INGENIC®
T31 IVS API Reference

Date: 2022-04

北京君正集成电路股份有限公司
Ingenic Semiconductor Co., Ltd.

INGENIC®

T31 IVS API Reference

Release history

Date	Revision	Change
2022-04	1.0	First release

Disclaimer

This documentation is provided for use with Ingenic products. No license to Ingenic property rights is granted. Ingenic assumes no liability, provides no warranty either expressed or implied relating to the usage, or intellectual property right infringement except as provided for by Ingenic Terms and Conditions of Sale.

Ingenic products are not designed for and should not be used in any medical or life sustaining or supporting equipment.

All information in this document should be treated as preliminary. Ingenic may make changes to this document without notice. Anyone relying on this documentation should contact Ingenic for the current documentation and errata.

Ingenic Semiconductor Co. Ltd

Add: Junzheng R&D Center, Phase II, Zhongguancun Software Park, Dongbeiwangxi Road, Haidian District, Beijing, China

Tel: 86-10-56345000

Fax: 86-10-56345001

Http: //www.ingenic.com

Content

1 IVS Intelligent Analysis

1.1 Module Introduction

IMP IVS calls the instantiated impivsinterface through the IVS general interface API to embed the intelligent analysis algorithm into the SDK to analyze the frame image in the SDK.

IMPIVSInterface is a general algorithm interface. The specific algorithm implements this interface and transmits it to IMP IVS achieving the purpose of running specific algorithms in the SDK.

A channel has and is only the carrier of a single algorithm instance. The specific implementation of the general algorithm interface must be passed to the specific channel to run the algorithm in the SDK.

IMPIVSInterface member param is the parameter of member function init.

IMP_ IVS will externally lock the frame parameter when it is passed to the member function ProcessAsync parameter. ProcessAsync must call IMP_IVS_ReleaseData after frame is used ,Then release the frame to avoid deadlock.

In addition to the above algorithm call through imp IVS bound in the data flow， It also provides a non binding way to implement algorithm call.That is, obtain the frame image of framesource channel and directly call the member function of IMPIVSInterface to realize algorithm processing.

1.2 Application Method

1.2.1 Algorithm usage of binding mode

Take the motion detection algorithm as an example. See the sample-move_c.c file for the specific implementation of the function. The steps are as follows:

(1) Initialize the system and directly call the sample_system_init() in the samples .The entire application only needs to initialize the system once.

(2) Initialize framesource,If the framesource channel used by the algorithm has been created, you

can directly use the created channel; If not created, you can call the sample_framesource_init(IVS_FS_CHN, &fs_chn_attr) in samples.

(3) Create IVS specific algorithm channel group.Multiple algorithms can share a channel group or separate channels Use channel group sample_ ivs_ move_ init()。

(4) Bind algorithm channel group and framesource channel group.

(5) Start framesource and algorithm.It is recommended that the algorithm channel number be consistent with the algorithm number, so that it can directly correspond to which algorithm is currently operated.

(6) Get algorithm results、 polling results,The acquisition result and release result must be strictly corresponding without interruption; Only when the polling result returns correctly will the obtained result be updated, otherwise the obtained result cannot be predicted.

(7) Free resources.

1.2.2 Algorithm usage of unbound mode

Take the motion detection algorithm as an example, see sample-Framesource- Algo. c for the detailed implementation of the function. The steps are as follows:

（1）Initialize the system and directly call the sample_system_init() in the samples .
（2）Initialize framesource,If the framesource channel used by the algorithm has been created, you can directly use the created channel; If not created, you can call the sample_framesource_init(IVS_FS_CHN, &fs_chn_attr) in samples.
（3）Initialize the interface handle, call interface->init() to initialize the algorithm and call IMP_FrameSource_SetFrameDepth () sets the frame cache.
（4）IMP_FrameSource_GetFrame, interface->preProcessSync, interface->processAsync, Interface->getResult gets algorithm result, interface->releaseResult releases algorithm result, IMP_FrameSource_ReleaseFrame releases frame.
（5）IMP_FrameSource_SetFrameDepth Cache depth back to 0, release cache, de-initialize interface handle, de-initialize framesouce and system.

2 API

2.1 API Reference

API name	Functon
IMP_IVS_CreateGroup	Create channel group
IMP_IVS_DestroyGroup	Destroy channel group
IMP_IVS_CreateChn	Create channels corresponding to IVS functions
IMP_IVS_DestroyChn	Destroy the channel corresponding to the IVS function handle
IMP_IVS_RegisterChn	Register channel to channel group
IMP_IVS_UnRegisterChn	Channel group injection and elimination channel
IMP_IVS_StartRecvPic	Channel starts receiving images
IMP_IVS_StopRecvPic	Channel stops receiving images
IMP_IVS_PollingResult	Blocking judge whether the intelligent analysis results calculated by IVS function can be obtained
IMP_IVS_GetResult	Obtain the intelligent analysis results calculated by IVS function
IMP_IVS_ReleaseResult	Release the result resource calculated by IVS function
int IMP_IVS_ReleaseData	Release the frame parameter passed to Datacallback
int IMP_IVS_GetParam	Get channel algorithm parameters
IMP_IVS_SetParam	Set channel algorithm parameters

Table 2-1 API function table

2.1.1 IMP_IVS_CreateGroup

【Function】

Create channel group.

【Grammar】

int IMP_IVS_CreateGroup(int GrpNum);

【Formal parameter】

Parameter name	Describe	Input/Output
GrpNum	Channel group number corresponding to IVS function	Input

【Return value】

return 0 success； no 0 fail.

【Dependence】

Head file：imp_ivs.h

Lib file：libimp.a / libimp.so

【NB】

None.

2.1.2 IMP_IVS_DestroyGroup

【Function】

Destroy the channel.

【Grammar】

int IMP_IVS_DestroyGroup(int GrpNum);

【Formal parameter】

Parameter name	Describe	Input/Output
GrpNum	Channel group number corresponding to IVS function	Input

【Return value】

return 0 success； no 0 fail.

【Dependence】

Head file：imp_ivs.h

Lib file：libimp.a / libimp.so

【NB】

None.

2.1.3 IMP_IVS_CreateChn

【Function】

Create channels corresponding to IVS functions.

【Grammar】

int IMP_IVS_CreateChn(int ChnNum, IMPIVSInterface *handler);

【Formal parameter】

Parameter name	Describe	Input/Output
ChnNum	Channel number	Input
handler	IVS function handle	Input

【Return value】

return 0 success；no 0 fail.

【Dependence】

Head file：imp_ivs.h

Lib file：libimp.a / libimp.so

【NB】

None.

2.1.4 IMP_IVS_DestroyChn

【Function】

Destroy the channel corresponding to the IVS function handle.

【Grammar】

int IMP_IVS_DestroyChn(int ChnNum);

【Formal parameter】

Parameter name	Describe	Input/Output
ChnNum	Channel number	Input

【Return value】

return 0 success； no 0 fail.

【Dependence】

Head file：imp_ivs.h

Lib file：libimp.a / libimp.so

【NB】

None.

2.1.5 IMP_IVS_RegisterChn

【Function】

Register channel to channel group.

【Grammar】

int IMP_IVS_RegisterChn(int GrpNum, int ChnNum);

【Formal parameter】

Parameter name	Describe	Input/Output
ChnNum	Channel number	Input
GrpNum	Channel group number corresponding to IVS function	Input

【Return value】

return 0 success； no 0 fail.

【Dependence】

Head file：imp_ivs.h

Lib file：libimp.a / libimp.so

【NB】

None.

2.1.6 IMP_IVS_UnRegisterChn

【Function】

Channel group injection and elimination channel.

【Grammar】

int IMP_IVS_UnRegisterChn(int ChnNum);

【Formal parameter】

Parameter name	Describe	Input/Output
ChnNum	Channel number	Input

【Return value】

return 0 success；no 0 fail.

【Dependence】

Head file：imp_ivs.h

Lib file：libimp.a / libimp.so

【NB】

Unregister channel number chnnum from channel group number grpnum..

2.1.7 IMP_IVS_StartRecvPic

【Function】

Channel starts receiving images.

【Grammar】

int IMP_IVS_StartRecvPic(int ChnNum);

【Formal parameter】

Parameter name	Describe	Input/Output
ChnNum	Channel number	Input

【Return value】

return 0 success；no 0 fail.

【Dependence】

Head file：imp_ivs.h

Lib file：libimp.a / libimp.so

【NB】

IVS function channel with channel number chnnum starts to receive images for intelligent analysis.

2.1.8 IMP_IVS_StopRecvPic

【Function】

The channel stops receiving images.

【Grammar】

int IMP_IVS_StopRecvPic(int ChnNum);

【Formal parameter】

Parameter name	Describe	Input/Output
ChnNum	Channel number	Input

【Return value】

return 0 success；no 0 fail.

【Dependence】

Head file：imp_ivs.h

Lib file：libimp.a / libimp.so

【NB】

The IVS function channel with channel number chnnum stops receiving images and pauses intelligent analysis.

2.1.9 IMP_IVS_PollingResult

【Function】

Blocking judge whether the intelligent analysis results calculated by IVS function can be obtained.

【Grammar】

int IMP_IVS_PollingResult(int ChnNum, int timeoutMs);

【Formal parameter】

Parameter name	Describe	Input/Output
ChnNum	Channel number	Input
timeout	Maximum waiting time, unit: ms; IMP_ IVS_ DEFAULT_ Timeouts: default waiting time in the Library: 0, no waiting; > 0, user set waiting time	Input

【Return value】

return 0 success；no 0 fail.

【Dependence】

Head file：imp_ivs.h

Lib file：libimp.a / libimp.so

【NB】

This polling function returns success only when the processasync function member in the parameter impivsinterface structure returns 0 when the channel is created, that is when the actual detection returns normally.

2.1.10 IMP_IVS_GetResult

【Function】

Obtain the intelligent analysis results calculated by IVS function.

【Grammar】

int IMP_IVS_GetResult(int ChnNum, void **result);

【Formal parameter】

Parameter name	Describe	Input/Output
ChnNum	Channel number	Input
result	The output result of the channel number corresponding to the IVS function returns the result pointer of the intelligent analysis algorithm corresponding to this channel. External users do not need to allocate space	Input

【Return value】

return 0 success； no 0 fail.

【Dependence】

Head file：imp_ivs.h

Lib file：libimp.a / libimp.so

【NB】

Output the corresponding results according to the channels bound by different IVS functions.

2.1.11 IMP_IVS_ReleaseResult

【Function】

Release the result resource calculated by IVS function.

【Grammar】

int IMP_IVS_ReleaseResult(int ChnNum, void *result);

【Formal parameter】

Parameter name	Describe	Input/Output
ChnNum	Channel number	Input

【Return value】

return 0 success； no 0 fail.

【Dependence】

Head file：imp_ivs.h

Lib file：libimp.a / libimp.so

【NB】

Release the output result resources according to the channels bound by different IVS functions.

2.1.12 IMP_IVS_ReleaseData

【Function】

Release the frame parameter passed to Datacallback.

【Grammar】

int IMP_IVS_ReleaseData(void *vaddr);

【Formal parameter】

Parameter name	Describe	Input/Output
vaddr	The virtual address of the freed space	Input

【Return value】

return 0 success； no 0 fail.

【Dependence】

Head file：imp_ivs.h

Lib file：libimp.a / libimp.so

【NB】

You must use this function to release the frame parameter passed to Datacallback or you will definitely cause a deadlock.

This interface is for the use of algorithm providers only, algorithm users do not need to concern.

2.1.13 IMP_IVS_GetParam

【Function】

Get channel algorithm parameters.

【Grammar】

int IMP_IVS_GetParam(int chnNum, void *param);

【Formal parameter】

Parameter name	Describe	Input/Output
ChnNum	Channel number corresponding to IVS function	Input
param	Algorithm parameter virtual address pointer	Input

【Return value】

return 0 success；no 0 fail.

【Dependence】

Head file：imp_ivs.h

Lib file：libimp.a / libimp.so

【NB】

Release the output result resources according to the channels bound by different IVS functions.

2.1.14 IMP_IVS_SetParam

【Function】

Set channel algorithm parameters.

【Grammar】

int IMP_IVS_SetParam(int chnNum, void *param);

【Formal parameter】

Parameter name	Describe	Input/Output
ChnNum	Channel number corresponding to IVS function	Input
param	Algorithm parameter virtual address pointer	Input

【Return value】

return 0 success； no 0 fail.

【Dependence】

Head file：imp_ivs.h

Lib file：libimp.a / libimp.so

【NB】

Release the output result resources according to the channels bound by different IVS functions.