



**PV2Way Communications I/O Development Guide**  
**OpenCORE 2.0, rev. 1**  
**Jan 22, 2009**

---



## Table of Contents

|   |          |
|---|----------|
| <b>1. Introduction.....</b>                             | <b>4</b> |
| <b>2. PVCommsIOnode .....</b>                           | <b>4</b> |
| 2.1. Two MIO Components / Two Ports .....               | 4        |
| 2.2. One MIO Component / One Port .....                 | 5        |
| 2.3. Two MIO Components / One Port.....                 | 6        |
| 2.4. One MIO Component / Two Ports.....                 | 7        |
| <b>3. Implementing Baseband Comm IO components.....</b> | <b>7</b> |
| 3.1. PvmiMIOControl Interface.....                      | 7        |
| 3.2. PvmiMediaTransfer and Data Transfer Models.....    | 7        |
| 3.3. PvmiCapabilityAndConfig Interface.....             | 8        |

## List of Figures

|   |   |
|---|---|
| Figure 1: Two MIO Components / Two Ports..... | 4 |
| Figure 2: One MIO Component / One Port .....  | 5 |
| Figure 3: Two MIO Components / One Port.....  | 6 |
| Figure 4: One MIO Component / Two Ports.....  | 7 |

## 1. Introduction

This document establishes guidelines for developing baseband communications I/O components to work with the PV2Way engine's PVCommsIONode. The PVCommsIONode serves to abstract the details (media transfer model, unidirectional/bidirectional operation, etc.) of device specific baseband communications from the PV2Way engine. A knowledge of the interfaces detailed in the reference documents is implied.

## 2. PVCommsIONode

The PVCommsIONode is designed to abstract device specific baseband communication details from the rest of the PV2Way Engine, and allow flexibility for both baseband side and PV2Way Engine side sending and receiving of data. Below are the following use case scenarios for the PVCommsIONode:

### 2.1. Two MIO Components / Two Ports

Scenario: Two unidirectional baseband components, two unidirectional ports

The PVCommsIONode establishes a MediaDataTransfer session with each baseband component. Data flowing from the baseband input component is relayed to the output port, and data flowing from the input port is relayed to the baseband media output component.

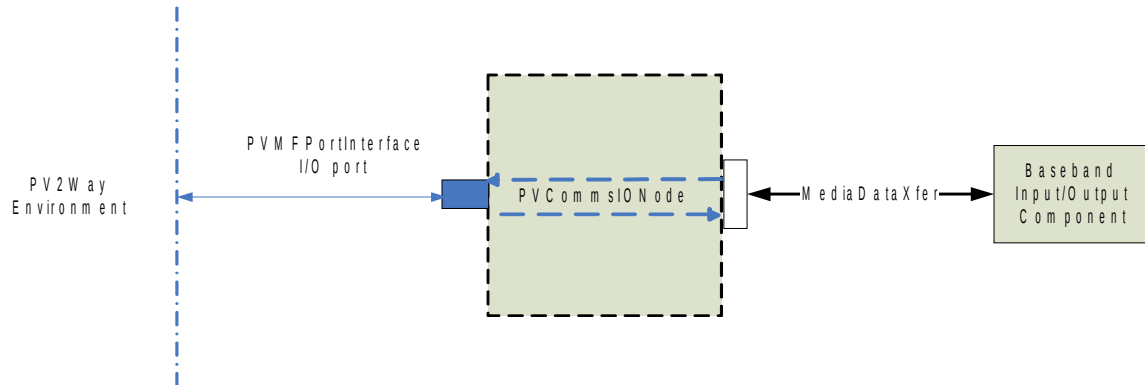


Figure 1: Two MIO Components / Two Ports

## 2.2. One MIO Component / One Port

Scenario: One bidirectional baseband component, one bidirectional port

The PVComm sIONode establishes a MediaDataTransfer session with a bidirectional baseband component. Data flowing from the baseband component is relayed and send out over the port interface, and data flowing in from the port interface is relayed to the baseband component.

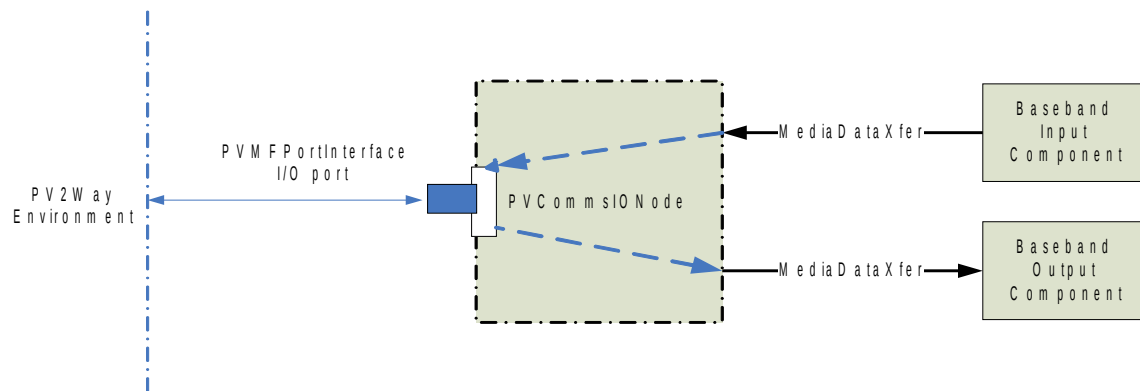


**Figure 2: One MIO Component / One Port**

## 2.3. Two MIO Components / One Port

Scenario: Two unidirectional baseband components, one bidirectional port

The PVCommsIONode establishes a MediaDataTransfer session with each baseband component. Data flowing from the baseband input component is relayed to the port, and sent out over the port interface, and data flowing in from the port interface is relayed to the baseband media output component.

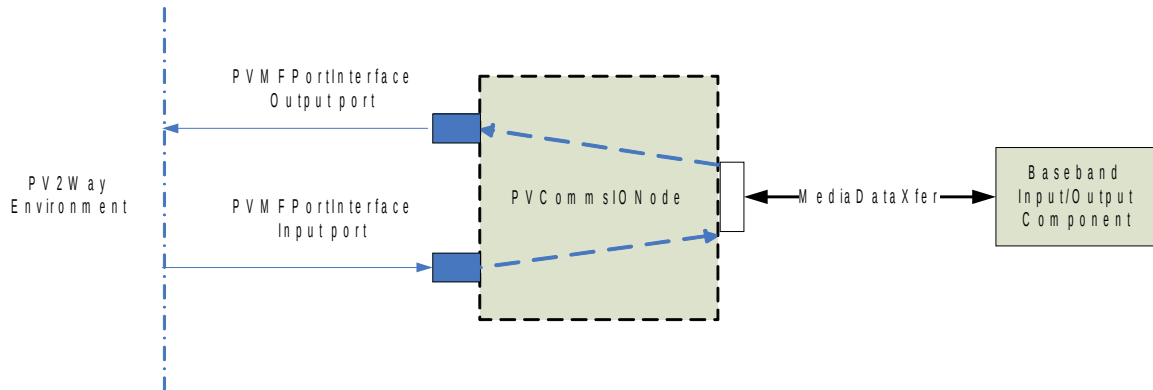


**Figure 3: Two MIO Components / One Port**

## 2.4. One MIO Component / Two Ports

Scenario 1: One bidirectional baseband component, two unidirectional ports

The PVCommsIONode establishes a MediaDataTransfer session with a bidirectional baseband component. Data flowing from the baseband component is relayed to the output port, and data flowing from the input port is relayed to the baseband component.



**Figure 4: One MIO Component / Two Ports**

An instance of PVCommsIONode should be created using the CPVCommsIONodeFactory class as detailed in the PV2Way API document.

## 3. Implementing Baseband Comm IO components

### 3.1. PvmiMIOControl Interface

All baseband comms components must implement the PvmiMIOControl interface. This provides a framework for the PVCommsIONode to programmatically start, stop and otherwise control the component. The component should return an implementation of the PvmiMediaTransfer interface via its ::CreateMediaTransfer() method. This PvmiMediaTransfer implementation is responsible for managing the actual transmission and reception of data.

### 3.2. PvmiMediaTransfer and Data Transfer Models

Currently, the PVCommsIONode supports the data push model for both input and output data. The PVCommsIONode is responsible for pushing output data to the baseband component, and the baseband component is responsible for pushing input data to the

PVCommsIONode. As this relates to the PvmiMediaTransfer interface, the baseband component should push input data by calling its peer's (PVCommsIONode) writeAsync() method. Accordingly, the PVCommsIONode should push output data to the baseband component by calling its peer's (baseband component) writeAsync() method.

Threading models for target environments may dictate that a pull model, requiring the use of the readAsync() methods, be used for data transfer and future support for this is planned.

### 3.3. PvmiCapabilityAndConfig Interface

All Baseband MIO components must implement the PvmiCapabilityAndConfig interface and expose a basic set of values that can be retrieved by a peer. At a minimum, a peer should be able to retrieve, and if necessary, set and enumerate values for the following keys:

|  |  |
|--|--|
| .../input_formats;valtype=int32          | If the component supports media output, it should allow a query on the current value for this key, and if more than one format is supported, it should allow enumeration and setting of this key as well. Format types are located in the file "pvmf_format_types.h". For H.324, PVMF_H223 format must be supported. |
| .../output_formats;valtype=int32         | If the component supports media input, it should allow a query on the current value for this key, and if more than one format is supported, it should allow enumeration and setting of this key as well. Format types are located in the file "pvmf_format_types.h". For H.324, PVMF_H223 format must be supported.  |
| .../input/transfer_model;valtype=uint32  | If the component supports media output, it should allow a query on the current value for this key. The key should be read-only, and a value of 0 indicates the component uses a data pull model for output data, 1 for data push. Currently, the PVCommsIONode requires that this value be 1.                        |
| .../output/transfer_model;valtype=uint32 | If the component supports media input, it should allow a query on the current value for this key. The key should be read-only, and a value of 0 indicates the component uses a data pull model for input data, 1 for data push. Currently, the PVCommsIONode requires that this value be 1.                          |