ADLINK Vision Software (AVS) for EOS-JNX Series

Function Reference Manual

Manual Revision: 1.0

Revision Date: February 17, 2022

Part No: 50M-00066-1000

LEADING EDGE COMPUTING

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Revision History

Revision	Release Date	Description of Change(s)
1.0	2022-02-17	Initial release

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Preface

Copyright © 2022 ADLINK Technology, Inc.

This document contains proprietary information protected by copyright. All rights are reserved. No part of this manual may be reproduced by any mechanical, electronic, or other means in any form without prior written permission of the manufacturer.

Disclaimer

The information in this document is subject to change without prior notice in order to improve reliability, design, and function and does not represent a commitment on the part of the manufacturer.

In no event will the manufacturer be liable for direct, indirect, special, incidental, or consequential damages arising out of the use or inability to use the product or documentation, even if advised of the possibility of such damages.

Environmental Responsibility

ADLINK is committed to fulfill its social responsibility to global environmental preservation through compliance with the European Union's Restriction of Hazardous Substances (RoHS) directive and Waste Electrical and Electronic Equipment (WEEE) directive. Environmental Protection is a top priority for ADLINK. We have enforced measures to ensure that our products, manufacturing processes, components, and raw materials have as little impact on the environment as possible. When products are at their end of life, our customers are encouraged to dispose of them in accordance with the product disposal and/or recovery programs prescribed by their nation or company.

Trademarks

Product names mentioned herein are used for identification purposes only and may be trademarks and/or registered trademarks of their respective companies.

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Conventions

Take note of the following conventions used throughout this manual to make sure that users perform certain tasks and instructions properly.



Additional information, aids, and tips that help users perform tasks.



Information to prevent ***minor*** physical injury, component damage, data loss, and/or program corruption when trying to complete a task.



Information to prevent ***serious*** physical injury, component damage, data loss, and/or program corruption when trying to complete a specific task.

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Table of Contents

Revision History ii

Preface iii

1 Introduction 1
- 1.1 Environment 1
- 1.2 AVS Installation 1
- 1.3 AVS Functions & Parameters 1
- 1.4 AVS Sample Programs 1
- 1.5 AVS Installation Folder 2
- 1.6 Sample Reference Code 2
- 1.6.1 Include Header File 2
- 1.6.2 Get Product Handle 3

2 AVS Parameters Reference 5
- 2.1 Product Name 5
- 2.2 Product Features 5
- 2.3 Edge Trigger 5
- 2.4 PoE 6
- 2.5 PoE Power Management 7
- 2.6 DIO Capabilities 7
- 2.7 DI Capabilities 8
- 2.8 DO Capabilities 8
- 2.9 Signal Edge Trigger Type 8
- 2.10 AVS Error Code 9

3 Function Library 11
- 3.1 List of Functions 11
- 3.2 Function Library 12
- 3.2.1 Common 12
- 3.2.2 Power over Ethernet 17
- 3.2.3 Digital Input/Output 24

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Important Safety Instructions..... 33

Getting Service 37

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

1 Introduction

ADLINK Vision Software (AVS) is the application programming interface (API) for the ADLINK EOS-JNX series of products. AVS allows developers to build applications in Linux via C/C++. Along with the APIs, AVS also includes product drivers and a set of sample programs and documentation.

1.1 Environment

Platform	NVIDIA Carmel ARM® v8.2 64-bit CPU
OS	Linux: Ubuntu 18.04 and above
Programming Language	Linux: C/C++

1.2 AVS Installation

Linux	Debian, DKMS are a must for AVS. $sudo apt-get install dkms $tar xvf **AVS-SDK_xxx.tar** $cd **AVS-SDK_xxx** $sudo dpkg -i AVS_arm64.deb

1.3 AVS Functions & Parameters

AVS Function	Description	Installation Folder
General SDK functions	Retrieves product handle and device information	\AVS_SDK.h
Feature functions	Basic functions	\Feature*.h
AVS Parameter Reference	AVS parameters are grouped by C/C++ enumerations	\Common\I_common.h

1.4 AVS Sample Programs

Type	Description
Console samples	Applications that can be run from a command prompt in Linux.

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

1.5 AVS Installation Folder

Type	Installation Folder	File	Description
Include	/usr/local/include/ADLINK/AVS	AVS_SDK.h	Header file required for General SDK functions
	/usr/local/include/ADLINK/AVS/Common/	I_common.h	Header file required for AVS parameters
	/usr/local/include/ADLINK/AVS/Feature/	PoE.h DIO.h	Header file required for basic functions
Library	/usr/local/lib/	libAVS_SDK.so libAVS_Product.so libAVS_CoreSDK.so	Exports API function definitions, required for C/C++ applications
Sample Program	/usr/local/src/ADLINK/AVS/Sample	Build Sample $cd /usr/local/src/ADLINK/AVS/Sample $make Execute Sample $cd /usr/local/src/ ADLINK/AVS/Sample/bin $sudo ./AVS_GetSDKInfo	

1.6 Sample Reference Code

1.6.1 Include Header File

This code sample shows how to include a header file.

```
#include "/usr/local/include/ADLINK/AVS/
     AVS_SDK.h"
#include "/usr/local/include/ADLINK/AVS/Common/
     I_common.h"
```

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

1.6.2 Get Product Handle

This code sample shows how to get the product handle and product Information.

```
using namespace std;
/*Show SDK Information*/
char buffer[256];
if (AVS_GetSDKInfo(buffer, sizeof(buffer)) ==
     AVS_FUN_SUCCESS)
cout << buffer << endl;
...
/*Show Product Information*/
unsigned int nums = AVS_GetProductNums();
if (nums <= 0)
{
     cout << "Find no device!" << endl;
     return 0;
}

void* handle;
unsigned int devIndex = 0;
handle = AVS_GetProductHandle(devIndex);
if (!handle)
{
     cout << "AVS_GetProductHandle Error\n" <<
     endl;
     return 0;
}

if (AVS_GetProductInfo(handle, buffer,
     sizeof(buffer)) == AVS_FUN_SUCCESS)
     cout << buffer << endl;
...
```

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

This page intentionally left blank.

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

2 AVS Parameters Reference

AVS parameters are grouped by enumerations and can be found in the header file “l_common”.

2.1 Product Name

ADLINK vision products supported by AVS SDK are grouped by **AVS_PRODUCT_TYPE**.

String Identifier	Product Name
AVS_PRODUCT_UNKNOWN	Unknown product
AVS_PRODUCT_EOS-JNX	EOS-JNX

2.2 Product Features

AVS parameters are grouped by **AVS_FEATURE_TYPE**.

String Identifier	Product Feature
AVS_FEATURE_POE	Power over Ethernet (PoE)
AVS_FEATURE_DIO	Digital Input / Output
AVS_FEATURE_NUMS	The number of the product feature

2.3 Edge Trigger

Edge Trigger parameters are defined by **AVS_ EDGETRIGGER_TYPE**.

String Identifier	Parameter
AVS_EDGETRIGGER_RISINGEDGE	Signal rising edge trigger
AVS_EDGETRIGGER_FALLINGEDGE	Signal falling edge trigger
AVS_EDGETRIGGER_CHANGINGEDGE	Signal changing edge trigger
AVS_EDGETRIGGER_NUMS	The number of the edge trigger type

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

2.4 PoE

PoE parameters are defined by **AVS_POEPROPERTY_TYPE**.

String Identifier	Parameter
AVS_POEPROPERTY_POEINITIALSTATE	Power sourcing equipment (PSE) initial power
AVS_POEPROPERTY_OVERTEMPERATURE	Check if the device temperature is over the high temperature (degrees C)
AVS_POEPROPERTY_CURRENTTEMPERATURE	Get the current temperature
AVS_POEPROPERTY_CONSUMEPOWER	The consumer power (Watt)
AVS_POEPROPERTY_REMAINPOWER	The left power (Watt)
AVS_POEPROPERTY_POWERBUDGET	The total power budget (Watt)
AVS_POEPROPERTY_NUMS	The number of the PoE property

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

2.5 PoE Power Management

PoE Power Management parameters are defined by **AVS_POEPORTPROPERTY_TYPE**.

String Identifier	Parameter
AVS_POEPORTPROPERTY_CURRENT	The current of a port (A)
AVS_POEPORTPROPERTY_VOLT	The voltage of a port (V)
AVS_POEPORTPROPERTY_POECLASS	The power class of a port
AVS_POEPORTPROPERTY_POWERGOOD	Check if the power is good on a port
AVS_POEPORTPROPERTY_LASTPOWERSTATE	Retrieve the last power state of a port. If the PoE status of a port is currently off, this function is used to retrieve the port's last PoE status, which is used to determine the PoE cut off status.
AVS_POEPORTPROPERTY_NUMS	The number of the PoE port properties

2.6 DIO Capabilities

DIO Capability parameters are defined by **AVS_DIOCAPABILITY_TYPE**.

String Identifier	Parameter
AVS_DIOCAPABILITY_DI	Support DI
AVS_DIOCAPABILITY_DO	Support DO
AVS_DIOCAPABILITY_NUMS	The number of the DIO capabilities

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

2.7 DI Capabilities

DI Capability parameters are defined by **AVS_DIODICAPABILITY_TYPE**.

String Identifier	Parameter
AVS_DIODICAPABILITY_GET	Support get state
AVS_DIODICAPABILITY_TRIGGERDETECTION	Support detect DI hardware trigger
AVS_DIODICAPABILITY_EDGETIRGGERTYPE	Support set/get DI edge trigger type
AVS_DIODICAPABILITY_NUMS	The number of the DIO's DI capabilities

2.8 DO Capabilities

DO Capability parameters are defined by **AVS_DIODOCAPABILITY_TYPE**.

String Identifier	Parameter
AVS_DIODOCAPABILITY_SET	Support set state.
AVS_DIODOCAPABILITY_NUMS	The number of the DIO's DO capabilities

2.9 Signal Edge Trigger Type

Signal Edge Trigger Type parameters are defined by **AVS_EDGETRIGGER_TYPE**.

String Identifier	Parameter
AVS_EDGETRIGGER_RISINGEDGE	Rising edge trigger
AVS_EDGETRIGGER_FALLINGEDGE	Falling edge trigger
AVS_EDGETRIGGER_CHANGINGEDGE	Falling and Rising edge triggers
AVS_EDGETRIGGER_NUMS	The number of the edge-triggered definitions

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

2.10 AVS Error Code

An AVS error code is defined by **AVS_FUN_RETURNVAL**. No error occurs if the API returns a value >= 0, or a negative value.

Error Code	Definition	Description
0	AVS_FUN_SUCCESS	Execution successful
-1	AVS_FUN_FAILED	Execution failed
-5	AVS_FUN_NOTDEFINED	Unsupported function
-11	AVS_FUN_INVALIDPARA	The parameters are invalid
-13	AVS_FUN_NOTINIT	The object is not initialized
-100	AVS_FUN_BUSY	The system is busy. The device has timed out, or is not currently able to reply.

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

This page intentionally left blank.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

3 Function Library

3.1 List of Functions

Category (Header File)	Function
Common (AVS_SDK.h)	AVS_GetProductNums()
	AVS_GetSDKInfo()
	AVS_GetProductHandle()
	AVS_GetProductInfo()
	AVS_GetProductType()
	AVS_GetProductSN()
	AVS_GetFeatureList()
Power over Ethernet (PoE.h)	AVS_PoEGetPortNums()
	AVS_PoESetPortEnable()
	AVS_PoEGetPortEnable()
	AVS_PoESetPowConsumCalcModel()
	AVS_PoEGetPowConsumCalcModel()
	AVS_PoESetProtectTempRange()
	AVS_PoEGetProtectTempRange()
	AVS_PoEGetProperty()
	AVS_PoEGetPortProperty()
Digital Input/Output (DIO.h)	AVS_DIOGetCapability()
	AVS_DIOGetDINums()
	AVS_DIOGetDONums()
	AVS_DIOGetDICapability()
	AVS_DIOGetDOCapability()
	AVS_DIOSetDOState()
	AVS_DIOSetDOStates()
	AVS_DIOGetDIState()
	AVS_DIOGetDIStates()
	AVS_DIOSetDITriggerCallback()
	AVS_DIOSetDIEdgeTrigger()
	AVS_DIOGetDIEdgeTrigger()

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

3.2 Function Library

AVS supports the C/C++ programming languages. The following are examples of the C/C++ library.

3.2.1 Common

These functions are used to get the product handle and retrieve device information.

3.2.1.1 AVS_GetProductNums

Function Name	AVS_GetProductNums
Purpose	This function returns the total number of products installed in the system.
Category	Common
Header File	AVS_SDK.h
Syntax (C/C++)	unsigned int AVS_GetProductNums()
Parameter(s)	
Return Code	
Returns the total number of devices if return value >= 0. See **Chapter 2 AVS Parameters Reference** for more information about return codes.	

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

3.2.1.2 AVS_GetSDKInfo

Function Name	AVS_GetSDKInfo
Purpose	This function returns the SDK version information (e.g., SDK version)
Category	Common
Header File	AVS_SDK.h
Syntax (C/C++)	int AVS_GetSDKInfo(char* data, unsigned int bufsize)
Parameter(s)	
data: A string buffer indicating the parameters and information provided. Each parameter and information is split by a break line character '\n'. The format is **Parameter Name**:Parameter Information '\n' For example: **SDK version**:1.3.6.0121 **bufsize**: Indicates the size of the "data" buffer. The suggested value is 256.	
Return Code	
Returns the total number of devices if return value >= 0. See **Chapter 2 AVS Parameters Reference** for more information about return codes.	

3.2.1.3 AVS_GetProductHandle

Function Name	AVS_GetProductHandle
Purpose	Initializes the specified device and gets its handle. This should be called before other functions except those with no handle parameter.
Category	Common
Header File	AVS_SDK.h
Syntax (C/C++)	void* AVS_GetProductHandle(unsigned int index)
Parameter(s)	
index: Indicates the number of devices, beginning at 0 for the number of the first card, with the second card 1, and so on. This value should be less than (total numbers of the cards -1). The total number of the cards is acquired from AVS_GetProductNums.	
Return Code	
No error occurs if the return value >= 0, or a negative value. See **Chapter 2 AVS Parameters Reference** for more information about return codes.	

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

3.2.1.4 AVS_GetProductInfo

Function Name	AVS_GetProductInfo
Purpose	This function returns all the product information. (e.g., product number)
Category	Common
Header File	AVS_SDK.h
Syntax (C/C++)	int AVS_GetProductInfo(void* handle, char* data, unsigned int bufsize)
Parameter(s)	
handle: Acquired from the return value of AVS_GetProductHandle. **data**: A string buffer indicating the parameters and information provided. Each parameter and information is split by a break line character '\n'. The format is **Parameter Name**:Parameter Information '\n' For example: **Product**:EOS-JNX **MCUVersion**:2.3 **PN**:91-64216-000E **SN**:0123456789 **PCBVersion**:1 **Bufsize**: Indicates the size of the "data" buffer. The suggested value is 256.	
Return Code	
No error occurs if the return value >= 0, or a negative value. See **Chapter 2 AVS Parameters Reference** for more information about return codes.	

3.2.1.5 AVS_GetProductType

Function Name	AVS_GetProductType
Purpose	Get the product type.
Category	Common
Header File	AVS_SDK.h
Syntax (C/C++)	int AVS_GetProductType(void* handle)
Parameter(s)	
handle: Acquired from the return value of AVS_GetProductHandle.	
Return Code	
Returns AVS_PRODUCT_TYPE. No error occurs if the return value >= 0, or a negative value. See **Chapter 2 AVS Parameters Reference** for more information about return codes.	

3.2.1.6 AVS_GetProductSN

Function Name	AVS_GetProductSN
Purpose	This function gets the product serial number (SN).
Category	Common
Header File	AVS_SDK.h
Syntax (C/C++)	int AVS_GetProductSN(void* handle, char* data, unsigned int bufsize)
Parameter(s)	
handle: Acquired from the return value of AVS_GetProductHandle.	
data: Pointer of an allocated character buffer into the function copies the string.	
Bufsize: Indicates the size of the “data” buffer. The suggested value is 32.	
Return Code	
No error occurs if the return value >= 0, or a negative value. See **Chapter 2 AVS Parameters Reference** for more information about return codes.	

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

3.2.1.7 AVS_GetFeatureList

Function Name	AVS_GetFeatureList
Purpose	This function gets the product feature list.
Category	Common
Header File	AVS_SDK.h
Syntax (C/C++)	int AVS_GetFeatureList(void* handle, AVS_FEATURE_TYPE* list, unsigned int listsize)
Parameter(s)	
handle: Acquired from the return value of AVS_GetProductHandle. **list**: Pointer of an allocated buffer into the function copies the string. **listsize**: Indicates the size of the “list” buffer. The suggested value is AVS_FEATURE_NUMS (AVS_FEATURE_NUMS is definded in I_common.h).	
Return Code	
No error occurs if the return value >= 0, or a negative value. See **Chapter 2 AVS Parameters Reference** for more information about return codes.	

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

3.2.2 Power over Ethernet

This feature is for SmartPoE, the PoE power management functions.

3.2.2.1 AVS_PoEGetPortNums

Function Name	AVS_PoEGetPortNums
Purpose	Retrieve the number of the power over Ethernet.
Category	Power over Ethernet
Header File	PoE.h
Syntax (C/C++)	int AVS_PoEGetPortNums(void* handle, unsigned int* nums)
Parameter(s)	
handle: Acquired from the return value of AVS_GetProductHandle. **nums**: Return the number of the power over Ethernet.	
Return Code	
No error occurs if the return value >= 0, or a negative value. See **Chapter 2 AVS Parameters Reference** for more information about return codes.	

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

3.2.2.2 AVS_PoESetPortEnable

Function Name	AVS_PoESetPortEnable
Purpose	Controls power auto/off.
Category	Power over Ethernet
Header File	PoE.h
Syntax (C/C++)	int AVS_PoESetPortEnable(void* handle, int* list, unsigned int listsize)
Parameter(s)	
handle: Acquired from the return value of AVS_GetProductHandle. **list**: An array for setting the power status of the ports. Assign a list to enable or disable each port. * 0 to disable power or other to enable power. **listsize**: The array must contain nums elements (nums is acquired from the return value of AVS_PoEGetPortNums).	
Return Code	
No error occurs if the return value >= 0, or a negative value. See **Chapter 2 AVS Parameters Reference** for more information about return codes.	

3.2.2.3 AVS_PoEGetPortEnable

Function Name	AVS_PoEGetPortEnable
Purpose	Retrieves PoE state.
Category	Power over Ethernet
Header File	PoE.h
Syntax (C/C++)	int AVS_PoEGetPortEnable(void* handle, int* list, unsigned int listsize)
Parameter(s)	
handle: Acquired from the return value of AVS_GetProductHandle. **list**: An array for receiving the power status of the ports. **listsize**: The array must contain nums elements (nums is acquired from the return value of AVS_PoEGetPortNums).	
Return Code	
No error occurs if the return value >= 0, or a negative value. See **Chapter 2 AVS Parameters Reference** for more information about return codes.	

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

3.2.2.4 AVS_PoESetPowConsumCalcModel

Function Name	AVS_PoESetPowConsumCalcModel
Purpose	Controls power budget mode.
Category	Power over Ethernet
Header File	PoE.h
Syntax (C/C++)	int AVS_PoESetPowConsumCalcModel(void* handle, int usePowClassBudget)
Parameter(s)	
handle: Acquired from the return value of AVS_GetProductHandle. **usePowClassBudget**: 0: calculate the Consumed Power Budget by PoE power consumption. 1: calculate the Consumed Power Budget by PoE Class.	
Return Code	
No error occurs if the return value >= 0, or a negative value. See **Chapter 2 AVS Parameters Reference** for more information about return codes.	

3.2.2.5 AVS_PoEGetPowConsumCalcModel

Function Name	AVS_PoEGetPowConsumCalcModel
Purpose	Retrieves power budget mode.
Category	Power over Ethernet
Header File	PoE.h
Syntax (C/C++)	int AVS_PoEGetPowConsumCalcModel(void* handle, int* usePowClassBudget)
Parameter(s)	
handle: Acquired from the return value of AVS_GetProductHandle. **usePowClassBudget**: 0: calculate the Consumed Power Budget by PoE power consumption. 1: calculate the Consumed Power Budget by PoE Class.	
Return Code	
No error occurs if the return value >= 0, or a negative value. See **Chapter 2 AVS Parameters Reference** for more information about return codes.	

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

3.2.2.6 AVS_PoESetProtectTempRange

Function Name	AVS_PoESetProtectTempRange
Purpose	Controls High Temperature (HT) and Recovery Temperature (RT). When HT is reached, automatically shuts down PoE power until the temperature sensor falls below RT.
Category	Power over Ethernet
Header File	PoE.h
Syntax (C/C++)	int AVS_PoESetProtectTempRange(void* handle, int high_temperature, int recovery_temperature)
Parameter(s)	
handle: Acquired from the return value of AVS_GetProductHandle. **high_temperature**: High temperature setting, more than or equal to 80°C, not exceeding 130°C, and at least 10°C higher than the Recovery Temperature setting. **recovery_temperature**: Recovery temperature setting, equal to or exceeding 70°C and at least 10°C lower than the high_temperature setting.	
Return Code	
No error occurs if the return value >= 0, or a negative value. See **Chapter 2 AVS Parameters Reference** for more information about return codes.	

3.2.2.7 AVS_PoEGetProtectTempRange

Function Name	AVS_PoEGetProtectTempRange
Purpose	Retrieves of High Temperature and Recovery Temperature.
Category	Power over Ethernet
Header File	PoE.h
Syntax (C/C++)	int AVS_PoEGetProtectTempRange(void* handle, int* high_temperature, int* recovery_temperature)
Parameter(s)	
handle: Acquired from the return value of AVS_GetProductHandle. **high_temperature**: High Temperature setting. **recovery_temperature**: Recovery Temperature setting.	
Return Code	
No error occurs if the return value >= 0, or a negative value. See **Chapter 2 AVS Parameters Reference** for more information about return codes.	

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

3.2.2.8 AVS_PoEGetProperty

Function Name	AVS_PoEGetProperty
Purpose	Retrieves PoE properties.
Category	Power over Ethernet
Header File	PoE.h
Syntax (C/C++)	int AVS_PoEGetProperty(void* handle, AVS_POEPROPERTY_TYPE property, float* value)
Parameter(s)	
handle: Acquired from the return value of AVS_GetProductHandle. **property & value**: Refer to the following table.	
Return Code	
No error occurs if the return value >= 0, or a negative value. See **Chapter 2 AVS Parameters Reference** for more information about return codes.	

Property	**Value**
AVS_POEPROPERTY_POEINITIALSTATE	0: PoE-PSE should be turned off when powered on 1: PoE-PSE should be turned on when powered on
AVS_POEPROPERTY_OVERTEMPERATURE	0: no interrupt 1: over temperature interrupt arise
AVS_POEPROPERTY_CURRENTTEMPERATURE	Thermal temperature in °C.
AVS_POEPROPERTY_CONSUMEPOWER	Consumed power budget in W.
AVS_POEPROPERTY_REMAINPOWER	Remaining available power in W.
AVS_POEPROPERTY_POWERBUDGET	Total Budget in W.
AVS_POEPROPERTY_NUMS	

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

3.2.2.9 AVS_PoEGetPortProperty

Function Name	AVS_PoEGetPortProperty
Purpose	Retrieves PoE port properties.
Category	Power over Ethernet
Header File	PoE.h
Syntax (C/C++)	int AVS_PoEGetPortProperty(void* handle, unsigned int port, AVS_POEPORTPROPERTY_TYPE property, float* value)
Parameter(s)	
handle: Acquired from the return value of AVS_GetProductHandle. **port**: Indicates port number from 0 to N-1. (N is the number of port) **property & value**: Refer to the following table.	
Return Code	
No error occurs if the return value >= 0, or a negative value. See **Chapter 2 AVS Parameters Reference** for more information about return codes.	

String Identifier	**Value**
AVS_POEPORTPROPERTY_CURRENT	Port PoE power current (unit: A)
AVS_POEPORTPROPERTY_VOLT	Port PoE power voltage (unit: V)
AVS_POEPORTPROPERTY_POECLASS	Port PoE Class
AVS_POEPORTPROPERTY_POWERGOOD	Port Power Good means the power is on and stable.
AVS_POEPORTPROPERTY_LASTPOWERSTATE	0: PoE power off 1: PoE power on

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

3.2.3 Digital Input/Output

The AVS SDK DIO feature is used to control and configure digital input and output. **AVS_DIOGetCapability()**, **AVS_DIOGetDICapability()** and **AVS_DIOGetDOCapability()** are used to check the supported functions of each product. The digital output can be a general digital output or a triggerable device in the AVS SDK.

3.2.3.1 AVS_DIOGetCapability

Function Name	AVS_DIOGetCapability
Purpose	Retrieves the DIO feature capability.
Category	Digital Input/Output
Header File	DIO.h
Syntax (C/C++)	int AVS_DIOGetCapability(void* handle, AVS_DIOCAPABILITY_TYPE* capList, unsigned int listsize)
Parameter(s)	
handle: Acquired from the return value of AVS_GetProductHandle. **caplist**: Assign a list to retrieve the capabilities. The value is defined by AVS_DIOCAPABILITY_TYPE. **listsize**: The number of list elements. The suggested list size is AVS_DIOCAPABILITY_NUMS.	
Return Code	
No error occurs if the return value >= 0, or a negative value. See **Chapter 2 AVS Parameters Reference** for more information about return codes.	

String Identifier	**Value**
AVS_DIOCAPABILITY_DI	Support DI
AVS_DIOCAPABILITY_DO	Support DO
AVS_DIOCAPABILITY_NUMS	Number of the DIO capabilities

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

3.2.3.2 AVS_DIOGetDINums

Function Name	AVS_DIOGetDINums
Purpose	Retrieves the number of the DI.
Category	Digital Input/Output
Header File	DIO.h
Syntax (C/C++)	int AVS_DIOGetDINums(void* handle, unsigned int* nums)
Parameter(s)	
handle: Acquired from the return value of AVS_GetProductHandle. **nums**: Returns the number of the DI.	
Return Code	
No error occurs if the return value >= 0, or a negative value. See **Chapter 2 AVS Parameters Reference** for more information about return codes.	

3.2.3.3 AVS_DIOGetDONums

Function Name	AVS_DIOGetDONums
Purpose	Retrieves the number of the DO.
Category	Digital Input/Output
Header File	DIO.h
Syntax (C/C++)	int AVS_DIOGetDONums(void* handle, unsigned int* nums)
Parameter(s)	
handle: Acquired from the return value of AVS_GetProductHandle. **nums**: Returns the number of the DO.	
Return Code	
No error occurs if the return value >= 0, or a negative value. See **Chapter 2 AVS Parameters Reference** for more information about return codes.	

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

3.2.3.4 AVS_DIOGetDICapability

Function Name	AVS_DIOGetDICapability
Purpose	Retrieves the DI Capability.
Category	Digital Input/Output
Header File	DIO.h
Syntax (C/C++)	int AVS_DIOGetDICapability(void* handle, unsigned int channel, AVS_DIODICAPABILITY_TYPE* capList, unsigned int listsize)
Parameter(s)	
handle: Acquired from the return value of AVS_GetProductHandle. **channel**: Assign a channel index **caplist**: Assign a list to retrieve the capabilities of each channel. The value is defined by AVS_DIODICAPABILITY_TYPE **listsize**: The number of list elements. The suggested list size is AVS_DIODICAPABILITY_NUMS.	
Return Code	
No error occurs if the return value >= 0, or a negative value. See **Chapter 2 AVS Parameters Reference** for more information about return codes.	

String Identifier	**Value**
AVS_DIODICAPABILITY_GET	Support get state ▶ AVS_DIOGetDIState() ▶ AVS_DIOGetDIStates()
AVS_DIODICAPABILITY_ TRIGGERDETECTION	Support detect DI hardware trigger ▶ AVS_DIOSetDITriggerCallback()
AVS_DIODICAPABILITY_ EDGETIRGGERTYPE	Support set/get DI edge trigger type ▶ AVS_DIOSetDIEdgeTrigger(), ▶ AVS_DIOGetDIEdgeTrigger()
AVS_DIODICAPABILITY_NUMS	The number of the DIO's DI capabilities

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

3.2.3.5 AVS_DIOGetDOCapability

Function Name	AVS_DIOGetDOCapability
Purpose	Retrieves the DO Capability.
Category	Digital Input/Output
Header File	DIO.h
Syntax (C/C++)	int AVS_DIOGetDOCapability(void* handle, unsigned int channel, AVS_DIODOCAPABILITY_TYPE* capList, unsigned int listsize)
Parameter(s)	
handle: Acquired from the return value of AVS_GetProductHandle. **channel**: Assign a channel index **caplist**: Assign a list to retrieve the capabilities of each channel. The value is defined by AVS_DIODOCAPABILITY_TYPE **listsize**: The number of list elements. The suggested list size is AVS_DIODOCAPABILITY_NUMS.	
Return Code	
No error occurs if the return value >= 0, or a negative value. See **Chapter 2 AVS Parameters Reference** for more information about return codes.	

String Identifier	**Value**
AVS_DIODOCAPABILITY_SET	Support set state ▶ AVS_DIOSetDOState() ▶ AVS_DIOSetDOStates()
AVS_DIODOCAPABILITY_NUMS	The number of the DIO's DO capabilities

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

3.2.3.6 AVS_DIOSetDOState

Function Name	AVS_DIOSetDOState
Purpose	Sets the DO state.
Category	Digital Input/Output
Header File	DIO.h
Syntax (C/C++)	int AVS_DIOSetDOState (void* handle, unsigned int channel, int state)
Parameter(s)	
handle: Acquired from the return value of AVS_GetProductHandle. **channel**: Assign a channel index **state**: The DO state should be 0 or 1.	
Return Code	
No error occurs if the return value >= 0, or a negative value. See **Chapter 2 AVS Parameters Reference** for more information about return codes.	

3.2.3.7 AVS_DIOSetDOStates

Function Name	AVS_DIOSetDOStates
Purpose	Sets all the DO states.
Category	Digital Input/Output
Header File	DIO.h
Syntax (C/C++)	int AVS_DIOSetDOStates(void* handle, int* statelist, unsigned int listsize)
Parameter(s)	
handle: Acquired from the return value of AVS_GetProductHandle. **statelist**: Assign a list to set all the DO states. The value is 0 or 1. **listsize**: The number of list elements. The suggested list size could be got from AVS_DIOGetDONums	
Return Code	
No error occurs if the return value >= 0, or a negative value. See **Chapter 2 AVS Parameters Reference** for more information about return codes.	

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

3.2.3.8 AVS_DIOGetDIState

Function Name	AVS_DIOGetDIState
Purpose	Gets the DI state.
Category	Digital Input/Output
Header File	DIO.h
Syntax (C/C++)	int AVS_DIOGetDIState(void* handle, unsigned int channel, int* state)
Parameter(s)	
handle: Acquired from the return value of AVS_GetProductHandle. **channel**: Assign a channel index **state**: The current DI state should be 0 or 1.	
Return Code	
No error occurs if the return value >= 0, or a negative value. See **Chapter 2 AVS Parameters Reference** for more information about return codes.	

3.2.3.9 AVS_DIOGetDIStates

Function Name	AVS_DIOGetDIStates
Purpose	Gets all the DI states.
Category	Digital Input/Output
Header File	DIO.h
Syntax (C/C++)	Int AVS_DIOGetDIStates(void* handle, int* statelist, unsigned int listsize)
Parameter(s)	
handle: Acquired from the return value of AVS_GetProductHandle. **statelist**: Assign a list to get all the DI states. The value is 0 or 1. **listsize**: The number of list elements. The suggested list size could be got from AVS_DIOGetDONums	
Return Code	
No error occurs if the return value >= 0, or a negative value. See **Chapter 2 AVS Parameters Reference** for more information about return codes.	

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

3.2.3.10 AVS_DIOSetDITriggerCallback

Function Name	AVS_DIOSetDITriggerCallback
Purpose	Sets the callback function. When DI is triggered, the function will be executed. The function may be supported when AVS_DIODICAPABILITY_TRIGGERDETECTION is in the list from AVS_DIOGetDICapability.
Category	Digital Input/Output
Header File	DIO.h
Syntax (C/C++)	int AVS_DIOSetDITriggerCallback(void* handle, AVS_DITRIGGER_CALLBACK callback)
Parameter(s)	
handle: Acquired from the return value of AVS_GetProductHandle. **callback**: The callback function. The function type is defined by AVS_DITRIGGER_CALLBACK. *void(*AVS_DITRIGGER_CALLBACK)(unsigned int index, int currentState)* index is the DI channel, currentState is the DI state when a trigger happened.	
Return Code	
No error occurs if the return value >= 0, or a negative value. See **Chapter 2 AVS Parameters Reference** for more information about return codes.	

3.2.3.11 AVS_DIOSetDIEdgeTrigger

Function Name	AVS_DIOSetDIEdgeTrigger
Purpose	Sets the edge trigger type of the DI.
Category	Digital Input/Output
Header File	DIO.h
Syntax (C/C++)	int AVS_DIOSetDIEdgeTrigger(void* handle, unsigned int channel, AVS_EDGETRIGGER_TYPE edgeTriggerType)
Parameter(s)	
handle: Acquired from the return value of AVS_GetProductHandle. **channel**: Assign a channel index **edgeTriggerType**: The edge trigger type. The value is defined by AVS_EDGETRIGGER_TYPE	
Return Code	
No error occurs if the return value >= 0, or a negative value. See **Chapter 2 AVS Parameters Reference** for more information about return codes.	

String Identifier	**Value**
AVS_EDGETRIGGER_RISINGEDGE	Signal rising edge trigger
AVS_EDGETRIGGER_FALLINGEDGE	Signal falling edge trigger
AVS_EDGETRIGGER_CHANGINGEDGE	Signal changing edge trigger

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

3.2.3.12 AVS_DIOGetDIEdgeTrigger

Function Name	AVS_DIOGetDIEdgeTrigger
Purpose	Gets the edge trigger type of the DI.
Category	Digital Input/Output
Header File	DIO.h
Syntax (C/C++)	int AVS_DIOGetDIEdgeTrigger(void* handle, unsigned int channel, AVS_EDGETRIGGER_TYPE* edgeTriggerType)
Parameter(s)	
handle: Acquired from the return value of AVS_GetProductHandle. **channel**: Assign a channel index **edgeTriggerType**: The edge trigger type. The value is defined by AVS_EDGETRIGGER_TYPE	
Return Code	
No error occurs if the return value >= 0, or a negative value. See **Chapter 2 AVS Parameters Reference** for more information about return codes.	

String Identifier	Value
AVS_EDGETRIGGER_RISINGEDGE	Signal rising edge trigger
AVS_EDGETRIGGER_FALLINGEDGE	Signal falling edge trigger
AVS_EDGETRIGGER_CHANGINGEDGE	Signal changing edge trigger

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Important Safety Instructions

For user safety, please read and follow all instructions, Warnings, Cautions, and Notes marked in this manual and on the associated device before handling/operating the device, to avoid injury or damage.

S'il vous plaît prêter attention stricte à tous les avertissements et mises en garde figurant sur l'appareil , pour éviter des blessures ou des dommages.

- Read these safety instructions carefully
- Keep the User's Manual for future reference
- Read the Specifications section of this manual for detailed information on the recommended operating environment
- The device can be operated at an ambient temperature of 50ºC
- When installing/mounting or uninstalling/removing device; or when removal of a chassis cover is required for user servicing:
  - Turn off power and unplug any power cords/cables
  - Reinstall all chassis covers before restoring power
- To avoid electrical shock and/or damage to device:
  - Keep device away from water or liquid sources
  - Keep device away from high heat or humidity
  - Keep device properly ventilated (do not block or cover ventilation openings)
  - Always use recommended voltage and power source settings
  - Always install and operate device near an easily accessible electrical outlet
  - Secure the power cord (do not place any object on/over the power cord)
  - Only install/attach and operate device on stable surfaces and/or recommended mountings
- If the device will not be used for long periods of time, turn off and unplug from its power source

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

- Never attempt to repair the device, which should only be serviced by qualified technical personnel using suitable tools
- A Lithium-type battery may be provided for uninterrupted backup or emergency power.



Risk of explosion if battery is replaced with one of an incorrect type; please dispose of used batteries appropriately.
Risque d'explosion si la pile est remplacée par une autre de type incorrect. Veuillez jeter les piles usagées de façon appropriée.

- The device must be serviced by authorized technicians when:
  - The power cord or plug is damaged
  - Liquid has entered the device interior
  - The device has been exposed to high humidity and/or moisture
  - The device is not functioning or does not function according to the User's Manual
  - The device has been dropped and/or damaged and/or shows obvious signs of breakage
- Disconnect the power supply cord before loosening the thumbscrews and always fasten the thumbscrews with a screwdriver before starting the system up
- It is recommended that the device be installed only in a server room or computer room where access is:
  - Restricted to qualified service personnel or users familiar with restrictions applied to the location, reasons therefor, and any precautions required
  - Only afforded by the use of a tool or lock and key, or other means of security, and controlled by the authority responsible for the location
- If PoE (Power over Ethernet) is enabled for the device, the system can ONLY be deployed indoors. Unless otherwise noted, the PoE system is NOT designed to withstand the rigors of outdoor use.

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

	**BURN HAZARD** Touching this surface could result in bodily injury. To reduce risk, allow the surface to cool before touching. ***RISQUE DE BRÛLURES*** *Ne touchez pas cette surface, cela pourrait entraîner des blessures.* *Pour éviter tout danger, laissez la surface refroidir avant de la toucher.*

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

This page intentionally left blank.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Getting Service

Ask an Expert: http://askanexpert.adlinktech.com

ADLINK Technology, Inc.
No. 66, Huaya 1st Rd., Guishan District
Taoyuan City 333411, Taiwan
Tel: +886-3-216-5088
Fax: +886-3-328-5706
Email: service@adlinktech.com

Ampro ADLINK Technology, Inc.
6450 Via Del Oro
San Jose, CA 95119-1208, USA
Tel: +1-408-360-0200
Toll Free: +1-800-966-5200 (USA only)
Fax: +1-408-600-1189
Email: info@adlinktech.com

ADLINK Technology (China) Co., Ltd.
300 Fang Chun Rd., Zhangjiang Hi-Tech Park
Pudong New Area, Shanghai, 201203 China
Tel: +86-21-5132-8988
Fax: +86-21-5132-3588
Email: market@adlinktech.com

ADLINK Technology GmbH
Hans-Thoma-Straße 11
D-68163 Mannheim, Germany
Tel: +49-621-43214-0
Fax: +49-621 43214-30
Email: emea@adlinktech.com

Please visit the Contact page at www.adlinktech.com for information on how to contact the ADLINK regional office nearest you:

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000