JPEG Decoder Driver Software User Guide

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

Revision History

VERSION	DATE	AUTHOR	CHANGE DESCRIPTION
0.1	23-September-16	Andrei Sin	First draft
0.2	14-November-16	Andrei Sin	Update with latest changes
0.3	20-December-16	Andrei Sin	Update with latest source code changes
1.0	03-July-17	Andrei Sin	Update for RTM
1.1	13-March-18	Loc Nguyen	Update for RTM 1.1: use low level driver command instead of ioctl command
1.2	15-Aug-18	Dat Vu Van	Update for RTM 1.2: Update 1.5 Document Location, 3.2.2 Usage, 4.2 File Structure

Table of Contents

1 Introduction

The purpose of this document is to describe the JPEG 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 JPEG 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 JPEG decoder driver software (SW) is intended for kernel space and standalone management of JPEG 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 JPEG 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 JPEG decoder user library. The provided user level API is explained in section 3.4.

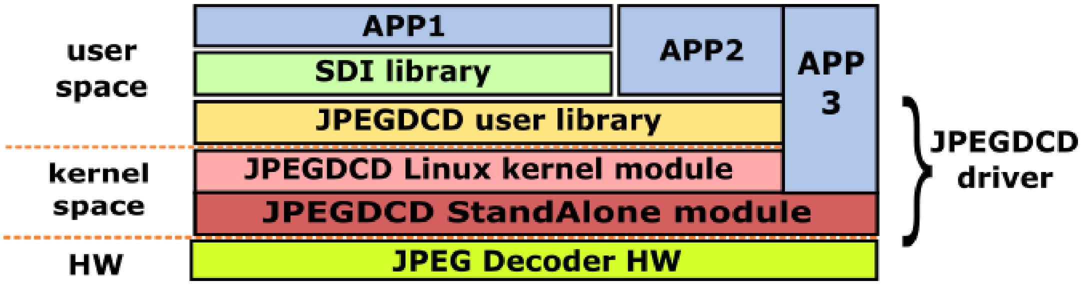

Figure 1: JPEG decoder driver software layout

3.1 Data Types

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

- Enumeration JPEGDCD_BOOL:
  Enumerates possible values for logical variables.
- Enumeration STREAM_ID :
  Enumerates possible values for stream id.
- Enumeration FIFO_LEVEL :
  Enumerates possible values for fifo level.
- Enumeration SAMPLE_PREC:
  Enumerates possible values for sample precision.
- Enumeration JPEGDCD_FRAMEST :
  Enumerates possible values for frame status.
- Enumeration COMPAT_MODE:
  Enumerates possible values for compatibility mode.
- Structure STREAM_DATA_CONFIG:

Describes the parameters of the data source.

- Structure STREAM_FIFO_STATUS:
  Describes the fifo stack status.
- Structure 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 STREAM_JPEG_PARAM:
  Describes the parameters for jpeg image (size, components sampling factors).
- Structure JPEGDCD_CFGERR:
  Describes the error that can occur (status and type).
- Structure JPEGDCD_DECODED_IMAGE:
  Describes the decoded image parameters (size, sampling factors, etc.).

3.2 Standalone

3.2.1 API Functions

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

Function	Description
`JPEGdcd_inputstream_cfg`	Sets the parameters of the data source for the specified stream.
`JPEGdcd_fifostatus_get`	Returns the status of the FIFO stack corresponding to the specified data stream.
`JPEGdcd_fifodata_get`	Returns the configuration data in the specified FIFO corresponding to the specified data stream.
`JPEGdcd_fifo_wtmklevel_set`	Sets the watermark level for all the FIFO stacks in use.
`JPEGdcd_output_cfg`	Sets for each component the buffer address and the number of lines per component used in the output circular component buffer.
`JPEGdcd_outsamplemode_set`	Sets the sample mode for the output buffer information: ON means all the registers are resampled at the frame done
`JPEGdcd_bankstride_set`	Sets the address offset of each start of macroblock.
`JPEGdcd_bankstride_get`	Returns the address offset of each start of macroblock.
`JPEGdcd_jpeg_configstart`	JPEG decoder HW needs a timeout for decoding operation. This function implements an HW

	workaround for setting the timeout (sets the timeout to 1 an after this it resets the JPEG decode). It shall be called before the other JPEG registers to be programmed. It shall be used before to program the JPEG core registers.
JPEGdcd_jpeg_timeoutset	Sets the JPEG error timeout (jpeg_errtimeout) measured in JPEG clock cycles. It shall be called after all the JPEG registers are programmed and before the JPEG decoder to be enabled.
JPEGdcd_jpeg_dcdenable	Sets the GO bit in order to start the decoding activity after all the JPEG registers are already configured according to the JPEG_BOOL input variable.
JPEGdcd_jpeg_streamdatacfg	Sets the image size and the components sampling factors for the selected input stream.
JPEGdcd_jpeg_streamnum_set	Sets the number of streams.
JPEGdcd_jpeg_streamnum_get	Returns the number of streams.
JPEGdcd_jpeg_mode_set	Sets the jpeg compatibility mode.
JPEGdcd_jpeg_mode_get	Returns the jpeg compatibility mode.
JPEGdcd_jpeg_swreset	Sets and resets the SW_RST bit resetting the Input DMA Wrapper, the Output DMA Wrapper and the JPEG Decoder Core.
JPEGdcd_jpeg_streamactive_get	Returns the id of the stream written in the memory at that moment.
JPEGdcd_jpeg_rstinterval_set	Sets the restart interval in terms of image lines.
JPEGdcd_jpeg_rstinterval_get	Returns the restart interval in terms of image lines.
JPEGdcd_jpeg_framest_get	Returns the status of the image to be decoded: ON means decoding frame is running, OFF means that was finished and REPEAT means that both flags were found set and are initialized for a new read operation.
JPEGdcd_jpeg_cfgerr_get	Returns the status of the JPEG configuration error flag and the value of the error byte.
JPEGdcd_jpeg_mcurow_get	Returns the MCU row end flag status and reset it (if it is set).
JPEGdcd_jpeg_dcdimage_get	Returns JPEG decoded image parameters.
JPEGdcd_interrupt_ctrl	Enables/disables interrupts according to mask set as parameter.
JPEGdcd_interrupt_get	Returns the interrupts which are enabled at the moment.

Table 3: JPEG decoder standalone driver API

3.2.2 Usage

The JPEG decoder interface can be configured using `JPEGdcd_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 `JPEGdcd_fifostatus_get`. To get data from a specific level in the FIFO stack, `JPEGdcd_fifodata_get` function can be used. User can set a watermark level for all the FIFO stacks in use, using `JPEGdcd_fifo_wtmklevel_set`. The output data be configured using `JPEGdcd_output_cfg`. Registers affected by configuring output data can be resampled at frame done using `JPEGdcd_outsamplemode_set` (using parameter JPEG_ON). Driver offers the possibility to access and change the address offset of each start of macroblock row using `JPEGdcd_bankstride_set` and `JPEGdcd_bankstride_get` functions. JPEG decoder hardware need to set a timeout for decoding operation. In order to do that, driver implements a workaround for the hw: `JPEGdcd_jpeg_configstart` should be invoked before programming the JPEG core registers. To set timeout `JPEGdcd_jpeg_timeoutset` function shall be used. Please notice that timeout should be set after all JPEG registers are programmed, just before starting the decode operation. To start decoding, `JPEGdcd_jpeg_dcdenable` function should be called. Driver can configure input data using `JPEGdcd_jpeg_streamdatacfg` function. The `JPEGdcd_jpeg_streamdatacfg` lets user to set image size and component sampling factors for desired input stream. JPEG decoder HW can use up to 4 input streams. To set the number of active streams user should call `JPEGdcd_jpeg_streamnum_set`. To get the number of active input stream `JPEGdcd_jpeg_streamnum_get` comes in handy. `JPEGdcd_jpeg_mode_set` and `JPEGdcd_jpeg_mode_get` should be used for interacting with JPEG HW compatibility mode. To perform a SW reset to JPEG decoder HW `JPEGdcd_jpeg_swreset` shall be used. `JPEGdcd_jpeg_streamactive_get` function can be used to get the id of the stream written in the memory at that moment. Driver can handle the restart interval in terms of image lines using `JPEGdcd_jpeg_rstinterval_set` and `JPEGdcd_jpeg_rstinterval_get`. To check the MCU row end flag status is completed `JPEGdcd_jpeg_mcurow_get` shall be called. To get the status of decoded image, `JPEGdcd_jpeg_framest_get` should be invoked. If an error occurs in the JPEG decoder HW, the `JPEGdcd_jpeg_cfgerr_get` can provide the status. Decoded image is provided by `JPEGdcd_jpeg_dcdimage_get` `(`with jpeg_decoded_image as parameter). Interrupts management is done via `JPEGdcd_interrupt_ctrl` and `JPEGdcd_interrupt_get`.

3.3 Kernel Space

3.3.1API Functions

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

LLD Command	Description
JPEGDCD_LLDCMD_INPUTSTREAM_CFG	Calls JPEGdcd_inputstream_cfg().
JPEGDCD_LLDCMD_FIFO_STATUS_GET	Calls JPEGdcd_fifostatus_get().
JPEGDCD_LLDCMD_FIFO_DATA_GET	Calls JPEGdcd_fifodata_get().
JPEGDCD_LLDCMD_FIFO_WATERMARK_SET	Calls JPEGdcd_fifo_wtmklevel_set().
JPEGDCD_LLDCMD_OUTSTREAM_CFG	Calls JPEGdcd_output_cfg().
JPEGDCD_LLDCMD_OUTSAMPLEMODE_SET	Calls JPEGdcd_outsamplemode_set().
JPEGDCD_LLDCMD_BANKSTRIDE_SET	Calls JPEGdcd_jpeg_bankstride_set().
JPEGDCD_LLDCMD_BANKSTRIDE_GET	Calls JPEGdcd_jpeg_bankstride_get().
JPEGDCD_LLDCMD_JPEG_CFGSTART	Calls JPEGdcd_jpeg_configstart().
JPEGDCD_LLDCMD_JPEG_TIMEOUT_SET	Calls JPEGdcd_jpeg_timeoutset().
JPEGDCD_LLDCMD_JPEG_DCDENABLE	Calls JPEGdcd_jpeg_dcdenable().
JPEGDCD_LLDCMD_JPEG_STREAMCFG_SET	Calls JPEGdcd_jpeg_streamdatacfg().
JPEGDCD_LLDCMD_JPEG_STREAMNUM_SET	Calls JPEGdcd_jpeg_streamnum_set().
JPEGDCD_LLDCMD_JPEG_STREAMNUM_GET	Calls JPEGdcd_jpeg_streamnum_get().
JPEGDCD_LLDCMD_JPEG_MODE_SET	Calls JPEGdcd_jpeg_mode_set().
JPEGDCD_LLDCMD_JPEG_MODE_GET	Calls JPEGdcd_jpeg_mode_get().
JPEGDCD_LLDCMD_JPEG_SW_RESET	Calls JPEGdcd_jpeg_swreset().
JPEGDCD_LLDCMD_JPEG_STREAM_GET	Calls JPEGdcd_jpeg_streamactive_get().
JPEGDCD_LLDCMD_JPEG_RSTINTERVAL_SET	Calls JPEGdcd_jpeg_rstinterval_set().
JPEGDCD_LLDCMD_JPEG_RSTINTERVAL_GET	Calls JPEGdcd_jpeg_rstinterval_get().
JPEGDCD_LLDCMD_JPEG_FRAMESTATUS_GET	Calls JPEGdcd_jpeg_framest_get().
JPEGDCD_LLDCMD_JPEG_DCDCFGERR_GET	Calls JPEGdcd_jpeg_cfgerr_get().
JPEGDCD_LLDCMD_JPEG_MCUROWEND_GET	Calls JPEGdcd_jpeg_mcurow_get().
JPEGDCD_LLDCMD_JPEG_DCDIMAGE_GET	Calls JPEGdcd_jpeg_dcdimage_get().
JPEGDCD_LLDCMD_IRQ_CONTROL	Calls JPEGdcd_interrupt_ctrl().
JPEGDCD_LLDCMD_IRQ_GET	Calls JPEGdcd_interrupt_get().

Table 4: JPEG Decoder kernel driver API

3.3.2 Usage

Linux kernel driver for JPEG 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 JPEG 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 JPEG decoder driver user level API mentioned in Table 5 is declared in `isp_jpegdec.h` and defined in `jpegdec_user.cpp` file. (see source code*[1]* for full definitions)

Function	Description
`JPEGDEC_Open`	Opens the special device file on Linux (“fsl_jpegdcd”).
`JPEGDEC_Close`	Closes the special device file on Linux (“fsl_jpegdcd”).
`JPEGDEC_ConfigStart`	Initializes JPEG decoder configuration. Should be invoked as the first call before the JPEG registers are configured.
`JPEGDEC_TimeoutSet`	Configures JPEG timeout. Should be called after all JPEG configuration was done, just before asserting GO bit.
`JPEGDEC_InConfig`	Configures JPEG decoder input.
`JPEGDEC_WtmLevelSet`	Sets up watermark level.
`JPEGDEC_OutConfig`	Configures JPEG decoder output.
`JPEGDEC_OutSampleModeSet`	Sets sample mode of the output related registers.
`JPEGDEC_BankStrideSet`	Modifies bank stride setup.
`JPEGDEC_StreamConfig`	Sets the image size and the components sampling factors.
`JPEGDEC_ModeSet`	Sets the number of streams used and appropriately the mode used.
`JPEGDEC_SwReset`	Invokes SW reset.
`JPEGDEC_RstIntervalSet`	Sets the stream reset interval in terms of number of image lines.
`JPEGDEC_Start`	Invokes decoding.
`JPEGDEC_Stop`	Stops decoding.
`JPEGDEC_IrqMaskSet`	Sets up IRQ mask.
`JPEGDEC_IrqMaskGet`	Reads current IRQ mask setup.

`JPEGDEC_CfgErrGet`	Reads configuration status register.

Table 5: JPEG decoder user library exported functions

4 High Level Design

4.1 System Decomposition

The JPEG 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 JPEG functionality in a user application is to use Sequencer graphs together with the SDI library services. The SDI library provides complete abstraction of the JPEG 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 JPEG decoder driver code is located in VSDK under s32v234_sdk/libs/isp/jpegdec folder. Internally it has the following structure:

- kernel
    - build-v234ce-gnu-linux-d – build folder for Linux kernel module
        - Makefile
    - common
        - include
            - jpegdcd_core.h - declaration of standalone driver functionality
            - jpegdcd_types.h – declaration of data types
            - io_core.h – definition of inline functions to read/write data from register on standalone
        - src
            - jpegdcd_core.c – standalone space driver related functionality
    - linux
        - include
            - io_linux.h - definition of inline functions to read/write data from register on linux
            - jpegdcd_linux.h – declaration of kernel space driver functionality
        - src
            - jpegdcd_linux.c – kernel space driver related functionality
- user
    - build-* – build folders for the Linux platform,
        - Makefile

    - src
        - jpegdec_user.cpp – definition of user space level public API.
    - BUILD.mk – defines build details
- Public headers (s32v234_sdk/include):
    - isp_jpegdec.h – declaration of user space level public API.