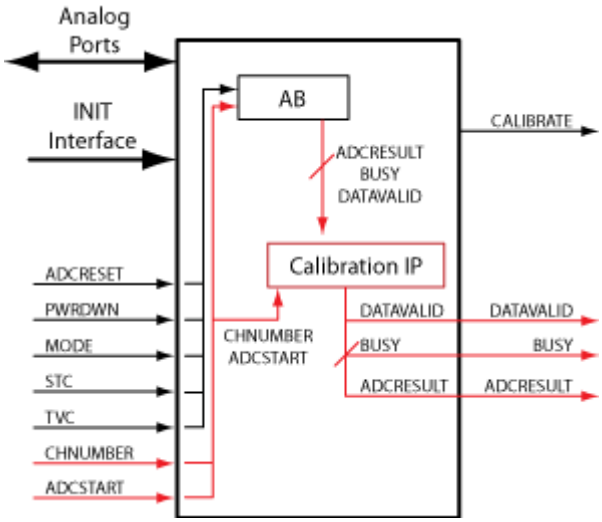


Figure 21 · System Diagram for ADC Only

See Also

[ASB Advanced Options - ASSC RAM]
[ASB Advanced Options - SMEV RAM]
[ASB Advanced Options - SMEV Status]
[ASB Advanced Options - ADC results]
[ASB Advanced Options - ACM Bus]

Analog System Builder Calibration Output Files

The ASB Calibration output files are placed in the common folder.

HDL Source Files (in <workspace_directory/<common>/vhdl or /verilog directory)

calibip.vhd/.v

calibip_brentkung_24.vhd/.v

calibip_clram.vhd/.v

capibip_compute_block.vhd/.v

calibip_ram512x9_afs.vhd/.v

calibip_ripple_24.vhd/.v

/smartgen/<corename>/<corename>_calibip_wrapper.v - Top level HDL wrapper for CalibIP

Memory Files (in /smartgen/<corename> directory)

<corename>_calibcoefficient.mem - Contains Flash Memory System Builder content for the M (GAIN) and C (OFFSET) coefficient section of Flash Memory System Builder.

This is imported by Flash Memory Builder to generate the Simulation Memory File. It is required to allow simulation to work properly, the M and C data in this file are populated with M=1 and C=0. Since the AB simulation model provides 'ideal' conversion results, no calibration is needed so populating the coefficient data with 1 and 0 emulates that scenario.

<corename>_calibip_wrapper.hex - Contains Flash Memory System Builder content for the CalibROM section of the Flash Memory System in IntelHex format.

<corename>_calibip_wrapper_R0C0.mem -Contains Flash Memory System Builder content for the CalibROM section of the Flash Memory System in binary format.

<corename>_calibrom.mem - Contains Flash Memory System Builder content for the CalibROM section of the Flash Memory System in the correct memory format. This is imported by the Flash Memory System Builder to generate the Simulation and Programming Memory File

ASB Advanced Options - ASSC RAM

If you choose to enable user access to ASSC RAM in the ASB Advanced Options dialog box, then you can read the contents of the ASSC RAM.

When performing slot processing, the ASSC stores the raw ADC Result value. The address locations for these values are exported in the log file created by the Analog System Builder.

The ASSC stores its result on a per-slot basis. Thus, if your sequencer samples a single channel multiple times, then the samples for that channel will be in multiple locations of the ASSC RAM. However, each sample result could potentially be different since it is sampled at a different time.

The ASSC continually overwrites each slot location every time it processes a sequence slot.

Each data is stored as a 12-bit value, so you must read two RAM locations to retrieve the value.

The exposed ASSC RAM ports are shown in the table below.

Port Name	Input/Output	Description
USER_ASSC_ADDR[8:0]	Input	**User RAM Address -** You can control these address signals and enable read access from the A-port of the 512x9 ASSC RAM. If unused, these signals must be tied to logic 0 or logic 1.
USER_ASSC_RD	Input	**User RAM Read Enable** - (Active high) You can control the control signal and enable read access from the A-port of the 512x9 ASSC dual-port RAM (you must connect to the ASSC_RAM_DO_A[8:0] port for read data). If unused, this signal must be tied to logic 0. Make sure that the ASSC_RAM_BUSY signal is inactive at logic 0 when this signal is first activated, otherwise, the data read from the A-port of the ASSC RAM will not be from the USER_ADDR[8:0] address.
USER_ASSC_RAM_BUSY	Output	**ASSC RAM Busy** - This output signal indicates that either the Init/Config Soft IP block or the System Monitor Evaluation Phase State Machine Soft IP block is busy accessing the A-port of the ASSC RAM. It is Active High. This signal can be used by user logic outside the analog interface soft IP blocks or can be left unconnected if unused.
ASSC_RAM_WR_BUSY_B	Output	**ASSC Busy Writing** - This active-high signal is for user status monitoring and indicates that the ASSC block is busy writing to the B port of its dual-port RAM. It is Active High.

Port Name	Input/Output	Description
ASSC_RAM_DOUT	Input	D_{out} of Port A of the ASSC RAM

The ASSC memory content report looks like this:

```
*******************************************************************************
                        ASSC Memory Content Report
*******************************************************************************
Slot              Channel   Address      Bits                          Value
-------------------------------------------------------------------------------
   0                volt1
                                    3|   [08:00]|        Raw ADC Result [08:00]
                                    4|   [02:00]|        Raw ADC Result [11:09]
```

See Also

- ASB Advanced Options
- ASB Advanced Options - SMEV RAM
- ASB Advanced Options - ADC results
- ASB Advanced Options - ACM Bus

ASB Advanced Options - SMEV RAM

If you choose to enable user access to SMEV RAM in the ASB Advanced Options dialog box, then you can read the contents of the SMEV RAM.

When processing channels the SMEV stores the digitally-filtered (i.e., averaged) ADC Result value into this RAM. The address locations for these values are exported in the log file created by Analog System Builder. Unlike the ASSC RAM, the SMEV stores its result on a per channel basis.

The SMEV will continually overwrite each channel location every time it processes a channel.

As with the ASSC RAM, the data is stored as a 12-bit value, so you must read two RAM locations to retrieve one value.

The exposed SMEV RAM ports are shown in the table below.

Port Name	Input/Output	Description
USER_EV_ADDR[8:0]	Input	**User RAM Address** - You can control these address signals and enable read access from the A-port of the 512x9 SMEV RAM. If unused, these signals must be tied to logic 0 or logic 1
USER_EV_RD	Input	**User RAM Read Enable** - You can control the control signal and enable read access from the A-port of the 512x9 SMEV dual-port RAM (you must connect to the EV_RAM_DO_A[8:0] port for read data). If unused, this signal must be tied to logic 0. Make sure that the USER_EV_RAM_BUSY signal is inactive at logic 0 while this signal is first activated, otherwise, the data read from the A-port of the SMEV RAM(s) will not be from the USER_EV_ADDR[EV_ASIZE-1:0] address.

Port Name	Input/Output	Description
USER_EV_RAM_BUSY	Output	**SMEV RAM Busy** - This output signal indicates that either the Init/Config Soft IP block or the SMTR Soft IP block is busy accessing the A-port of the SMEV RAM. This signal can be used by user logic external to the analog interface soft IP blocks or can be left unconnected if unused.
EV_RAM_WR_BUSY_B	Output	**SMEV Busy Writing** - This active-high signal is for user status monitoring and indicates that the SMEV block is busy writing to the B port of its dual-port RAM.
SMEV_RAM_DOUT	Input	Dout of Port A of the SMEV RAM

The SMEV memory content report looks like this:

```
********************************************************************************
                        SMEV Memory Content Report
********************************************************************************
                  Channel   Address      Bits                          Value
--------------------------------------------------------------------------------
                    volt1
                                 75|   [08:00]|   Averaged ADC Result [08:00]
                                 76|   [02:00]|   Averaged ADC Result [11:09]
--------------------------------------------------------------------------------
```

See Also

- ASB Advanced Options
- ASB Advanced Options - ASSC RAM
- ASB Advanced Options - ADC results
- ASB Advanced Options - ACM Bus

ASB Advanced Options - SMEV Status

This option exposes status signals from the SMEV; these are useful for monitoring the current processing state of the SMEV.

The exposed SMEV Status ports are shown in the table below.

Port Name	Input/Output	Description
EV_DONE	Output	SMEV Done – Indicates that the SMEV is done for the current channel.
EV_EVFLAG	Output	SMEV Active – Indicates that the SMEV is currently in effect for the channel that has just been sampled by the ASSC.
EV_CHHOLD[4:0]	Output	Channel Value – The channel that the SMEV is currently processing

ASB Advanced Options - ADC results

This option exposes the RESULT port from the AB. It represents the ADC result value that was currently sampled. The exposed port is described in the table below.

The ADC Result bus is a decimal representation of the voltage value for Voltage, Current, and Differential Voltage peripherals. For a temperature peripheral the bus is in degrees Kelvin. See ASB - Calculating a threshold for more information.

Port Name	Input/Output	Description
ADC_RESULT[11:0]	Output	**ADC Result** - These signals comprise the conversion result from the ADC. In 12-bit mode, the ADC uses 11:0 In 10-bit mode, the ADC uses 11:2; 1:0 are grounded In 8-bit mode, the ADC uses 11:4; 3:0 are grounded

See Also

ASB Advanced Options
ASB Advanced Options - ASSC RAM
ASB Advanced Options - SMEV RAM
ASB Advanced Options - ACM Bus
ASB - Calculating a threshold

ASB Advanced Options - ADC status

This option exposes the status ports from the AB.

The exposed ports are:

Port Name	Input/Output	Description
CALIBRATE	Output	ADC Calibration – Indicates the ADC calibration is currently in effect.
BUSY	Output	ADC Busy – Indicates that an Analog conversion is in effect. When this transitions from logic 1 to logic 0, valid conversion data is available on ADC_RESULT.
SAMPLE	Output	ADC Sample – If logic 1 then analog input is sample. Refer to the Fusion datasheet for exact timing of this signal from the AB.

ASB Advanced Options - ACM Bus

This option provides access to the ACM Address and Data signals to update the values of the counter and match register.

Note: Exercise Caution: The ACM address bus is shared by the Analog System and the Analog system configuration could be overwritten if it accesses an incorrect address.

This option exposes the ACM Bus interface allowing users to read / write the ACM registers. Refer to Designing with Analog System Builder for the suggested connectivity scheme when using this bus.

There are 2 options to accessing the ACM Bus:

1. Independent, or
2. Part of Init/CFG interface

ACM Bus Independent

If you select this option, generates a MUX internally to multiplex between the Init/Cfg interface and your user interface to the ACM.

Microsemi recommends this option when directly accessing the ACM. The ports that are exposed when selecting this option are:

Port Name	Input/Output	Description
ACMCLK	Input	Clock for ACM interface. The max frequency for this clock is 10 MHz; it must be the same frequency that is used during Initialization from the Flash Memory System.
ACMRDATA_I[7:0]	Output	Data read from the ACM.
ACMADDR[7:0]	Input	Address interface of the ACM.
ACMWDATA[7:0]	Input	Write data to the ACM.
ACMWEN	Input	Write enable to the ACM.

ACM Bus Part of Init/CFG Interface

If you select this option, you must create your own multiplexors between the Init/CFG interface and your user logic. The INIT_DONE signal from the Flash Memory System can be used to indicate when the Init/CFG operation is complete. This signal should be used to select the MUX.

The ports related to this interface are:

Port Name	Input/Oputput	Description
ACMCLK	Input	Clock for ACM interface. The max frequency for this clock is 10 MHz; it must be the same frequency that is used during Initialization from the Flash Memory System.
ACMRDATA_I[7:0]	Output	Data read from the ACM.
INIT_ADDR[7:0]	Input	The ACM bus is shared with the Initialization interface of the Flash Memory System. This port is always exposed; the address of the ACM uses this port as part of the Initialization interface. You must multiplex your ACM address with the Initialization address into this port. The selection of the multiplexer can be based on the INIT_DONE from the Flash Memory System.
INIT_DATA[7:0]	Input	The ACM bus is shared with the Initialization interface of the Flash Memory System. This port is always exposed; the data written into the ACM uses this port as part of the Initialization interface.

Port Name	Input/Oputput	Description
		You must multiplex your ACM write data with the Initialization data into this port. The selection of the multiplexer can be based on the INIT_DONE from the Flash Memory System.
INIT_ACM_WEN	Input	The ACM bus is shared with the Initialization interface of the Flash Memory System. This port is always exposed; the data write enable of the ACM uses this port as part of the Initialization interface. You must multiplex your ACM write enable with the Initialization write enable into this port. The selection of the multiplexer can be based on the INIT_DONE from the Flash Memory System.

Accessing RTC Registers

When reading the RTC count or match register, which operates in the XTLCLK domain, the appropriate 40-bit value is first copied to a capture register through clock synchronization circuitry, if and ony if the least significant byte of that set of register is addressed. Higher-order bytes of the same set of registers captured with the LSB can then be read on immediately later read cycles. Higher-order bytes of that set of registers can be read in any order but must be read before switching to a different set of registers to ensure data consistency.

For example, RTC counter address ranges from 0x40 to 0x44, register 0x40 must be accessed first before accessing addresses 0x41, 0x42, 0x43, and 0x44 to get the full 40-bit value.

See the Fusion Datasheet for the detailed register description and how to setup the CTRL/STAT register for RTC read and write.

See Also

ASB Advanced Options

ASB Advanced Options - ASSC RAM

ASB Advanced Options - SMEV RAM

ASB Advanced Options - ADC results

ACM Register Map in Fusion Datasheet: ACM Address Decode Table for Analog Quad

ASB Advanced Options - ACM Clock

This option exposes the ACM Clock to the top level.

See Designing with the Analog System to view Microsemi's recommendations on clock schemes with Fusion designs.

The port that will be exposed when using this option is:

Port Name	Input/Output	Description
ACMCLK	Input	Clock for ACM interface. The max frequency for this clock is 10 MHz; it must be the same frequency that is used during Initialization from the Flash Memory System.

ASB - Calculating a Threshold

This section describes the ASB threshold conversion logic. These equations are performed by ASB and do not need to be employed by you unless you are directly monitoring the raw ADC RESULT values.

Scaling Factor is determined based upon the selected prescaler range. Refer to the Prescaler range topic for a list of prescaler range values.

The value of each LSB bit is determined by dividing VREF by 2^12. This calculation is always performed with 12-bit mode, the masking determines the resolution (see below).

Voltage Monitor

Calculated Threshold = (UserThreshold(V) * ScalingFactor * Value of each LSB)

Current Monitor and Differential Monitor

The Differential Voltage Monitor uses the same block as the Current Monitor so it has the same gain.

```
Calculated Threshold =
(user threshold(A)*
Resistor Value(Ohm)*
Gain Applied by Current Monitor)*
(Value of each LSB bit)
```

Gain = 10 for Current Monitor and Differential Voltage Monitor

Then masking is performed; see below.

Temperature Monitor

The temperature value from the ADC is in degrees Kelvin. For example, using the internal VREF of 2.56V and 10-bit resolution, each LSB of the ADC result = 1K. This is generalized for internal VREF to:

8-bit mode -> 1LSB = 4K

10-bit mode -> 1LSB = 1K

12-bit mode -> 1LSB = .25K

For example:

8-bit mode -> ADC LSB = 01001010b (74) = 74 * 4 = 296K

10-bit mode -> ADC LSB = 0100101010b (298) = 298 * 1 = 298K

12-bit mode -> ADC LSB = 000100101010 (298) = 298 * .25 = 74.5K

The general equation is:

K = 2.30258 * 0.0000087248 (derived from diode equation)

Note: These constants are derived from

Calculated Threshold = (user threshold (K)) * (K) * (Gain Applied by Temperature Monitor)

Gain = 12.5 for Temperature Monitor

Then masking is performed (see below).

Masking on Resolution

The calculated threshold is masked depending upon the resolution.

- 12-bit resolution - No bits are masked
- 10-bit resolution - Bits 0 & 1 are set to ‘0’
- 8-bit resolution - Bits [3:0] are set to ‘0’

Prescaler Range

The prescaler range is determined from the maximum input voltage field. See the table below to calculate your prescaler range.

Scaling Factor: Pad to ADC Input	LSB for 8-bit conversion (mV)	LSB for 10-bit conversion (mV	LSB for 12-bit conversion (mV	Full-Scale Voltage	Range Name (V)
0.15625	64	16	4	16.368	16
0.3125	32	8	2	8.184	8
0.625	16	4	1	4.092	4
1.25	8	2	0.5	2.046	2
2.5	4	1	0.25	1.023	1
5.0	2	0.5	0.125	0.5115	0.5
10.0	1	0.25	0.0625	0.25575	0.25
20.0	0.5	0.125	0.03125	0.127875	0.125

If the maximum input voltage is greater than the given range, it will select the higher range.

The corresponding ranges for negative polarity is the same.

The prescaler logic of the AB has a settling time of 10µs (max). It is an application-dependent setting and must be accounted for by you via the acquisition and hold time for each channel. A recommended default value is inserted by the ASB when configuring a new Voltage Monitor but it may be reduced with a possible reduction in sampling accuracy. See the Fusion datasheet for information.

Acquisition Time

The required settling/sampling time for this channel. It is the amount of time the Sample and Hold circuit in the ADC charges the capacitor with the input analog signal. The characteristics of your system and monitoring requirements will determine this value. Note also that this value has a direct correlation to the achievable sampling rate of the system.

The prescaler logic of the AB has a settling time of 10 µs (max). It is an application-dependent setting and must be accounted for by you via the acquisition and hold time for each channel. A recommended default value is inserted when configuring a new Voltage Monitor but it may be reduced with the possible reduction in sampling accuracy. See the Fusion datasheet for information.

See Also

Sampling rate in Analog System Builder

ASB Channel Mapping

In Analog System Builder you must specify names for each peripheral you configure. For example, you can name your Voltage Monitor “MYVOLT” and it the name will appear in your netlists as “MYVOLT”.

However, the analog block (AB) names Analog Ports AV0, AV1, … , AC0, AC1, … AT0, AT1, etc. Also, each port is physically tied to a specific pin on the package. For example, AV0 is physical pin 99 on the package; AV1 is physical pin 103 on the package, etc.

The physical pin placement is communicated directly through the netlist. So, when Designer imports a netlist with an AB, it infers the pin placement from the netlist.

Thus, the core has to map your logical channel into the physical channel on the AB.

Note: If you modify the die or package of a project the ASB automatically reverts all user-assigned pins to UNASSIGNED.

Mapping Requirements

The mapping rules are as follows:

- Voltage Monitors can be placed on AV, AC, or AT pads; Voltage Monitors placed on AV and AC pads can only be of the range -10.5 to 12V. Voltage Monitors on AT pads can only have prescalers in the 4V and 16V range.
- Current Monitors can be placed only on AC pads with the additional requirement that the adjacent AV pad is also available. This is due to the fact that a Current Monitor is an AC-AV pair.
- Temperature Monitors can be placed only on AT pads
- Gate Drivers can be placed only on AG pads
- Digital Inputs can be placed on AV, AC, or AT pads
- Internal channels do not have physical pin mappings, they reside purely on-chip.
- See the Fusion datasheet for the number of peripherals available for each device.

Designing with the Analog System

The Analog System is initialized via the Analog Configuration MUX (ACM). The ACM supports a maximum frequency of 10 MHz. Therefore, Analog System initialization is limited to less than 10 MHz.

The PLL and NGMUX can be used as needed to enable initialization to run at less than 10MHz and normal operation at greater than 10MHz.

See Also

Analog System clocks

ASB and FMB basic configuration

ASB with RTC

ASB with ACM access

Analog System Clocks

The following is a short summary of the important clocks when you use and configure the Analog System.

Initialization Clock (SLOWCLK)

System initialization frequency. Any save operations occur at this frequency.

The INIT IP was designed to operate only at slow frequencies for initialization and save operations. A large 64 to 1 data mux exists in this design so as the number of clients increase, the achievable frequency decreases.

Microsemi recommends that you keep the initialization frequency below 10Mhz. This is an application-dependent decision unless you are accessing the ACM port interface.

The initialization frequency must be less than 10 Mhz when you use the Flash Memory Block to initialize the ASB (because it accesses the ACM port interface).

ACM Clock

Analog Configuration MUX frequency. This block resides in the AB library macro. Its purpose is to house the registers for Analog I/O configuration and RTC configuration.

The ACM port interface has a max frequency of 10 Mhz. This is a silicon requirement and must be adhered to whenever using the ACM port interface.

If the ACM is already configured (during initialization) and the interface is not used during normal operation, it is safe to connect this clock to a frequency greater than 10 Mhz, however the ACM is inoperable in this state. The AB library macro reports warning messages during simulation about the frequency being exceeded but they can be safely ignored at this stage.

System Clock (FASTCLK)

This is the frequency that is entered into ASB dialog box. It is the frequency at which the ASB and FMB run during normal execution (after initialization is complete).

ASB needs this information because it calculates the ADC Clock from this frequency (see ADC Clock below).

ADC Clock

All ADC conversions operate at this frequency. Silicon requirements limit it to less than 10 Mhz. This has no relationship to the ACM Clock.

This clock is completely internal to the AB block. ASB sets this clock. The maximum ideal frequency for the ADC is based on the System Frequency entered in the ASB during Analog System configuration. This is based on the acquisition times entered for each peripheral.

There are a limited number of dividers available in the AB macro for this clock calculation, specifically 0, 4, 8, 12, 16, etc. This implies that certain System Clock settings result in faster ADC Clock frequencies. For example, a System Clock frequency of 40 Mhz enables a maximum possible 10 Mhz ADC Clock, whereas a 50 Mhz System Frequency results in a slower ADC Clock.

Real Time Counter (RTC) Clock

The RTC Clock is the clock frequency for the Real Time Counter logic on the chip. This is modeled inside the AB library macro.

The RTC Clock is exposed on the ASB if the RTC peripheral is configured within the tool (i.e. not configured externally). Connect this port to the CLKOUT of the XTLOSC macro if in use, otherwise it must be grounded.

This clock has no relationship with any of the other clocks listed here.

Flash Memory Block (FMB) User Clock

When using the FMB with a data storage client, an extra clock port (USRCLK) is exposed on the top level. This clock has a maximum frequency of 100 Mhz because of the NVM.

This clock is intended as the frequency at which the user accesses the NVM. It is muxed with the INIT IP functionality. The arbitration gives INIT IP highest priority.

This clock has no relationship with any of the other clocks listed here.

See Also

Designing with Analog System

ASB and FMB basic configuration

ASB with RTC

ASB with ACM access

Analog System Builder and Flash Memory Block Basic Configuration

The diagram below illustrates the recommended system configuration when using the Analog System Builder and Flash Memory Block (FMB). The ASB and FMB shown below are the default exported systems when using basic ASB features and using FMB with an Analog client only.

You can use a PLL to create the two clock frequencies necessary (FASTCLK and SLOWCLK), the GLA and GLC of this PLL must then be fed into a NGMUX for clock switching. During initialization the SLOWCLK is used to drive both FMB and ASB subsystems, and after initialization the clock switches to the faster clock (as shown in the figure below).

This clock switching is required because of the silicon requirement on the ACM interface. However, once the initialization of the ACM is complete, the ACMCLK can be driven with the faster clock. The library model issues warnings indicating that this clock is being driven faster than 10 Mhz but these warnings can be ignored after the initialization phase, as long as the ACM will not be accessed while being clocked greater than 10 MHz.

If these warnings are unwanted the ACMCLK can be tied directly to the SLOWCLK. To export the ACMCLK open the Analog System Builder Advanced Options dialog box and export the ACM clock.

If you export the ACM clock, the system exports an ACMCLK port for the Analog System. GLC on the PLL can then be directly connected to the ACMCLK. This eliminates the simulation warning messages.

The INIT_DONE from the FMB is used for the selection of the NGMUX.

In the following diagrams “AS IP” and “AS RAM” consist of the Analog System Soft IP modules and their associated RAM blocks.

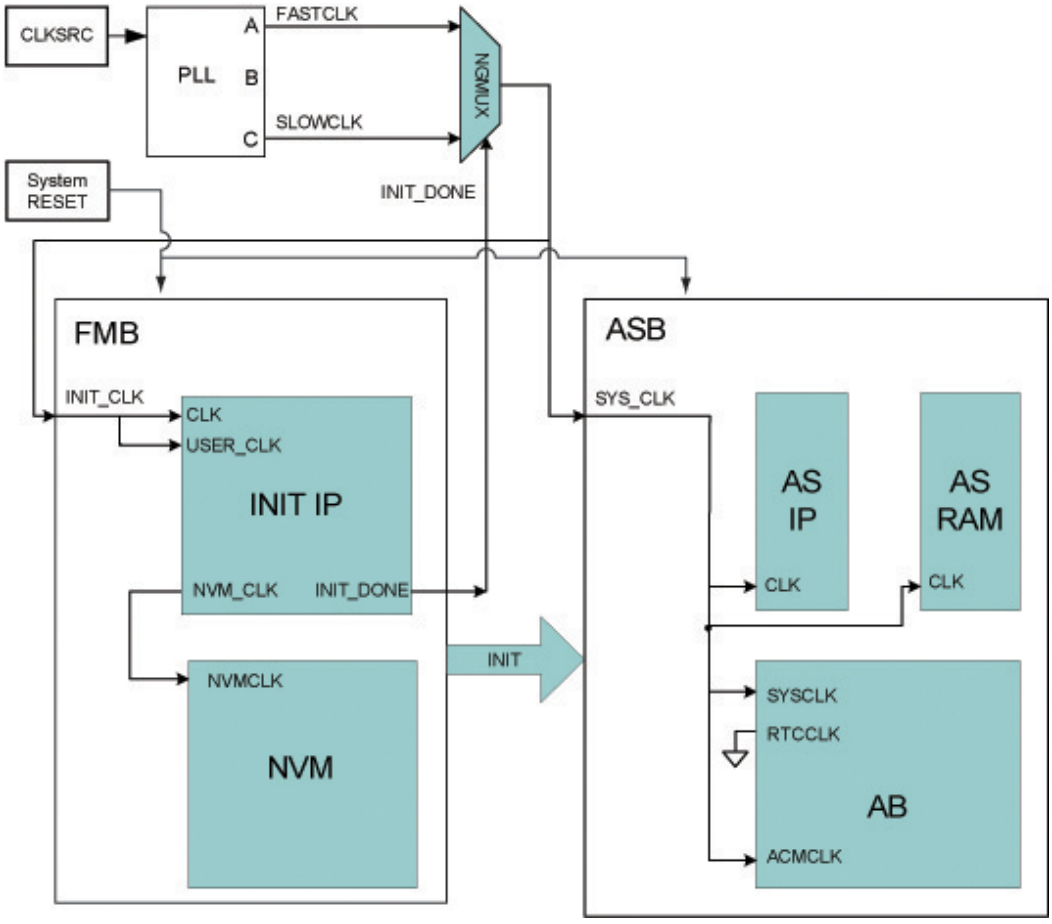


Figure 22 · ASB and FMB Basic Configuration

Analog System Builder with Real Time Counter (RTC)

When using the RTC in the Analog System Builder (ASB), an extra RTCCLK is exposed on the ASB core that must be properly connected.

This port requires a connection from the CLKOUT of the XTLOSC. The output signals RTCXTLSEL and RTCXTLMODE must also be connected from the ASB module to the XTLOSC module.

The diagram below illustrates this use model.

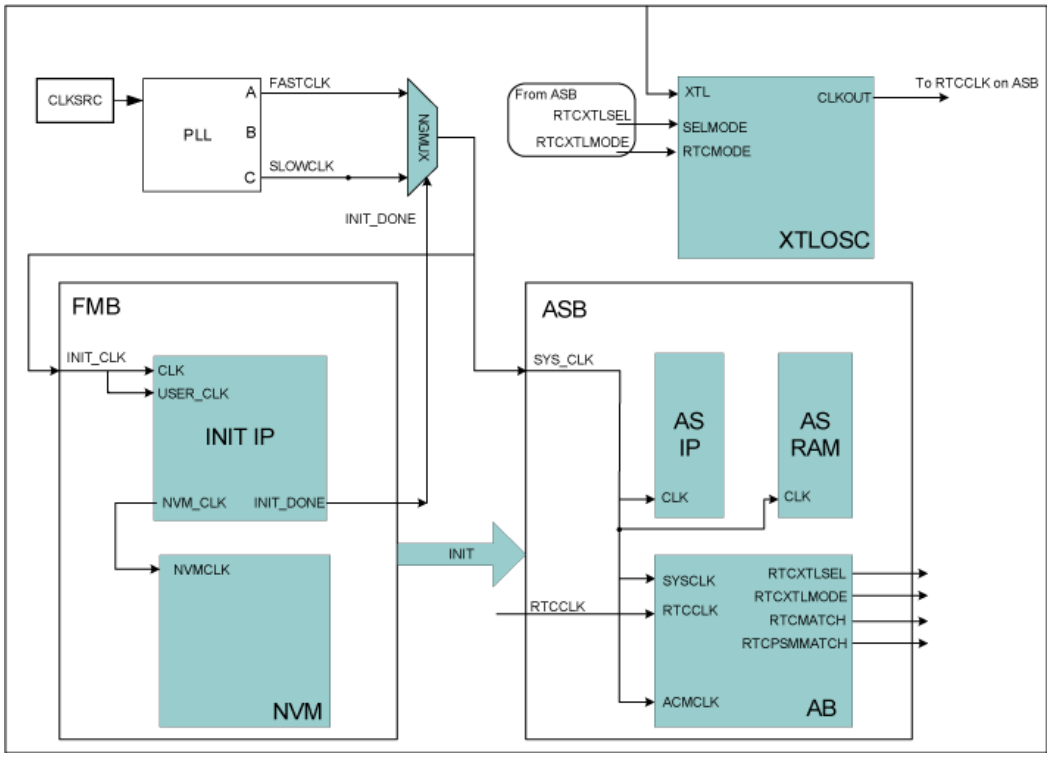


Figure 23 · Analog System Builder with RTC

Analog System Builder with ACM Access

This use model is effective if you require access to the ACM during normal operation. The primary use is to access the Real Time Counter (RTC) registers for calendar or real-time applications. Alternatively, the ACM can be used to access the Analog I/O configurations.

In this configuration, the ASB contains additional ports to access the ACM. Refer to ASB Advanced Options – ACM Bus for more information.

The interface to the ACM must be run at less than 10Mhz, and the User block must run at the same clock frequency as SLOWCLK.

Another option is to place some synchronization logic in between your own block and the ACM interface. This enables you to run your own logic at a clock frequency independent of the ACM. shown below describes the setup if you select the ACM Bus as part of the Init/CFG interface.

If you choose the ACM Bus Independent option (in Advanced Options) then the multiplexor shown below is instantiated directly in the Analog System block (ASB below) and you need to hook directly onto the ACM Bus interface.

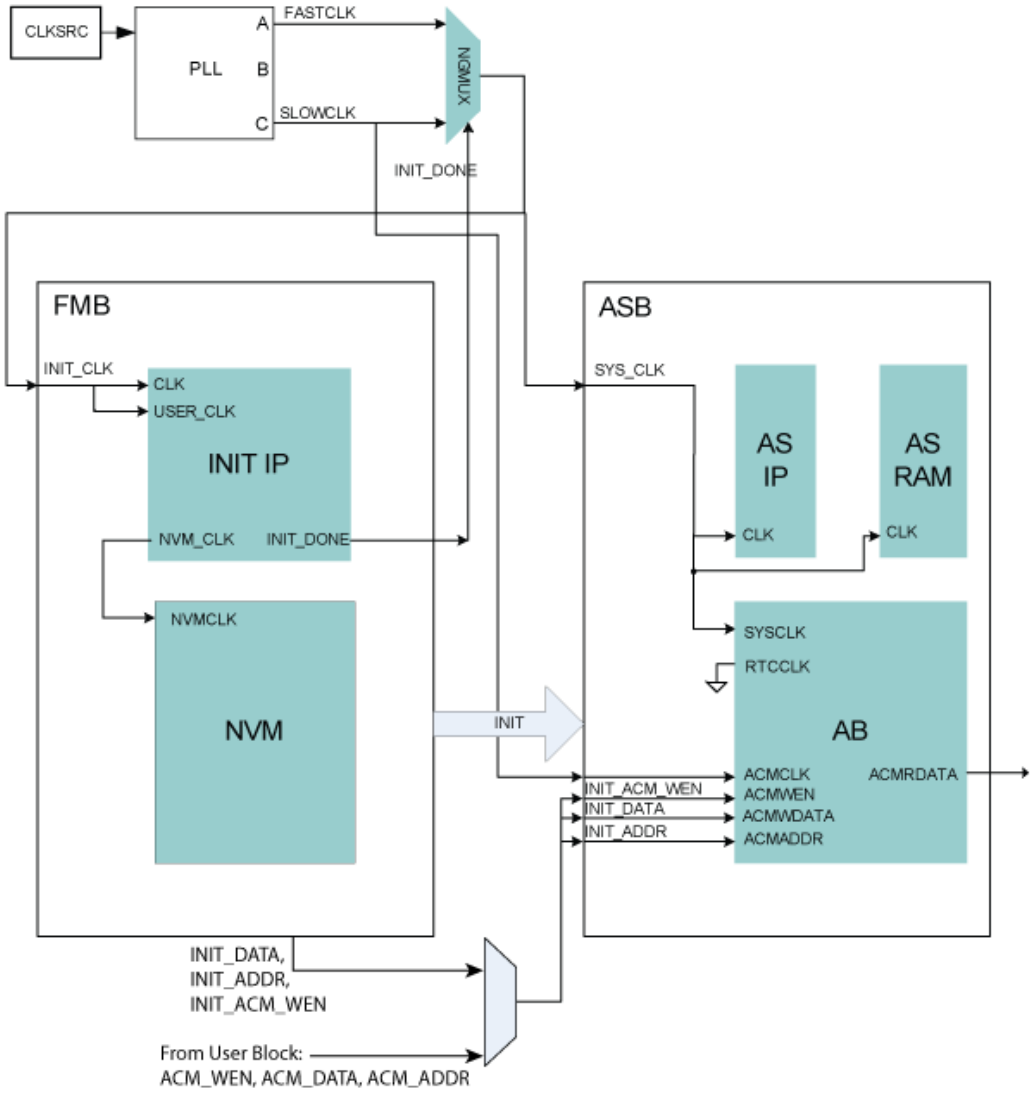


Figure 24 · Analog System Builder with ACM Access

Analog System Builder Peripherals

Current Monitor

The current monitor in Fusion measures current by measuring the differential voltage across a resistor between a pair of Voltage and Current Input channels. This peripheral requires two channels, one of type V and one of type C, and they must be on adjacent package pins.

The differential voltage is multiplied by 10x before it is applied to the ADC; there is no pre-scaling on the differential voltage measurement. The difference in voltages must be less than the value of Vref, external or internal. You must choose an external resistor that satisfies this condition.

The differential amp gain in the current monitor is 10X. ASB assumes a series resistor because it is being used to measure current. The differential amplifier measures the potential/voltage drop (256 mV max if the reference is 2.56V) across the resistor, which is proportional to the current flowing in the direction AV -> AC (I = (change in voltage)/resistance)). The voltage channel in the pair can be used as a voltage monitor to measure the actual voltage that is connected to the Voltage channel.

See the Configuring Current, Voltage, and Temp peripherals for information on the digital filtering factor, Acquisition time, and Comparison Flag Specifications.

Controls unique to this peripheral are the External resistor and >Maximum voltage.

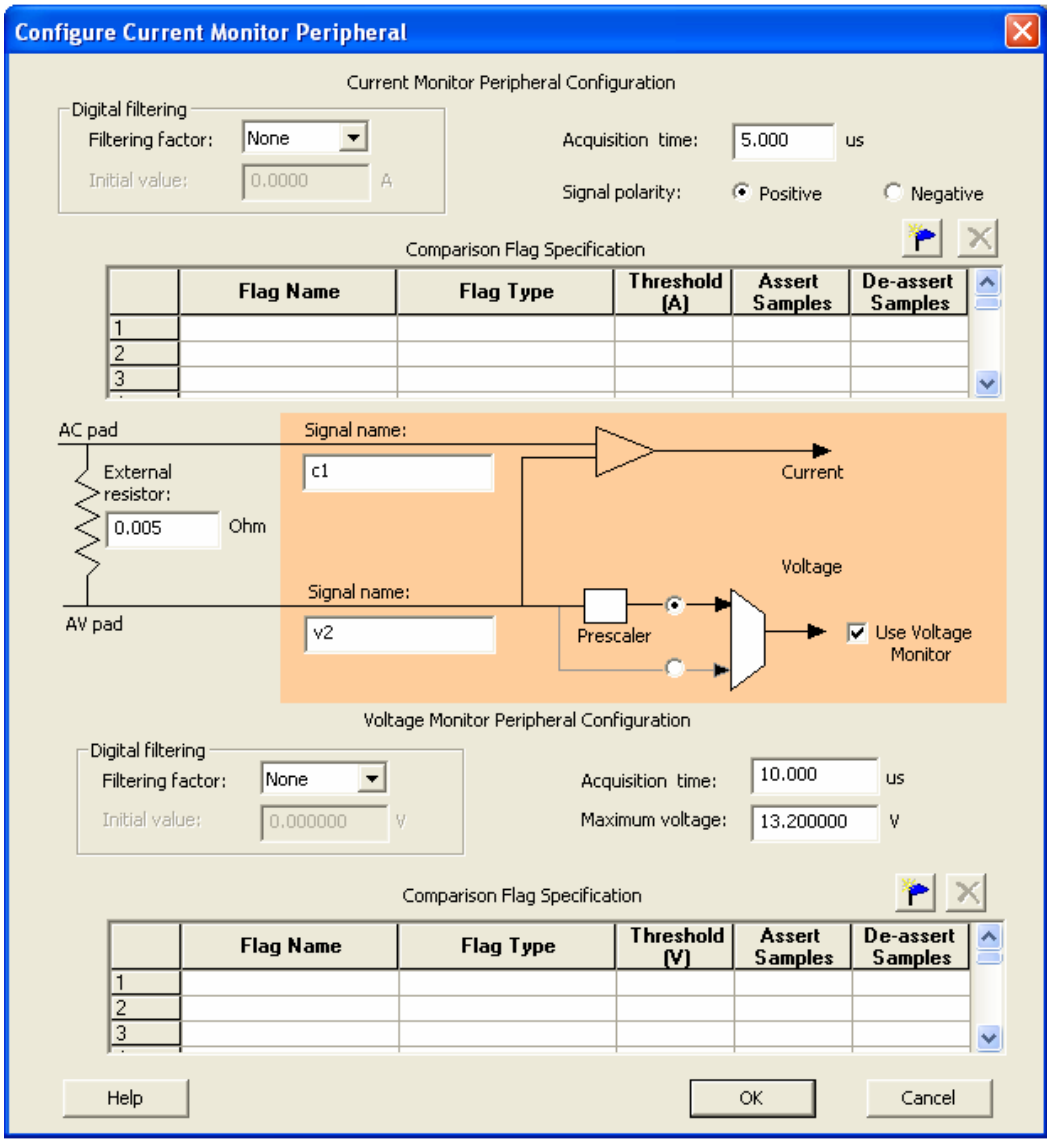


Figure 25 · Configure Current Monitor Peripheral Dialog Box

External Resistor - The value of the resistor that is connected across the Current-Voltage pair, external to the device. ASB uses this value to convert the thresholds into voltages.

Polarity - Sets the polarity to Positive or Negative. The associated Voltage values must match this polarity. ASB returns a warning if the Voltage values do not match the polarity. When you select a negative polarity, the prescaler option on the associated Voltage Monitor defaults to the "Prescaler" path.

Maximum voltage - The maximum anticipated voltage measured by this Voltage Monitor peripheral pad. The range is -10.5V to +16V (the voltage range is NOT bipolar). The ADC is capable of measuring a voltage range of 0 - Vref. For the Internal voltage reference, this value is 2.56V. ASB automatically configures the prescaler in the AB Analog Block for this peripheral to maximize the available voltage range. ASB also post-scales the digital result of ADC conversion so that it returns a result in your specified range.

Differential Voltage Monitor

The differential voltage monitor uses the same components as the Current monitor.

The differential voltage monitor in Fusion measures the differential voltage between a pair of Voltage and Current Input channels. This peripheral requires two channels, one of type V and one of type C, and they must be on adjacent package pins.

The differential amp gain in the differential voltage monitor is 10x. The differential amplifier measures the potential/voltage drop (256 mV max if the reference is 2.56V) across the two inputs.

The voltage channel in the pair can be used as a voltage monitor to measure the actual voltage that is connected to the Voltage channel.

See Configuring Current, Voltage, and Temp peripherals for information on the digital filtering factor, Acquisition time, and Comparison Flag Specifications.

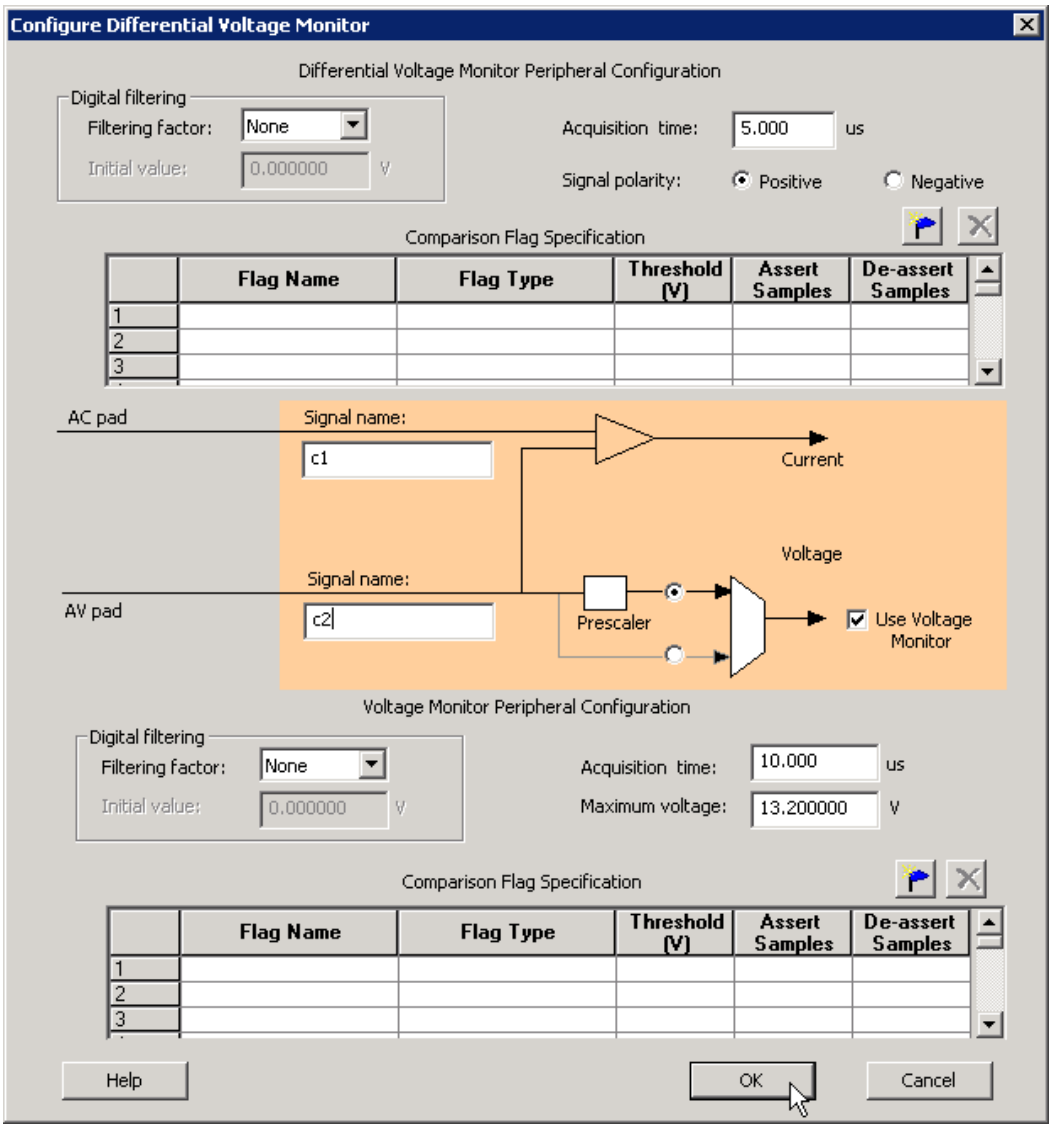


Figure 26 · Differential Voltage Monitor

Polarity - The polarity of the peripheral can be set to Positive or Negative. The associated Voltage values must match this polarity. The ASB returns a warning if the voltage values do not match. When you select a negative polarity, the prescaler option on the associated Voltage Monitor defaults to the "Prescaler" path.

Maximum voltage - The maximum anticipated voltage measured by this Voltage Monitor peripheral pad. The range is -10.5V to +16V (the voltage range is NOT bipolar). The ADC is capable of measuring a voltage range of 0 - V_{ref}. For the Internal voltage reference, this value is 2.56V. ASB automatically configures the prescaler in the Analog Block for this peripheral to maximize the available voltage range.

Direct Digital Input

You can use the Direct Digital Input to configure the unused analog channels or peripherals as digital inputs. Specify the input and output signal name and add the inputs to your sampling sequence. The digital input can be up to 12V but is limited to a frequency of 10 MHz.

Typical delay for the digital input is 10ns.

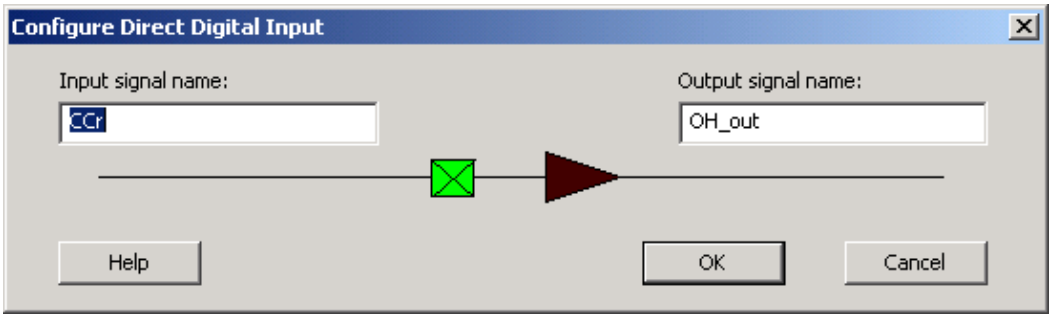


Figure 27 · Configure Direct Digital Input Dialog Box

Gate Driver

The Gate Driver is the analog output coming from the analog system. You can use it to drive the gate of an external MOSFET on or off. The Gate Driver is designed to work with external MOSFETs as a configurable current sink or source.

The figure below shows the Gate Driver dialog box.

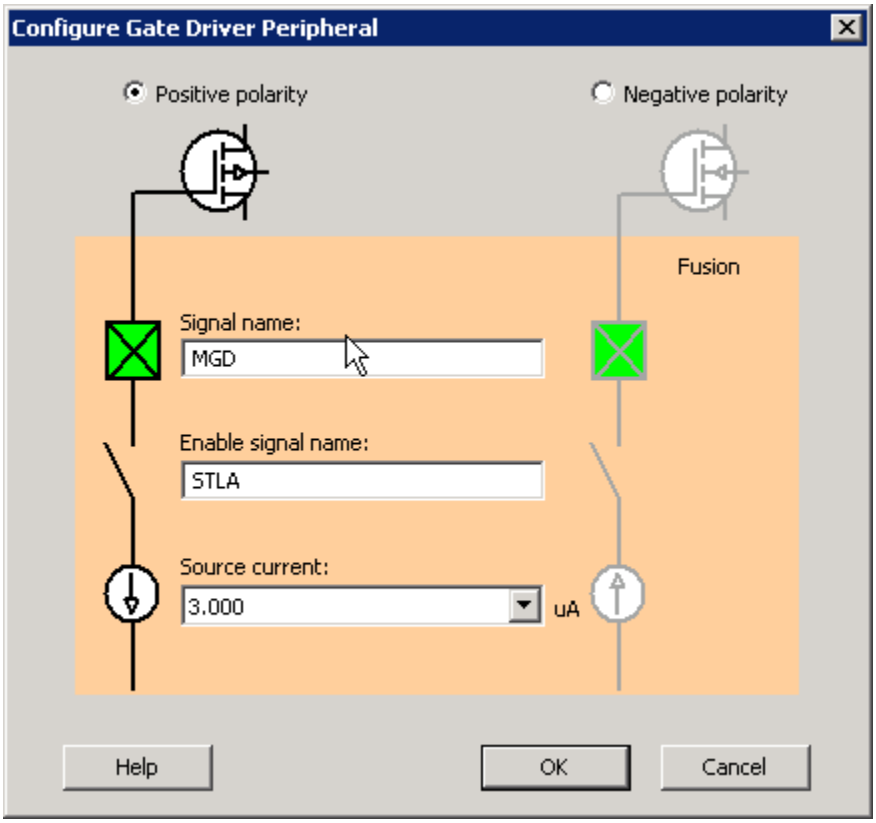


Figure 28 · Output Gate Driver Dialog Box

Gate Driver Polarity

Indicates the type of MOSFET controlled by the gate driver (NMOS or PMOS).

PMOS controls positive voltage supply and NMOS controls negative voltage supply. The AG pad can work with either P-Channel (pulling down to ground) or N-Channel (pulling up to ground) devices.

Signal Name

The name of the signal assigned to this gate driver. It appears as the final port name in the generated system.

Enable Name

The name of the signal that enables the gate driver

Source Current

The amount of current the device is expected to source. The Gate Driver supports selectable current drive levels: 1 µA, 3 µA, 10 µA, 30 µA, and 25mA.

Internal Temperature Monitor

The internal temperature channel measures the internal die temperature (as shown in the figure below). The peripheral is mapped to logical channel 31 and does not have an associated physical pin.

See the Configuring Current, Voltage, and Temp peripherals for information on the digital filtering factor, Acquisition time, and Comparison Flag Specifications.

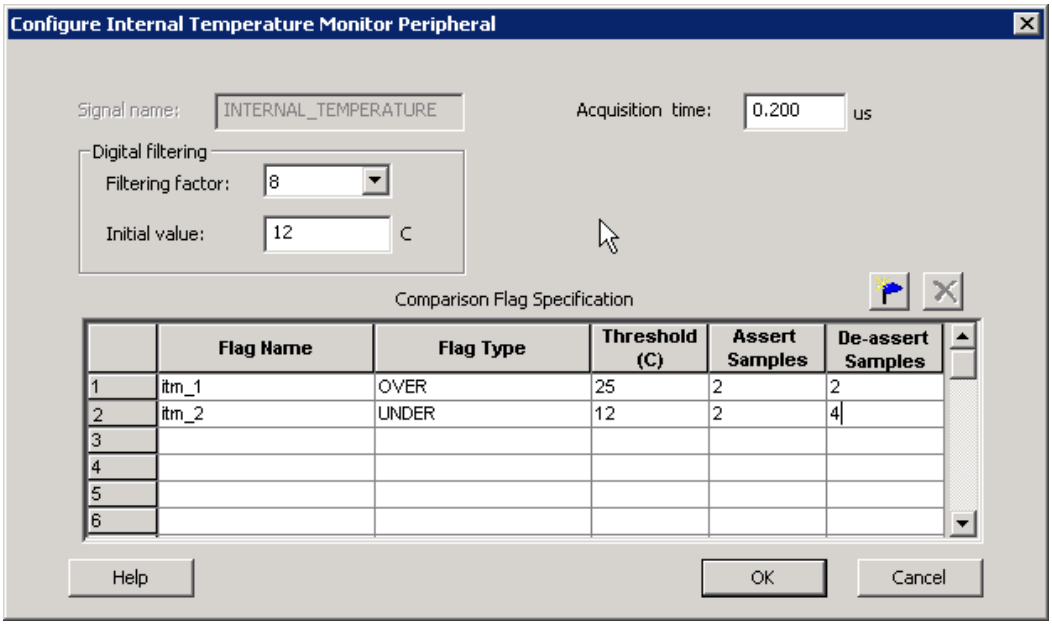


Figure 29 · Internal Temperature Monitor Dialog Box

Internal Voltage Monitor

The internal voltage channel measures the internal 1.5V power supply. The Maximum Input Voltage for this peripheral is fixed (as shown in the figure below). The peripheral is mapped to logic channel 0 and does not have an associated physical pin.

See the Configuring Current, Voltage, and Temp peripherals for information on the Digital filtering factor, Acquisition time, and Comparison Flag Specifications.

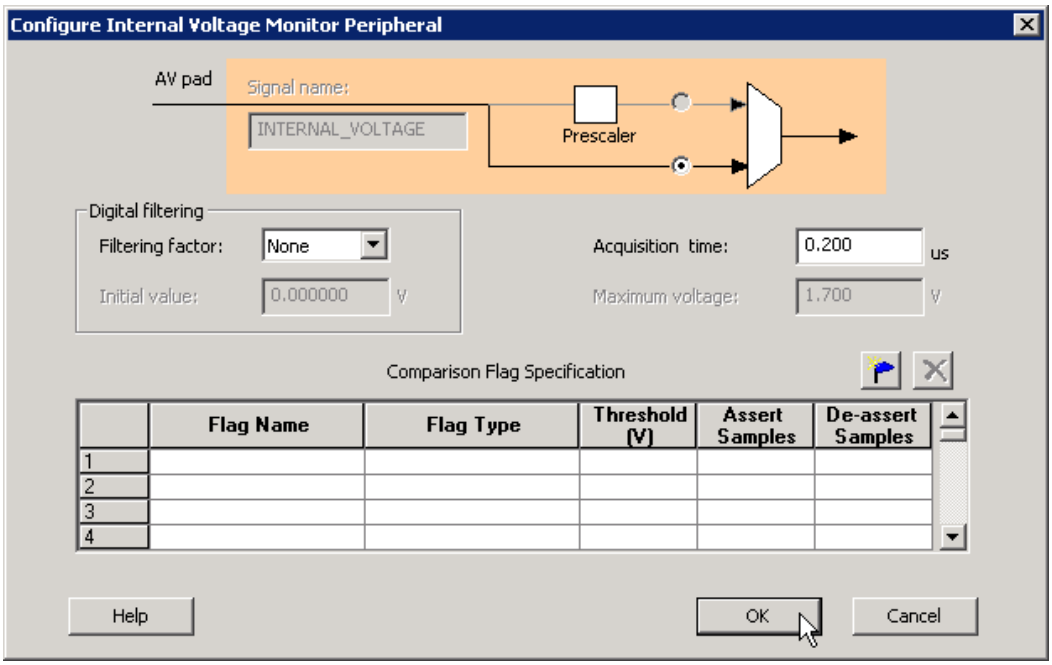


Figure 30 · Internal Voltage Monitor Dialog Box

Real Time Counter

The on-chip crystal oscillator circuit works with an off-chip crystal to generate a high-precision clock. It has an accuracy of 100 ppm (0.01%) and is capable of providing system clocks for Fusion peripherals and the other system clock networks, both on- and off-chip. When combined with the on-chip CCC/PLL blocks, a wide range of clock frequencies can be created to support various design requirements.

The Real Time Counter inside the Analog Block has the following features:

- The MATCH signal on the output of the system asserts when the value in the counter matches the value specified in the match register. Also, there is an optional output RTCPSMMATCH that is triggered on match. The RTCPSMMATCH signal must be connected to the RTCPSM macro so that the Voltage Regulator activates when the Match signal is asserted.
- If you use the RTC, RTCCLK must be driven by the External Crystal Oscillator driving the Fusion device and the mode of the Crystal oscillator must be controlled by the RTC.
- Displays two different views: Alarm or Custom view. The alarm time specified in the RTC dialog relates to the deassertion of one RTCMATCH signal to the reassertion of the signal.

See Designing with Analog System Builder for more information on connectivity.

The figure below shows the Real Time Counter dialog box.

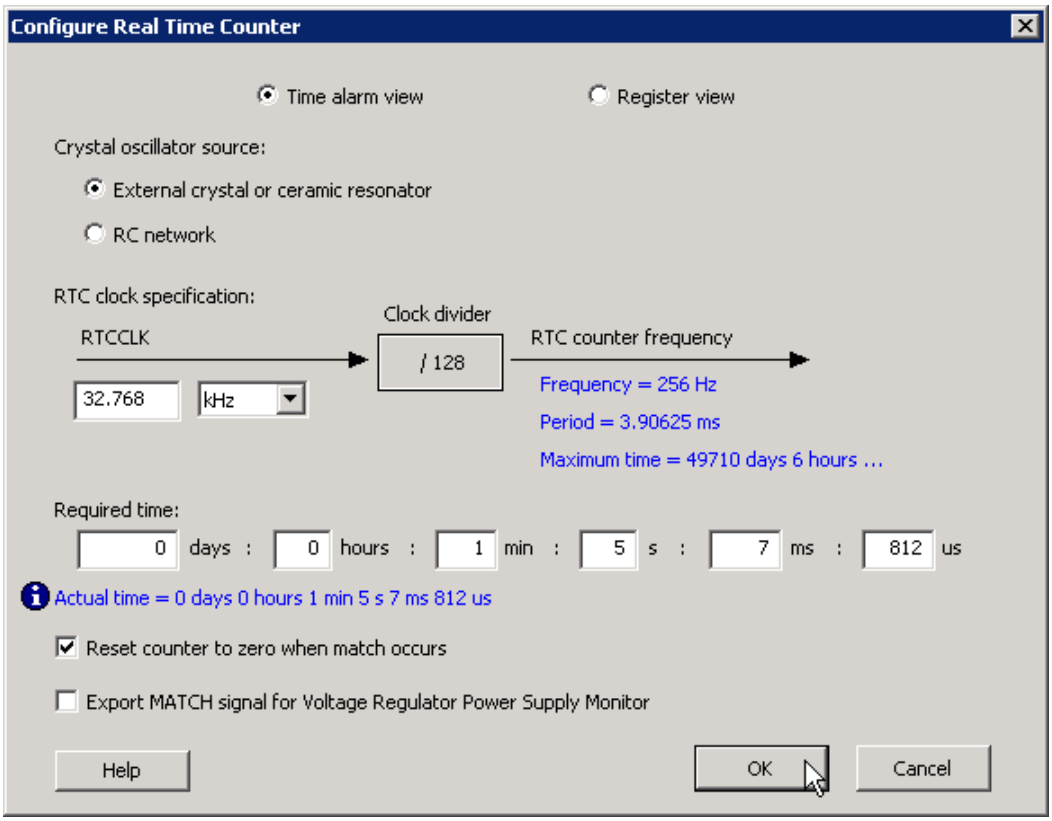


Figure 31 · Real Time Counter Dialog Box

Time Alarm View - This view enables you to set time-based configuration of the RTC. The required time can be entered in days, hours, minutes, seconds, milliseconds, and microseconds. Granularity of achievable time values is based on the frequency you set. The actual time used is shown in blue.

The Maximum Time shows the total representable time with the 40-bit register in the RTC and the specified clock frequency.

Note: This mode disables the use of the initial value setting of the RTC. It is set automatically to 0.

You can specify a required time, ASB displays an actual time. The actual time reflects what is actually achievable based on the clock frequency of the RTC.

For example, when the clock frequency is 32.768 kHz, each clock tick equals ~3.9ms. If you enter 4ms and 100µs for a required time, ASB approximates using the next achievable time. In this example, 7.8ms (shown in the actual time field).

Register View - This view allows you to configure the RTC in its direct hardware representation. Also, this mode enables you to configure the initial value setting.

Initial Value - You may specify a different counter start value. The default is zero. This also is a 40 binary or a 10 bit hex value.

Match with Register Value - The MATCH signal asserts when the counter is equal to this register value (40-bit binary or 10-bit hex).

Crystal Oscillator source

External crystal or ceramic oscillator – Oscillator is configured to work with an external crystal or ceramic resonator.

RC network - Oscillator is configured to work with an external resistor-capacitor network.

RTC clock specification

RTCCLK – The frequency of the CLKOUT from the crystal oscillator

Based on this frequency and the crystal oscillator mode selection, ASB automatically configures the mode, as shown in the table below:

Mode	Recommended Capacitor	Frequency Range (in MHz)
LOW_GAIN	100 pf	0.032 to 0.20
MEDIUM_GAIN	100 pf	0.21 to 2.0
HIGH_GAIN	15 pf	2.1 to 20.0

The RTCLK is divided internally by 128 to generate the frequency at which the real time counter operates. The frequency and period are shown in blue in the dialog box.

Reset count to zero when match occurs- Resets the counter to zero once the match occurs. This can be disabled for an application that measures elapsed time.

Export Match signal for Voltage Regulator Power Supply Monitor - Asserts the RTCPSMMATCH to activate the Voltage Regulator Power Supply Monitor (VRPSM).

If you wish to update the match and counter registers of the RTC dynamically, you must use ASB Advanced Options and select theACM Bus checkbox.

RTC Signals

All signals are active high.

Name	Type	Direction
RTCCLK	INPUT	RTC Clock; must be driven by the XTLOUT of the XTLOSC macro
RTCXTLSEL	OUTPUT	Crystal Oscillator select; must be connected to the XTLSEL on the XTLOSC macro
RTCXTLMODE[1:0]	OUTPUT	Crystal Oscillator mode select; must be connected to RTCXTLMODE on the XTLOSC macro
RTCMATCH	OUTPUT	Match signal when match value reached
RTCPSMMATCH	OUTPUT	Match signal when match value reached; must be connected to RTCMATCH of VRPSM macro. Exposed only if "Export Match signal for Voltage Regulator Power Supply Monitor" is enabled.

The following timing diagram shows accessing the ACM for the RTC values. In the diagram, the RTC counter is disabled. Then, the RTC match register is read and finally the match register is overwritten with a new value. The RTC is then re-enabled. Refer to the Fusion Datasheet - Real Time Counter section for more information.

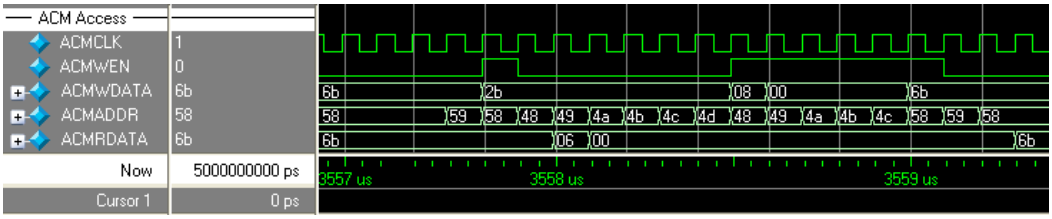


Figure 32 · ASB RTC Access

Temperature Monitor

When used in conjunction with an external bipolar transistor, the Temperature Monitor is designed to measure temperature of an external location. A temperature monitor circuit can be very sensitive to system noise.

See the Configuring Current, Voltage, and Temp peripherals for information on the Digital filtering factor, Acquisition time, and Comparison Flag Specifications.

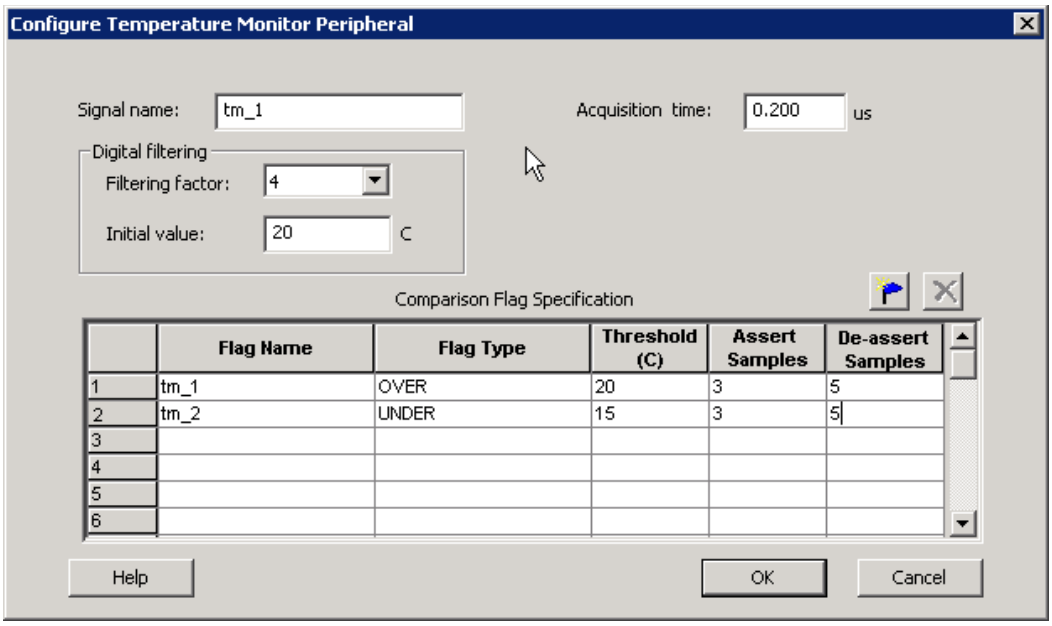


Figure 33 · Configure Temperature Monitor Peripheral Dialog Box

Voltage Monitor

The Voltage Monitor contains a 2-channel analog multiplexer that allows an incoming analog signal to be routed directly to the analog-to-digital converter, or allows the signal to be routed to a prescaler circuit before being sent to the ADC.

See the Configuring Current, Voltage, and Temp peripherals for information on the Digital filtering factor, Acquisition time, and Comparison Flag Specifications.

Use Prescaler or Direct Analog

Each Analog channel has an associated prescaler circuit which can prescale the input signal to a appropriate value supported by the ADC. Using the prescaler circuit imposes certain requirements on the acquisition time of the peripheral. The two radio buttons in front of the multiplexer determine if the Voltage monitor uses prescaling or not.

If the input value is less than the ADC reference voltage and high accuracy is critical for the application, choose the direct analog path. The prescaler circuit could introduce potential gain errors or offset errors. If resolution is more important than accuracy for input voltage ranges lower than the ADC reference voltage, choose the prescaler path.

The prescaler logic of the AB has a settling time of 10µs (max). It is an application-dependent setting and must be accounted for by you via the acquisition and hold time for each channel. A recommended default value is inserted by ASB when configuring a new Voltage Monitor but it may be reduced with the possible reduction in sampling accuracy. See the Fusion datasheet for information.

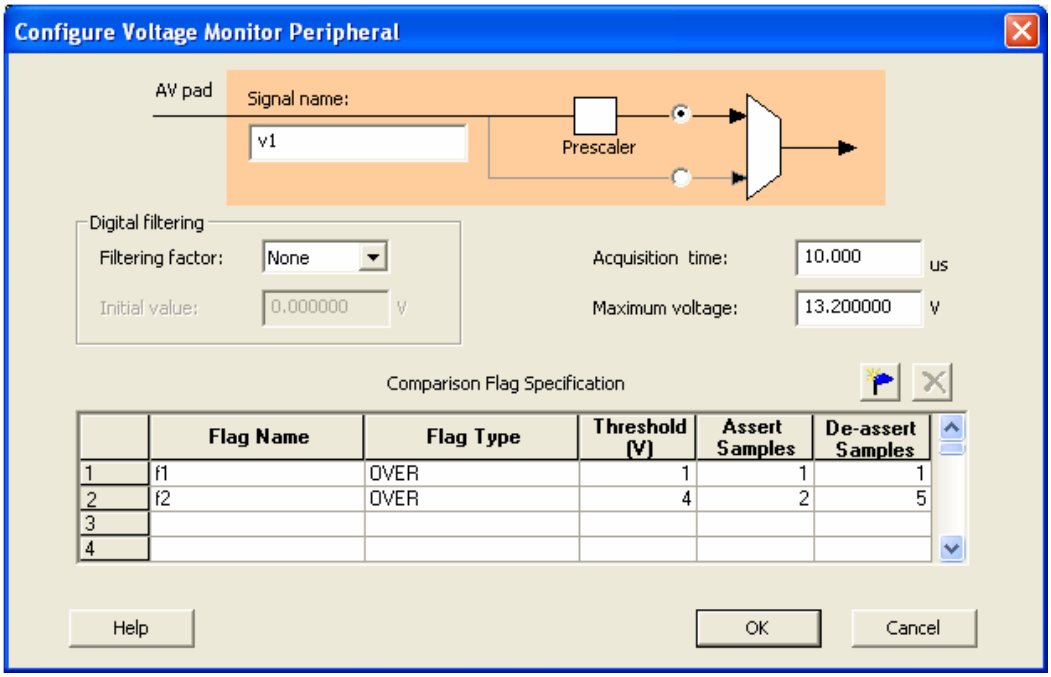


Figure 34 · Configure Voltage Monitor Dialog Box

Maximum Voltage

The maximum anticipated voltage measured by this Voltage Monitor peripheral pad. The range is -10.5V to +16V (the voltage range is NOT bipolar). The ADC is capable of measuring a voltage range of 0 - Vref. For the Internal voltage reference, this value is 2.56V. ASB automatically configures the prescaler in the AB Analog Block for this peripheral to maximize the available voltage range.

Note: Setting the Maximum Voltage to a negative value means that the voltage values driving this signal must be negative. The flag threshold evaluation for negative voltages is on a mathematical scale.

For example, if an OVER -3V flag was configured, the flag would assert whenever the input voltage is -1, -2, up to the threshold value. The flag would deassert when the value is -4, -5, etc.

Configuring Current, Differential Voltage, Temperature, and Voltage peripherals

The Current, Differential Voltage, Temperature, and Voltage peripherals are all configured the same way. Also, the effects of averaging are the same for all peripherals in the Analog System Builder.

Minor variations, such as Maximum Voltage in the Voltage monitor, are explained in the help topic for that peripheral.

Signal name is the name of the signal as you want it to appear in the main Analog System Builder dialog box.

Digital filtering

Once the ADC finishes converting the Analog Signal to a digital value, it filters (averages) the resulting digital output. Digital filtering is a single-pole low-pass filter built in soft gates; you can use it to improve the signal-to-noise ratio. If the ADC input data is very erratic, the filtering will smooth out the input and reduce the noise.

The filtered value is calculated using the following equation:

$Filtering_result_n = filtering_result_{n-1} + (ADC_Result_n / filtering_factor) – (filtering_result_{n-1} / filtering_factor)$

If the Digital filtering factor is set to 1 it is ignored.

In some cases, where the inputs are very low frequency, and the electrical environment is not very noisy, it may be possible to proceed without any special filtering of input analog signals. However, in most

applications it is desirable to at least implement a simple post-conversion digital filter inside the FPGA by oversampling and averaging several results to reduce the effects of random noise in the conversion signal path and improve overall accuracy. This simple averaging is automatically handled in the software by setting the Digital Filtering factor in the Analog System Builder (ASB) to specify how many samples are averaged (When the factor = F, F samples are averaged together.)

For situations where greater accuracy is required, an external analog filter may be needed to eliminate non-random and out-of-band noise sources. If an analog filter is not used to restrict the input signal content to the band-of-interest, any out-of-band signals or noise will be aliased into the conversion result as random in-band noise.

Some applications (for example, those that require frequency detection) may need both external analog filtering to limit out-of-band effects, and more sophisticated digital processing such as a multi-tap Finite Impulse Response (FIR) filter. A wide variety of digital filtering methods are available through the FPGA gates available in a Fusion Device.

Initial Filtering Value - The initial filtering value enables you to specify the starting value for the averaging function (Filtering Result[0]). This enables you to 'seed' your filtering function so that there are no erroneous values produced during the beginning of operation.

If you do not use an initial filtering value, the filtering function always starts with FilteringResult[0] = 0, thereby skewing the results towards 0 during the first range of samples.

Range - The Initial value range for the digital filter is identical to the threshold range for the peripheral.

The system instantiates the logic required to perform averaging as soon as there is at least one channel in the system that requires averaging. There is no extra logic penalty for averaging the other channels of the system. See the figures below for a graphical representation of the effect of digital filtering on a signal.

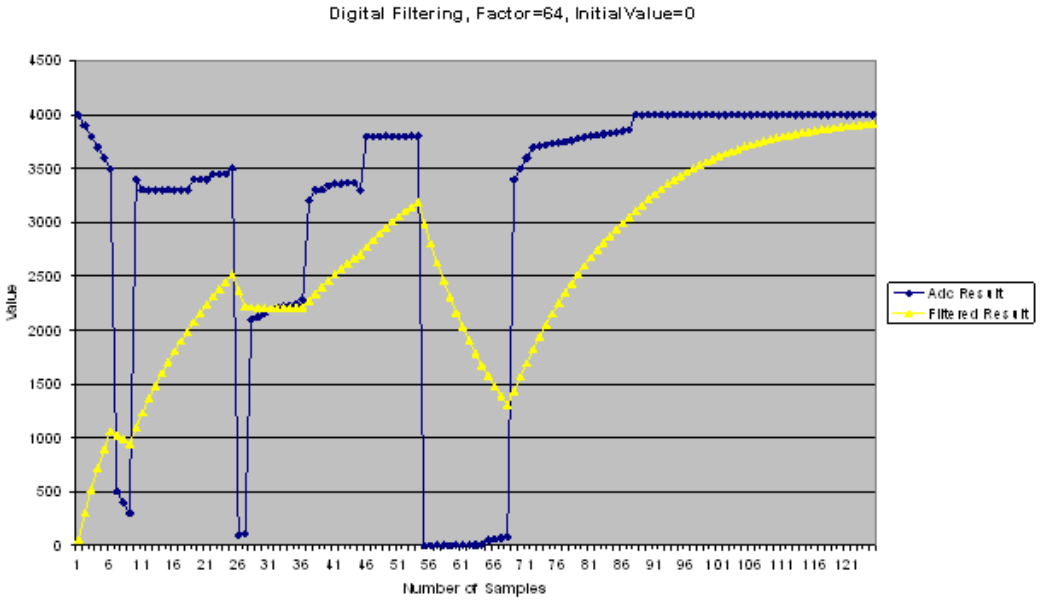


Figure 35 · Effect of Averaging on Voltage, Initial Digital Filtering Value = 0

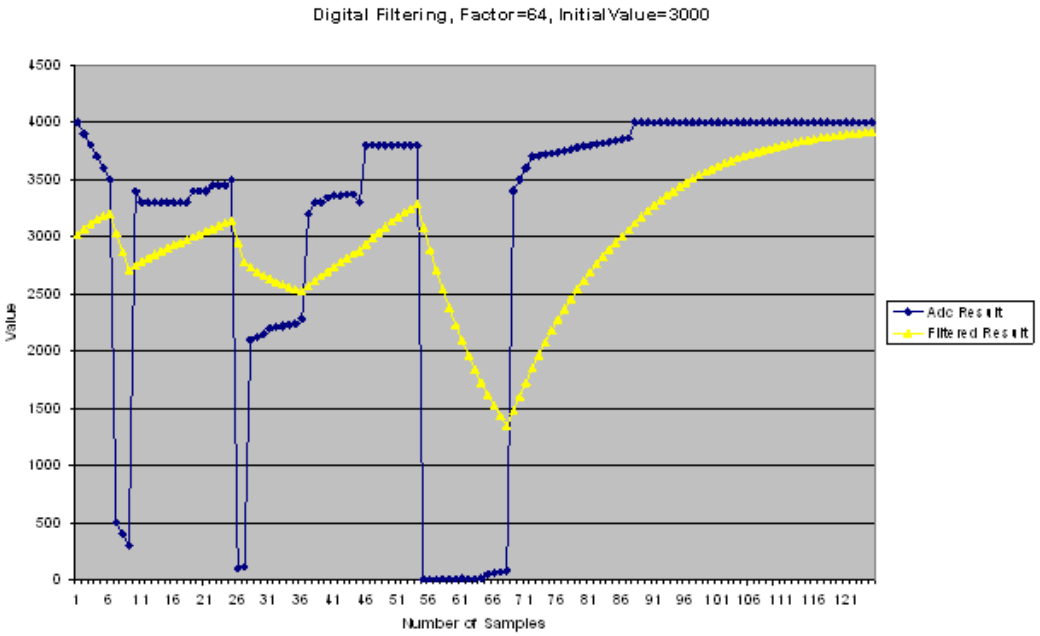


Figure 36 · Effect of Averaging on Voltage, Initial Digital Filtering Value = 3000

Acquisition Time

The required settling/sampling time for this channel. It is the amount of time the Sample and Hold circuit in the ADC charges the capacitor with the input analog signal. The characteristics of your system and monitoring requirements will determine this value. Note also that this value has a direct correlation to the achievable sampling rate of the system.

The prescaler logic of the AB has a settling time of 10µs max. It is an application-dependent setting and must be accounted for by you via the acquisition and hold time for each channel. A recommended default value is inserted by ASB when configuring a new Voltage Monitor but it may be reduced, with a possible reduction in sampling accuracy. See the Fusion datasheet for information.

ASB evaluates the required acquisition times for all peripherals and the system frequency to compute the maximum possible ADC clock frequency. Only certain divider factors exist to create the ADC Clock; because of this and peripheral acquisition times, certain system frequencies result in a faster ADC Clock.

Signal Polarity - Current and Differential Voltage Peripherals ONLY

Use this option to change polarity. The associated voltage monitor must be configured with the same polarity.

During simulation, a negatively configured peripheral must be driven by negative voltage values.

Comparison Flag Specification

Once the software calculates the average, you can further process the result by comparing the result to a given threshold and deciding under what conditions to assert the comparison flags.

Select a flag and click the **Delete** button to delete it.

Click the **Add Flag** button to add more flags to the system.

Flag Name - This is the name of the flag. This will appear as the output port name in the final output. It will be prefixed with the signal name with which it is associated, to group the input and outputs together.

Flag Type - You can choose to assert the flag when the signal is either under a given threshold or over a given threshold.

Threshold - Threshold value. The figure below shows the effect of the threshold on a given signal.

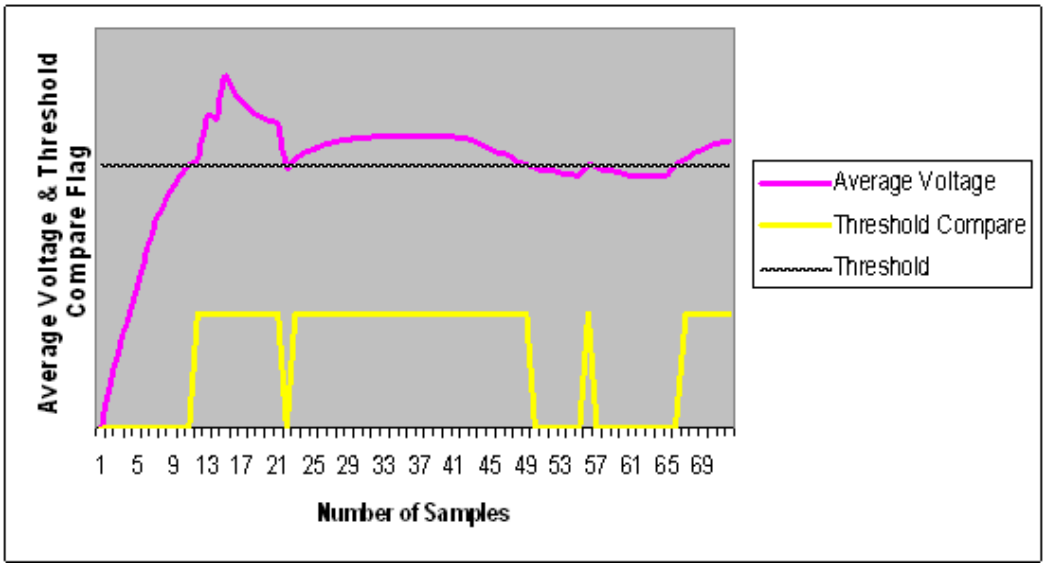


Figure 37 · Threshold Comparison

Assert Samples - The number of consecutive samples on this channel that reach or exceed the threshold for the flag to assert. This can be a glitch removal feature. If it is set to 1, the final flag is identical to the comparison result.

For example, if your Assert Sample value is 5 and the threshold is set at 3.0V, the channel must reach or exceed 3.0V five times in a row on this channel for the flag to assert. If your voltage values are less than 3.0V, the flag will not assert. The figure below shows the effect of glitch removal on a given signal.

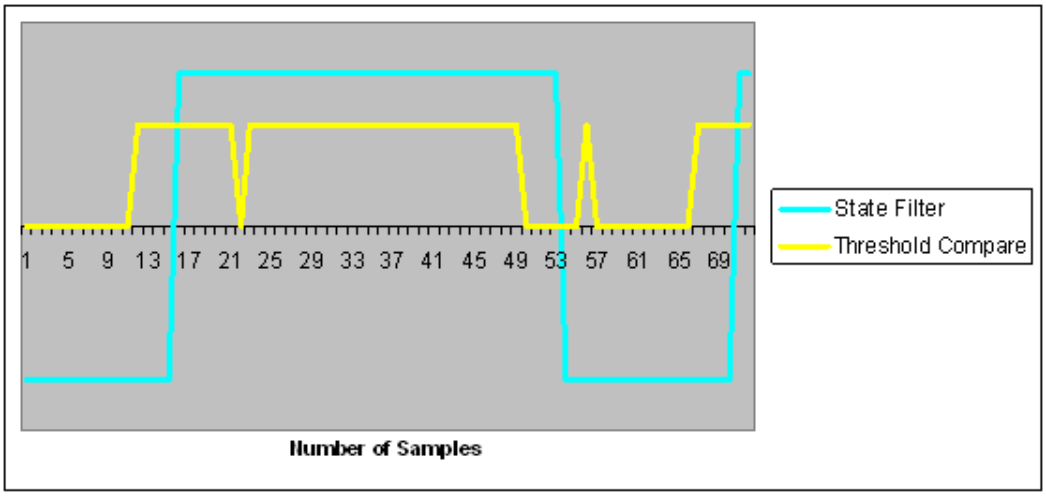


Figure 38 · Glitch Removal

De-assert Samples - The number of consecutive samples of this channel that are not above the threshold required for the flag to de-assert once it has been asserted. This is a glitch removal feature. If this value is set to 1, the final flag is identical to the comparison result.

For example, if your de-assert Sample value is 10 and the threshold is set at 3.3V, you must have 10 consecutive samples that go below 3.3V in order for the flag to de-assert

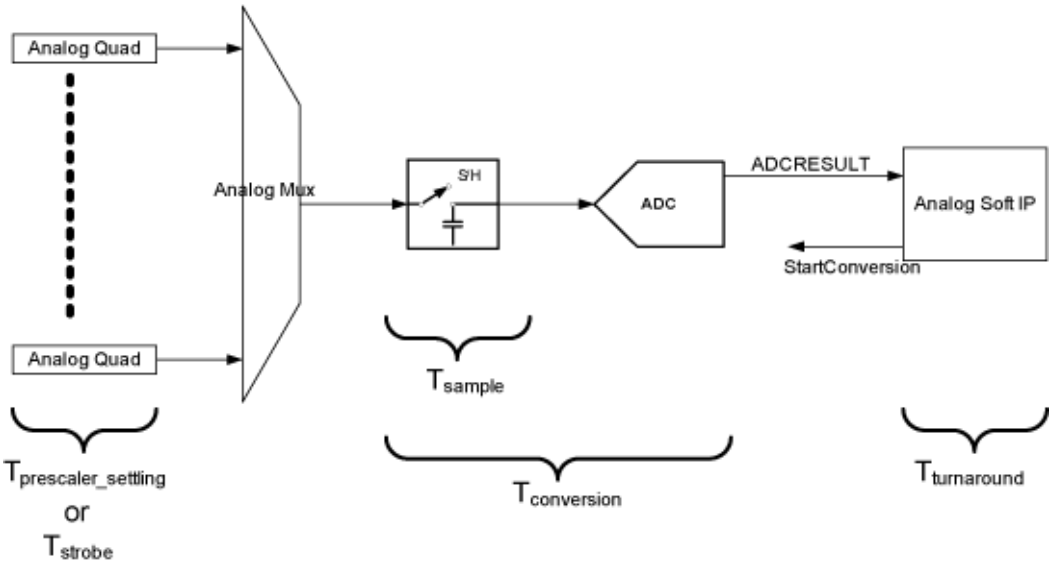
Sampling Rate in Analog System Builder

For ADC's that perform one sample per conversion, the throughput rate is also referred to as the sampling rate – this is the case for Fusion. Sampling Rate or frequency is the rate at which the ADC acquires or samples the analog input and converts it to digital data. It is specified as samples per second (S/s) or Hertz (Hz).

The sampling rate is typically the inverse of the ADC Conversion Time. For example, an ADC that takes 10 microseconds to acquire and convert an analog signal to a digital value will be able to generate about 100,000 samples per second. In the case where only a single channel is being sampled, the channel's sampling rate is equal to the total system sampling rate.

However, in the case where the sampling sequence contains multiple channels and where a channel may be sampled multiple times in relation to another channel, the sampling rate computation becomes more involved.

The diagram below shows a silicon view of the analog signal path.



Tprescaler_settling: Applicable only in Voltage Monitor peripheral when prescaler circuit is used

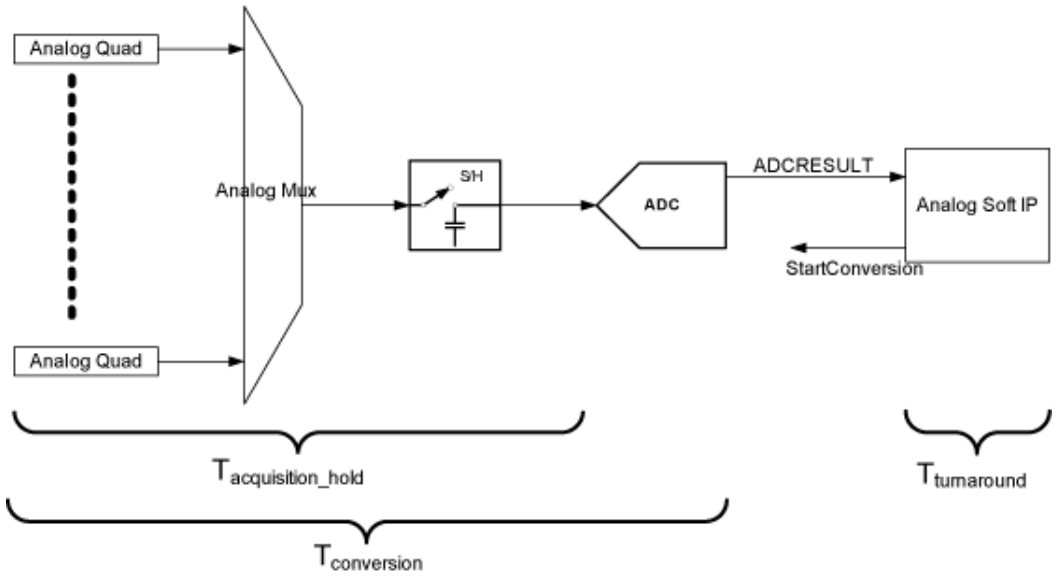
Tstrobe: Applicable only in Current, Temperature, or Differential peripherals. See Sequencing in Fusion for more information.

Tsample: Time for sample and hold circuit to sample the analog input voltage into the input capacitor

Tconversion: Conversion time of ADC

Tturnaround: How fast a client of the ADC can process data and give another start conversion signal. In this case this falls onto the ASSC IP.

To simplify the user interface experience, ASB utilizes a simpler view of the signal path, shown below:



Notice that the prescaler, strobe, and sample time have been condensed into the Tacquisition_hold time. This is the value that is specified by you during configuration of the peripheral. Legal ranges are enforced by ASB based upon requirements of the silicon.

The following sections describe the sampling rate computation for ASB. ASB enables you to specify the minimum sampling time and minimum sampling rate. ASB then reports the total system sampling rate and the actual sampling rate of each channel.

Conversion Time Calculation

The conversion time is a sum of:

$T_{conv} = T_{sync_read} + t_{sample} + -t_{distrib} + t_{cal} + t_{sync_write}$

$T_{sync_read} = T_{sync_write}$ = synchronization time = sys_clk

$T_{acquisition_hold} = T_{sample} + (T_{prescaler_settling}$ or $T_{strobe}) = (2 + stc) * adc_clk$

The stc is a factor of the user's acquisition and hold time

$T_{distrib}$ = charge distribution time = adc_resolution * adc_clock

T_{cal} = calibration time = 2 * adc_clk

ADC Clock Calculation

The ADC Clock period is calculated by:

(4 * (1 + clock divider setting)) / System Clock Period

The ADC Clock period has a maximum possible frequency of 10Mhz.

ASB automatically computes values for the sample time control (STC), clock divider setting (TVC), and ADC Clock Period based on your sample time requirement. Only certain divider factors exist to create the ADC Clock; because of the divider factors and peripherals' acquisition times, certain system frequencies result in a faster ADC Clock.

The goal of ASB is to meet the specified acquisition and hold time requirements with the highest possible ADC Clock frequency, which implies a low TVC value and high STC value.

See Designing with Analog System Builder for a discussion of the clocks involved in an Analog System design.

Sampling Rate Calculation for Fusion

The inverse of the conversion time is the sampling rate for that channel. However, the sampling rate reported by ASB includes the time that the ASSC can process the data and assert another start conversion

signal. This time is referred to as ‘turnaround time’. With no wait states, the ASSC takes 10 system clock cycles for turnaround time.

Therefore, in the case of Fusion, the sampling rate of a single channel is:

1 / [(ASSC turnaround time) + (channel conversion time)]

ASSC Turnaround Time - The number of flags for a channel may increase the turnaround time of the ASSC. Increasing the number of flags will potentially reduce your max sampling rate.

General Formula for Sampling Rate

The general formula for system and per-channel sampling rate is as follows:

Total Sampling Rate = Total # of Samples / Total Conversion Time of all Samples

Channel Sampling Rate =
((Total Number of Samples for Channel) /
(Total Number of all Samples)) *
(Total Sampling Rate)

The method for arriving at this formula is explained in the examples below.

Example: Equal Weight and Equal Conversion Time

All Channels have a conversion time of 2µs as shown in the figure below.

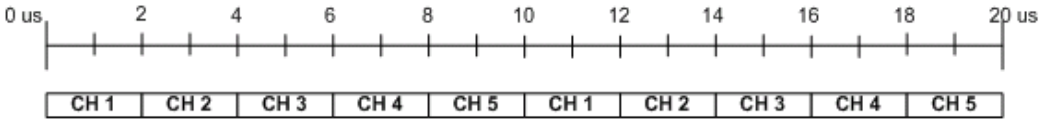


Figure 39 · Equal Weight and Equal Conversion Time

In this case we have 10 samples which take a total of 20µs.
Thus, our total system sampling rate is: 10 / 20µs = 500 kS / s

Channel1 Sampling Rate: 2µs / 10µs = .20 * 500 kS /s = 100 kS /s

Channel2 Sampling Rate: 2µs / 10µs = .20 * 500 kS /s = 100 kS /s

Channel3 Sampling Rate: 2µs / 10µs = .20 * 500 kS /s = 100 kS /s

Channel4 Sampling Rate: 2µs / 10µs = .20 * 500 kS /s = 100 kS /s

Channel5 Sampling Rate: 2µs / 10µs = .20 * 500 kS /s = 100 kS /s

Example: Unequal Weight and Equal Conversion Time

In this example, the channels are not equally weighted in the sampling sequence,as shown in the figure below.

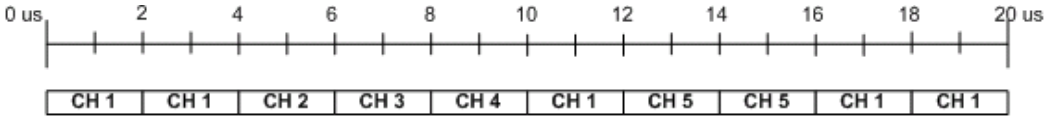


Figure 40 · Unequal Weight and Equal Conversion Time

In this case we have 10 samples that take a total of 20µs, giving us a total system sampling rate of 500 kS/s (as above). However, the individual channel sampling rates are different.

Channel1 Sampling Rate: 5µs / 10µs = .5 * 500 kS /s = 250 kS /s

Channel2 Sampling Rate: 1µs / 10µs = .1 * 500 kS /s = 50 kS /s

Channel3 Sampling Rate: 5µs / 10µs = .1 * 500 kS /s = 50 kS /s

Channel4 Sampling Rate: 5µs / 10µs = .1 * 500 kS /s = 50 kS /s

Channel5 Sampling Rate: 2µs / 10µs = .20 * 500 kS /s = 100 kS /s

Example: Unequal Weight and Unequal Conversion Time

In this example, channels have different conversion times and are not equally weighted in the sampling sequence, as shown in the figure below.

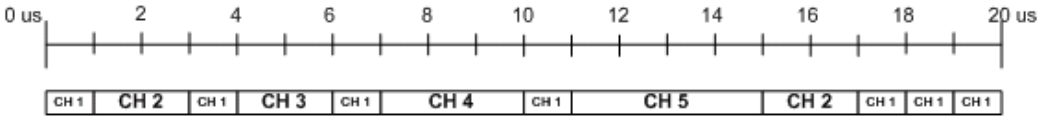


Figure 41 · Unequal Weight and Unequal Conversion Time

In this case we have 12 samples in 20µs, giving us a total system sampling rate of 600 kS/s.

Channel1 Sampling Rate: 7 / 12 = .583 * 600 kS /s = 349 kS /s

Channel2 Sampling Rate: 2 / 12 = .166 * 600 kS /s = 99.6 kS /s

Channel3 Sampling Rate: 1 / 12 = .083 * 600 kS /s = 49.8 kS /s

Channel4 Sampling Rate: 1 / 12 = .083 * 600 kS /s = 49.8 kS /s

Channel5 Sampling Rate: 1 / 12 = .083 * 600 kS /s = 49.8 kS /s

Wait States and the Analog System Controller

The Analog System Controller automatically inserts wait states to prevent collisions during processing. The wait states are inserted to ensure this condition:

Current conversion time + ASSC processing time > = previous sample's (SMEV + SMTR) processing time

Thus a wait state is calculated as:

(Current conversion time + ASSC processing time) – previous sample's (SMEV + SMTR) processing time

This extra time is inserted into the sample rate calculation, thereby decreasing the total sample rate; when wait states are inserted, the calculated sample rate is only an approximation. For example:

Single channel, continuous loop:

System clock = 40 MHz

ADC clock = 10 MHz

Acquisition time = 0.2µs

Resolution = 8-bit

No Flags

Tconv = tsync_read + tsample + tdistrib + tcal + tsync_write

(25 ns) + (2*100 ns) + (8*100 ns) + (2*100 ns) + (2*25 ns) = 1.25µs

Add turnaround time:

1.25µs + 0.25µs = 1.5µs

Sampling Rate = 1 / 1.5µs = 666 ksps

Sequencing in Fusion

The ADC has a strobe signal per temperature or current channel that must be asserted to initiate the conversion process. This strobe signal has strict timing pulse-width requirements; the strobe signal must stay on and then off for certain periods of time.

The requirement is that a strobe signal must stay low for a minimum of 5µs after a high-to-low transition.

Also, there is a high time requirement:

- 5µs high for current strobe
- 10µs high for temperature strobe

For temperature channels, the sample sequencer automatically inserts a 5µs delay before the ADC is started. During this time the strobe signal for that particular channel is asserted. This is to ensure that the temperature monitor block inside the ADC has settled to the correct value. After this 5µs has elapsed, the ADC sampling occurs. The length of this sampling period is the acquisition time specified in the peripheral configuration dialog.

Note: If the Analog System is configured in the ADC only mode, these strobe requirements need to be handled manually.

After the sampling period is completed, then ADC conversion begins.

Because of the automatic insertion of the 5µs delay, the minimum acquisition time for temperature is only required to be 5µs. Using the minimum acquisition time + the 5µs delay satisfies the minimum strobe requirement for temperature monitoring, as shown in the figure below.

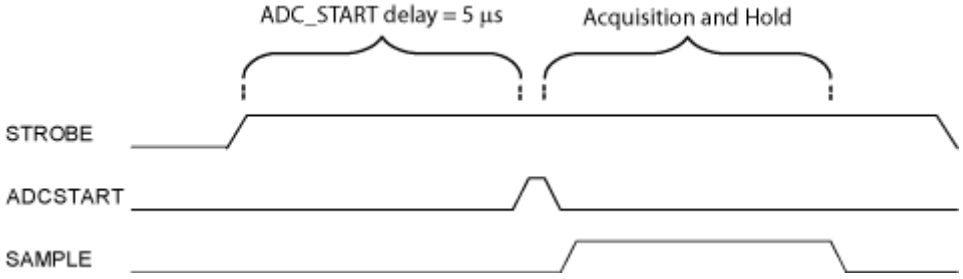


Figure 42 · ADC_START delay and Acquisition and HOLD Time

The ASSC generates these strobe signals; it has no concept of elapsed time, past samples, or future samples. A strobe pulse is activated for the entire ASSC processing time for a particular channel, and deactivated as soon as it moves to another channel. Thus, this requirement must be accounted for by you and/or ASB during the sequencing.

You must satisfy the following conditions to meet the requirement:

- A channel that requires a strobe cannot be sampled again until 5µs has elapsed
- A channel that requires a strobe must have a conversion time + ASSC processing time >= 5µs

The second condition is automatically handled because you enter a MINIMUM required sampling time. ASB can then ensure a minimum 5µs conversion time for temperature and current channels by adjusting the STC and ADC clock periods.

Flash Memory System Builder

Welcome to the Flash Memory System Builder

The Flash Memory System Builder enables you to configure the entire flash memory block. You can:

- Add analog system initialization data
- Add initialization clients
- Add Fusion RAM clients that require initialization
- Partition non-volatile memory for data access
- Specify the sizes of the partitions
- Specify the memory contents for partitions

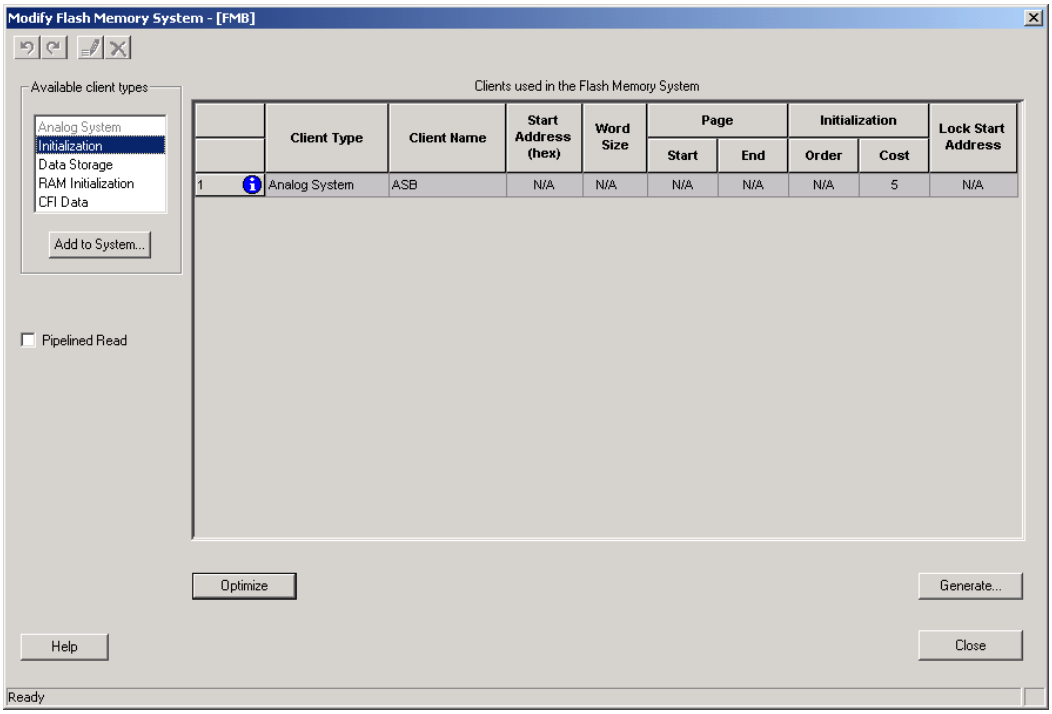


Figure 43 · Flash Memory Block Builder

The Flash Memory Block Builder supports the following clients:

- Analog System
- Initialization
- Data Storage
- Ram Initialization
- CFI Data

Each client spans a minimum of one page (128 bytes) and can go up to 2048 pages, based on the number of free pages available. The analog system itself does not take any of the regular pages; it is stored entirely in the reserved pages.

The System Grid in the GUI provides an overview of the system.

Client Type - The type of the client that is added to the system.

Client Name - The name of the client. It must be unique across the system.

Start Address - The decimal or HEX address at which the client starts. It must be on a page boundary. Clients cannot have overlapping start addresses.

Word Size - Word size of the client, in bits

Page Start - This is the page on which the start address begins or ends. You can modify either the start address or the start page.

Page End- This is computed based on the word size and the number of words in the client.

Initialization Order - The order in which the clients' required initialization is written with the data from the Flash Memory System. If an analog system is present, it will be initialized first. If any clients must save their data to the Flash Memory System, the save order is the same as the Initialization Order.

Initialization Cost - The number of Initialization Enables used by a given client. An Analog System client has an Init Cost of 4. A RAM client has an Init Cost equal to the number of RAM blocks. A regular initialization client has an Init Cost of 1. The data storage client has no Init Cost.

The total init cost must be less than or equal to 64.

Lock Start Address - You can specify this option if you do not want the system to change your start address for any reason.

Optimize - Click this button to resolve the conflicts on overlapping base addresses for clients. This operation will not modify the base addresses for any clients that have their base addresses locked. It also de-fragments the memory.

Analog System Client

You can load the configuration file generated by the Analog System Builder into the Flash Memory System Builder. Once loaded, all the analog system components can be initialized by the Flash Memory System at start up.

The Add Analog System Client opens with a blank field for the Configuration file. The Modify Analog System Client opens with the field filled in. Click the dropdown menu and select your Analog System core from the list.

Note: Initialization of the Analog System must occur at less than 10 Mhz; the Analog Configuration MUX (ACM) inside the Analog Block runs at a maximum of 10 Mhz. The ACM contains the configuration data for the Analog channels. If you initialize at more than 10 MHz, you will overload the MUX.

Note: If necessary, a PLL and NGMUX can be used in conjunction to support driving the Analog System at a slow initialization frequency and a fast operating frequency.

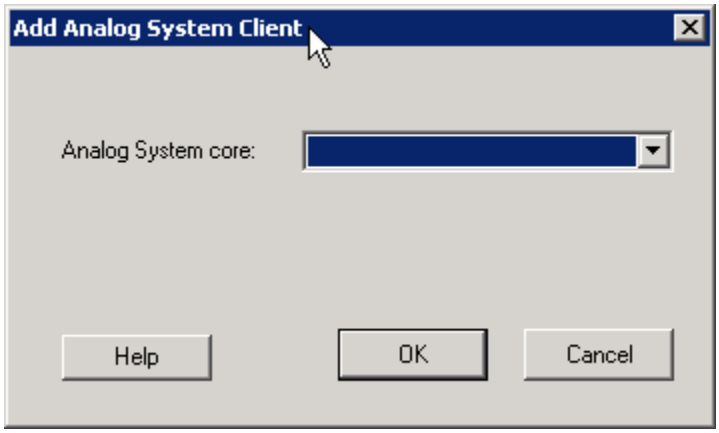


Figure 44 · Add Analog System Client Dialog Box

Data Storage Client

The Data Storage Client enables you to create a partition in the Flash Memory System and specify the memory content for that partition. You can access the partition directly via the Flash Memory System

busses. The Add Data Storage Client dialog box opens with blank fields; the Modify Data Storage Client displays any values you have already set.

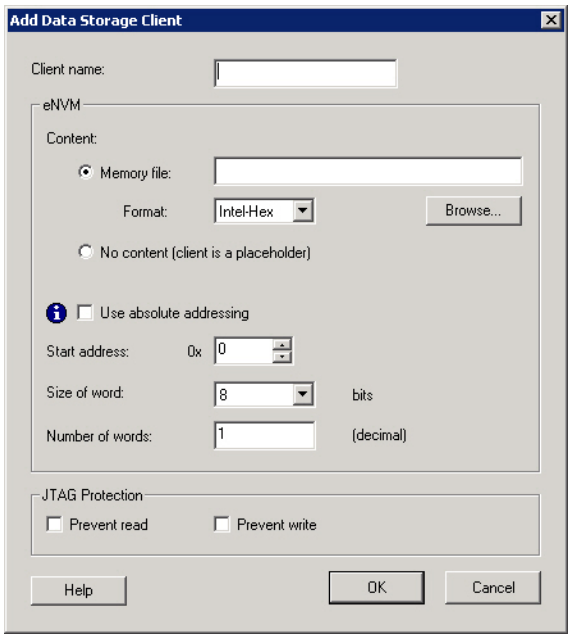


Figure 45 · Add Data Storage Client Dialog Box

Client name- Name of the client; the value you enter is attached before the select and enable names to group all the control signals for that client.

eNVM Content Description

Content - Specify the memory content that you want to be loaded into Flash Memory. You may choose one of the two following options:

- **Memory File -** You need to select a file on disk that matches one of the following memory file formats – Intel-Hex, Motorola-S, Microsemi-S or Microsemi-Binary.
- **No content -** The client is a place holder. You will be available to load a memory file using FlashPro/FlashPoint.

Use absolute addressing - Lets the memory content file dictate where the client is placed in the flash memory block. The addressing in the memory content file for the client becomes absolute to the whole flash memory block. Once you choose the absolute addressing option, the software extracts the smallest address from the memory content file and uses that address as the start address for the client.

Start Address - The memory location of the content loaded in the eNVM.

Size of Word - Word size, in bits, of the initialized client; can be either 8, 16 or 32.

Number of words - Number of words of the client.

JTAG Protection

Click the checkbox to protect your JTAG from read and write.

CFI Data

Used to store the query data for CoreCFI. The data is stored in a reserved page location. This client does not take up any of the 2048 pages in the Flash Memory.

Use the dialog box to specify the location of the memory file for CFI query data (as shown in the figure below). This memory file is provided to you when you obtain the CoreCFI IP. The memory file format is Microsemi Binary. You are not expected to modify this file.

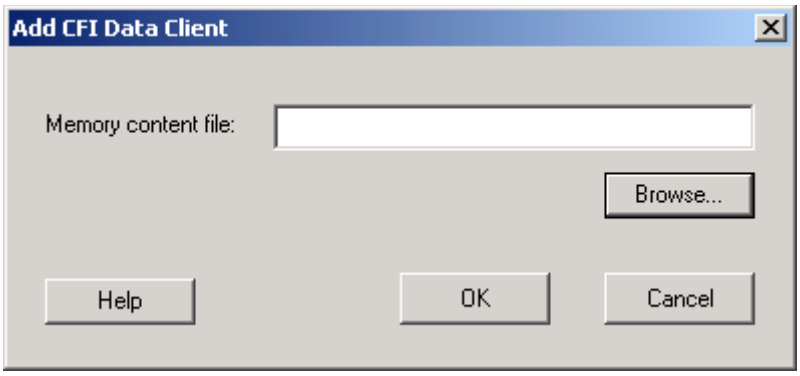


Figure 46 · CFI Data Dialog Box

Output is top_level.vhd and top_level.efc if you use only the CFI Data client. The EFC file is used to program the Flash Memory System Builder.

For more documentation on connecting the top-level to CoreCFI, see the CoreCFI datasheet at http://www.actel.com/ipdocs/CoreCFI_HB.pdf.

Initialization Client

The Flash Memory System Builder initializes all the clients with the data stored in the Flash Memory when the system is powered-up. You must assert the INIT_POWER_UP signal to High to power up the system. You can use the Initialization Client to set initial values of the RAM/FIFO (such as a table or list of filter coefficients, MAC addresses, etc.) and ROM emulation.

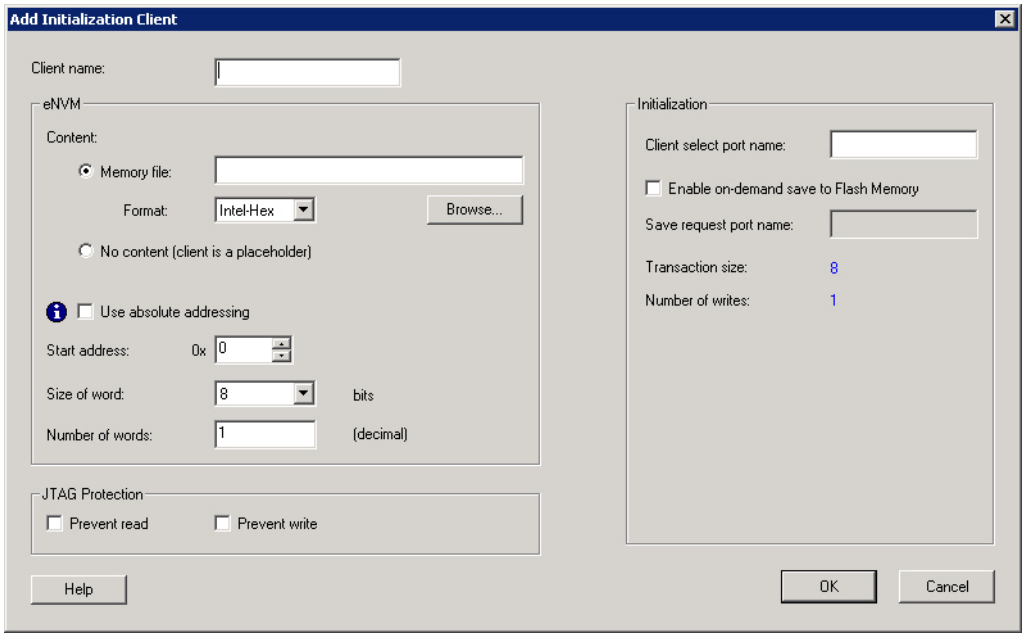


Figure 47 · Add Initialization Client Dialog Box

Client Name - Name of the client; the value you enter is attached before the select and enable names to group all the control signals for that client.

eNVM

Content - Specify the memory content that you want to be loaded into Flash Memory. You may choose one of the two following options:

- **Memory File -** You need to select a file on disk that matches one of the following memory file formats – Intel-Hex, Motorola-S, Microsemi-S or Microsemi-Binary.
- **No content -** The client is a place holder. You will be available to load a memory file using FlashPro/FlashPoint.

Use absolute addressing - Lets the memory content file dictate where the client is placed in the flash memory block. The addressing in the memory content file for the client becomes absolute to the whole flash memory block. Once you choose the absolute addressing option, the software extracts the smallest address from the memory content file and uses that address as the start address for the client.

Start Address - The memory location of the content loaded in the eNVM.

Size of Word - Word size, in bits, of the initialized client; can be either 8, 16 or 32.

Number of words - Number of words of the client.

Initialization

Target address - The address of your storage element in terms of the Cortex-M3 system memory map. Certain regions of the system memory map are not allowed to be specified for this client because they contain reserved system blocks. The tool will inform you of the legal regions for your client.

Transaction size - The size of the APB transfers when the data is copied from the eNVM memory region to the target destination by the Microsemi SoC system boot code.

Number of writes - The number of APB transfers when the data is copied from the eNVM memory region to the target destination by the Microsemi SoC system boot code. This field is automatically computed by the tool based on the eNVM content information (size and number of words) and the destination transaction size.

JTAG Protection

Click the checkbox to protect your JTAG from read and write.

RAM Initialization client

You must create a RAM with Initialization core before it can be imported here.

When you generate a RAM with Initialization for Fusion, the data for the RAM can be loaded into the Flash Memory System Builder such that at power-up the data is loaded into the RAM from the flash memory.

The difference between this and the regular Initialization Client is that the cascading of multiple RAM blocks is handled automatically. The RAM Initialization client takes as many initialization clients as there are RAM blocks in the cascaded RAM.

Start Address (hexadecimal only) - Sets the starting address for the RAM Initialization Client.

RAM core - RAM with Initialization core you wish to import.

JTAG Protection

Click the checkbox to protect your JTAG from read and write.

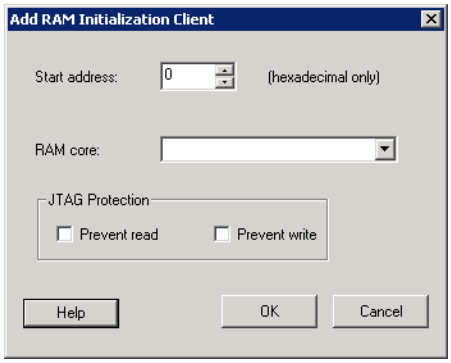


Figure 48 · RAM Initialization Client Dialog Box

Flash Memory Block with Data Storage Client

In this scenario, the Flash memory block (FMB) is configured with a data storage client, which exposes the USER_* interface of the Flash Memory system. This enables complete access to the Flash Memory System interface.

In this case, the FMB serves two purposes, Initialization and Save functionality as well as allowing access to the Flash Memory System. The arbitration between these two functions is handled inside the INIT IP, the arbitration is a simple priority scheme; if any initialization or save activity is in progress then the ownership is given to the INIT IP and removed from the USER.

This implies that the user must be certain that an initialization or save activity is not triggered during their Flash Memory System access; if so, data may be corrupted. Furthermore, a user must also monitor the INITDONE signal to ensure that the INITIP is not busy before attempting to access the Flash Memory System.

In a typical situation, the INITCLK and USRCLK are connected together and exposed on the FMB top level as a single clock port (INIT_CLK). In this case, the USRCLK is exposed and the user can connect any frequency less than 100Mhz (max frequency of the Flash Memory System) to this port. It has no relationship with any of the other clocks in the system.

To avoid any clock dependency issues between the 2 frequencies any CLIENT_UPDATE (for save) requests given to the FMB should be held until the INIT_DONE is deasserted, indicating that it has accepted the transaction.

Flash Memory System Output Files

When you click **Generate,** the software generates the following files; they are saved in your components directory.

HDL Source Files

<user_name>.vhd/.v – Top level design that combines all the blocks together

<user_name>_init_wrapper.vhd/.v – Initialization and configuration instantiation wrapper for this design

<common>/<Vhdl>/<Verilog>/initcfg.vhd/v - Soft IP

<common>/<Vhdl>/<Verilog>/initcfg_xa.vhd/v – Soft IP

<common>/<Vhdl>/<Verilog>/initcfg_xb.vhd/v – Soft IP

<common>/<Vhdl>/<Verilog>/initcfg_xc.vhd/v – Soft IP

<common>/<Vhdl>/<Verilog>/initcfg_xd.vhd/v -Soft IP

<common>/<Vhdl>/<Verilog>/initcfg_xe.vhd/v – Soft IP

<common>/<Vhdl>/<Verilog>/initcfg_xf.vhd/v – Soft IP

<common>/<Vhdl>/<Verilog>/numbits.vhd/v – package file

Memory Files

<user_name>.mem - Non-volatile memory file

<user_name>.efc - Contains all the Embedded Flash Memory partitions defined by each client of the Flash Memory System. FlashPoint uses the EFC file to create the STAPL file that programs the embedded Flash memory.

The Flash Memory System Builder creates only one EFC file per Embedded Flash Memory System. A Fusion device may contain from 1-4 embedded Flash memories. You must use the FlashPoint user interface to associate each embedded Flash memory in the design with an EFC file.

Configuration Files

<user_name>.cfg – Captures information about the settings that were specified for the system.

<user_name>.gen – Core project file; stores information about your Flash Memory System so that you can save your settings.

<user_name>.cxf – Core configuration file that contains information required by Libero SoC for file management.

Log Files

<user_name>.log

Memory File Formats in Flash Memory System Builder

The following memory file formats are available as input files into the Flash Memory System Builder:

- Microsemi BINARY
- INTEL-HEX
- MOTOROLA S-record
- MICROSEMI-HEX

An example of how to interpret the memory content is listed below.

Microsemi BINARY

The simplest memory format. Each memfile contains as many rows as there are words. Each row is one word, where the number of binary digits equals the word size in bits. This format has a very strict syntax. The word size and number of rows must match exactly. The file extension is MEM; for example, file1.mem.

Example: Depth 6, Width is 8

```
01010011
11111111
01010101
11100010
10101010
11110000
```

INTEL-HEX

Industry standard file. Extensions are HEX and IHX. For example, file2.hex or file3.ihx.

A standard format created by Intel. Memory contents are stored in ASCII files using hexadecimal characters. Each file contains a series of records (lines of text) delimited by new line, '\n', characters and each record starts with a ':' character. For more information regarding this format, refer to the Intel-Hex Record Format Specification document available on the web (search Intel Hexadecimal Object File for several examples).

The Intel Hex Record is composed of five fields and arranged as follows:

```
:llaaaatt[dd...]cc
```

Where:

- : is the start code of every Intel Hex record
- ll is the byte count of the data field
- aaaa is the 16-bit address of the beginning of the memory position for the data. Address is big endian.
- tt is record type, defines the data field:

- 00 data record
- 01 end of file record
- 02 extended segment address record
- 03 start segment address record (ignored by Microsemi SoC tools)
- 04 extended linear address record
- 05 start linear address record (ignored by Microsemi SoC tools)

- [dd...] is a sequence of n bytes of the data; n is equivalent to what was specified in the ll field
- cc is a checksum of count, address, and data

Example Intel Hex Record:

```
:10000000112233445566778899FFFA
```

Where 11 is the LSB and FF is the MSB.

MOTOROLA S-record

Industry standard file. File extension is S, such as file4.s

This format uses ASCII files, hex characters, and records to specify memory content in much the same way that Intel-Hex does. Refer to the Motorola S-record description document for more information on this format (search Motorola S-record description for several examples). The RAM Content Manager uses only the S1 through S3 record types; the others are ignored.

The major difference between Intel-Hex and Motorola S-record is the record formats, and some extra error checking features that are incorporated into Motorola S.

In both formats, memory content is specified by providing a starting address and a data set. The upper bits of the data set are loaded into the starting address and leftovers overflow into the adjacent addresses until the entire data set has been used.

The Motorola S-record is composed of 6 fields and arranged as follows:

```
Stllaaaa[dd...]cc
```

Where:

- S is the start code of every Motorola S-record
- t is record type, defines the data field
- ll is the byte count of the data field
- aaaa is a 16-bit address of the beginning of the memory position for the data. Address is big endian.
- [dd...] is a sequence of n bytes of the data; n is equivalent to what was specified in the ll field
- cc is the checksum of count, address, and data

Example Motorola S-Record:

```
S10a00001122334455667788 99FFFA
```

Where 11 is the LSB and FF is the MSB.

Microsemi-HEX

A simple address/data pair format. All the addresses that have content are specified. Addresses with no content specified will be initialized to zeroes. The file extension is AHX, such as filex.ahx. The format is:

```
AA:D0
AA:D1
AA:D2
```

Where AA is the address location in hex, and D# is the data at that AA address.

The only valid addressing scheme for .ahx is one byte per line and the addresses are byte addresses. A memory file content specified as address followed by 16 or 32 bit data is not supported at this time.

The data size must match the word size. Example: Depth 6, Width is 8

```
00:FF
01:AB
02:CD
03:EF
04:12
05:BB
```

All other addresses will be zeroes.

Interpretation of the Memory Content

Absolute vs Relative Addressing

In Relative Addressing, the addresses in the memory content file did not determine where the client was placed in memory. You specify the location of the client by entering the start address. This becomes the 0 address from the memory content file perspective and the client is populated accordingly.

For example, if we place a client at 0x80 and the content of the memory file is as follows:

```
Address: 0x0000 data: 0102030405060708
Address: 0x0008 data: 090A0B0C0D0E0F10
```

Then the first set of bytes of this data is written to address 0x80 + 0000 in the flash memory block. The second set of bytes is written to address 0x80 + 0008 = 0x88, and so on.

Thus the addresses in the memory content file are relative to the client itself. Where the client is placed in memory is secondary.

For absolute addressing (available in the Initialization Client and the Data Storage Client), the memory content file dictates where the client is placed in the flash memory block. So the addressing in the memory content file for the client becomes absolute to the whole flash memory block. Once you enable absolute addressing option, the software extracts the smallest address from the memory content file and uses that address as the start address for the client.

Data Interpretation Example

The following examples illustrate how the data is interpreted for various word sizes:

For the given data: FF 11 EE 22 DD 33 CC 44 BB 55 (where 55 is the MSB and FF is the LSB)

For 32-bit word size:

```
  0x22EE11FF (address 0)
  0x44CC33DD (address 1)
0x000055BB (address 2)
```

For 16-bit word size:

```
0x11FF (address 0)
0x22EE (address 1)
0x33DD (address 2)
0x44CC (address 3)
0x55BB (address 4)
```

For 8-bit word size:

```
  0xFF (address 0)
0x11 (address 1)
 0xEE (address 2)
 0x22 (address 3)
0xDD (address 4)
0x33 (address 5)
0xCC (address 6)
0x44 (address 7)
0xBB (address 8)
0x55 (address 9)
```

For 9-bit word size:

```
0x11FF -> 0x01FF (address 0)
0x22EE -> 0x00EE (address 1)
0x33DD -> 0x01DD (address 2)
 0x44CC -> 0x00CC (address 3)
 0x55BB -> 01BB (address 4)
```

Notice that for 9-bit, that the upper 7-bits of the 2-bytes are ignored.