H264 Decoder Driver Software User Guide

ABSTRACT:
This is the Software User Guide Document for H264 Decoder Driver.
KEYWORDS:
User Guide
APPROVED:

Revision History

VERSION	DATE	AUTHOR	CHANGE DESCRIPTION
0.1	28-September-16	Andrei Sin	First draft
0.2	20-February-17	Andrei Sin	Update API
1.0	03-July-17	Andrei Sin	Update for RTM
1.1	14-December-17	Andrei Sin	Update H264dcd_dcd_cfg* description
1.2	13-March-18	Loc Nguyen	Update for RTM 1.1: use low level driver command instead of ioctl command
1.3	15-Aug-18	Dat Vu Van	Update for RTM 1.2: update section 3.1 Data Types, 3.4.1 API Function, 4.2 File Structure, section 1.5 Document Location

Table of Contents

H264 Decoder Driver Software User Guide 1

1 Introduction 4

1.1 Purpose 4

1.2 Audience Description 4

1.3 References 4

1.4 Definitions, Acronyms and Abbreviations 4

1.5 Document Location 5

2 General Description 6

3 Functional Description 7

3.1 Data Types 7

3.2 Standalone 9

3.3 Kernel Space 12

3.4 User Space 14

4 High Level Design 16

4.1 System Decomposition 16

4.2 File Structure 16

1 Introduction

The purpose of this document is to describe the H264 Decoder driver interface. It is intended to serve as a reference source during the development of VSDK based application.

1.1 Purpose

The purpose of this document is to define H264 Decoder driver internal behavior and user space interface. It is intended to serve as a reference source during the driver implementation and future use. For exact definitions and implementation details please check references and source code.

1.2 Audience Description

This document is intended for internal use by S23V234 Vision SDK developers.

1.3 References

Id	*Title*	*Location*
[1]	*SDI SW User Guide*	*Vision sdk install dir, folder:* s32v234_sdk\docs\drivers
[2]	*S32v234 Reference Manual*	Available on demand

Table 1: References

1.4 Definitions, Acronyms and Abbreviations

Term/Acronym	*Description*
API	*Application Programming Interface*
FIFO	*First In First Out*
HW	*Hardware*
IP	*Intellectual Property*
SDI	*Sensor Data Interface library*
SoC	*System on Chip*
SW	*Software*

Table 2: Acronyms

1.5 Document Location

This document is available in VisionSDK directory structure at the following location:
`VisionSDK: s32v234_sdk/docs/drivers`

2 General Description

The H264 decoder driver software (SW) is intended for kernel space and standalone management of H264 decoder HW module, which is designed to be part of the S32V234 SoC. An integral part of the driver is also a user space library providing an API for the user applications. This API wraps the kernel space interface of the driver by LLD commands

3 Functional Description

The H264 decoder driver SW has 3 layers (see Figure 1).

The first layer is standalone driver and implements functionality using all HW resources. Internal behavior of this layer will be described in detail in section 3.1.

The second layer operates in kernel space and implements functionality using first layer. The behavior of the kernel space layer is described in section 3.3.

The third layer is implemented as a user space abstraction layer for the kernel driver API. This layer is designated as H264 decoder user library. The provided user level API is explained in section 3.4.

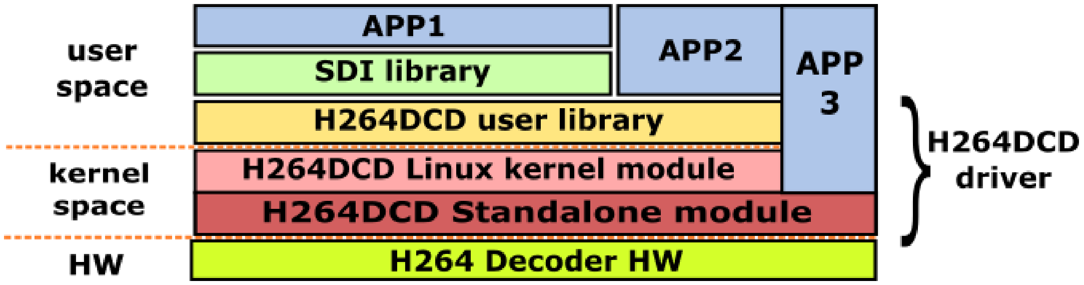

Figure 1: H264 decoder driver software layout

3.1 Data Types

The H264 decoder standalone driver introduces the following data types and containers (see source code for full definitions):

- Enumeration H264DCD_BOOL:
  Enumerates possible values for logical variables.
- Enumeration H264DCD_STREAM_ID:
  Enumerates possible values for stream id.
- Enumeration H264DCD_FIFO_LEVEL:
  Enumerates possible values for fifo level.
- Enumeration H264DCD_OUTPUT_BIT_WIDTH:
  Enumerates possible values for output bit width.
- Enumeration H264DCD_SAMPLE_PREC:
  Enumerates possible values for color component sample precision.
- Enumeration H264DCD_COL_FORMAT:
  Enumerates possible values for color format.

- Enumeration H264DCD_DATA_MODE

  Enumerates possible the data mode used for Chroma Cb and Cr components
- Enumeration H264DCD_DATAFLOW_MODE:
  Enumerates possible values for flow mode.
- Enumeration H264DCD_CH_STATUS:
  Enumerates possible values for channel control.
- Enumeration H264DCD_MEM:
  Enumerates possible values for memory types.
- Enumeration H264DCD_CH_CTRL

  Enumerates possible channel status.
- Enumeration H264DCD_CH_STATUS:
  Enumerates possible values for channel status.
- Structure H264DCD_STREAM_DATA_CONFIG:
  Describes the parameters of the data source.
- Structure H264DCD_STREAM_DATA_STATUS:
  Describes the packet status.
- Structure H264DCD_STREAM_FIFO_STATUS:
  Describes the fifo stack status.
- Structure H264DCD_FIFO_WTM:
  Describes the fifo stack watermark level.
- Structure H264DCD_TIMEOUT_ENABLE:
  Describes the timeout status (enable/disable).
- Structure H264DCD_OUTPUT_BUFFCFG:
  Describes the parameters for each component the buffer address and the number of lines per component used in the output circular component buffer.
- Structure H264DCD_OUTPUT_SAMPLEMODE:
  Describes the status of resample mode.
- Structure H264DCD_OUTPUT_COLOURDATA:
  Describes the parameters for data stream (color format, resample precision).
- Structure H264DCD_OUTPUT_BKSTRIDE:
  Describes the parameters for bankstride (address offset and feature enablement flag).
- Structure H264DCD_CH_CONTROL:
  Describes the status of a specific channel.
- Structure H264DCD_INFRAME_SIZE:
  Describes the frame size.
- Structure H264DCD_DECOD_CONFIG:
  Describes the memory configuration (memory type and deblocking filer flag for Mode1)
- Structure H264DCD_DECOD_THRLEVELS:

Describes the threshold levels for decoder core and frame based cycle counter.

- Structure H264DCD_DECOD_REFMEM:
  Describes the memory type for a stream.
- Structure H264DCD_DECOD_CHANNELST:
  Describes the channel status.
- Structure H264DCD_PIC_PARAMS:
  Describes the parameters for the decoded image (size, sampling precision, etc).
- Structure H264DCD_PIC_STATUS:
  Describes the parameters for decoded image status.
- Structure H264DCD_PIC_GENPARAMS:
  Describes the general parameters for decoded image (picture order, status, etc).
- Structure H264DCD_IRQ_TIMEOUT:
  Describes the timeout interrupt. (flag and status)
- Structure H264DCD_IRQ_ERROR:
  Describes the error interrupt status. (flag and status)
- Structure H264DCD_IRQ_STREAM:
  Describes the status for all interrupts that can occur.

3.2 Standalone

3.2.1 API Functions

This section, Table 3, describes functionality exported by the H264 decoder standalone driver module. It is used by upper SW layers (Linux environment – kernel and user space). See source code for full definitions.

Function Command	Description
H264dcd_inputstream_cfg	Sets the parameters of the data source (data packet address and size) for the specified stream.
H264dcd_fifostatus_get	Returns the status of the FIFO stack corresponding to the specified data stream.
H264dcd_packetstatus_get	Returns the packet status in the specified FIFO level corresponding to the specified data stream.
H264dcd_fifo_wtmklevel_set	Sets the watermark level for the FIFO stack used by the specified input stream.
H264dcd_fifo_clear	Resets the read/write pointers of the FIFO stack for the specified input stream.
H264dcd_timeoutset	Sets the timeout in terms of decoder clock (the used

	value is multiplied by 512). If timeout is enabled for a data stream the stream is switched automatically to the next available stream when the timeout expires.
`H264dcd_timeouten`	Enables the timeout functionality for the selected data stream.
`H264dcd_swreset`	Software reset for H264 decoder.
`H264dcd_output_cfg`	Sets for the selected data stream, the components buffer addresses and the number of lines per component used in the luma output circular component buffer.
`H264dcd_outsamplemode_set`	Enables/disables resample mode.
`H264dcd_outcolourdata_set`	Sets the color format and sampling precision for the selected data stream.
`H264dcd_flowmode_set`	Sets the data flow mode.
`H264dcd_flowmode_get`	Gets the data flow mode.
`H264dcd_bankstride_set`	Sets the address offset of each start of macroblock and enable or disable this feature.
`H264dcd_bankstride_get`	Returns the address offset of each start of macroblock and the enable flag of this functionality.
`H264dcd_dcd_chcontrol`	Stops the selected channel and switching it in idle mode
`H264dcd_dcd_inframesz`	Sets the coded image size in terms of macroblocks.
`H264dcd_dcd_cfgset`	Sets the type of the used memory SRAM or DDR and enable/disable the deblock filtering. Note: Deblock filter cannot be enabled in constrained baseline modes. Also, deblock filtering is not enabled when buffers are located in DDR due to the high usage of DDR bandwidth.
`H264dcd_dcd_cfgget`	Returns the type of the used memory SRAM or DDR and deblock filtering status.
`H264dcd_thrlevels_set`	Sets the 3 threshold levels for the decoder core, frame based cycle counter.
`H264dcd_thrlevels_get`	Returns the 3 threshold levels for the decoder core, frame based cycle counter.
`H264dcd_refmemaddress_set`	Sets the address of the memory zone where the decoder reads/stores the reference data for the selected channel.

H264dcd_refmemaddress_get	Returns the address of the memory zone where the decoder reads/stores the reference data for the selected channel.
H264dcd_chstatus_get	Returns the status of the selected decoder channel.
H264dcd_pictureparam_get	Returns the parameters of the decoded image for the selected channel.
H264dcd_picturestatus_get	Returns the status of the decoded image for the selected channel valid (ON) or not (OFF).
H264dcd_picgenparams_get	Returns the general parameters of the decoded image: the picture order count value, status (new POC cycle or not) and the maximum number of ref frames in the buffer.
H264dcd_interrupt_ctrl	Enables/disables interrupts.
H264dcd_interrupt_get	Returns which of the interrupts are enabled at the moment.
H264dcd_timeoutirq_get	Returns the timeout interrupt flag and the timeout status byte. If the flag is set the function resets it.
H264dcd_errorirq_get	Returns the decoding error interrupt flag and the 4 channel status bytes. If the flag is set the function resets it.
H264dcd_streamirq_get	Returns the status of the interrupt flags (for all the interrupt causes) for the selected data stream. It shall reset all the interrupt flags which are already set.

Table 3: H264 decoder standalone driver API

3.2.2 Usage

The H264 decoder interface can be configured using H264dcd_inputstream_cfg to set input data address and size. Pointer to the input data and its size is stored in a FIFO buffer. The level of the FIFO stack can be checked using H264dcd_fifostatus_get. To get data status from a specific level in the FIFO stack, H264dcd_packetstatus_get function can be used. User can set a watermark level for all the FIFO stacks in use, using H264dcd_fifo_wtmklevel_set and clear the FIFO using H264dcd_fifo_clear. The output data be configured using H264dcd_output_cfg. Registers affected by configuring output data can be resampled at frame done using H264dcd_outsamplemode_set (using parameter H264_ON). Also, the color format and resampling precisions can be configured with H264dcd_outcolourdata_set function. H264 decoder HW needs a timeout for decoding operations. H264dcd_timeoutset and H264dcd_timeouten will set the desired timeout and enables this feature. To perform a SW reset to H264 decoder HW `H264dcd_swreset` shall be used.

H264 decoder can work in 3 modes. Interacting with decoder working mode is done by `H264dcd_flowmode_set` and `H264dcd_flowmode_get`. Driver offers the possibility to access and change the address offset of each start of macroblock row using `H264dcd_bankstride_set` and `H264dcd_bankstride_get` functions. Other H264 settings that user can change are frame dimensions (`H264dcd_dcd_inframesz)`, the type of the used memory (`H264dcd_dcd_cfgset` / `H264dcd_dcd_cfgget`), threshold level for decoder core and frame based cycle counter (`H264dcd_thrlevels_set` / `H264dcd_thrlevels_get`) and the address of the memory zone where the decoder reads/stores the data (`H264dcd_refmemaddress_set` / `H264dcd_refmemaddress_get`). Driver can perform the management of the input channel: `H264dcd_chstatus_get` returns the status of decoder channel; `H264dcd_dcd_chcontrol` – starts/stops the selected input channel. Decoded picture management is implemented by `H264dcd_pictureparam_get` - provides the parameters for the decoded image such as sampling precision, size and the amount of luma samples which must be cropped on the right side to obtain the original picture resolution; `H264dcd_picturestatus_get` – returns the status of the decoded image (if is valid or not) and `H264dcd_picgenparams_get` – which provides the generic parameters for decoded image (order count value, status (new POC cycle or not) and the maximum number of ref frames in the buffer).

User can check what interrupts are enable/ disable using `H264dcd_interrupt_get/ H264dcd_streamirq_get` and can enable / disable specific interrupt using `H264dcd_interrupt_ctrl`. Timeout interrupts and error can be check with `H264dcd_timeoutirq_get` and `H264dcd_errorirq_get` functions.

3.3 Kernel Space

3.3.1API Functions

This section, Table 4, describes functionality exported by the H264 decoder driver module at kernel space level (see source code for full definitions).

LLD Command	Description
`H264DCD_LLDCMD_INPUTSTREAM_CFG`	Calls H264dcd_inputstream_cfg().
`H264DCD_LLDCMD_FIFO_STATUS_GET`	Calls H264dcd_fifostatus_get().
`H264DCD_LLDCMD_PCK_STATUS_GET`	Calls H264dcd_packetstatus_get().
`H264DCD_LLDCMD_FIFO_WATERMARK_SET`	Calls H264dcd_fifo_wtmklevel_set().

H264DCD_LLDCMD_FIFO_CLEAR	Calls H264dcd_fifo_clear().
H264DCD_LLDCMD_TIMEOUT_SET	Calls H264dcd_timeoutset().
H264DCD_LLDCMD_TIMEOUT_ENABLE	Calls H264dcd_timeouten().
H264DCD_LLDCMD_SW_RESET	Calls H264dcd_swreset().
H264DCD_LLDCMD_OUTSTREAM_CFG	Calls H264dcd_output_cfg().
H264DCD_LLDCMD_OUTSAMPLEMODE_SET	Calls H264dcd_outsamplemode_set().
H264DCD_LLDCMD_OUTCOLOURDATA_SET	Calls H264dcd_outcolourdata_set().
H264DCD_LLDCMD_DATAFLOWMODE_SET	Calls H264dcd_flowmode_set().
H264DCD_LLDCMD_DATAFLOWMODE_GET	Calls H264dcd_flowmode_get().
H264DCD_LLDCMD_BANKSTRIDE_SET	Calls H264dcd_bankstride_set().
H264DCD_LLDCMD_BANKSTRIDE_GET	Calls H264dcd_bankstride_get().
H264DCD_LLDCMD_DCD_CH_STOP	Calls H264dcd_dcd_chcontrol().
H264DCD_LLDCMD_DCD_INFRAMESIZE_SET	Calls H264dcd_dcd_inframesz().
H264DCD_LLDCMD_DCD_CFG_SET	Calls H264dcd_dcd_cfgset().
H264DCD_LLDCMD_DCD_CFG_GET	Calls H264dcd_dcd_cfgget().
H264DCD_LLDCMD_DCD_THRLEVELS_SET	Calls H264dcd_thrlevels_set().
H264DCD_LLDCMD_DCD_THRLEVELS_GET	Calls H264dcd_thrlevels_get().
H264DCD_LLDCMD_DCD_REFMEMORY_SET	Calls H264dcd_refmemaddress_set().
H264DCD_LLDCMD_DCD_REFMEMORY_GET	Calls H264dcd_refmemaddress_get().
H264DCD_LLDCMD_CH_STATUS_GET	Calls H264dcd_chstatus_get().
H264DCD_LLDCMD_PICDCD_PARAM_GET	Calls H264dcd_pictureparam_get().
H264DCD_LLDCMD_PICDCD_STATUS_GET	Calls H264dcd_picturestatus_get().
H264DCD_LLDCMD_PICDCD_GENPARAMS_GET	Calls H264dcd_picgenparams_get().
H264DCD_LLDCMD_IRQ_CONTROL	Calls H264dcd_interrupt_ctrl().
H264DCD_LLDCMD_IRQ_GET	Calls H264dcd_interrupt_get().
H264DCD_LLDCMD_IRQ_TIMEOUT_GET	Calls H264dcd_timeoutirq_get().
H264DCD_LLDCMD_IRQ_ERROR_GET	Calls H264dcd_errorirq_get().
H264DCD_LLDCMD_IRQ_STREAMSTATUS_GET	Calls H264dcd_streamirq_get().

Table 4: H264 Decoder kernel driver API

3.3.2 Usage

Linux kernel driver for H264 decoder HW is based on standalone version. All LLD commands are mapped 1:1 with the standalone version. For more information please check 3.2.2 section.

3.4 User Space

The H264 decoder driver SW includes a user space library to abstract the kernel space driver from user applications. The user space library invokes the kernel space functionality described in the previous section.

3.4.1 API Functions

The H264 decoder driver user level API mentioned in Table 5 is declared in `isp_h264dec.h` and defined in `h264dec_user.cpp` file.

Function	Description
`H264DEC_Open`	Opens the special device file on Linux ("h264dcd")
`H264DEC_Close`	Closes the special device file on Linux ("h264dcd")
`H264DEC_InConfig`	Configures H264 decoder input (data address and data size).
`H264DEC_WtmLevelSet`	Sets up watermark level.
`H264DEC_TimeoutSet`	Configures H264 timeout.
`H264DEC_TimeoutEnable`	Enables H264 timeout.
`H264DEC_SwReset`	Invokes SW reset.
`H264DEC_OutConfig`	Configures H264 decoder output.
`H264DEC_OutSampleModeSet`	Sets sample mode of the output related registers.
`H264DEC_OutColorDataSet`	Sets the color format and component samples precision.
`H264DEC_DataFlowModeSet`	Set the data flow mode.
`H264DEC_BankStrideSet`	Set the bank stride.
`H264DEC_InFrameSizeSet`	Configures encoded frame size in terms of macroblocks.
`H264DEC_ChannelStop`	Stops the selected channel switching it to idle mode.
`H264DEC_DeblockSet`	Sets up deblocking and the memory type (SRAM/DDR) used for reference data.
`H264DEC_ThreshLevelsSet`	Sets up 3 thresholding levels for the decoder core.
`H264DEC_RefMemorySet`	Sets up reference memory address.

H264DEC_FifoStatusGet	Gets status of specified channel fifo.
H264DEC_IrqMaskSet	Sets up IRQ mask.
H264DEC_IrqMaskGet	Reads current IRQ mask setup.

Table 5: H264 decoder user library exported functions

3.4.2 Usage

The H264 Decoder driver provides a userspace library which expose the main functionality. To use this userspace interface, the application should include "isp_h264dec.h" and link the static library.

Before configuring the H264 decoder hardware, H264DEC_Open() should be called. When the application exits, H264DEC_Close() should be called to release used resources. H264DEC_InConfig() provides the input data address and size for H264 decoder. User can set a watermark level for all the FIFO stacks in use, using H264DEC_WtmLevelSet() . The output data be configured using H264DEC_OutConfig(). Registers affected by configuring output data can be resampled at frame done using H264DEC_OutSampleModeSet() (using parameter H264_ON). Also, the color format and resampling precisions can be configured with H264DEC_OutColorDataSet() function. Data flow mode for decoder can be set using H264DEC_DataFlowModeSet(). H264 decoder HW needs a timeout for decoding operations. H264DEC_TimeoutSet() and H264DEC_TimeoutEnable() will set the desired timeout and enables this feature. To perform a SW reset to H264 decoder HW H264DEC_SwReset() shall be used.

Library offers the possibility to set different parameters for H264 decoder: address offset of each start of macroblock row using H264DEC_BankStrideSet (), frame dimensions with H264DEC_InFrameSizeSet(), threshold level for decoder core and frame based cycle counter with H264DEC_ThreshLevelsSet(), address of the memory zone where the decoder reads/stores the data with H264DEC_RefMemorySet()and the memory type (SRAM/DDR) used for reference data using H264DEC_DeblockSet(). The channel decoding management is done by using H264DEC_ChannelStop() function.

Userspace library allows to manage interrupts using H264DEC_IrqMaskSet() and H264DEC_IrqMaskGet().

4 High Level Design

4.1 System Decomposition

The H264 decoder driver belongs to the complex data preprocessing subsystem of the s32v234 SoC that is wrapped and controlled by the SDI library. Part of this subsystem is visualized in Figure 1. For more information about SDI and data preprocessing please refer to *[1]*.

The preferred way to use the H264 functionality in a user application is to use Sequencer graphs together with the SDI library services. The SDI library provides complete abstraction of the H264 decoder driver interface and thanks to utilization of the Sequencer HW the data flow management load for the host CPU is minimized.

4.2 File Structure

The H264 decoder driver code is located in VSDK under s32v234_sdk/libs/isp/h264dec folder. Internally it has the following structure:

- kernel
    - build-v234ce-gnu-linux-d – build folder for Linux kernel module
        - Makefile
    - common
        - include
            - h264dcd_core.h - declaration of standalone driver functionality
            - h264dcd_linux.h – declaration of kernel space driver functionality
            - io_core.h – definition of inline functions to read/write data from register on standalone
        - src
            - h264dcd_core.c – standalone space driver related functionality
    - linux
        - include
            - h264dcd_types.h - declaration of data types
            - io_linux.h - definition of inline functions to read/write data from register on linux
        - src
            - h264dcd_linux.c – kernel space driver related functionality

- user
    - build-* – build folders for the Linux platform,
        - Makefile
    - src
        - h264dec_user.cpp – definition of user space level public API.
- BUILD.mk – defines build details
- Public headers (s32v234_sdk/include):
    - isp_h264dec.h – declaration of user space level public API.