



# **OpenMAX™ Integration Layer Extension**

## **Extended Callback Events**

Version 1.0.0

Copyright © 2010 The Khronos Group Inc.

May 18, 2010  
Document version 1.0.0.0

Copyright © 2005-2010 The Khronos Group Inc. All Rights Reserved.

This specification is protected by copyright laws and contains material proprietary to the Khronos Group, Inc. It or any components may not be reproduced, republished, distributed, transmitted, displayed, broadcast, or otherwise exploited in any manner without the express prior written permission of the Khronos Group. You may use this specification for implementing the functionality therein, without altering or removing any trademark, copyright or other notice from the specification, but the receipt or possession of this specification does not convey any rights to reproduce, disclose, or distribute its contents, or to manufacture, use, or sell anything that it may describe, in whole or in part.

Khronos Group grants express permission to any current Promoter, Contributor or Adopter member of Khronos to copy and redistribute UNMODIFIED versions of this specification in any fashion, provided that NO CHARGE is made for the specification and the latest available update of the specification for any version of the API is used whenever possible. Such distributed specification may be reformatted AS LONG AS the contents of the specification are not changed in any way. The specification may be incorporated into a product that is sold as long as such product includes significant independent work developed by the seller. A link to the current version of this specification on the Khronos Group website should be included whenever possible with specification distributions.

Khronos Group makes no, and expressly disclaims any, representations or warranties, express or implied, regarding this specification, including, without limitation, any implied warranties of merchantability or fitness for a particular purpose or non-infringement of any intellectual property. Khronos Group makes no, and expressly disclaims any, warranties, express or implied, regarding the correctness, accuracy, completeness, timeliness, and reliability of the specification. Under no circumstances will the Khronos Group, or any of its Promoters, Contributors or Members or their respective partners, officers, directors, employees, agents or representatives be liable for any damages, whether direct, indirect, special or consequential damages for lost revenues, lost profits, or otherwise, arising from or in connection with these materials.

SAMPLE CODE and EXAMPLES, as identified herein, are expressly depicted herein with a “grey” watermark and are included for illustrative purposes only and are expressly outside of the Scope as defined in Attachment A - Khronos Group Intellectual Property (IP) Rights Policy of the Khronos Group Membership Agreement. A Member or Promoter Member shall have no obligation to grant any licenses under any Necessary Patent Claims covering SAMPLE CODE and EXAMPLES.

Khronos and OpenMAX are trademarks of the Khronos Group Inc. Bluetooth is a registered trademark of the Bluetooth Special Interest Group. RealAudio and RealVideo are registered trademarks of RealNetworks, Inc. Windows Media is a registered trademark of Microsoft Corporation.

# Contents

- 1 OVERVIEW .....4
- 1.1 INTRODUCTION .....4
- 1.2 DEPENDENCY .....4
- 1.3 EXTENSION DEFINITIONS .....4
  - 1.3.1 *Event definition*.....5
  - 1.3.2 *Index definition*.....5
  - 1.3.3 *Data Structure Definition* .....5

# 1 Overview

## 1.1 Introduction

OpenMAX IL standard currently defines nine types of standard events:

- `OMX_EventCmdComplete`
- `OMX_EventError`
- `OMX_EventMark`
- `OMX_EventPortSettingsChanged`
- `OMX_EventBufferFlag`
- `OMX_EventResourcesAcquired`
- `OMX_EventComponentResumed`
- `OMX_EventDynamicResourcesAvailable`
- `OMX_EventPortFormatDetected`

None of the above is suitable for real-time events that signal changes in run-time controls (e.g., auto-focus events). Specifically, `OMX_EventPortSettingsChanged` is only described in the standard for situations that require the client to disable the corresponding port before taking any other action. Additionally, while the IL event enumerated type `OMX_EVENTTYPE` has a dedicated range for vendor extensions, IL clients have no way of knowing beforehand, if a component supports a given extension event.

This extension provides a solution by defining one new event type for signaling a change in any IL parameter or config.

## 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 Extension Definitions

The extension consists of three parts: a new event type, a new config index for enabling callback events, and a data structure to be used with the new index.

### 1.3.1 Event definition

This extension introduces a new event type `OMX_EventIndexSettingChanged` that can be used for signaling changes in IL params, configs, and vendor extensions.

When receiving this event the event handler parameters are

eEvent (in <code>OMX_CoreExt.h</code> )	nData1	nData2	pEventData
<code>OMX_EventIndexSettingChanged</code>	Port index (may also be <code>OMX_ALL</code> )	Param or config index	NULL

This event signals a change in the parameter or config with index 'nData2' in component port 'nData1'. Naturally, the index in nData2 may be an index that is specific to a regular vendor extension as well as one of the standard IL indices.

IL parameter and config data structures typically have multiple fields of parameters. If a component supports `OMX_EventIndexSettingChanged` for a particular index, it shall send the event to the IL client each time at least one of the fields the in corresponding struct changes value.

When receiving the event, the client should call `OMX_GetParameter()` or `OMX_GetConfig()` as appropriate to get the new settings.

### 1.3.2 Index definition

To avoid backwards compatibility problems, callbacks are always disabled when a component is first created. To enable or disable callbacks during the component lifecycle, the IL client uses a new extension index `OMX_IndexConfigCallbackRequest`, with a corresponding data structure `OMX_CONFIG_CALLBACKREQUESTTYPE`.

OpenMAX IL Index (in <code>OMX_IndexExt.h</code> )	Corresponding OpenMAX IL Structures
<code>OMX_IndexConfigCallbackRequest</code>	<code>OMX_CONFIG_CALLBACKREQUESTTYPE</code>

### 1.3.3 Data Structure Definition

`OMX_CONFIG_CALLBACKREQUESTTYPE` structure is used for enabling and disabling callbacks using the `OMX_EventIndexSettingChanged` event.

`OMX_CONFIG_CALLBACKREQUESTTYPE` is defined in `OMX_CoreExt.h` as follows.

```
typedef struct OMX_CONFIG_CALLBACKREQUESTTYPE {
    OMX_U32 nSize;
    OMX_VERSIONTYPE nVersion;
    OMX_U32 nPortIndex;
    OMX_INDEXTYPE nIndex;
    OMX_BOOL bEnable;
} OMX_CONFIG_CALLBACKREQUESTTYPE;
```

### 1.3.3.1 Parameters

The parameters for `OMX_CONFIG_CALLBACKREQUESTTYPE` are defined as follows.

- `nPortIndex` is the port the setting applies to (can be `OMX_ALL`).
- `nIndex` is the index the client changes the callback setting for.
- `bEnable` either enables (`OMX_TRUE`) or disables (`OMX_FALSE`) the callbacks; the default value is `OMX_FALSE`.

### 1.3.3.2 Functionality

Callback settings can be changed and queried using the standard `OMX_SetConfig()` and `OMX_GetConfig()` functions, respectively. Callback settings are fully independent of any other settings on the component, including component state.

Component implementations shall fail the `OMX_SetConfig()` call for `OMX_IndexConfigCallbackRequest` with `OMX_ErrorUnsupportedIndex` if it does not support callbacks at all, and with `OMX_ErrorUnsupportedSetting` if it does not support callbacks for the particular index, but supports callbacks for some other settings.