1 Overview

1.1 Introduction
Currently, IL standard lacks a mechanism to synchronize multiple configs (run-time settings) such that they are all applied atomically. To introduce this functionality, this extension defines two new indices and a new structure.

1.2 Dependency
This extension is written against the wording of:
OpenMAX IL 1.1.2 Specification
Document Version 1.1.2.0
September 1, 2008

1.3 Functionality

1.3.1 Operation Modes
This extension introduces two modes of operation: immediate mode (current behavior) and deferred mode.

In the deferred mode, all config settings are cached but not applied until the client sends an explicit commit config (see next section). At that point in time, all the settings that have not been applied yet shall be applied atomically. In case the client sets the component back to the immediate mode, while there are pending (not applied) settings, all pending settings shall be discarded, and the component moves back to the immediate mode without any changes to its runtime settings.

At the time of commit all pending settings are applied simultaneously and atomically; the order of the individual OMX_SetConfig() calls shall not affect the end result. However, if the IL client sends the same setting multiple times before the commit, the last of such settings shall override the earlier settings.

In case the IL client calls OMX_GetConfig() on a setting that is currently pending, the result will reflect the current active value of that setting, not the value in queue.

Trying to set the component to the deferred mode, when a component already is in the deferred mode is an error condition; OMX_ErrorInvalidMode shall be returned. OMX_ErrorInvalidMode is a new error code proposed as a part of this extension.

1.3.2 Explicit commit
A component that supports the extension for commit mode selection, shall also support the explicit commit extension index. By using this index, the IL client can apply
(commit) settings cached in the component. The component will remain in the deferred mode after the commit config.

Commit on a component that is in the immediate mode is an error condition; OMX_ErrorInvalidMode shall be returned. Commit on a component that is in the deferred mode but does not have configs pending, has no effect.

1.3.3 Error handling in the deferred mode

If an OMX_SetConfig() fails (returns an error code) while a component is in the deferred mode, the next commit shall fail with OMX_ErrorBadParameter. It is also possible that separate configs have interdependencies, e.g., a particular setting of one config may restrict the valid value range of another setting. In such cases the individual OMX_SetConfig() calls can fail with an error code only if the component will not support a particular setting in any situation. If all the individual OMX_SetConfig() calls have succeeded, but the component determines the combination invalid, it shall signal the IL client by returning OMX_ErrorBadParameter from the commit OMX_SetConfig() call. When a commit fails, all settings cached in the component shall be discarded.

1.4 Extension Definitions

The extension consists of four parts: two new config indices and the data structures to be used with these indices.

1.4.1 Index Definitions

<table>
<thead>
<tr>
<th>OpenMAX IL Index (in OMX_IndexExt.h)</th>
<th>Corresponding OpenMAX IL Structures</th>
</tr>
</thead>
<tbody>
<tr>
<td>OMX_IndexConfigCommitMode</td>
<td>OMX_CONFIG_COMMITMODETYPE</td>
</tr>
<tr>
<td>OMX_IndexConfigCommit</td>
<td>OMX_CONFIG_COMMITTYPE</td>
</tr>
</tbody>
</table>

1.4.2 OMX_CONFIG_COMMITMODETYPE

An IL component that supports the deferred mode of operation, shall support the extension index OMX_IndexConfigCommitMode, with the corresponding data structure OMX_CONFIG_COMMITMODETYPE. This structure is used for setting the component to work in deferred (bDeferred=OMX_TRUE) or in immediate (bDeferred=OMX_FALSE) mode. The default is the immediate mode. This setting can be changed using the OMX_SetConfig() function, and the current state can be queried using the OMX_GetConfig() function.

OMX_CONFIG_COMMITMODETYPE is defined in OMX_ComponentExt.h as follows.
typedef struct OMX_CONFIG_COMMITMODETYPE {
    OMX_U32 nSize;
    OMX_VERSIONTYPE nVersion;
    OMX_BOOL bDeferred;
} OMX_CONFIG_COMMITMODETYPE;

1.4.2.1 Parameters
The parameters for OMX_CONFIG_CALLBACKREQUESTTYPE are defined as follows.

- `bDeferred` either enables (OMX_TRUE) or disables (OMX_FALSE) the deferred mode of operation; the default value is OMX_FALSE.

1.4.3 OMX_CONFIG_COMMITTYPE
A component that supports the extension for commit mode selection, shall also support the explicit commit extension index OMX_IndexConfigCommit, with the corresponding data structure OMX_CONFIG_COMMITTYPE.

OMX_CONFIG_COMMITTYPE is defined in OMX_ComponentExt.h as follows.

typedef struct OMX_CONFIG_COMMITTYPE {
    OMX_U32 nSize;
    OMX_VERSIONTYPE nVersion;
} OMX_CONFIG_COMMITTYPE;

1.4.3.1 Parameters
There are no parameters except for the standard fields. The functionality is fully determined by the index OMX_IndexConfigCommit.

1.4.4 Error Definition
This extension adds a new error code OMX_ErrorInvalidMode. The error code is defined as follows.

<table>
<thead>
<tr>
<th>Field name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OMX_ErrorInvalidMode</td>
<td>The component is currently in a mode that is incompatible with the control being applied.</td>
</tr>
</tbody>
</table>