Software On-Screen Display on C674x

Application Programming Interface

and

User's Guide

For TMS320DM81xx-based Reference Design Kits

Texas Instruments

MPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		**Applications**	
Amplifiers	amplifier.ti.com	Audio	www.ti.com/audio
Data Converters	dataconverter.ti.com	Automotive	www.ti.com/automotiveDSP
	dsp.ti.com	Broadband	www.ti.com/broadband
Clocks and Timers	www.ti.com/clocks	Digital Control	www.ti.com/digitalcontrol
Interface	interface.ti.com	Medical	www.ti.com/medical
Logic	logic.ti.com	Military	www.ti.com/military
Power Mgmt	power.ti.com	Optical Networking	www.ti.com/opticalnetwork
Microcontrollers	microcontroller.ti.com	Security	www.ti.com/security
RFID	www.ti-rfid.com	Telephony	www.ti.com/telephony
RF/IF and ZigBee® Solutions	www.ti.com/lprf	Video & Imaging	www.ti.com/video
		Wireless	www.ti.com/wireless

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265

Contents

1 Introduction

DVR RDK on TI81xx supports On-Screen Display algorithm running on C674x DSP. This application note serves as guidance to use SWOSD from demo/user layer. This includes the memory allocation, filling the memory with right YUV data, configuring the OSD at user layer.

2 Application Programming Interface On-Screen Display

This section describes the OSD's macros and data structures accessible from user space to interface with OSD algorithm. These elements are defined in `<DVR_RDK>/dvr_rdk/mcfw/interfaces/link_api/algLink.h`. algLink is common links for algorithm on DSP side and wraps today links for OSD and SCD algorithm.

2.1 OSD Macros

The following Macros are defined and supported for osd by algLink.

2.1.1 ALG_LINK_OSD_MAX_WINDOWS

ALG_LINK_OSD_MAX_WINDOWS is maximum number of OSD windows per frame supported by algorithm. This is used by algorithm to set the maximum size of various arrays in its internal state structure. Application can change this value but need to make sure to rebuild the DSP firmware dvr_rdk_fw_c6xdsp.xe674. Default value supported by algorithm is maximum 10 OSD windows per frame.

2.1.2 ALG_LINK_OSD_MAX_CH

ALG_LINK_OSD_MAX_CH is maximum channel numbers supported by OSD Link. If input queue to OSD link has channel numbers greater than ALG_LINK_OSD_MAX_CH, it asserts. User can modify this value if required as per application use case but again will need to rebuild the algLink and create new firmware binary dvr_rdk_fw_c6xdsp.xe674 for C6x DSP.

2.1.3 ALG_LINK_OSD_CMD_SET_CHANNEL_WIN_PRM

ALG_LINK_OSD_CMD_SET_CHANNEL_WIN_PRM is system link command to change run time osd window parameters.

2.2 OSD Structures

The following public structures are defined and supported to interface with OSD algorithm

2.2.1 struct AlgLink_OsdWindowPrm

AlgLink_OsdWindowPrm defines the OSD window parameters that user need to set at create time and can modify run time to configure and control OSD algorithm.

```
typedef struct {
  UInt8  *addr[2][2];
  UInt32 format;
  UInt32 startX;
  UInt32 startY;
  UInt32 width;
  UInt32 height;
  UInt32 lineOffset;
  UInt32 globalAlpha;
  UInt32 transperencyEnable;
  UInt32 enableWin;
} AlgLink_OsdWindowPrm;
```

In this table W and H stands for width and height of video window

Variable Name	Type	Range	Description
addr	UInt8 *[x][y]	-	*Array of OSD buffer pointer address. In this array , y represents two separate buffer pointers for Y and UV when input data is in YUV420 semi planar format. When input data is YUV422i, [x][1] is ignored. While size of x is 2, OSD algorithm only uses [0][y] buffer pointers.*
format	U32	--	*SYSTEM_DF_YUV422I_YUYV or SYSTEM_DF_YUV420SP_UV. This macro is defined in mcfw/interfaces/link_api/system_const.h* *If input video frame is in YUV422 interleaved format, then set the OSD window format as SYSTEM_DF_YUV422I_YUYV, ex- captured frame buffer. If input video frame is in YUV42o semi-planar format, then set the OSD window format as SYSTEM_DF_YUV420SP_UV, ex- encoder video buffer.*
startX	U32	[0,W-width]	*X co-ordinate of OSD window in pixels, relative to start of video window, must be multiple of 2. In the range specified' width' is width of OSD window*
startY	U32	[0,H-height]	*Y co-ordinate of OSD window in pixels, relative to start of video window. In the range specified 'height' is height of OSD window,*
width	U32	[4,W]	*OSD window width*
height	U32	[1,H]	*OSD window height in lines*
lineOffset	U32	[1, 30]	*OSD window buffer pitch in pixels, must be >= width, must be multiple of 4, recommended to be multiple of 32 for efficiency*

Variable Name	Type	Range	Description
globalAlpha	U32	[0,128]	*8-bit global Alpha Value, used only if Alpha window is not enabled. Set to 0 to disable alpha blend. 0: Min Alpha, show only video, 128: Max Alpha, show only Graphics*
transperencyEnable	U32	[0,1]	*Boolean flag to enable/disable transparency.* *1: enable transperency, 0: disable transperency*
enableWin	U32	[0,1]	*Boolean flag to enable/disable OSD for this window.* *0:disable OSD, 1: enableOSD*

2.2.2 struct AlgLink_OsdWindowPrm

AlgLink_OsdWindowPrm defines the osd parameters that can be changed dynamically for channel id chId. User need to pass this entire structure to algLink each time any parameter is modified.

```
typedef struct AlgLink_OsdChWinParams
{
    UInt32 chId;
    UInt32 numWindows;
    AlgLink_OsdWindowPrm winPrm[ALG_LINK_OSD_MAX_WINDOWS];
}AlgLink_OsdChWinParams;
```

Variable Name	Type	Range	Description
chId	UInt32	-	*Channel ID: A 32-bit number used to uniquely identify the OSD channel. Serves as a internal handle.*
numWindows	U32	[0, ALG_LINK_OSD_MAX_WINDOWS]	*Maximum number of OSD window possible to be overlayed/blended seperately on video frame*
winPrm	AlgLink_OsdWindowPrm[]	--	*Array of osd window parameters for each channel. Set this pointer to a valid, allocated array to set the OSD during set-up. Up to ALG_LINK_OSD_MAX_WINDOWS windows can be defined per channel*

2.2.3 struct AlgLink_ OsdChCreateParams

AlgLink_ OsdChCreateParams defines the set of parameters used at creation time to configure the OSD instance.

```
typedef struct
{
    UInt32 maxWidth;
    UInt32 maxHeight;
    AlgLink_OsdChWinParams chDefaultParams;
} AlgLink_OsdChCreateParams;
```

Variable Name	Type	Range	Description
maxWidth	U32	[4, W]	*The maximum possible width of an input OSD window that needs to be overlayed. Size can range from 4 to W where W is the width of video frame on top of which the OSD data needs to be overlayed*
maxHeight	U32	[]	*The maximum possible height of OSD window. This is currently not used and ignored by OSD algorithm.*
chDefaultParams	AlgLink_OsdChWinParams	---	*Default values of OSD windows parameters for particular channel.*

3 OSD Usage in Application

To use OSD feature, application needs to follow these steps at mcfw layer and user application (demo) layer.

3.1 At McFW Layer

At McFw layer, create the link chains for your use case including algLink (with enableOSDAlg flag set to TRUE) without having any specific information on individual OSD window parameters including valid memory allocation for the OSD windows.

Initialize the OSD create time parameters AlgLink_OsdChCreateParams::maxWidth and AlgLink_OsdChCreateParams::maxHeight per your use case max window size for all the channels. Set the channel ids of individual channel in AlgLink_OsdChCreateParams::chDefaultParams::chId. At this time, no information on other parameters of AlgLink_OsdChCreateParams::chDefaultParams is available, so set the AlgLink_OsdChCreateParams::chDefaultParams::numWindows value to 0. This will ensures that algLink state is created at the time of link chains creation with all necessary memories required but will not do any processing as number of windows set to 0.

Refer to file `<DVR_RDK>/dvr_rdk/mcfw/src_linux/mcfw_api/usecases/multich_vcap_venc_vdis.c` for initializations of osdLink create time parameter.

Below is relevant code snippet for setting create time parameters for algLink structure settings from multich_vcap_venc_vdis.c file for quick reference.

```
  if(enableOsdAlgLink)
    {
        int chId;

        for(chId = 0; chId < ALG_LINK_OSD_MAX_CH; chId++)
        {
            AlgLink_OsdChWinParams * chWinPrm =
                            dspAlgPrm.osdChCreateParams[chId].chDefaultParams;

            /* set osd window max width and height */
            dspAlgPrm.osdChCreateParams[chId].maxWidth  =
            EXAMPLE_OSD_WIN_MAX_WIDTH;
            dspAlgPrm.osdChCreateParams[chId].maxHeight =
            EXAMPLE_OSD_WIN_MAX_HEIGHT;

            chWinPrm->chId = chId;
            chWinPrm->numWindows = 0;
        }
    }
```

Note that to save upon DDR BW, osd algorithm does in-place overlay meaning that OSD window data is copied to the video input buffer and no separate output buffer is created. Since OSD data is overwritten to the input video buffer, if use case requires original input captured data (without OSD) for encoding purpose, make sure that algLink is part of the chain after encoding is done.

3.2 At User Application Layer (Demo Layer)

User can allocate buffers for OSD windows at user layer, own and manage it. Since user owns the memory and has access to OSD buffer pointers, user can change the OSD window content run time. Memory for OSD window needs to be allocated from shared region non cached memory pool SR0. Memory has to be from SR0 pool as only osd window pointers are passed to the OSD algorithm running on DSP. No physical copy of OSD data is done on DSP side. To learn about SR0 pool, refer to 5.2 Memory Map Details of DVR_RDK_McFW_UserGuide.pdf.

Memory allocation can be done using Vsys_allocBuf() function call. This function call can only be made after System_init() call has been invoked in McFw layer. This constraint is because necessary ipc layer initialization is required before requesting for memory from SR0 buffer pool. System_init() call is part of Vsys_create() module in case you are following example sample application as demonstrated in <DVR_RDK>/dvr_rdk/demos/mcfw_api_demos/mcfw_demo/demo_vcap_venc_vdec_vdis.c file.

After Vsys_Create()/System_init() call is made, user can configure osd windows and allocate buffer from SR0 pool. Following are the steps -

- Allocate the memory for OSD windows
- Set following OD windows details of each channel
    - startX
    - startY
    - width
    - height
    - lineOffset
    - globalAlpha
    - transperencyEnable
    - enableWin
    - format
    - addr
- Fill the memory with OSD data

- Pass the OSD windows parameters to algLink dynamically using Vcap_setDynamicParamChn() API

Note that when data is in YUV422 interleaved format, only addr[0][0] need to be allocated and initialized. When data is in YUV420 semi planar format, both addr[0][0] (for Y plane) and addr[0][1] (for UV plane) needs to be allocated and initialized.

If application use case is not limited by DDR BW and for simplification purpose, integrator wants to use one single OSD window rather using multiple windows; it can be achieved by following settings as below

`AlgLink_OsdChWinParams::numWindows` as 1
`AlgLink_OsdChWinParams::winPrm[chId]::startX` as 0
`AlgLink_OsdChWinParams::winPrm[chId]::startY` as 0
`AlgLink_OsdChWinParams::winPrm[chId]::width` as video frame width
`AlgLink_OsdChWinParams::winPrm[chId]::Height` as video frame height
`AlgLink_OsdChWinParams::winPrm[chId]::lineOffset` as osd buffer pitch

4 Run Time OSD Window Parameter Change

OSD window parameters for each channel belonging to structure AlgLink_OsdChWinParams can be changed run time. AlgLink supports only one single cmd ALG_LINK_OSD_CMD_SET_CHANNEL_WIN_PRM for all parameter changes for osdLink. osdLink copies entire AlgLink_OsdChWinParams structure passed to it by A8 in it's internal state, **so user should make sure that all other window parameters that did not require changes have valid original data**.

OSD is part of Video Capture (VCap) sub-system in McFW. OSD windows run time parameters can be changed either using video capture Vcap_setDynamicParamChn() API with command as VCAP_OSDWINPRM or directly using System_linkControl() API.

Below is sample vcap McFW API usage for run time osd parameter changes. &params is pointer to updated AlgLink_OsdChWinParams structure with channle id as chId. Refer to `Demo_captureSettings()` module in `<DVR_RDK>/dvr_rdk/demos/mcfw_api_demos/mcfw_demo/demo_capture.c` file for further usage.

```
Vcap_setDynamicParamChn(chId, &params, VCAP_OSDWINPRM);
```

Example usage for direct System_linkControl is as shown below. Refer to module `Vcap_getDynamicParamChn()` in file `<DVR_RDK>/dvr_rdk/mcfw/src_linux/mcfw_api/ti_vcap.c` file for further usage.

```
System_linkControl(dspAlgId,
                                   ALG_LINK_OSD_CMD_SET_CHANNEL_WIN_PRM,
                                   &osdChWinPr,
                                  sizeof(AlgLink_OsdChWinParams),
                                   TRUE
                                   );
```

5 Alpha Blending

The global alpha value is valid for the overlay windows. User can set the value from 0 to 128. The value entered is U1Q7 format where the MSB is integer value and other seven bits are fraction.

User Input	Hex Equivalent	Decimal Interpretation	Alpha Blending Percentage	Output seen on Display
0	0x00	0.0	0%	OSD window is seen 100% and video behind the OSD window is not seen
64	0x40	0.5	50%	OSD window is blended 50%
128	0x80	1.0	100%	OSD window is not seen and video is fully seen

6 Privacy Mask Implementation Using OSD

Privacy Mask can be implemented using this OSD feature. Following steps are needed for Privacy Mask feature:

- Allocate memory for the overlay content as described in Section 3.2
- Fill the memory buffer with a fixed pattern (grey color or black color or any other pattern). **User need to ensure that the correct YUV data is filled for the pattern, based on YUV422ILE or YUV420SP format**
- User can set the same buffer address of the overlay mask for all the windows for a channel. The same buffer address can be used across multiple channels.
- User need to configure the following run time parameters to set the correct mask region on the video data

  `AlgLink_OsdChWinParams::winPrm[chId]::startX` as 0
  `AlgLink_OsdChWinParams::winPrm[chId]::startY` as 0
  `AlgLink_OsdChWinParams::winPrm[chId]::width` as video frame width
  `AlgLink_OsdChWinParams::winPrm[chId]::Height` as video frame height
  `AlgLink_OsdChWinParams::winPrm[chId]::lineOffset` as osd buffer pitch