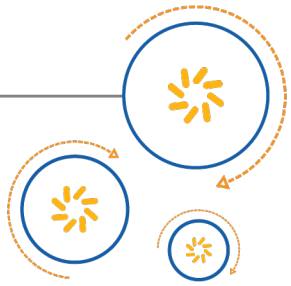
Qualcomm Technologies International, Ltd.



ADK Applications Configuration XML Definitions

Reference Guide

80-CT541-1 Rev. AD

October 17, 2017

Confidential and Proprietary – Qualcomm Technologies International, Ltd.

NO PUBLIC DISCLOSURE PERMITTED: Please report postings of this document on public servers or websites to DocCtrlAgent@qualcomm.com.

Restricted Distribution: Not to be distributed to anyone who is not an employee of either Qualcomm Technologies International, Ltd. or its affiliated companies without the express approval of Qualcomm Configuration Management.

Not to be used, copied, reproduced, or modified in whole or in part, nor its contents revealed in any manner to others without the express written permission of Qualcomm Technologies International, Ltd.

Qualcomm is a trademark of Qualcomm Incorporated, registered in the United States and other countries. Other product and brand names may be trademarks or registered trademarks of their respective owners.

This technical data may be subject to U.S. and international export, re-export, or transfer ("export") laws. Diversion contrary to U.S. and international law is strictly prohibited.

Qualcomm Technologies International, Ltd. (formerly known as Cambridge Silicon Radio Limited) is a company registered in England and Wales with a registered office at: Churchill House, Cambridge Business Park, Cowley Road, Cambridge, CB4 0WZ, United Kingdom.
Registered Number: 3665875 | VAT number: GB787433096

© 2017 Qualcomm Technologies International, Ltd. All rights reserved.

Revision history

Revision	Date	Description
1	March 2017	Initial release. Alternative document number CS-00346862-UG
2	12 April 2017	Minor edits
3	25 April 2017	Minor corrections
AD	October 2017	Added to the Content Management System. DRN updated to use Agile Number. No change to the technical content

Contents

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Tables

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

1 XML overview

This document describes the XML definition used by the Configuration Build Scripts as part of the ADK Application Configuration Architecture. It contains sections for both the Global Configuration Definition and the Module Definition XML.

ADK Applications Configuration Architecture Overview and *ADK Application Configuration System* describe the ADK Application Configuration Architecture.

1.1 Common Definition XML

Table 1-1 lists all Common Definition Section XML elements, used by both the Global and Module XML files.

Table 1-1 Common Definition XML overview

XML Element	Section	Purpose
<DefineGroup>	Define Group	Container element for the definition section.
<DefineList>	Define List	Define a named list containing enum or bitfield values.
<enum>	Enumerated Type definition	Define an enum key, value pair.
<bitfield>	Bitfield Mask Bit definition	Define a bitfield key, value pair.
<List>	Include List	Include a named list in another XML element.
<DefinePattern>	Pattern definition	Define a named pattern containing different configuration items, referenced from a pattern array.
<DefineStruct>	Structure Definition	Define a named structure of configuration items.
<ConfigItem>	Configuration Item	Define a configuration item.
<ConfigArray>	Array Definition	Define an array of basic configuration items.
<ConfigStruct>	Structure Instance Definition	Define an instance of a named structure.

1.2 Global XML

In addition to the elements listed in Table 1-1, the Global XML file also uses the elements listed in Table 1-2.

Table 1-2 Global XML overview

XML Element	Section	Purpose
<GlobalData>	Global Data	Container element for the Global XML data.
<ConfigGroup>	Configuration Group XML	Container for a set of related configuration items.

1.3 Module XML

In addition to the elements listed in Table 1-1, the Module XML file also uses the elements listed in Table 1-3.

Table 1-3 Module XML overview

XML Element	Section	Purpose
<ModuleData>	Module Data	Container element for the Module XML data.
<ConfigGroup>	Configuration Group	Container for a set of related configuration items.
<ConfigItem>	Configuration Item	Configuration item declaration including default value.
<ConfigArray>	Array Declaration	Declaration of an array of basic configuration items including default values.
<ArrayElementConfigItem>	Array Item	Define the default value for a specific element in an array.
<ConfigStruct>	Structure Declaration	Declaration of a named structure including default values.
<StructElementConfigItem>	Structure Item	Define the default value for a specific element in a structure.
<ConfigPatternArray>	Configuration Pattern Array	Declaration of an array of a named pattern including default values.
<PatternArrayRow>	Configuration Pattern Array Row	Define the default values for a single pattern in a pattern array.
<PatternArrayConfigItem>	Configuration Pattern Array Row item	Define the default value of a specific element in a pattern.
<BuildVariant>	Build Variant	Include build specific default values.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

2 Common Definition XML

The definition section of both the Global and Module XML specify the symbols, Patterns and Types referenced by the rest of the Configuration Set XML definition.

2.1 Define Group

The Define Group tag (<DefineGroup>), is used to distinguish the XML definition section.

Attributes

Not applicable for this element.

Child Elements

- 0 or more <DefineList> elements
- 0 or more `<DefinePattern>` elements
- 0 or more `<DefineStruct>` elements

Define Group: Example XML code

```
<DefineGroup>
    <DefineList
        Id=”User Event”
        ShortId="user_event">
        <!--N.b. content abridged in this example!-->
    </DefineList>
    <DefinePattern
        ShortId="LED"
        PatternName=”LED_pattern>
        <!--N.b. content abridged in this example!-->
    </DefinePattern>
</DefineGroup>
```

2.2 Define List

Use the Define List XML element (<DefineList>), to specify reusable Configuration Set list data once.

Rather than explicitly re-defining XML fragments every time they are required, the Define List is used to factorise these elements out of the Configuration Set definitions, allowing them to be referenced

symbolically, similar to the way a macro pre-processor manages symbol expansion, prior to compilation of source code.

Once defined using a <DefineList> element, List Definitions can be referenced from any of the following scopes:

- The Define Group
  - Patterns
- Configuration Groups
  - Configuration Items

The List Definition is analogous to the definition of an enumerated type in C, for example:

```
typedef enum {} enum_name
```

Use the `<List>` XML element to reference the list definition.

Define List: Attributes

Table 2-1 DefineList Attributes

Attribute Name	Expected Type	Requirement	Contains Subfields
Id	String	Mandatory	No
ShortId	Symbolic Name		

Define List: Id

The `Id` attribute specifies the text string that defines the display name for the list and must be specified, for example: `Id="User Event"`

Define List: ShortId

The `ShortId` attribute defines the text string, following the rules for a C identifier, used as the symbolic name in the Application for this list and must be specified, for example: `ShortId="user_event"`.

All `ShortId` entries match the name used for the data in the Application.

NOTE If the `ShortId` of a `DefineList` in a **Module Definition** is the same as the `ShortId` of a `DefineList` in the **Global Definition**, the elements of the Module `DefineList` will be appended to the Global `DefineList` resulting in a single `DefineList` for the Configuration Set.

Define List: Child Elements

The Define List must include at least one `<enum> or <bitfield>` element; element types cannot be mixed within a single List.

- 0 or more `<enum>` elements
- 0 or more `<bitfield>` elements

Define List: Example XML

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
<DefineList ShortId="Audio Instance">
    <enum
        key="Audio Instance 0"
        value="0"/>
    <enum
        key="Audio Instance 1"
        value="1"/>
    <enum
        key="Audio Instance 2"
        value="2"/>
</DefineList>
```

2.3 Enumerated Type definition

The <enum> XML element defines the key-value pairs that constitute enumerated types in the Configuration Tool.

Enum: Attributes

Table 2-2 <enum> Attributes

Attribute Name	Expected Type	Requirement	Contains Subfields
Key	String	Mandatory	No
Value	Integer		
Disable	String	Optional	Yes
Desc			No

Enum: key

The `key` attribute is used to specify the symbolic name for the value of the enumerated type and must be specified, for example: `key=”DISABLED”`.

Enum: value

The `value` attribute is used to specify the integer value associated with the displayed name and must be specified, for example: `value=”0x04”`.

Enum: disable

Use the `disable` attribute to specify items in a Pattern Array row using a comma de-limited list to be disabled if the `enum` Configuration Item is set to `value`.

The following code example defines a `DefineList` that should be used in a Pattern Array that also contains the following items:

- Speed Factor
- Speed Action
- Color

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

- Filter to Cancel
- LED to Use

In this example, if the `enum` Configuration Item for a given row is set to 1 then the items with `Id` set to `Speed Factor`, `Speed Action`, `Colour` and `LED to Use` will be disabled for the same row.

```
<DefineList
    ShortId="filter_type_options"
    Id="Filter Type Options"
    <enum
        key="DISABLED"
        value="0"
        disable="Speed Factor,Speed Action,Colour,Filter to Cancel,LED to
Use"/>
    <enum
        key="CANCEL"
        value="1"
        disable="Speed Factor,Speed Action,Colour,LED to Use"/>
    …
</DefineList>
```

Enum: desc

The optional `desc` attribute is used to provide information, but has no functional role.

```
<DefineList
    Id="Tone">
    <enum
        key="No Tone"
        value="0"
        desc="No Tone Configured"/>
    <enum
        key="Middle 0"
        value="1"
        desc="300ms G5"/>
    <enum
        key="ring_twilight"
        value="2"
        desc="Ring Tone"/>
</DefineList>
```

Enum: Child elements

The <enum> XML element does not support Child elements.

Enum: Example XML

```
<enum
    key="..."
    value="0x00"/>
<enum
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
    key="Power On"
    value="0x01"/>
<enum
    key="Power Off"
    value="0x02"/>
```

2.4 Bitfield Mask Bit definition

The <bitfield> XML element defines the key-value pairs to attribute labels to numbered masking bits.

Bitfield Mask Bit: Attributes

Table 2-3 Bitfield Attributes

Attribute Name	Expected Type	Requirement	Contains Subfields
Key	String	Mandatory	No
Value	Integer		

Bitfield Mask Bit: key

The `key` attribute is used to specify the symbolic name for the value of the enumerated type and must be specified, for example: `key=”DISABLED”`.

Bitfield Mask Bit: value

The `value` attribute specifies the bit number of the masking bit associated with the displayed name and must be specified, for example: `value=”5”`.

Bitfield Mask Bit: Child elements

The <bitfield> XML element does not support Child elements.

Bitfield Mask Bit: Example XML

```
<bitfield
    key="AAC"
    value="1"/>
<bitfield
    key="aptX"
    value="3"/>
<bitfield
    key="aptX Low Latency"
    value="4"/>
```

2.5 Include List

The <List> XML element references a previously defined key-value pairs list.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Include List: Attributes

Table 2-4 list Attributes

Attribute Name	Expected Type	Requirement	Contains Subfields
Use	String	Mandatory	No
Prefix	Integer	Optional	

Include List: use

The `use` attribute specifies the symbolic name for the list and must be specified, for example: `use=”user_events”`.

Include List: prefix

The `prefix` attribute specifies a value to prefix the values contained in the referenced list, for example: `prefix=”0x40”`.

For example, if the list was 8-bit values long, an 8-bit prefix value is added to each list value, as bits 9 to 15.

Include List: Child elements

The <List> XML element does not support Child elements.

Include List: Example XML

```
<ConfigItem
    Id="Event"
    Desc="User Event or System Event associated with the configuration."
    Size=”16"
    Type="enum"
    ShowOrdered="true">
    <!--Concatenates User Events (+ a 0x40 prefix) and System Events into a
single enum type Config Item-->
    <List
        use="User Event"
        prefix="0x40"/>
    <List use="System Event"/>
</ConfigItem>
```

2.6 Pattern definition

Use the <DefinePattern> XML element to specify the Configuration Items contained within a Pattern.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Pattern Definition: Attributes

Table 2-5 DefinePattern Attributes

Attribute Name	Expected Type	Requirement	Contains Subfields
PatternName	String	Mandatory	No
ShortId	Symbolic Name		

Pattern Definition: PatternName

The `PatternName` attribute specifies the text string representing the display name for the pattern body and must be specified, for example: `PatternName=”AT Command Buffer”`.

Pattern Definition: ShortId

The `ShortId` attribute defines the text string, following the rules for a C identifier, used as the symbolic name for the Pattern Body and must be specified, for example: `ShortId=”user_event”`.

Pattern Definition: Child elements

The <DefinePattern> XML element must contain at least one `<ConfigItem> (`Other than `AsciiString)`, `<ConfigArray> or <ConfigStruct>` element; it is possible to include any number of elements in any combination, as required.

- 0 or more `<ConfigItem>` elements, of any type other than `AsciiString`
- 0 or more `<ConfigArray>` elements
- 0 or more `<ConfigStruct>` elements

Pattern Definition: Example XML

```
<DefinePattern
    PatternName="AT Commands Map"
    ShortId="at_commands_events">
    <ConfigItem
        Id="AT Cmd Event"
        ShortId="event"
        Desc="User Event or System Event associated with the configuration."
        Type="enum"
        ShowOrdered="true"
        Size="16"/>
        <!--N.b. content abridged in this example!-->
</DefinePattern>
```

2.7 Structure Definition

The <DefineStruct> XML element is used to specify a group of Configuration Items that can have multiple instances by referring to the definition.

During the Configuration Build process, the `DefineStruct` XML elements will be replaced with the corresponding set of Configuration Items.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Structure Definition: Attributes

Table 2-6 DefineStruct Attributes

Attribute Name	Expected Type	Requirement	Contains Subfields
Id	String	Mandatory	No
ShortId	Symbolic Name		

Structure Definition: Id

The `Id` attribute specifies the text string that defines the display name for the struct and must be specified, for example: `Id="Audio Mic Parameters"`.

Structure Definition: ShortId

The `ShortId` attribute specifies the text string, following the rules for a C identifier, used as the symbolic name for the struct and must be specified, for example: `ShortId="audio_mic_params"`.

Structure Definition: Child elements

The <DefineStruct> XML element must contain at least one; `<ConfigItem>, <ConfigArray> or <ConfigStruct> element of any type;` it is possible to include any number of elements in any combination, as required.

- 0 or more `<ConfigItem>` elements
- 0 or more `<ConfigArray>` elements
- 0 or more `<ConfigStruct>` elements

Structure Definition: Example XML

```
<DefineStruct
    Id="Audio Mic Parameters"
    ShortId="audio_mic_params">
    <ConfigItem
        Id="Bias Drive Voice Mic A"
        ShortId="bias_config"
        Desc="Configuration bias"
        Type="enum"
        Size="2"/>
        <!--N.b. content abridged in this example!-->
</DefinePattern>
```

2.8 Configuration Item

The <ConfigItem> XML element represents the fundamental structure for the Configuration Set and is used to define the format of data within pattern and structure definitions, as well as stand-alone Configuration Items.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

ConfigItem: Attributes

Table 2-7 ConfigItem Attributes

Attribute Name	Expected Type	Requirement	Contains Subfields
Id	String	Mandatory	No
ShortId	Symbolic Name		
Type	String		
Desc			
Size	Integer	Optional	
MaxStrLenBytes			
PresentHex	Boolean		
Multiplier	Integer		
EnableTrackBar	Boolean		
Range	String		Yes
ConfigGroupPath	XPath string		No
ShowOrdered	Boolean		

ConfigItem: Id

The `Id` attribute specifies the text string representing the display name for the Configuration Item and must be specified, for example: `Id="Audio Volume"`.

ConfigItem: ShortId

The `ShortId` attribute specifies the text string, following the rules for a C identifier, used as the symbolic name for the Config Item and must be specified, for example: `ShortId="audio_volume"`.

ConfigItem: Type

The `Type` attribute defines the Configuration Item data type and must be specified, for example: `Type="uint"`.

Table 2-8 ConfigItem Type Attribute values

Type Assigned	Comment
Bool	Simple Boolean: ‘True’, ‘False’
Int	16-bit signed 2's complement integer
Uint	16-bit unsigned integer
Ulong	32-bit unsigned integer
AsciiString	A 'packed' ASCII String An array of 16-bit words, where each word contains two ASCII characters.
Enum	Up to 16-bit unsigned integer with a set of possible values Similar to a C enum declaration.
Bitfield	Up to 16-bit bitmask A set of possible bits that can be set according to a list of bit numbers.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

ConfigItem: Desc

The <Desc> attribute must be specified, even though it only provides help information and has no functional role.

For example:

```
Desc="IMPORTANT: the number of languages supported by the Audio Prompt engine
if included in the build. To use Audio Prompts in the Sink device, the
'Number of Languages Supported by Audio Prompts' must be set to at least 1."
```

ConfigItem: Size

Except for `AsciiString`, the `Size` attribute is required by all element types, to define the data field width, for example: `Size=”16”`.

ConfigItem: PresentHex

The <`PresentHex`> attribute is only applicable to <`enum`> and <`bitfield`> types; `PresentHex=”true”` displays the specified value in hexadecimal; ‘False’ displays the value as a decimal.

ConfigItem: Multiplier

If the Configuration Item type is an `int`, `uint` or `ulong` integer; then the <`Multiplier`> scaling factor attribute is used minimize the number of Bits required to cover the widest possible range of number values.

For example, Non-unitary values can be set to individual numerical values of an integer, for example; a 2-bit unsigned number can have the values 0, 1, 2 and 3. Specifying a Multiplier of 10 (`Multiplier=”10”`), causes these values to be treated as 0, 10, 20 and 30, respectively.

ConfigItem: EnableTrackerBar

If the Configuration Item type is an `int`, `uint` or `ulong` integer, use `EnableTrackBar=”true”` to display an additional mouse-controlled Track Bar control, alongside the normal Numeric Up-Down control.

`EnableTrackBar=”false”` disables the Track Bar.

ConfigItem: Range

If the Configuration Item is of `int`, `uint` or `ulong` type, use the `Range` attribute to define the valid range for that Item, for example `Range=”Min=0,Max=100”`.

Table 2-9 Range Attribute Subfields

Attribute Name	Expected Type	Requirement	Contains Subfields
Min	Integer	Mandatory	Specifies the minimum valid signed decimal integer value for the specified Configuration
Max			Specifies the maximum valid signed decimal integer value (Greater than the Min value), for the specified Configuration

ConfigItem: ConfigGroupPath

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

The optional <ConfigGroupPath> attribute contains the XPath string used to represent the unique `ShortId` or `Id` path location within the Configuration set, used to place the Configuration Item during the Configuration Build process.

The <ConfigGroupPath> attribute uses the following format:

```
ConfigGroupPath=”./[@ShortId='user_interfaces']/[@ShortId='led']”
```

ConfigItem: ShowOrdered

If the Configuration Item is of the enumerated <enum> type, use the `ShowOrdered=”true”` attribute to display all possible display names, as an alphabetically ordered list.

NOTE If the <`ShowOrdered>` attribute is not included or set ‘False’, then the displayed names are in XML Definition order, that is, the order of the Child elements encapsulated by the <`ConfigItem>`.

ConfigItem: Child elements

The `<ConfigItem>` element can only contain; `<List>`, `<bitfield> or <enum>` Child elements, as defined by the <`Type>` attribute.

ConfigItem with bitfield > Type Attribute

Any <`ConfigItem>` with <`Type>` attribute set to <`bitfield>` must contain at least one `<List>` or `<bitfield>` element; any number of either element type can be mixed in the same <`ConfigItem>`.

ConfigItem with enum > Type Attribute

Any <`ConfigItem>` with <`Type>` attribute set to <`enum>` must contain at least one `<List>` or `<enum>` element; any number of either element type can be mixed in the same <`ConfigItem>`.

ConfigItem Example XML: Unsigned Integer

The following example stores an unsigned integer in one byte.

```
<ConfigItem
    Id="Link Loss Interval [s]"
    ShortId="link_loss_interval"
    Desc="The time interval, in seconds, at which the headset will make
reconnection attempts following a link loss."
    Type="uint"
    Size=”8"/>
```

ConfigItem Example XML: Enumerated Type

The following example demonstrates the use of the <ConfigItem> XML tag to encapsulate child <enum> elements.

```
<ConfigItem
    Id="Route Digital Audio Interface Output 0"
    ShortID="route_digital_audio_interface_0"
    Desc=""
    Type="enum"
    Size=3">
    <enum key="None" value="0"/>
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
    <enum key="Primary" value="1"/>
    <enum key="Secondary" value="2"/>
    <enum key="Subwoofer" value="3"/>
    <enum key="Aux" value="4"/>
</ConfigItem>
```

ConfigItem Example XML: String

The following example illustrates an ASCII string, with the Store allocated to hold the maximum string length (16).

```
<ConfigItem
    Id="One Touch Dial Phone Number"
    Desc="If this configuration item is programmed with a phone number, this
will be sent to the Audio Gateway when a one-touch dial call event occurs on
the device."
    Type="AsciiString"
    MaxStrLenBytes=16"/>
```

ConfigItem Example XML: Bool

The following example demonstrates use of single-Bit storage for small Configuration Items, such as Boolean flags.

```
<ConfigItem
    Id="Analogue Wired Input is Stereo"
    ShortId="stereo"
    Desc="When set, this checkbox controls whether a wired analogue input is
connected through to the DSP as a Stereo stream. If it remains unchecked,
wired analogue audio will be connected as a Mono source. Note: this
Configuration Item is only relevant for wired analogue input routing, not
I2S, S/PDIF or any other wired sources."
    Type="bool"
    Size=1"/>
```

2.9 Array Definition

Use the `ConfigArray` XML element to define an array of Configuration Items. This will generate individual `ConfigItem` XML elements in the generated Configuration Set and an array in the generated source code for use by the application.

Array Definition: Attributes

Table 2-10 ConfigArray Attributes

Attribute Name	Expected Type	Requirement	Contains Subfields
Id	String	Mandatory	No
ShortId	Symbolic Name		
Type	String		

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 2-10 ConfigArray Attributes (cont.)

Attribute Name	Expected Type	Requirement	Contains Subfields
Desc			
ArraySize	Integer		
ElementType	String		
ElementStruct		Optional	
Range			Yes

Array Definition: Id

The `Id` attribute specifies the text string that represents the display name for the group and must be specified, for example: `Id="Audio Volume"`.

Array Definition: ShortId

The `ShortId` attribute defines the text string, following the rules for a C identifier, used to represent the symbolic name for the Config Array and must be specified, for example: `ShortId="audio_volume`.

Array Definition: Type

The `Type` attribute defines the data type of the Config Array and is always equal to `array, for example: Type="array"`.

Array Definition: Desc

The `Desc` attribute is used to provide help information and must be specified, even though it has no functional role.

For example:

```
Desc="IMPORTANT: the number of languages supported by the Audio Prompt engine
if included in the build. To use Audio Prompts in the Sink device, the
'Number of Languages Supported by Audio Prompts' must be set to at least 1."
```

Array Definition: ArraySize

The `ArraySize` attribute defines the quantity of `ConfigItem` XML elements to create for the `ConfigArray` and must be specified, for example: `ArraySize="16"`.

Array Definition: ElementType

The `ElementType` attribute defines the data type for all `ConfigArray` elements and must always be specified, for example: `ElementType="uint"`

Table 2-11 ConfigArray ElementType Attribute values

Type Assigned	Comment
Int	A 16-bit signed 2's complement integer
Uint	A 16-bit unsigned integer
Enum	Up to 16-bit unsigned integer with a set of possible values Similar to a C enum declaration

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Table 2-11 ConfigArray ElementType Attribute values (cont.)

Type Assigned	Comment
Bitfield	Up to 16-bit bitmask, with a set of possible bits, set according to a list of bit numbers.
struct	Use `DefineStruct` referenced in the `ElementStruct` attribute

Array Definition: ElementStruct

Use the `ElementStruct` attribute to reference a `DefineStruct` by its symbolic name, when `ElementType="struct"`.

Array Definition: Range

If the Configuration Item is of `int`, `uint` or `ulong` type, use the optional `Range="Min=0,Max=100"` attribute to define the valid range for any Integer-based Configuration Item.

Table 2-12 Range Attribute Subfields

Attribute Name	Expected Type	Requirement	Contains Subfields
`Min`	Integer	Mandatory	Specifies the minimum value of this Configuration Item integer as a signed decimal number.
`Max`	Integer	Mandatory	Specifies the maximum value of this Configuration Item integer as a signed decimal number, which cannot be less than the Min value.

Array Definition: Child elements

The <ConfigArray> element can only contain; `<List>`, `<bitfield> or <enum>` Child elements, as defined by the <`Type`> attribute.

ConfigArray with Attribute Type: bitfield

Any <`ConfigArray`> with <`Type`> attribute set to <`bitfield`> must contain at least one `<List>` or `<bitfield>` element; any number of either element type can be mixed in the same <`ConfigArray`>.

ConfigArray with Attribute Type: enum

Any <`ConfigArray`> with <`Type`> attribute set to <`enum`> must contain at least one `<List>` or `<enum>` element; any number of either element type can be mixed in the same <`ConfigArray`>.

Array Definition Example XML: 8 unsigned integers

The following example illustrates eight unsigned integers.

```
<ConfigArray
    Id="Link Loss Intervals [s]"
    ShortId="link_loss_intervals"
    Desc="Possible link loss intervals according to state."
    Type="array"
    ElementType="uint"
    ArraySize="8"/>
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Array Definition Example XML: 3 Structures

The following example illustrates three sets of a structure.

```
<ConfigArray
    Id="Link Loss Intervals [s]"
    ShortId="link_loss_intervals"
    Desc="Possible link loss intervals according to state."
    Type="array"
    ElementType=”struct”
    ElementStruct=”channel_configuration”
    ArraySize=”3"/>
```

2.10 Structure Instance Definition

Use the <ConfigStruct> XML element to include the structure definition from a DefineStruct within another definition.

Structure Instance: Attributes

Table 2-13 ConfigStruct Attributes

Attribute Name	Expected Type	Requirement	Contains Subfields
Id	String	Mandatory	No
ShortId	Symbolic Name		
Type	String		
Desc			
Struct		Optional	

Structure Instance: Id

The Id attribute defines the text string used to represent the display name for the Config Struct and must be specified, for example: Id=”Audio Volume”.

Structure Instance: ShortId

The ShortId attribute defines the text string, following the rules for a C identifier, used to represent the symbolic name for the Config Struct and must be specified, for example: ShortId=”audio_volume”.

Structure Instance: Type

The Type attribute defines the data type of the Config Struct, which is always struct.

Structure Instance: Desc

The <Desc> attribute must be specified, even though it only provides help information and has no functional role.

For example:

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
Desc="IMPORTANT: the number of languages supported by the Audio Prompt engine
if included in the build. To use Audio Prompts in the Sink device, the
'Number of Languages Supported by Audio Prompts' must be set to at least 1."
```

Structure Instance: Struct

Use the `Struct` attribute to reference a `DefineStruct` by its symbolic name.

Structure Instance: Child elements

The `<ConfigStruct>` XML element does not support Child elements.

Structure Instance: Example XML

```
<ConfigStruct
    Id="Link Loss configuration 0"
    ShortId="link_loss_config0"
    Desc="Default link loss config"
    Type="struct"
    Struct=”link_loss_configuration” />
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

3 Global Definition XML

The Global Definition XML provides all definitions referenced by more than one module XML and defines a skeleton layout for configuration items, represented by the Configuration Group structure, see .

The following XML elements are included in the Global Definition XML file.

3.1 Global Data

The <GlobalData> XML element is considered to be the basic container for a Global Definition set.

Global Data: Attributes

Not applicable for this element.

Global Data: Child elements

The Global Data XML element forms the container for the rest of the Global Definition Set and contains both `<DefineGroup>` and <ConfigGroup> elements.

- 0 or 1 `<DefineGroup>` element, See Define Group.
- 0 or more <ConfigGroup> elements See Configuration Group XML.

NOTE The <DefineGroup> must be listed as the first child element, since <DefineGroup> definitions must occur before they are used in any <ConfigGroup> sections.

Global Data: Example XML

```
<GlobalData>
    <DefineGroup/>
    <ConfigGroup
        Id="Audio"
        ShortId=”audio”
        Node="Basic">
        <!--N.b. content abridged in this example!-->
    </ConfigGroup>
</GlobalData>
```

3.2 Configuration Group XML

Use the `<ConfigGroup>` XML element to define a Config Group structure for modules, onto which definition elements, such as; `ConfigGroup`, `ConfigItem` or `ConfigPatternArray` attach.

The Configuration Group structure forms a skeleton Configuration Set Tree, for the Configuration Definition data.

ConfigGroup: Attributes

Table 3-1 <ConfigGroup> Attributes

Attribute Name	Expected Type	Requirement	Contains Subfields
Id	String	Mandatory	No
ShortId	Symbolic Name		
EnableControl	String	Optional	Yes
Node		Mandatory	No

ConfigGroup: Id

The `Id` attribute specifies the text string used as a display name for the group and must be specified.

ConfigGroup: ShortId

The `ShortId` attribute specifies a text string, following the rules for a C identifier, used as the symbolic name for the group and must be specified.

ConfigGroup: EnableControl

The optional `EnableControl` attribute indicates when the Configuration Group is either enabled or disabled by another Configuration Item and informs the Configuration Tool when to enable or disable Config Groups, in its GUI.

Table 3-2 EnableControl Attributes subfields

Attribute Subfield Name	Expected Type	Requirement
ValueToEnable	Integer	Mandatory
ConfigItemId	String	

The following XML example requires a <`ConfigItem`> in the Configuration Set, with id `"Drive PIO/BIAS Selection for Voice Mic A"`, to control whether the items under the different `ConfigGroup` nodes are enabled in the Configuration Tool.

If the `"Drive PIO/BIAS Selection for Voice Mic A"` equals '0' then:

- All items in "`Voice Mic A Drive PIO`" are enabled, and
- All items in "`Voice Mic A Bias`" are disabled

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

- And the reverse if `ConfigItem = 1`.

```
<ConfigGroup
    Id="Voice Mic A Drive PIO"
    ShortId="voice_mic_A_drive_PIO"
    Node="Basic"
    EnableControl="ValueToEnable=0,ConfigItemId=Drive PIO/BIAS Selection
for Voice Mic A"/>
<ConfigGroup
    Id="Voice Mic A Bias"
    Node="Basic"
    EnableControl="ValueToEnable=1,ConfigItemId=Drive PIO/BIAS Selection
for Voice Mic A"/>
```

Configuration Group: Node

The `Node` attribute defines how the Configuration Group is presented in the Configuration Tool and must be specified, for example: `Node=”Basic”`.

Possible values for this attribute are:

Table 3-3 Configuration Group Nodes

Node	Use
`Basic`	– Regular, basic container for all other XML elements
`Array`	– Container for a Pattern Array **NOTE** Pattern Array Configuration Groups must only contain a single `ConfigPatternArray` element.
`AudioPrompts`	– A special case container specifically for the Audio Prompt Configuration Items used in the ADK Sink Application. **NOTE** This container must only contain the specific XML elements referenced by the Configuration Tool.
`Expert`	XML elements within this node are not displayed and cannot be modified using the Configuration Tool. For example, Expert is used to store on-chip Application session data in the Configuration Store.

ConfigGroup: Child elements

The `<ConfigGroup>` Child elements are Node-specific.

Child elements for Basic, AudioPrompts or Expert Nodes

The `<ConfigGroup>` element can contain any number of `<ConfigGroup>` elements, including none.

Child elements for Array Nodes

Not applicable for this node in a Global Definition XML.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

ConfigGroup: Example XML

```
<ConfigGroup
    Id="Peer Device Support"
   ShortId="peer_device_support"
   Node="Basic">
   <ConfigGroup
        Id="Custom Peer Device Service UUID Global"
        ShortId="custom_peer_device_service_uuid_g"
        Node="Array">
   </ConfigGroup>
   <ConfigGroup
        Id="ShareMe"
        ShortId="shareme"
        Node="Basic">
   </ConfigGroup>
   <ConfigGroup
        Id="True Wireless Stereo"
        ShortId="true_wireless_stereo"
        Node="Basic">
        <ConfigGroup
            Id="Audio Routing"
            ShortId="audio_routing"
            Node="Basic">
        </ConfigGroup>
        <ConfigGroup
            Id="Device Trim"
            ShortId="device_trim"
            Node="Basic">
        </ConfigGroup>
    </ConfigGroup>
</ConfigGroup>
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

4 Module Definitions XML

Module Definitions XML provide definitions specific to a module and contains a collection of Configuration Items or Pattern Arrays, which are generally placed on the Global Definition XML group structure.

This section describes the XML elements that can be included in a Module Definition XML file.

4.1 Module Data

The <ModuleData> XML element is considered to be the basic container for a Module Definition set.

Module Data: Attributes

Table 4-1 ModuleData Attributes

Attribute Name	Expected Type	Requirement	Contains Subfields
Name	Symbolic Name	Mandatory	No

Module Data: Name

The `Name` attribute must be used to define the text string that represents the symbolic name for the module and must always be specified, for example: `Name="sink_anc"`.

NOTE The Config Build process uses this attribute to name the module header files generated during the build. By default, each module produces a header file called `<name>_config_def.h`, for example, `sink_anc_config_def.h`.

Module Data: Child elements

The <ModuleData> XML element can contain either `<DefineGroup>` or `<ConfigGroup>` elements.

For example:

- 0 or 1 <DefineGroup> elements, See)
- 1 or more <ConfigGroup> elements, See)

NOTE If the <DefineGroup> is present, it must be listed as the first child element, since the definitions within it must occur before they are used in any <ConfigGroup> sections.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Module Data: Example XML

```
<ModuleData Name=”sink_anc”>
    <DefineGroup/>
    <ConfigGroup
        Id="Audio"
        ShortId=”audio”
        Node="Basic">
        <!--N.b. content abridged in this example!-->
    </ConfigGroup>
</ModuleData>
```

4.2 Configuration Group

The <ConfigGroup> XML element is used to organize groups of related configuration data into an **Item Tree** format in Module Definitions.

NOTE The <ConfigGroup> Configuration Data groupings are often different to those displayed within the Configuration Tool GUI Tree View. Therefore, to enable re-sorting, the Config Build process re-directs Items to a skeleton Configuration Set Tree, defined in the Global Definition.

In addition to the Global Definition <ConfigGroup> properties (see), a Module Definition <ConfigGroup> also support the following additional properties.

ConfigGroup: Attributes

All the attributes of a Global Definition `ConfigGroup` apply.

ConfigGroup: Child elements

The Child elements of the `<ConfigGroup>` element are Node-specific.

Child elements for Basic, AudioPrompts or Expert Nodes

The `<ConfigGroup>` element can contain any number or combination of: <ConfigGroup>, `<ConfigArray>`, <ConfigItem> or <ConfigStruct> elements, including none.

- 0 or more <ConfigGroup> elements
- 0 or more <ConfigArray> elements
- 0 or more <ConfigItem> elements
- 0 or more <ConfigStruct> elements

Child elements for Array Nodes

The `<ConfigGroup>` element can contain any number of <ConfigPatternArray> elements, including none.

ConfigGroup: Basic Node Example XML

This code example illustrates a Basic Node ConfigGroup.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
<ConfigGroup
    Id="Peer Device Support"
   ShortId="peer_device_support"
   Node="Basic">
   <ConfigGroup
        Id="Custom Peer Device Service UUID Global"
        ShortId="custom_peer_device_service_uuid_g"
        Node="Array">
   </ConfigGroup>
   <ConfigItem
        Id="ShareMe"
        ShortId="shareme"
        Desc=”ShareMe state”
        Type="bool"
        Size=”1”
        DefaultValue=”true”>
   </ConfigGroup>
   <ConfigGroup
        Id="True Wireless Stereo"
        ShortId="true_wireless_stereo"
        Node="Basic">
        <ConfigGroup
            Id="Audio Routing"
            ShortId="audio_routing"
            Node="Basic">
        </ConfigGroup>
        <ConfigGroup
            Id="Device Trim"
            ShortId="device_trim"
            Node="Basic">
        </ConfigGroup>
    </ConfigGroup>
</ConfigGroup>
```

ConfigGroup: Array Node Example XML

This code example illustrates an Array Node ConfigGroup.

```
<ConfigGroup
    Id="Audio Prompts Support"
   ShortId="audio_prompts_support"
   Node="Array">
   <ConfigPatternArray
        Pattern=” audio_prompts_config_type"
        Id="Audio Prompt"
        ShortId="audio_prompts_config "
        MaxNumPatterns="128">
</ConfigGroup>
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

4.3 Configuration Item

The <ConfigItem> XML element is used as the fundamental structural building block for the Configuration Set and to define both Application data format and default data in a `ConfigGroup`.

NOTE During the Configuration Build process, any XML elements not natively supported by the Configuration Tool, such as `ConfigArray` or `ConfigStruct`, are also converted into a Configuration Items.

Attributes

In addition to the attributes of a `DefineGroup ConfigItem` (see), the following also apply to <ConfigItem>:

Table 4-2 ConfigItem Attributes

Attribute Name	Expected Type	Requirement	Contains Subfields
DefaultValue	In accordance with Type	Mandatory	No

ConfigItem: DefaultValue

The `DefaultValue` attribute defines the value used by the Application when no default value for the Build Variant is available and must be specified, for example: `DefaultValue=”16”`.

NOTE The assigned value is dependent on the `ConfigItem` data type.

ConfigItem: Child elements

In addition to the Child elements defined in Configuration Item, a <`ConfigItem`> may also contain any number of <`BuildVariant`> elements, including none.

ConfigItem Example XML: Unsigned Integer

The following example is used to store an unsigned integer in one byte, with a default of `255` or `0xff`.

```
<ConfigItem
    Id="Link Loss Interval [s]"
    ShortId="link_loss_interval"
    Desc="The time interval, in seconds, at which the headset will make
reconnection attempts following a link loss."
    Type="uint"
    Size=”8"
    DefaultValue=”0xff”/>
```

ConfigItem Example XML: Enumerated Type

This example demonstrates the encapsulation of child <enum> elements by the <ConfigItem> XML tag, where, `DefaultValue` is specified as either a numeric value or the corresponding key value, from the <`enum`> definition.

```
<ConfigItem
    Id="Route Digital Audio Interface Output 0"
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
    ShortID="route_digital_audio_interface_0"
    Desc=""
    Type="enum"
    Size=3"
    DefaultValue="Secondary">
    <enum key="None" value="0"/>
    <enum key="Primary" value="1"/>
    <enum key="Secondary" value="2"/>
    <enum key="Subwoofer" value="3"/>
    <enum key="Aux" value="4"/>
</ConfigItem>
```

ConfigItem > Example XML > : ASCII String

This example demonstrates how a Store is allocated to hold the maximum length of an ASCII string and defaulted to an appropriate phone number.

```
<ConfigItem
    Id="One Touch Dial Phone Number"
    Desc="If this configuration item is programmed with a phone number, this
will be sent to the Audio Gateway when a one-touch dial call event occurs on
the device."
    Type="AsciiString"
    MaxStrLenBytes=16"
    DefaultValue=”08005355356”/>
```

ConfigItem Example XML: Single-bit Bool

This example demonstrates single-bit storage, as used with the smallest Configuration Items, such as Boolean flags.

```
<ConfigItem
    Id="Analogue Wired Input is Stereo"
    ShortId="stereo"
    Desc="When set, this checkbox controls whether a wired analogue input is
connected through to the DSP as a Stereo stream. If it remains unchecked,
wired analogue audio will be connected as a Mono source. Note: this
Configuration Item is only relevant for wired analogue input routing, not
I2S, S/PDIF or any other wired sources."
    Type="bool"
    Size=1"
    DefaultValue=”true”/>
```

Configuration Item Example XML: bitfield Type

This example demonstrates encapsulation of <bitfield> Child elements by the <ConfigItem> XML tag, using a default setting that combines bits 1, 3 and 4.

NOTE `Specify <DefaultValue>` as either a numeric value or the corresponding key values, obtained from the `bitfield` definition. The empty string, or 0, represents no bits set.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
<ConfigItem
    Id="Route Digital Audio Control 0"
    ShortID="route_digital_audio_control_0"
    Desc=""
    Type="bitfield"
    Size=5"
    DefaultValue="Primary | Subwoofer | Aux">
    <bitfield key="None" value="0"/>
    <bitfield key="Primary" value="1"/>
    <bitfield key="Secondary" value="2"/>
    <bitfield key="Subwoofer" value="3"/>
    <bitfield key="Aux" value="4"/>
</ConfigItem>
```

4.4 Array Declaration

In addition to the properties that form part of a `ConfigArray` configuration item, see , the following properties also apply.

Array Declaration: Attributes

As for a <`ConfigArray`> see .

Array Declaration: Child elements

In addition to the <`ConfigArray`> Child elements; one or more `<ArrayElementConfigItem>` elements are included, each corresponding to an element in the array.

Array Declaration Example XML: Add unsigned integers

The following example adds eight unsigned integers to the Configuration Set.

```
<ConfigArray
    Id="Link Loss Intervals [s]"
    ShortId="link_loss_intervals"
    Desc="Possible link loss intervals according to state."
    Type="array"
    ElementType=”uint”
    ArraySize=”8"/>
    <ArrayElementConfigItem
        …
        DefaultValue=”24”/>
    <!--N.b. content abridged in this example!-->
    <ArrayElementConfigItem
        …
        DefaultValue=”0x42”/>
</ConfigArray>
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Array Declaration Example XML: 3 Structures

This example demonstrates three sets of a structure.

```
<ConfigArray
    Id="Link Loss Intervals [s]"
    ShortId="link_loss_intervals"
    Desc="Possible link loss intervals according to state."
    Type="array"
    ElementType=”struct”
    ElementStruct=”channel_configuration”
    ArraySize=”3"/>
    <ArrayElementConfigItem
        …
        <StructElementConfigItem
    <!--N.b. content abridged in this example! Included to show relationships
between elements.-->
    <ArrayElementConfigItem
        …
        DefaultValue=”0x42”/>
        <StructElementConfigItem
</ConfigArray>
```

4.5 Array Item

Use the <ArrayElementConfigItem> XML element to define a default value for an item within a `ConfigArray`.

Array Item: Attributes

Table 4-3 <ArrayElementConfigItem> Attributes

Attribute Name	Expected Type	Requirement	Contains Subfields
Id	String	Mandatory	No
ShortId	Symbolic Name		
Desc	String		
DefaultValue		Optional	

Array Item: Id

The `Id` attribute specifies a text string representing the display name for the array element and must be specified.

Array Item: ShortId

The `ShortId` attribute specifies a text string, following the rules for a C identifier, used to represent the symbolic name for the array element and must be specified.

Array Item: Desc

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

The Desc attribute provides help information and must be specified, even though it has no functional role.

Array Item: DefaultValue

If the Config Array ElementType is **not** <struct>: use the DefaultValue attribute to define the default value for the Application, for example: DefaultValue=”16”.

NOTE Value format is dependent on the Config Array data type.

Array Item: Child elements

The <ArrayElementConfigItem> XML element can contain any number of <BuildVariant> elements, including none or, if the ConfigArray ElementType is <struct>, one <StructElementConfigItem> element for each corresponding <struct> element.

NOTE If the ArrayElementConfigItem XML element is being used as a direct or indirect child of a PatternArrayRow XML element then BuildVariant child elements are not allowed as build specific values for a Pattern Array have to be specified at PatternArrayRow granularity.

Array Item Example XML: Unsigned integers

This example demonstrates one element of a <uint> Config Array.

```
<ArrayElementConfigItem
    Id="Link Loss configuration 0"
    ShortId="link_loss_config0"
    Desc="Default link loss config"
    DefaultValue="0x200"/>
```

Array Item Example XML: Structures

This example demonstrates one element of a Config Array with a structure containing a <uint> and an <int>.

```
<ArrayElementConfigItem
    Id="Link Loss configuration 0"
    ShortId="link_loss_config0"
    Desc="Default link loss config">
    <StructElementConfigItem
        …
        DefaultValue=”34”/>
    < StructElementConfigItem
        …
        DefaultValue=”-56”/>
</ArrayElementConfigItem>
```

4.6 Structure Declaration

In addition to the properties defined as part of a <ConfigStruct> definition (see), when using a <ConfigStruct> configuration item, the following properties also apply.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Structure Declaration: Attributes

The <ConfigStruct> attributes apply, see .

Structure Declaration: Child elements

In addition to the Child elements (see), a ConfigStruct can also have one or more <StructElementConfigItem> Child elements, each corresponding to a structure element.

NOTE Each element in the structure is defined by the order of the Child elements, that is, the first child element represents the first element defined in the structure, and so on.

Structure Declaration: Example XML

```
<ConfigStruct
    Id="Link Loss configuration 0"
    ShortId="link_loss_config0"
    Desc="Default link loss config"
    Type="struct"
    Struct=”link_loss_configuration”>
    <StructElementConfigItem
        …
        DefaultValue=”34”/>
    <!--N.b. content abridged in this example!-->
    < StructElementConfigItem
        …
        DefaultValue=”0x56”/>
</ConfigStruct>
```

4.7 Structure Item

A <StructElementConfigItem> XML element defines a default value for an item within a ConfigStruct.

Structure Item: Attributes

Table 4-4 <StructElementConfigItem> Attributes

Attribute Name	Expected Type	Requirement	Contains Subfields
Id	String	Mandatory	No
ShortId	Symbolic Name		
Desc	String		
DefaultValue		Optional	

Structure Item: Id

The Id attribute specifies a text string used as the display name for the array element and must be specified.

Structure Item: ShortId

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

The ShortId attribute specifies a text string, following the rules for a C identifier, used as the symbolic name for the array element and must be specified.

Structure Item: Desc

The Desc attribute provides help information and must be specified, even though it has no functional role.

Structure Item: DefaultValue

The DefaultValue attribute defines the Application default value and must be specified if the corresponding ConfigStruct element is not another ConfigStruct or a ConfigArray.

NOTE Value format is dependent upon the corresponding element <ConfigStruct> data type.

Structure Item: Child elements

Embedded Child element type is dependent upon the corresponding element defined in the ConfigStruct element type.

Where a Child element can be:

1 or more *<StructElementConfigItem>*	ONLY if the corresponding embedded element is defined as a ConfigStruct
1 or more *<ArrayElementConfigItem>*	ONLY if the corresponding embedded element is defined as a ConfigArray
0 or more *<BuildVariant>*	For any corresponding embedded element type

NOTE If the StructElementConfigItem XML element is being used as a direct or indirect child of a PatternArrayRow XML element then BuildVariant child elements are not allowed as build specific values for a Pattern Array have to be specified at PatternArrayRow granularity.

Structure Item Example XML: Unsigned Integers

This example demonstrates a ConfigStruct containing two elements.

```
<ConfigStruct
    Id="Silence Detect Instance"
    ShortId="SilenceDetSettings"
    Desc=" "
    Type="struct"
    Struct="silence_detect_settings">
    <StructElementConfigItem
        Id="Threshold"
ShortId="threshold_default"                ConfigGroupPath="./
[@ShortId='audio']/[@ShortId='routing']/
[@ShortId='silence_detection_features']"
        Desc="..."
        DefaultValue="0x000A" />
    <StructElementConfigItem
        Id="Trigger Time [s]"
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
ShortId="trigger_time_default"        ConfigGroupPath="./[@ShortId='audio']/
[@ShortId='routing']/[@ShortId='silence_detection_features']"
        Desc="..."
        DefaultValue="0x0258" />
</ConfigStruct>
```

Structure Item Example XML: Embedded <ConfigStruct>

This example demonstrates a <ConfigStruct> with the first element in the structure being an embedded <ConfigStruct>.

```
<ConfigStruct
    Id="Anc Mic Params"
    ShortId="anc_mic_params_r_config"
    Desc="Anc Mic Params"
    Type="struct"
    Struct="anc_mic_params_r_config">
    <StructElementConfigItem
        Id="Audio Params Mic A"
        ShortId="mic_a"
        Desc="Audio Params Mic A. ">
        <StructElementConfigItem
            Id="Bias Drive ANC Mic A"
            ShortId="bias_r_config_a"
            Desc=" "
            DefaultValue="Mic Bias 0"/>
        …
    </StructElementConfigItem>
</ConfigStruct>
```

4.8 Configuration Pattern Array

Use the <ConfigPatternArray> XML element to create arrays of Patterns, as defined using the <DefinePattern> element, See .

Configuration Pattern Array: Attributes

Table 4-5 <ConfigPatternArray> attributes

Attribute Name	Expected Type	Requirement	Contains Subfields
Id	String	Mandatory	No
ShortId	Symbolic Name		
Pattern	String		
MaxNumPatterns	Integer	Optional	
FixedNumPatterns			
Presentation			
ConfigGroupPath	XPath string		

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Configuration Pattern Array: Id

The `Id` attribute specifies the text string used to represent the display name for the pattern array and must be specified.

Configuration Pattern Array: ShortId

The `ShortId` attribute specifies the text string, following the rules for a C identifier, used to represent the symbolic name for the Config Pattern Array and must be specified.

Configuration Pattern Array: Pattern

The `Pattern` attribute specifies the Pattern to make array instances.

Configuration Pattern Array: MaxNumPatterns

Use the `MaxNumPatterns` attribute to specify the maximum number of elements to be available at any given time.

NOTE This attribute:

- Must be specified if <`FixedNumPatterns`> is not defined
- Cannot be present if <`FixedNumPatterns`> is defined

Configuration Pattern Array: FixedNumPatterns

Use the `FixedNumPatterns` attribute to specify an exact number of elements in a fixed-length array.

NOTE This attribute:

- Must be specified if <`MaxNumPatterns`> is not defined
- Cannot be present if <`MaxNumPatterns`> is defined

Configuration Pattern Array: Presentation

If the number of Patterns is fixed, use the <`Presentation`> attribute set to `"ReadOnlyHeader"` to indicate that the first element in the pattern should be treated as a Read-Only header for the pattern row.

Configuration Pattern Array: ConfigGroupPath

The optional <ConfigGroupPath> attribute contains the XPath string used to represent the unique `ShortId` or `Id` path location within the Configuration set, used to place the Pattern Array during the Configuration Build process.

The <ConfigGroupPath> attribute uses the following format:

```
ConfigGroupPath="./[@ShortId='user_interfaces']/[@ShortId='led']"
```

Configuration Pattern Array: Child elements

The Configuration Pattern Array contains a number of <`PatternArrayRow`> elements, each corresponding to a Pattern in the Pattern Array; this number can be zero.

Config Pattern Array Example XML: Variable Length (≤ 8 pattern) Array

The following example can be used for up to eight Pattern instances, but only includes one <`PatternArrayRow`>, by default.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

```
<ConfigPatternArray
    Id="Event Filter"
    ShortId="pEventFilters"
    Pattern="LEDFilter”
    MaxNumPatterns=”8">
    < PatternArrayRow
    … />
</ConfigPatternArray>
```

Config Pattern Array Example XML: Fixed Length Array (2-pattern)

The following example is used for two-pattern instances and must include two <PatternArrayRow> elements.

```
<ConfigPatternArray
    Id="Event Filter"
    ShortId="pEventFilters"
    Pattern="LEDFilter”
    MaxNumPatterns=”2">
    < PatternArrayRow
    … />
    < PatternArrayRow
    … />
</ConfigPatternArray>
```

4.9 Configuration Pattern Array Row

Use the <PatternArrayRow> XML element to encapsulate a collection of default data for one instance of a <DefinePattern>, as if it is a row of data in the <ConfigPatternArray>.

Config Pattern Array Row: Attributes

Table 4-6 <PatternArrayRow> Attributes

Attribute Name	Expected Type	Requirement	Contains Subfields
Id	String	Mandatory	No
ShortId	Symbolic Name		
Node	String		

Config Pattern Array Row: Id

The Id attribute specifies the text string used to represent the display name for the pattern array and must be specified.

Config Pattern Array Row: ShortId

The ShortId attribute specifies the text string, following the rules for a C identifier, used to represent the symbolic name for the Config Pattern Array and must be specified.

Config Pattern Array Row: Node

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

The `Node` attribute must be specified and must be set to `"Basic"`.

Config Pattern Array Row: Child elements

The <PatternArrayRow> XML element must contain at least one `<PatternArrayConfigItem>` Child element, where each child corresponds to a unique Configuration Item, defined in the Pattern.

The corresponding Configuration Item element in the Pattern is defined by the order of the Child element; that is, the first child element represents the first element defined in the pattern.

Config Pattern Array Row Example XML: Variable length Pattern Array (≤ 8 rows)

The following example illustrates a variable length Pattern Array able to contain up to eight rows, but only using one <`PatternArrayRow>`, by default.

```
<ConfigPatternArray
    Id="Look Up Table Array IR"
    ShortId="lookupTable"
    Pattern="irLookupTableConfig"
    MaxNumPatterns="8">
    <PatternArrayRow Id="Row1" ShortId="bt_row1" Node="Basic">
        <PatternArrayConfigItem …/>
    </PatternArrayRow>
</ConfigPatternArray>
```

Config Pattern Array Row Example XML: Fixed Length (2-row) Pattern Array

The following example illustrates a fixed length Pattern Array containing two rows, each having a single <`PatternArrayConfigItem>` element.

```
<ConfigPatternArray
    Id="Button Translation" ShortId="button_translation"
    Pattern="button_translation_type"
    FixedNumPatterns="2"
    Presentation="ReadOnlyHeader">
    <PatternArrayRow Id="Row1" ShortId="bt_row1" Node="Basic">
        <PatternArrayConfigItem …/>
    </PatternArrayRow>
    <PatternArrayRow Id="Row2" ShortId="bt_row2" Node="Basic">
        <PatternArrayConfigItem …/>
    </PatternArrayRow>
</ConfigPatternArray>
```

4.10 Configuration Pattern Array Row item

The `<PatternArrayConfigItem>` XML element defines the default data for an Item within a <`PatternArrayRow>`.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Configuration Pattern Array Row item: Attributes

Table 4-7 <PatternArrayConfigItem> Attributes

Attribute Name	Expected Type	Requirement	Contains Subfields
ShortId	Symbolic Name	Mandatory	No
DefaultValue	String		

Configuration Pattern Array Row item: ShortId

The `ShortId` attribute defines the text string, following the rules for a C identifier, used to define the symbolic name for the Configuration Pattern Array and must be specified.

Configuration Pattern Array Row item: DefaultValue

The additional `DefaultValue` attribute defines the Application default value, where the value format is dependent upon the corresponding <`DefinePattern`> element data type.

Configuration Pattern Array Row item: Child elements

Not applicable

Config Pattern Array Row item Example XML: Variable length Array

The following example illustrates a variable length Pattern Array (see), that contains a three <`PatternArrayConfigItem`> element:

```
<ConfigPatternArray
    Id="Look Up Table Array IR"
    ShortId="lookupTable"
    Pattern="irLookupTableConfig"
    MaxNumPatterns="8">
    <PatternArrayRow Id="Row1" ShortId="bt_row1" Node="Basic">
        <PatternArrayConfigItem
            ShortId="input_id"
            DefaultValue="vb0" />
        <PatternArrayConfigItem
            ShortId="ir_code"
            DefaultValue="0x0C" />
        <PatternArrayConfigItem
            ShortId="remote_address"
            DefaultValue="0" />
    </PatternArrayRow>
</ConfigPatternArray>
```

4.11 Build Variant

The `BuildVariant` XML element is used to change default Configuration Item values to make them Build-specific, as required.

`BuildVariant` values can either; **Not Match** a specific Build, be a **Default Match**, be a **Partial Match** or an **Exact Match** for the Build.

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Build Variant values are:

- Not used by the Build if it does **Not Match**
- Used by the Build if it is an **Exact Match**
- Used by the Build if it is a **Partial Match,** but only if an Exact Match is not found.
- Used by the Build if it is a **Default Match**, but only if an Exact or Partial Match is not found.

NOTE A Partial Match occurs when either the `HwVariant` or the `SwVariant` is an Exact Match for a given Build and the other variant is a wildcard option. A Default Match is when both `HwVariant` and `SwVariant` are the wildcard option. For example, for a Build with:

hw variant = `hw_var`

sw variant = `sw_var`

- <BuildVariant HwVariant=”All” SwVariant=”All”> **Default Match**
- <BuildVariant HwVariant=”hw_var” SwVariant=”All”> **Partial Match**
- <BuildVariant HwVariant=”hw_var” SwVariant=”sw_var”> **E > xact Match**

Build Variant Attributes

Table 4-8 <BuildVariant> Attributes

Attribute Name	Expected Type	Requirement	Contains Subfields
HwVariant	String	Mandatory	No
SwVariant			
Value		Optional	

Build Variant: HwVariant

The `HwVariant` attribute specifies a text string that matches the hardware variant to which the default value is to apply and must be specified.

The string can specify one or more specific variants, for example:

- **HwVariant = `”hw_var1 hw_var2”`** – **Matches both `hw_var1` and `hw_var2`**
- HwVariant = `”All”` – Matches any `hw` variant

NOTE The minus symbol (-), is used to remove specific variants from the set of `All`, for example:

- `HwVariant=”All –hw_var1”` – Matches any hw variant except `hw_var1`

Build Variant: SwVariant

The `SwVariant` attribute can take the same format as the mandatory hw variant, but must be specified, since it defines the software variant to which the default value is to apply, for example: `SwVariant=”Headset-Gaming”`.

Build Variant: Value

MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION

Use the `Value` attribute to define the default value for the build matching this `BuildVariant`, unless specifying default values for a <`ConfigPatternArray>, where` a simple data value is not sufficient.

Value format is dependent on the data type and must be specified.

Build Variant: Child elements

If specifying build variant values for a `ConfigPatternArray` then the <BuildVariant> element requires at least one `<PatternArrayRow>` element representing the build specific values for a complete row of data in the pattern array.

Build Variant Example XML: Unsigned integers

This example illustrates one element of a <`uint`><`ConfigArray>`.

```
<ArrayElementConfigItem
    Id="Link Loss configuration 0"
    ShortId="link_loss_config0"
    Desc="Default link loss config"
    DefaultValue="0x200">
    <BuildVarint
        HwVariant="H13179v2_H13478v2"
        SwVariant="All"
        Value="0x022E"/>
```

Build Variant Example XML: ConfigPatternArray

This example illustrates a <`ConfigPatternArray>` row containing two items.

```
<BuildVariant HwVariant="All" SwVariant="Soundbar-Subwoofer">
    <PatternArrayRow
        Id="Tone Config Row 20"
        ShortId="tone_config_row_20"
        Node="Basic" >
        <PatternArrayConfigItem
            ShortId="Event"
            DefaultValue="0x4085"/>
        <PatternArrayConfigItem
            ShortId="Tone"
            DefaultValue="0x44" />
    </PatternArrayRow>
</BuildVariant>
```


MAY CONTAIN U.S. AND INTERNATIONAL EXPORT CONTROLLED INFORMATION