TW2880P-BC2-GR Chip Application Note

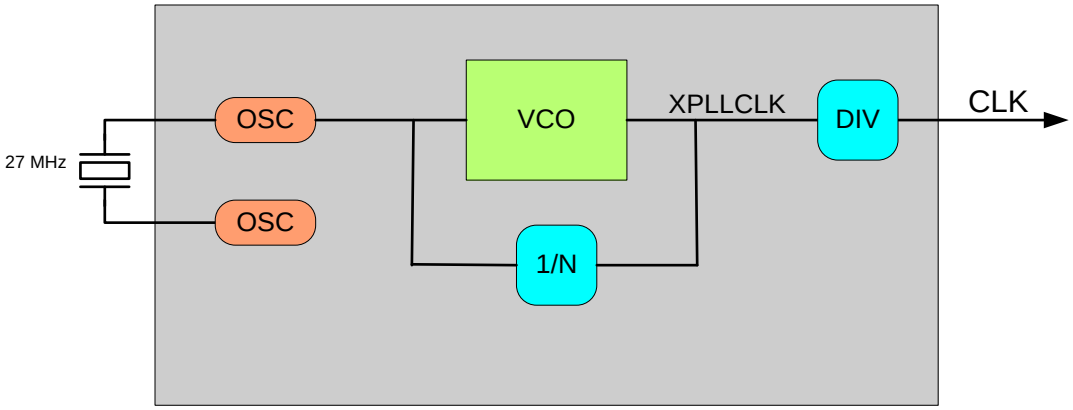
Table of Contents

List of Figures

List of Tables

Section 1: Clockgen and PLL

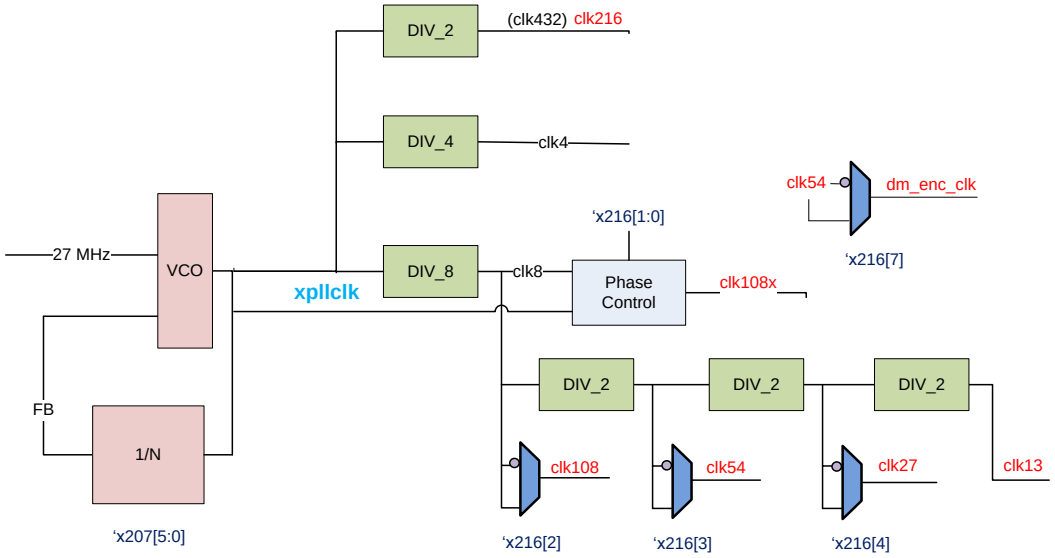
Introduction



TW2880C has three clock domains, they are, system clock domain, memory clock domain and video clock domain. Each clock domain support different kind of functional units. The clocks are generated from three different free running PLLs. The high-speed clock after the VCO stage will go through a series divider and phase select before reach the final circuit. Now we will walk through each clock domain in detail.

SCLK

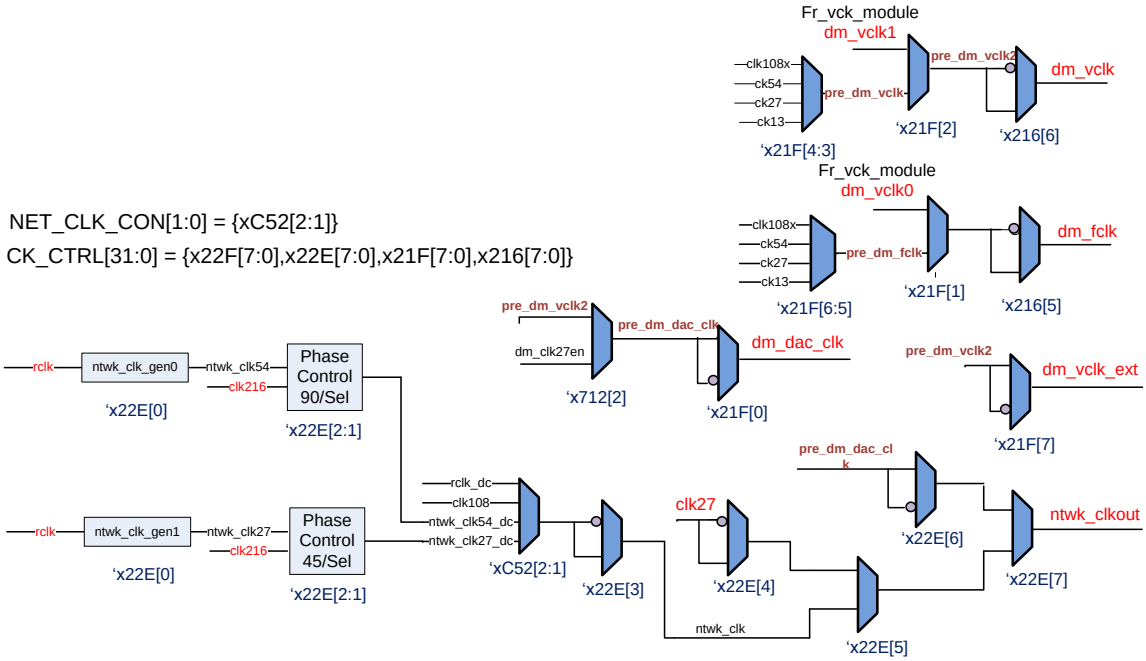
System clock is used throughout the TW2880. The idea is we will also use clocks with multiple of 27 in this clock group. The reason is obvious; 27 MHz clock is the data clock for the BT.656 standard. Because of these characteristics, some video decoders or CVBS output sections will use this clock group.



Clock Listing

12 clocks are generated from this clock group.

- 108 MHz system clock
- 54, 27, 13.5 MHz system using in input / output blocks
- 108 MHz system clock with phase control, used in recording output unit.
- High speed 216 MHz scaler clock (can be 432 MHz)
- Dual monitor TV encoder clock
- Dual monitor video clock
- Dual monitor fast clock for scaler
- Dual monitor DAC clock
- Dual monitor external video clock for VGA
- Network port output clock



Register Setting for SPLL

In normal cases, [0x207] bit[5:0] is designed to have default value of 5'd31 as this will make xpllclk 864 MHz and after divided by 8 circuits will create 108 MHz system clock.

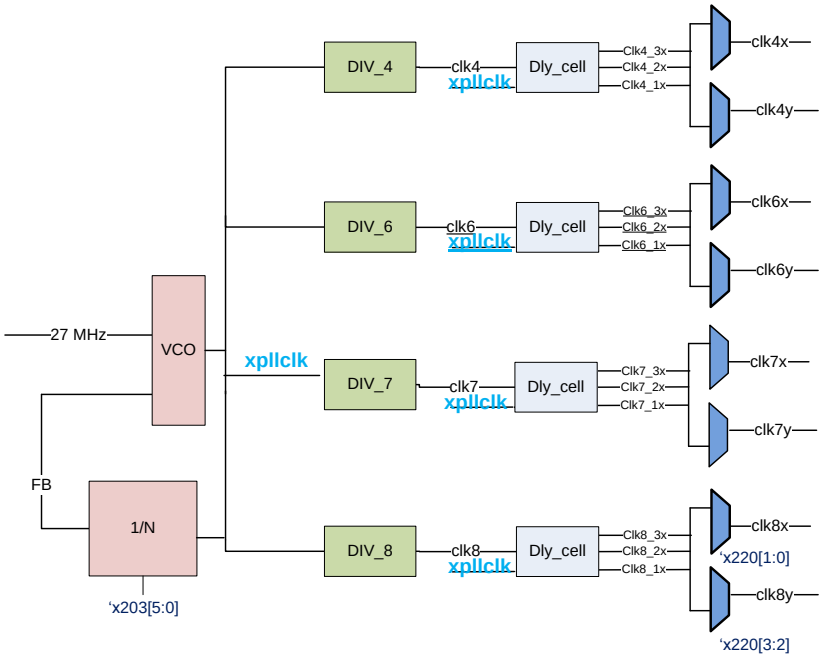
[0x216] bit 2, 3, 4 are used to select the phases of 108 MHz, 54 MHz, and 27 MHz clock.

[0x216] bit 1, 0 are used to select the phases of 108 MHz clock, this to adjust the record port clock / data relationship.

[0x21F] bit 2:1 are used to select the source of dual monitor clock, if CVBS is wanted, we should use the clock generated from the SCLK group. If display mode other than 27 MHz related then we should set these two bits to one and use clock generated from VCLK group.

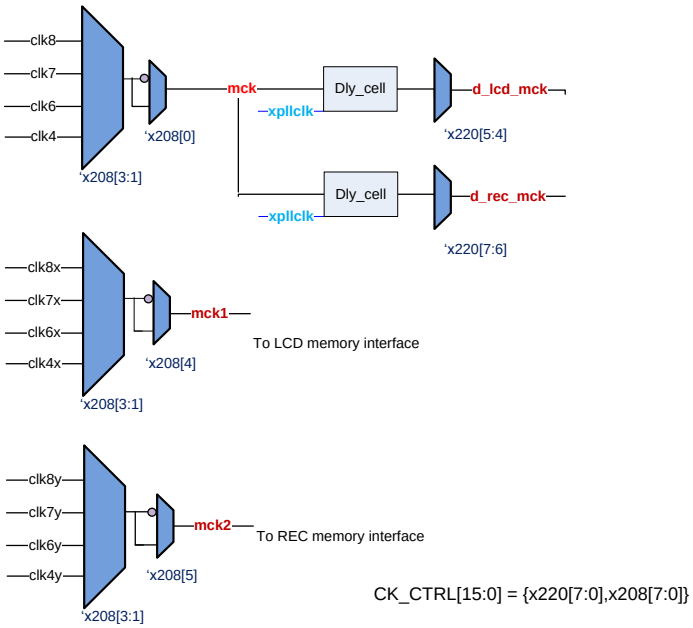
MCLK

Introduction



The memory clock range supported in TW2880C is between 133 – 200 MHz. User needs to program the multiplier register x203[5:0] and choose the desired divider to generate frequency. One thing needs to remember is the larger the divider, the more steps in the delay control. Five clocks are needed to adjust in a TW2880C system. They are:

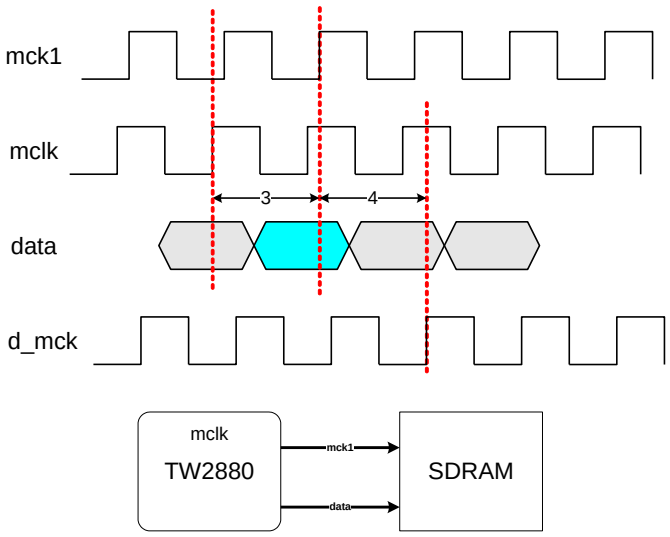
- Master clock for internal memory related blocks
- Clock for external SDRAM to use (Display side)
- Clock for external SDRAM to use (Recording side)
- Delayed version of display memory clock for latching incoming data
- Delayed version of recording memory clock for latching incoming data



Master Clock Calculation

Follow are the examples of popular master clock frequencies based on SDRAM speed grade.

For 133 MHz (-7.5 ns)				
27	19	4	128.25	
27	29	6	130.50	
27	34	7	131.14	
27	39	8	131.63	**Select**
For 166 MHz (-6 ns)				
27	24	4	162.00	
27	37	6	166.50	**Select**
27	43	7	165.86	
27	48	8	162.00	
For 175 MHz (-5 ns)				
27	26	4	175.50	
27	39	6	175.50	**Select**
27	45	7	173.57	
27	52	8	175.50	
For 200 MHz (-5 ns)				
27	29	4	195.75	
27	44	6	198.00	
27	52	7	200.57	
27	59	8	199.13	**Select**

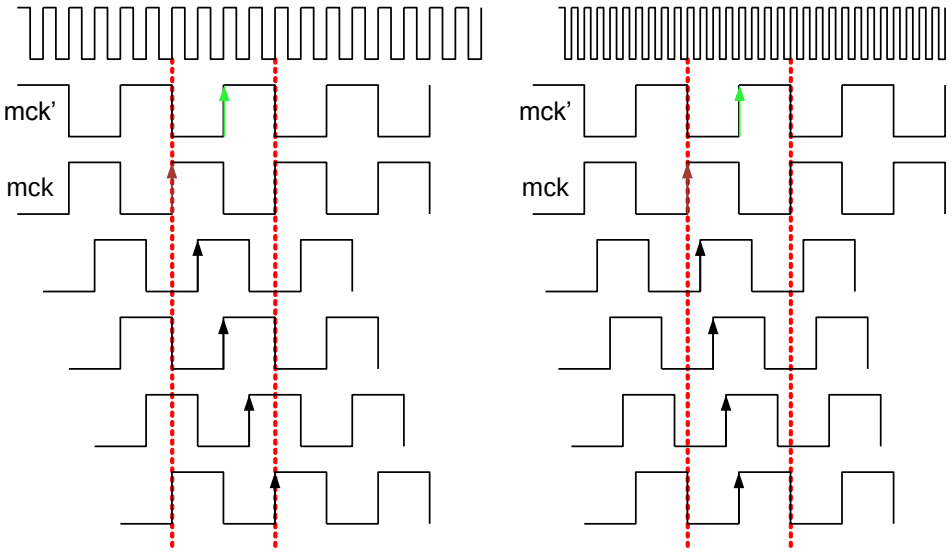


Clock Relationship

Two derivative clocks need to be adjusted to make the memory system work:

1. TW2880 to SDRAM: command, write data, controlled by 0x220[1:0] on the display, 0x220[3:2] for the record.
2. SDRAM to TW2880: read data, controlled by 0x220[5:4] on the display, 0x220[7:6] for the record.

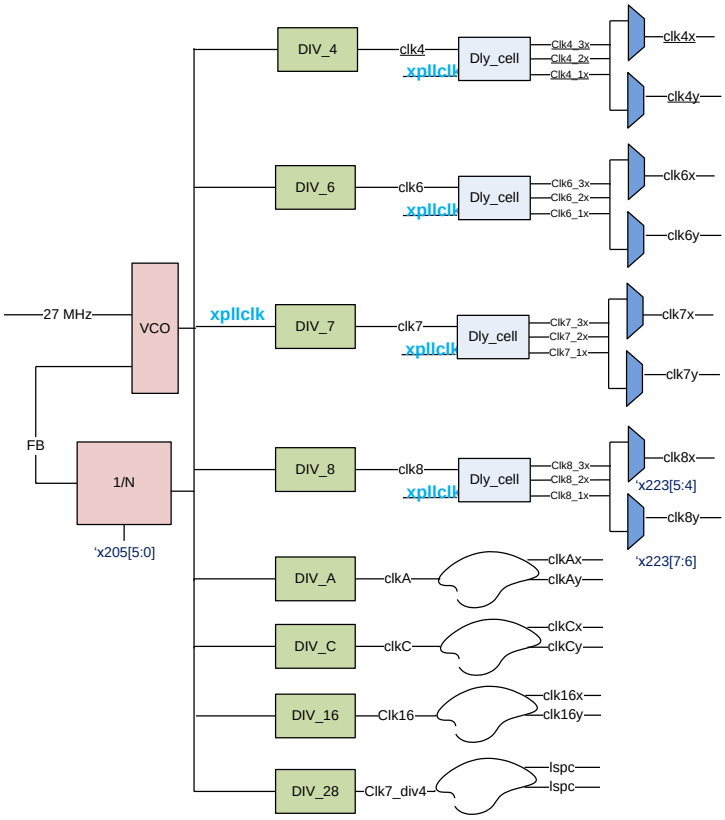
The steps are determined by divider, for example, divided by 4 you have only four steps, divided by 8, 8 steps.

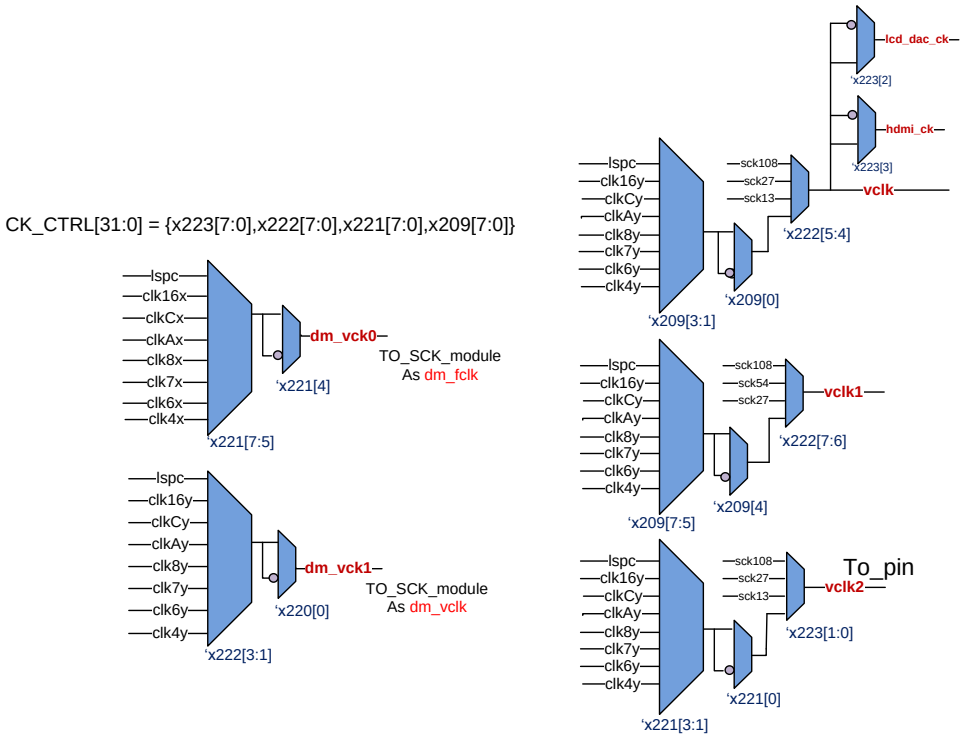


VCLK

The VCLK in TW2880C has the most complicated clock tree in the three as many exact frequencies are needed. To use it user needs to program the multiplier register x205[5:0] and choose the desired divider to generate frequency. 8 dividers are provided to generate correct clock for display. All together, seven clocks are generated:

- Clock for internal video related clock (vclk)
- Clock for VGA DAC
- Clock for HDMI block
- VCLK1 (not used)
- Clock for digital interface
- Two other dual monitor clock mux with SCLK





Popular Main Display Clocks

From the table shown in the following, TW2880C clock generation module can support most VESA standard resolution for the main display and HDMI TV Standard by selecting the proper VCK_N and VCK_Q values.

	Res	FRS	PCLK	VCK_N	VCK_Q	PCLK2	Diff
4:3	640x480	50Hz	19.75	23	28	22.18	-2.43
	640x480	60Hz	23.88	25	28	24.11	-0.23
	800x600	50Hz	31.13	32	28	30.86	0.27
	800x600	60Hz	38.13	40	28	38.57	-0.45
	1024x768	50Hz	51.75	23	12	51.75	0.00
	1024x768	60Hz	64.13	38	16	64.13	0.00
	1280x960	50Hz	83.00	37	12	83.25	-0.25
	1280x960	60Hz	102.00	38	10	102.60	-0.60
	1400x1050	50Hz	99.75	37	10	99.90	-0.15
	1400x1050	60Hz	122.50	32	7	123.43	-0.93
	1600x1200	50Hz	132.38	39	8	131.63	0.75
	1600x1200	60Hz	160.88	24	4	162.00	-1.13
	1600x1200r	60Hz	130.38	29	6	130.50	-0.13
	848x480	60Hz	31.50	33	28	31.82	-0.32
	1064x600	60Hz	51.00	30	16	50.63	0.38
	1280x720	50Hz	60.38	27	12	60.75	-0.38
	1280x720	60Hz	74.38	33	12	74.25	0.13
	1360x768	50Hz	69.50	31	12	69.75	-0.25
	1360x768	60Hz	84.63	25	8	84.38	0.25
	1704x960	50Hz	110.25	41	10	110.70	-0.45
	1704x960	60Hz	134.88	30	6	135.00	-0.13
	1864x1050	50Hz	133.50	30	6	135.00	-1.50
	1864x1050	60Hz	163.25	24	4	162.00	1.25
	1864x1050r	60Hz	131.13	34	7	131.14	-0.02
	1920x1080	50Hz	141.38	42	8	141.75	-0.38
	1920x1080	60Hz	172.73	38	6	171.00	1.72
	1920x1080r	60Hz	138.63	36	7	138.86	-0.23
16:10	768x480	50Hz	23.63	24	28	23.14	0.48
	768x480	60Hz	28.63	30	28	28.93	-0.30
	960x600	50Hz	37.00	38	28	36.64	0.36
	960x600	60Hz	45.88	27	16	45.56	0.31
	1152x720	60Hz	67.25	25	10	67.50	-0.25
	1680x1050	50Hz	120.13	31	7	119.57	0.55
	1680x1050	60Hz	147.00	38	7	146.57	0.43
	1680x1050r	60Hz	119.13	44	10	118.80	0.33
	1728x1080	60Hz	155.50	23	4	155.25	0.25
	1728x1080r	60Hz	125.75	28	6	126.00	-0.25
	1920x1200	50Hz	158.00	41	7	158.14	-0.14
	1920x1200	60Hz	193.13	43	6	193.50	-0.38
	1920x1200r	60Hz	154.13	40	7	154.29	-0.16
5:4	1280x1024	50Hz	89.38	33	10	89.10	0.28
	1280x1024	60Hz	108.88	24	6	108.00	0.88
15:9	1280x768	50Hz	65.13	29	12	65.25	-0.13
	1280x768	60Hz	80.13	24	8	81.00	-0.88
HDMI	1920x1080p	60Hz	148.50	44	8	148.50	0.00
	1920x1080i	30Hz	74.25	44	16	74.25	0.00

Dual Monitor Setting

Dual monitor's clock setting is a little bit different from the main display's settings. As mentioned before in the SCLK group, dual monitor block can take SCLK as clock source if CVBS output is needed. This is done by setting [0x21F] bit 2:1 to zero. When dual monitor block is running at frequencies other than multiples of 27 MHz, for example, driving a progressive VGA monitor, you need to set [0x21F] bit 2:1 to one and select the output from VCLK VCO. The VCO frequency will be the same for both displays and only the dividers are different. For example, the

main display is set at 1080P and the Dual monitor is set at 1280x1024 resolution. Then, select the SCLK (108 MHz) as the dual monitor video clock and set the main display frequency as listed in the following Table.

MAIN DISPLAY RESOLUTION	FRAME RATE	MAIN PCLK	VCK_N	VCK_Q	DUAL MON. RESOLUTION	DUAL MON. PCLK	REG SETTING
1920x1080P	60Hz	148.50	44	8	800x600x60Hz	42.4 MHz	x205= 2B (VCK_N) x209= 66 (div by 8) x221= C6 (div by 28) x222= 0C x216= 02 x21f= 06 (dm sel VCK)
1920x1080P	60Hz	148.50	44	8	1280x1024x60Hz	108	x205= 2B (VCK_N) x209= 66 (div by 8) x221= C6 x222= 0C x216= 02 x21f= 78 (dm sel SCK)
1280x1024	60Hz	108	32	8	1024x768x70Hz	72	x205= 1F (VCK_N) x209= 66 (div by 8) x221= A6 (div by 12) x222= 0A x216= 02 x21f= 06 (dm sel VCK)

When the Dual monitor is used to drive an analog TV, the frequency setting is simply by choosing the 54 Mhz as the dm_vck output. The registers x21F[4:2]='x4

Using SCLK Clock Group For Dual Monitor Clock

Usually, the SCLK frequency should not be changed either for system, record ports, or SPOT displays stability.

The performance of the above ports as well as the host bandwidth will also be affected if SCLK frequency is changed. For Dual monitor, if SCLK clock is used, the registers x21F need to be set properly.

Example

As an illustration, if the main display is in 1080p mode so the output frequency is 148.5 MHz, the dual monitor is VGA with 640x480@72Hz. The dual monitor clock is 31.5 MHz. Therefore, in addition to correcting the RGB register settings, the clock gen registers setting are:

MCLK REGISTERS

REG ADDRESS	DEFAULT SETTING	NEW SETTING	COMMENTS
x202	‘x12		MCLK M
x203	‘x23		MCLK N
x208	‘x02		MCLK CTRL[7:0]
x220	‘x00		MCLK CTRL[15:8]

Default setting is 162 MHz.

VCLK REGISTERS

REG ADDRESS	DEFAULT SETTING	NEW SETTING	COMMENTS
x204	‘x0d		VCLK M
x205	‘x27	‘x20	VCLK N
x209	‘x88	‘x02	VCLK CTRL[7:0]
x221	‘x88	‘xc0	VCLK CTRL[15:8]
x222	‘x00	‘xcc	VCLK CTRL[23:16]
x223	‘x00	‘x00	VCLK CTRL[31:24]

The main display frequency is calculate as 27 * 33 / 6 = 148.5 MHz

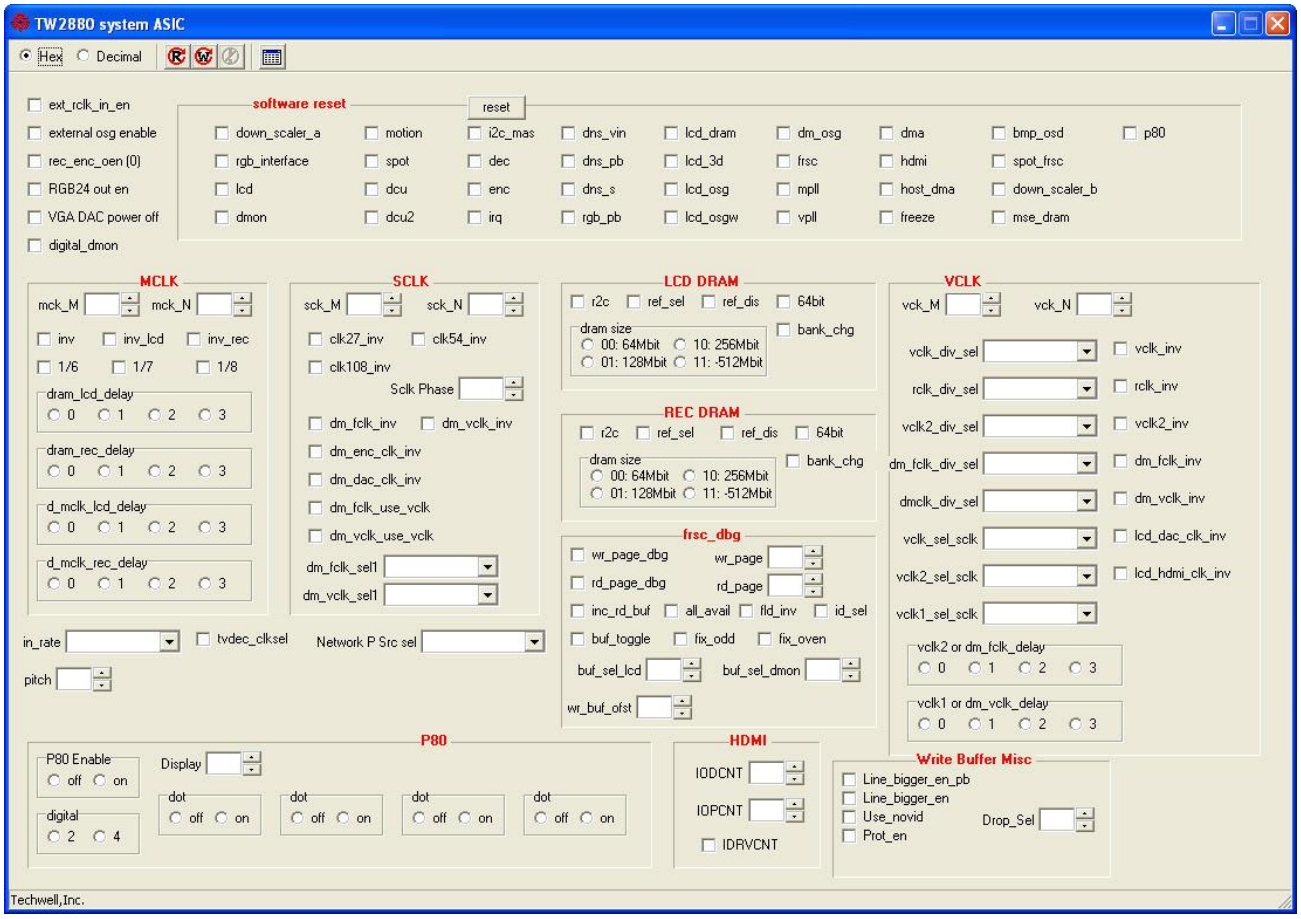
The Dual monitor frequency is selected as 27 * 33 / 28 = 31.8 MHz

VCLK REGISTERS

REG ADDRESS	DEFAULT SETTING	NEW SETTING	COMMENTS
x206	‘x0f		SCLK M
x207	‘x1f		SCLK N
x216	‘x00		SCLK CTRL[7:0]
x21F	‘x78	‘x07	SCLK CTRL[15:8]
x22E	‘x00		SCLK CTRL[23:16]
x22F	‘x00		SCLK CTRL[31:24]
X712	‘x00	‘x04	DM_LCD

Techwell Terminal Tool Setting

Layout of the CFG File



Explanation

In the Terminal, there are bold red characters, which describe the functions along with registers values that can be selected or white boxes that may be filled in to control the behavior of the functions.

- In the software reset section, a checked box would reset the specified module until the box is un-checked.
- In the MCK section, the mck_M is for the mck duty cycle control. The mck_N is the multiplier to the MPLL with the 27 MHz oscillator input, the PLL frequency output equals (mck_N + 1) x 27 MHz. The MPLL output frequency must less 1200 MHz and higher than 600 MHz for the MPLL to operate stable. After the mck_N is set, then final mck frequency is derived from check one of the division 1/6, 1/7, or 1/8. By properly choose the mck_N value and the division, the optimal MCK frequency can be acquired.
- In the MCK section, there are four other delay control selections, which are used for timing control to the DRAM interface. Two MCK outputs delay control such as dram_lcd_delay and dram_rec_delay, are used to clock phase delay respect to data and control signals. When delay 0 is check, the MCK output and the 64b data and control signals are aligned as the chip layout timing. If the PCB timing is not ideal and needs to be adjusted, then the MCK phase control can be set to 90, 180, or 270 degrees with respect to the data and control signals.
- There are two d_mclk_lcd_delay and d_mclk_rec_delay, which are used to adjust the input data to be latched by the TW2880. If 0 is checked, the data are latched by the MCK, otherwise, the data are latched by the delayed MCK.

- In the SCLK section, the M, N values are set as the MCK. The SCLK is set to 108 MHz as the default and should not be changed. Others boxes can be checked or un-checked to tune the inputs or outputs timing as needed. There are some boxes called dm_enc_clk_inv ..., which are used by the DUAL MONITOR module timing control.
- In the VCLK section, the M, N values are set as the MCK. All the boxes in the section are used to control the data timing respect to the VCLK or Dual_Monitor VCLK.
- In the LCD_DRAM or REC_DRAM section, dram size indicates the DRAM size used on the board.
- The 64-bit box is checked indicates the TW2880 interface to DRAM in 64-bit bus. Otherwise, it is 32-bit bus. Other buttons are for internal use only. The SDRAM controller default value should be good enough for everyday use.
- In the frsc_dbg section, the options for write pages and read pages can be controlled for debug purpose.
- In the P80 section. This is the LED control module used for debugging purposes.
- In the HDMI section, the setting is to control the HDMI output. The IODCNT control the output currents, the IOPCNT control the De-Emphasis and IDRVCNT Disable the De-emphasis if the box is checked.
- In the Write Buffer Misc section, these registers are used to protect SDRAM off-screen memory from being overwritten by run away RGB write FIFO process.

Section 2: PCB Layout Guide

Introduction

TW2880 is a complicated VLSI device whose inputs and outputs include several high frequency signal groups. To achieve the best result, the traces and associated discrete components need to carefully designed, placed and connected. To further complicate the board design, there are several power rails used either by digital or by analog functions. This guide served as a general reference for the board designer of TW2880.

Placement Suggestions

The first suggestion in designing TW2880 related PCB is clearly identifying the major functions that you want to include in this board. The second step involves planning the input / output connectors in a way such that do not let the signal trace crossed by traces in other groups if you can, whether it is signal traces or power traces. These are very important steps for getting a clean video output because crosstalk noise between the groups can easily destroy a board.

For the people not familiar with the term, crosstalk is the unwanted coupling of signals between parallel traces. To reduce crosstalk in TW2880 related boards, use dual-stripline layouts, which have two signal layers next to each other, route all traces perpendicular, increase the distance between the two signal layers, and minimize the distance between the signal layer and adjacent plane. Use the following steps to reduce crosstalk in either microstrip or stripline layouts:

- Widen spacing between signal lines as much as routing restrictions will allow. Try not to bring traces closer than three times the dielectric height.
- Design the transmission line so that the conductor is as close to the ground plane as possible. This technique will couple the transmission line tightly to the ground plane and help decouple it from adjacent signals.
- Use differential routing techniques where possible, especially for critical nets (i.e., match the lengths as well as the gyrations that each trace goes through).
- If there is significant coupling, route single-ended signals on different layers orthogonal to each other.

Minimize parallel run lengths between single-ended signals. Route with short parallel sections and minimize long, coupled sections between nets. Crosstalk also increases when two or more single-ended traces run parallel and are not spaced far enough apart. The distance between the centers of two adjacent traces should be at least four times the trace width. To improve design performance, lower the distance between the trace and the ground plane to under 10 mils without changing the separation between two traces.

Signal Integrity

For a single-ended trace, like clock transmission line, it could be improved using the following guidelines:

- Keep clock traces as straight as possible. Use arc-shaped traces instead of right-angle bends.
- Do not use multiple signal layers for clock signals.
- Do not use via in clock transmission lines. Via can cause impedance change and reflection.
- Place a ground plane next to the outer layer to minimize noise. A “grow to fill” function in the layout tool provides exactly this. If you use an inner layer to route the clock trace, sandwich the layer between reference planes.
- Terminate clock signals to minimize reflection.
- Use point-to-point clock traces as much as possible.

Power Regulator and Noise Filtering

TW2880 has 5 voltage tails for analog and digital functions. To get the best possible result but still keep the power consumption down, we suggestion using the switching regulator in the beginning of the power network and switching to LDO in the end to reduce the switching noise. This is especially true if the power is used for analog function. To

decrease the low frequency (below 1 kHz) noise caused by the power supply, filter the noise on power lines at the point where the power connects to the PCB and to each device. Place a 100 µF electrolytic capacitor where the power supply lines enter the PCB and after the first stage voltage regulator VCC signal. (Capacitors not only filter low-frequency noise from the power supply, but also supply extra current when many outputs switch simultaneously in a circuit.)

To filter power supply noise, use a non-resonant, surface-mount ferrite bead large enough to handle the current in series with the power supply. Place a 10 to 100 µF bypass capacitor next to the ferrite bead. (If proper termination, layout, and filtering eliminate enough noise, you do not need to use a ferrite bead.) The ferrite bead acts as a short for high frequency noise coming from the VCC source. Any low frequency noise is filtered by a large 10 µF capacitor after the ferrite bead. Usually, elements on the PCB add high-frequency noise to the power plane. To filter the high-frequency noise at the device, place decoupling capacitors as close as possible to each VCC and GND pair.

Power Distribution

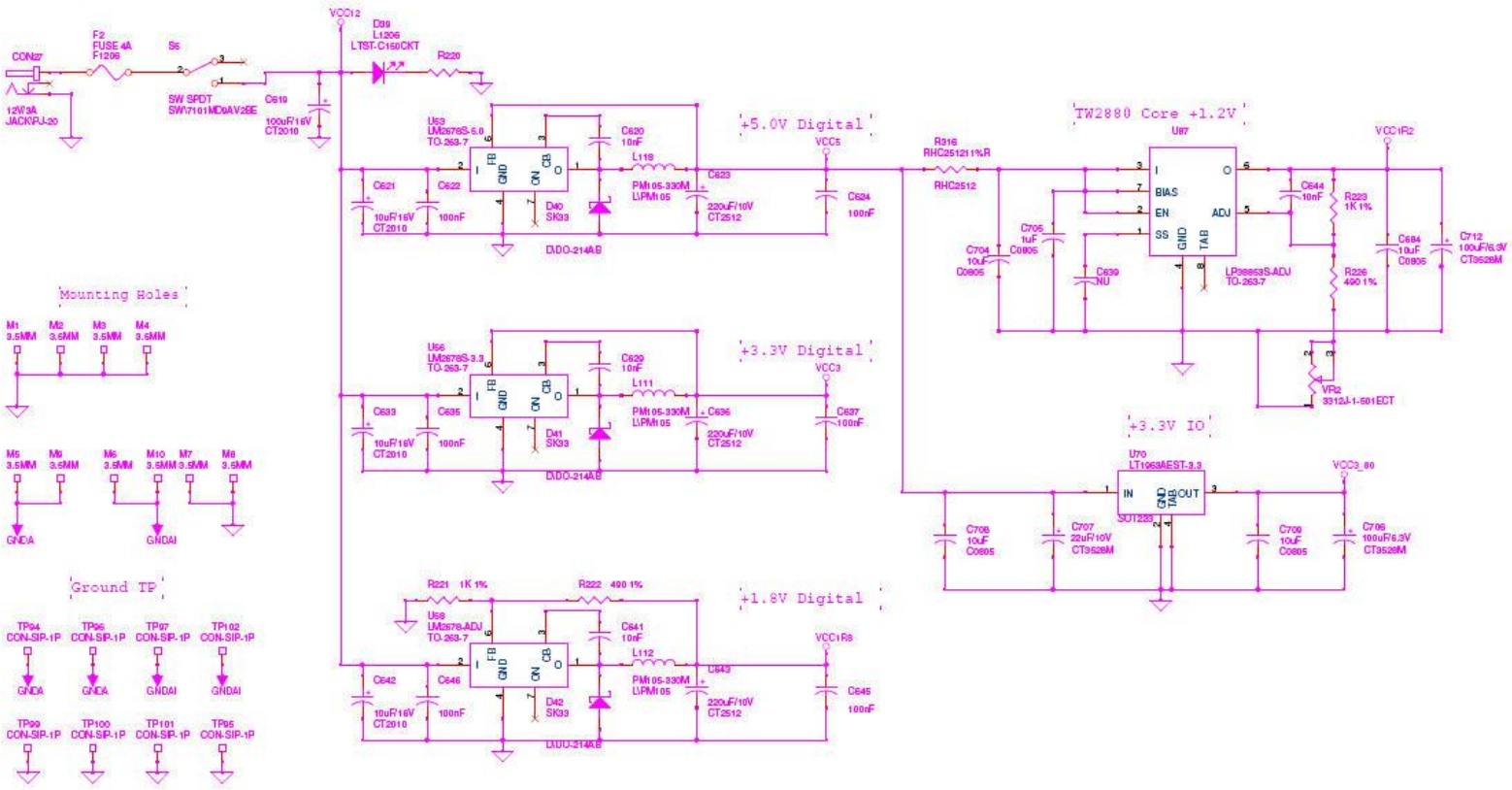
You can distribute power throughout the TW2880 PCB with either power planes or a power bus network. When designing TW2880 related PCB, a multi-layer PCBs that consist of two or more metal layers that carry VCC and GND to TW2880 is highly recommended. Because the power plane covers the full area of the PCB, its DC resistance is very low. The power plane maintains VCC and distributes it equally to all devices while providing very high current-sink capability, noise protection, and shielding for the logic signals on the PCB. It is recommended to use lower planes to distribute power. The power bus network, which consists of two or more wide metal traces that carry VCC and GND to devices, is often used on two-layer PCBs and is less expensive than power planes. When designing with power bus networks, be sure to keep the trace widths as wide as possible. The main drawback to using power bus networks is significant DC resistance. It is recommended to separate analog and digital power planes. For fully digital systems that do not already have a separate analog power plane, it can be expensive to add new power planes. However, you can create partitioned islands (split planes).

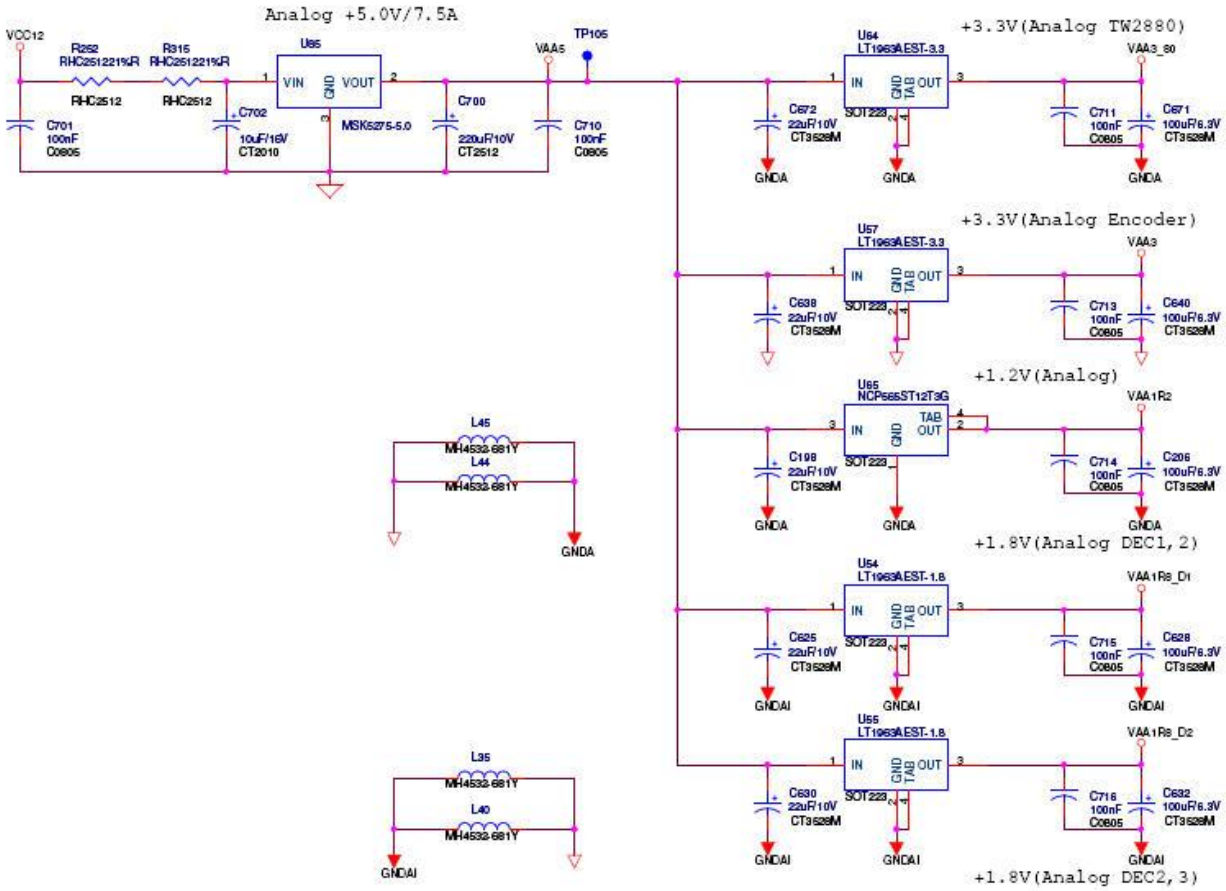
If your system shares the same plane between analog and digital power supplies, there may be unwanted interaction between the two circuit types. The following suggestions will help to reduce noise:

- For equal power distribution, use separate power planes for the analog (PLL) power supply. Avoid using trace or multiple signal layers to route the PLL power supply.
- Use a ground plane next to the PLL power supply plane to reduce power-generated noise.
- Place analog and digital components only over their respective ground planes.
- Use ferrite beads to isolate the PLL power supply from digital power supply.

TW2880 Power Rails

There are eleven voltage sources in a TW2880 HQ EV board. There are 5V digital, 3.3V digital, 3.3V TW2880 analog, 3.3V I/O, 1.8V digital, 1.8V analog, 1.2V TW2880 core, 1.2V TW2880 analog, 3.3V analog encoder, 1.8V analog decoder x2. We used a buck-switching regulator to create power source from external 12V DC adapter. In the final stage, we use many LDO to get the desired analog voltage. Please reference to the next two schematics. Please pay special attention to all analog power supplies to TW2880 and the I/O video chip, as this will determine the final visual effect.


+5.0V Digital
+3.3V Digital
+1.8V Digital
TW2880 Core +1.2V
+3.3V IO
Mounting Holes
Ground TP
LTST-C150CKT
SW SPDT
JACKPJ-20
LM2678S-5.0
LM2678S-3.3
LM2678-ADJ
LP38853S-ADJ
LT1963AEST-3.3
3312J-1-501ECT
PM105-330M
DIDO-214AB
SOT223


Analog +5.0V/7.5A
VCC12
R252
RHC2512
R315
RHC2512
C701
100nF
C0805
U85
VIN
GND
VOUT
MSK5275-5.0
C702
10uF/16V
CT2010
C700
220uF/10V
CT2512
TP105
VAA5
C710
100nF
C0805
+3.3V(Analog TW2880)
U64
LT1963AEST-3.3
IN
GND
TAB
OUT
SOT223
C672
22uF/10V
CT3528M
VAA3_80
C711
100nF
C0805
C671
100uF/6.3V
CT3528M
GNDA
+3.3V(Analog Encoder)
U57
LT1963AEST-3.3
C638
22uF/10V
CT3528M
VAA3
C713
100nF
C0805
C640
100uF/6.3V
CT3528M
+1.2V(Analog)
U65
NCP565ST12T3G
TAB
C198
22uF/10V
CT3528M
VAA1R2
C714
100nF
C0805
C206
100uF/6.3V
CT3528M
L45
MH4532-681Y
L44
MH4532-681Y
+1.8V(Analog DEC1,2)
U54
LT1963AEST-1.8
C625
22uF/10V
CT3528M
VAA1R8_D1
C715
100nF
C0805
C628
100uF/6.3V
CT3528M
GNDAI
U55
LT1963AEST-1.8
C630
22uF/10V
CT3528M
VAA1R8_D2
C716
100nF
C0805
C632
100uF/6.3V
CT3528M
L35
MH4532-681Y
L40
MH4532-681Y
+1.8V(Analog DEC2,3)

SDRAM

Introduction

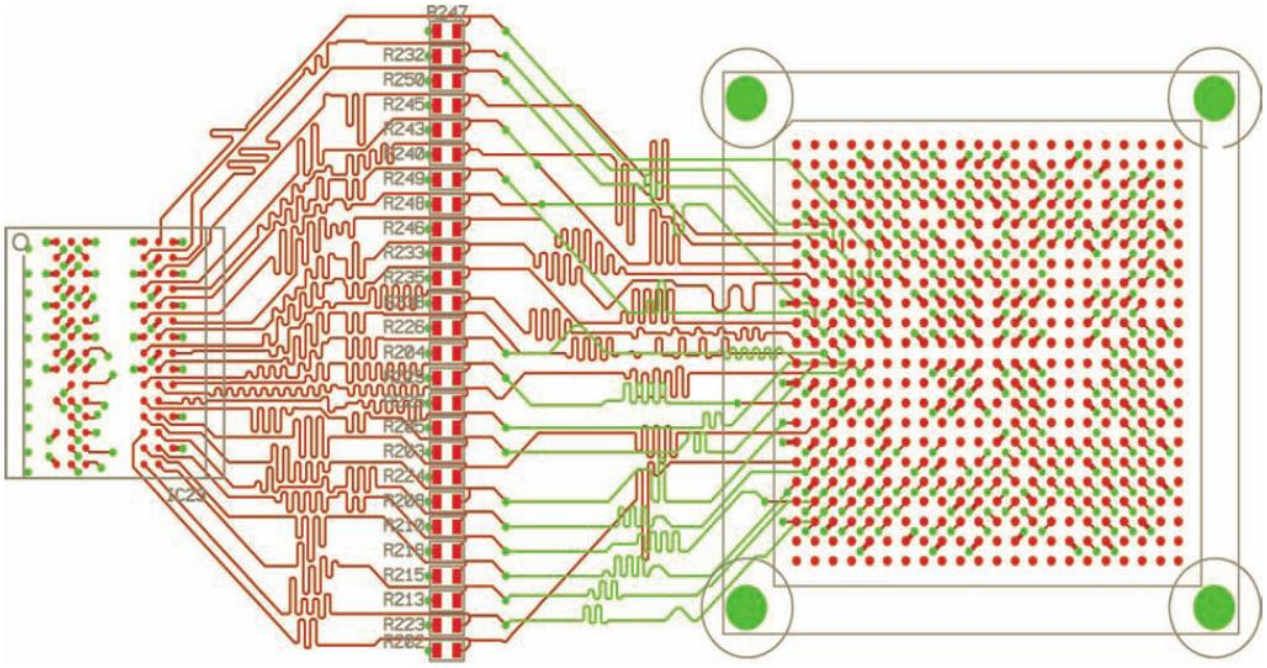
TW2880 has two 64-bit memory sub-systems, each can support up to 64M Byte SDRAM. To form a system, you can select x16 or x32 width SDRAM. In addition, to support all features in the chip, at least -6 speed grade device is needed. Use -5 device if you can find them.

Termination Resistors

Because the operating frequency is pretty high for SDR operation, certain rules need to be observed for trouble free results. The clock trace on the board needs to wider and short with respect to other trace. Secondly, all address / control lines to the SDRAM should be terminated. We suggest values of 22 ohms to soften the rising / falling edge of the signals.

Equi-Length Line Rule

TW2880 has various read / write timing adjustments built in to let user to program to fit different SDRAM into TW2880's environment. However, one rule the layout engineer needs to follow is the equi-length rule. Each address line and data line need to hand matched to similar total trace length in the board as the following diagram suggestive.



DAC

Introduction

TW2880 has 10 DACs built into the chip. Three DACs are used for main VGA output; three DACs are used for secondary VGA / CVBS and S-video. The remaining four DACs are used for SPOT monitors. The following is a suggestion of how to get good video quality and is applicable to every DAC on the chip.

Power Supply

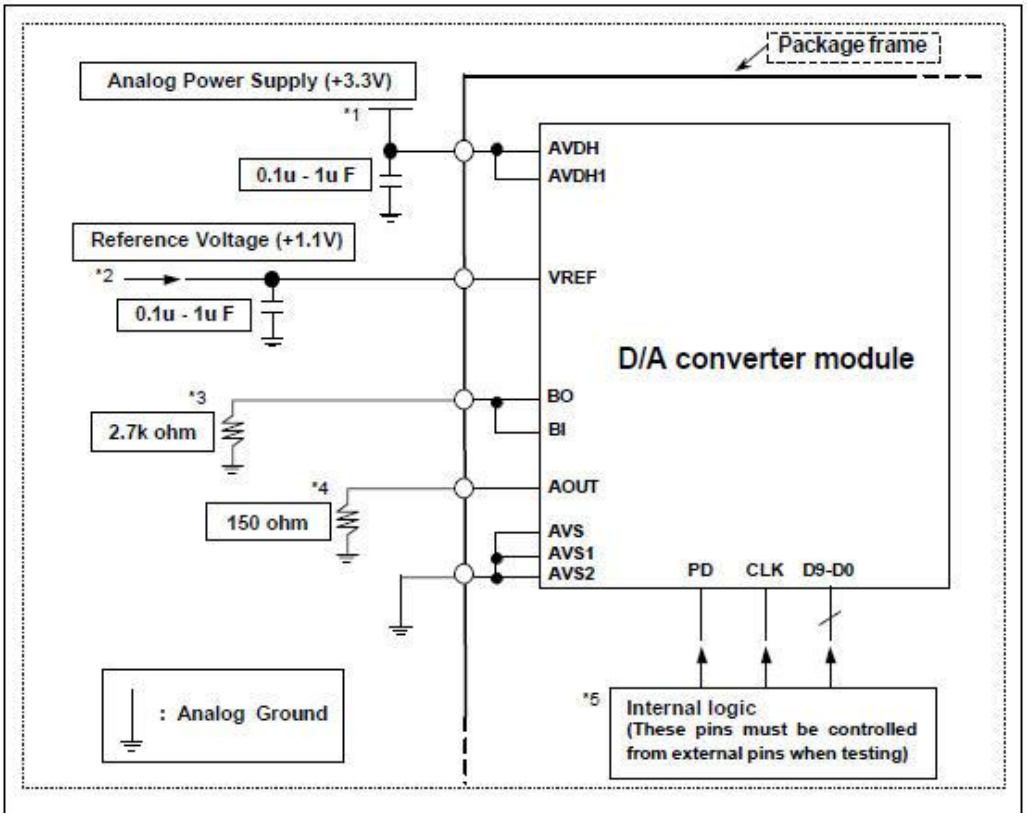
For accurate operation of D/A converter, we must pay special attention to the noise of analog power supply. The wiring impedance is the most important factor for its accuracy. It is recommended to use the high frequency type ceramic capacitor for decoupling to the analog ground. The decoupling capacitor must be as close as possible to the TW2880 to keep lead lengths to an absolute minimum.

The voltage of 1.1V must be drawn from a clean voltage source or the video result will be bad. This is especially true is high frequency mode is desired. This voltage source also needs to be de-coupled by using the high frequency type ceramic capacitor and the location should be close to the chip.

Proper Termination

Mismatched impedance between DAC and the monitor causes video signals to reflect back and forth along the lines, which will cause the annoying ringing effect at the TV or monitor. The ringing reduces the dynamic range of the receiver and can cause false triggering. To eliminate reflections, the impedance of the source (ZS) must equal the impedance of the trace (Zo), as well as the impedance of the load (ZL). The loading in TW2880 application is 37.5R as we are using the standard 75R double terminated scheme to cut down reflection.

Connection Example



PCB Layout Considerations

The TW2880 is dedicated video VLSI with many integrated functions. To complement the excellent performance of the TW2880, it is imperative that great care be given to the PCB layout. The diagram on the previous page shows a recommended connection diagram for the TW2880. The layout should be optimized for lowest noise on the TW2880 power and ground lines. This can be achieved by shielding the digital inputs and providing good decoupling. Shorten the lead length between groups of VAA and GND pins to minimize inductive ringing.

It is recommended to use at least 6-layer printed circuit board with a single ground plane. The ground and power planes should separate the signal trace layer and the solder side layer. Noise on the analog power plane can be further reduced by using multiple decoupling capacitors (see diagram on the previous page). Optimum performance is achieved by using 0.1 µF and 0.01 µF ceramic capacitors. Individually decouple each VAA pin to ground by placing the capacitors as close as possible to the device with the capacitor leads as short as possible, thus minimizing lead inductance. If a high frequency switching power supply is used, pay close attention to reducing power supply noise. A dc power supply filter (Murata BNX002) provides EMI suppression between the switching power supply and the main PCB.

RECOMMENDED ROUTING/LAYOUT RULES

- Do not run analog and digital signals in parallel.
- Use separate analog and digital power planes to supply power.
- Traces should run on top of the ground plane at all times.
- No trace should run over ground/power splits.
- Avoid routing at 90-degree angles.
- Minimize clock and video data trace length differences

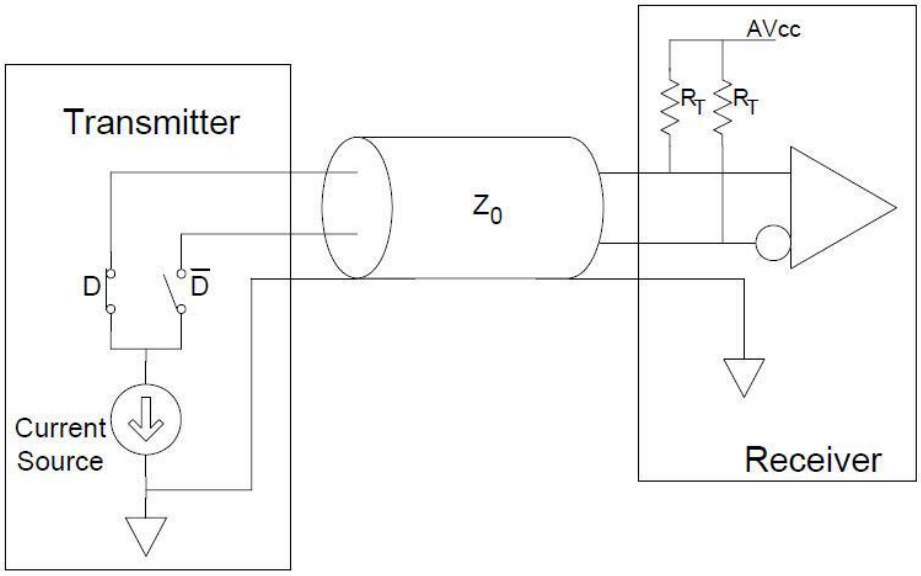
HDMI

General Description

The guidelines in this chapter apply to the following listed signals of TW2880' HDMI transmitter. These signals are TMDS (Transition Minimized Differential Signaling) and are open-drain outputs. Therefore, these signals need to be pulled up to 3.3 V power supply via resistors of 50Ω at the receiver side.

- EXCP
- EXCN
- EXP0
- EXN0
- EXP1
- EXN1
- EXP2
- EXN2

The conceptual schematic of a TW2880 enabled transmitter / receiver pair is shown below. TMDS technology uses current drive to develop the low voltage differential signal at the Sink side of the DC-coupled transmission line. The link reference voltage AV_{CC} sets the high voltage level of the differential signal, while the low voltage level is determined by the current source of the HDMI Source and the termination resistance at the Sink. **The termination resistance (R_T) and the characteristic impedance of the cable (Z_0) must be matched.**



Signal Integrity

The basic rules related to the layout of transmission lines on a printed circuit board are explained below.

IMPEDANCE CONTROL

The characteristics impedance of the transmission lines must be differential impedance of 100Ω±10% as a rule. For proper characteristics impedance, use the strip line or microstrip line structure. Study which of them should be used on a case-by-case basis, considering the package and the numbers of the pins and macros.

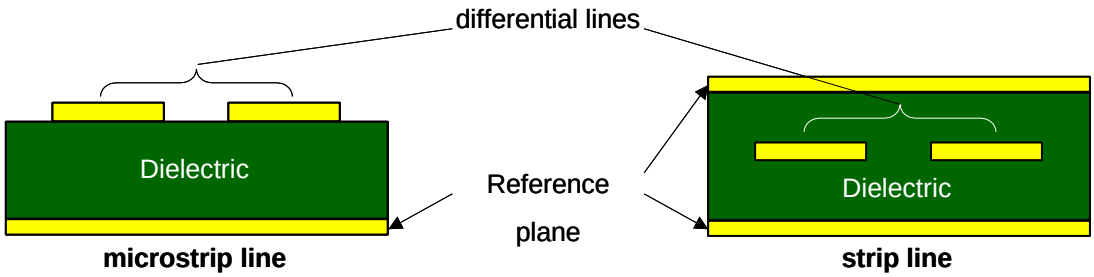


FIGURE 1. DIFFERENTIAL LINE STRUCTURES

45°BENDS

At corner areas, keep the bending degrees of the transmission lines up to 45°.

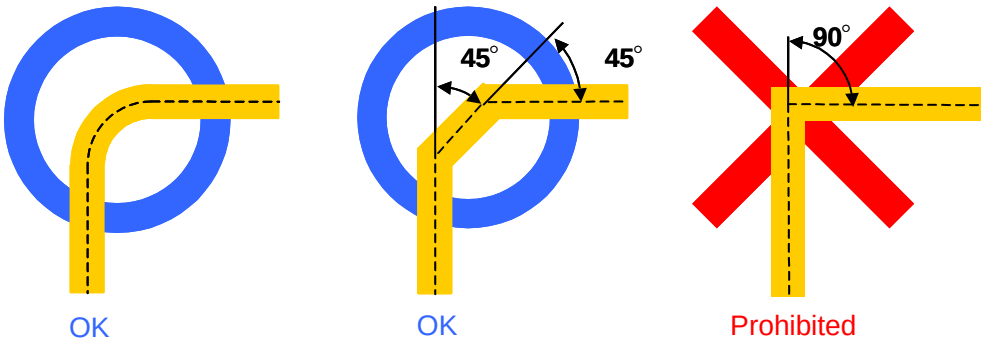


FIGURE 2. CORNER PATTERNS

Also at corner areas, keep the same space between the differential lines.

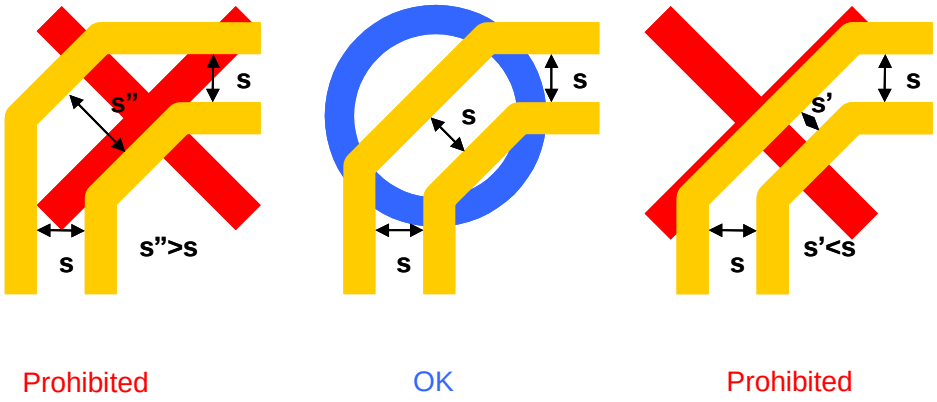


FIGURE 3. SPACE BETWEEN DIFFERENTIAL LINES AT CORNER AREAS

SKEW CONTROL

Eliminate the skew between the clock channel and the data channels. If an inter-channel skew exists between the clock and data within the LSI package, correct the wiring of the lines on the printed circuit board. When meander lines are used, keep at least 5W of spaces between the meander line patterns.

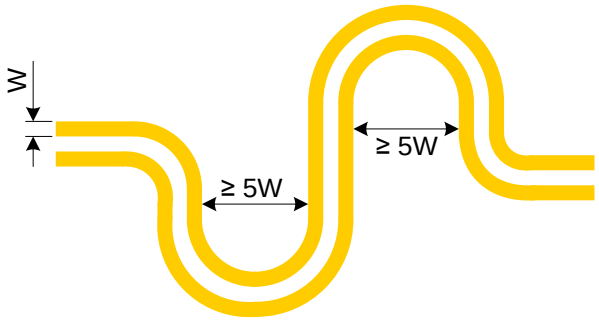


FIGURE 4. MEANDER LINES

For the inter-channel skew between the clock and data, refer to the specification documents for each product.

Concerning the skews occurring in single-ended areas of the differential lines around the BGA, adjust them within each single-ended area. Do not adjust the skew between the differential lines occurring in the corner areas.

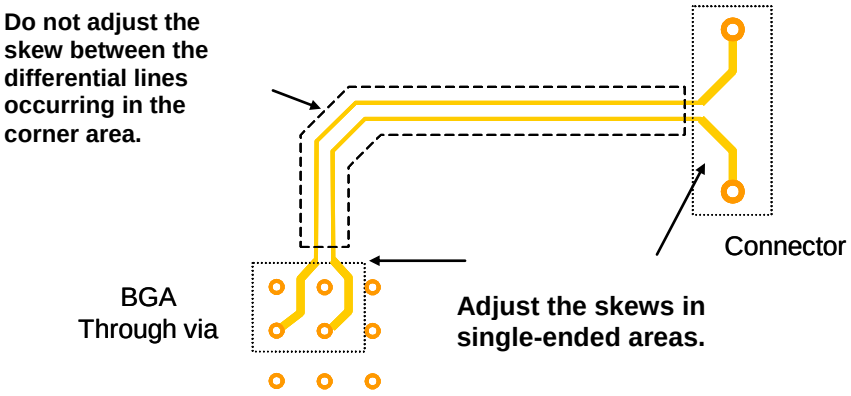


FIGURE 5. ADJUSTMENT OF SKEW BETWEEN DIFFERENTIAL LINES

SYMMETRICAL DESIGN

When providing the patterns in the peripheral areas of the differential lines and connecting parts to the differential lines, provide and connect them in such a way that they are symmetrically provided and connected with respect to the centerline between the differential lines. When shielding the differential lines, shield both the clock channel and the data channels using the same architecture.

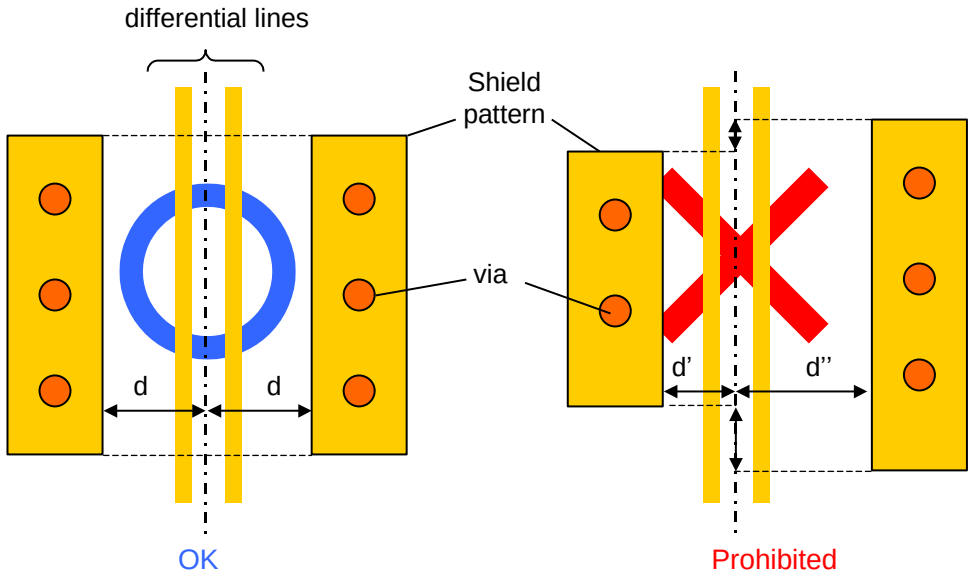


FIGURE 6. SYMMETRICAL ARCHITECTURE OF SHIELD PATTERNS

Power and Ground

POWER AND GND PLANES

Use planes, not wires, for power supply and GND. Laying the power supply and GND planes in layers produces capacitive coupling, which functions also as a decoupling capacitor reducing power supply noises.

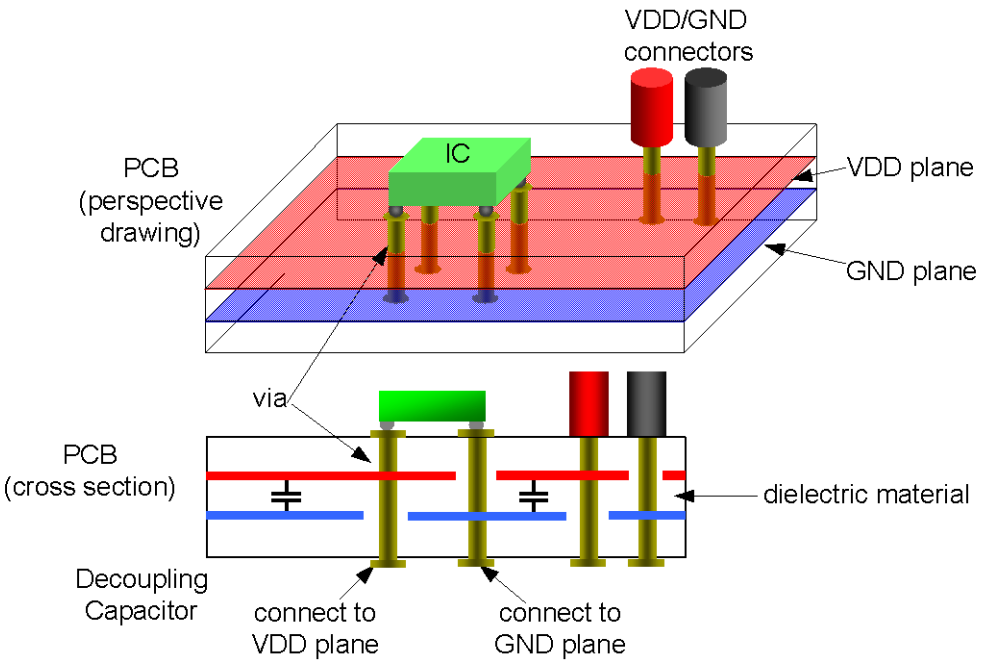


FIGURE 7. SUPPLY OF POWER AND GND BY PLANES, AND DECOUPLING CAPACITOR PRODUCED BY INTERLAYER DIELECTRIC MATERIAL

PLANE ISOLATION

Divide the power supply and GND planes into an analog (VDN, VDU, VDP, VSN) area and a digital (VDI, VDE, VSS) area respectively according to function. In each area, mount large capacitance capacitors close to the power supply and GND connectors, and mount small capacitance capacitors close to the ICs.

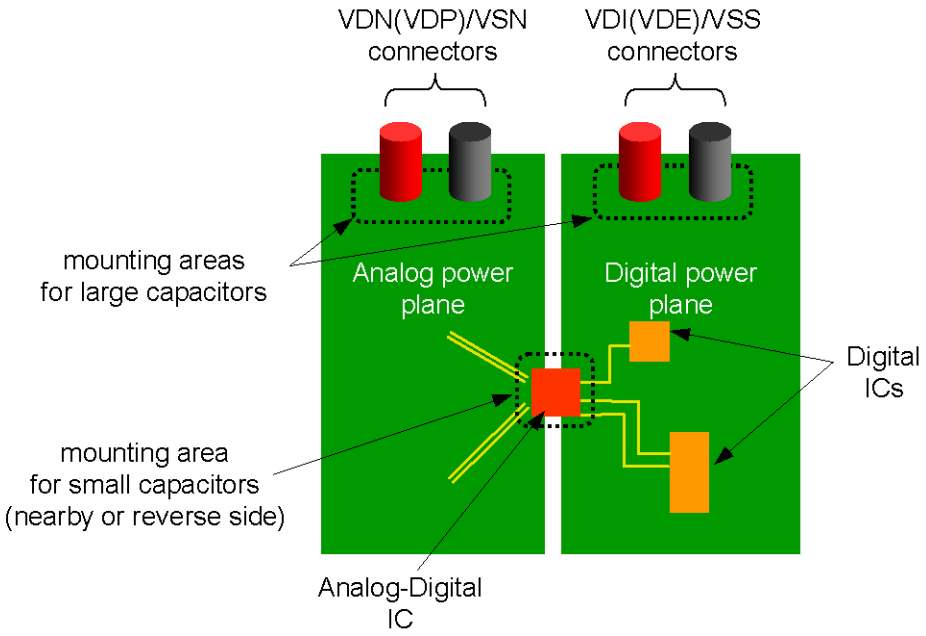
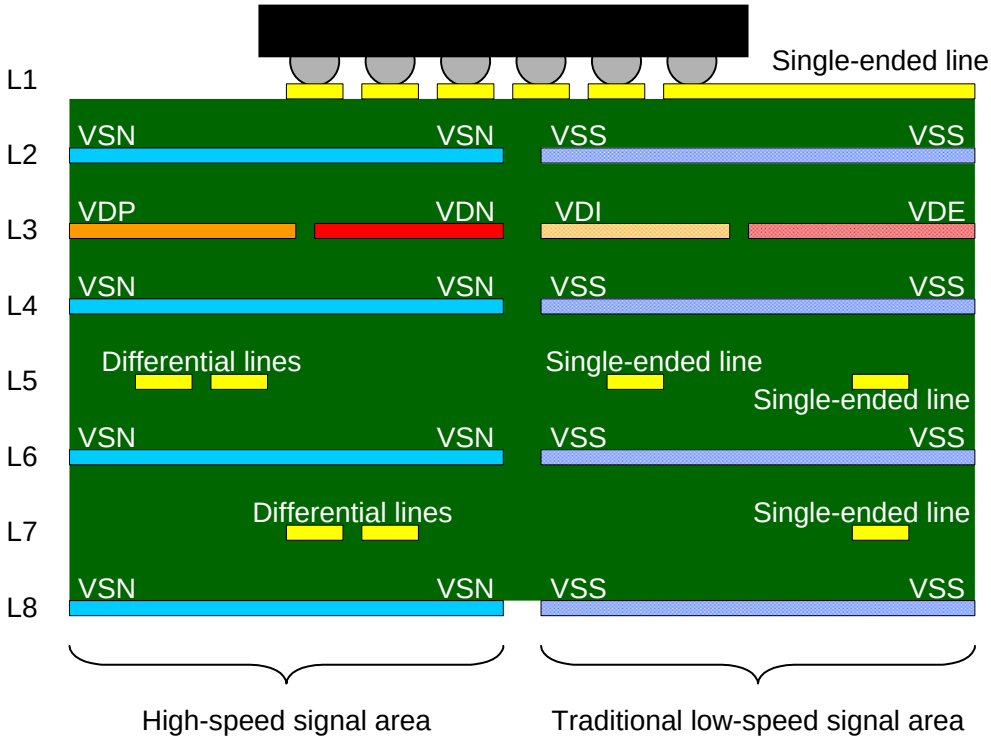


FIGURE 8. ISOLATION OF DIGITAL AND ANALOG PLANES

RECOMMENDATION OF LAYER STRUCTURE

Figure 9 shows an example of the layer structure for an 8-layer printed circuit board, where a chip with high-speed I/O macros implemented is mounted on the layer L1. In this structure, the following points are considered.

- Power and GND plane coupling (See “Power and GND Planes” on page 35 in “Section 2: PCB Layout Guide”)
- Power isolation (See “Plane Isolation” on page 36 in “Section 2: PCB Layout Guide”)
- Implementation of differential lines on lower layers



*** It is assumed that in the PCB above, the signal lines are provided not using through-vias. When using through-vias, not to cause stubs, connect the signals through the layers between the top layers (L1/L2) and the bottom layers (L7/L8).**

FIGURE 9. EXAMPLE OF LAYER STRUCTURE FOR 8-LAYER PRINTED CIRCUIT BOARD

RECOMMENDATION OF POWER SUPPLY PIN CONNECTIONS

Recommended power supply pin connections are shown in Figure 10.

- Power and ground planes should be used in the PCB.
- Bypass capacitors should be placed near the LSI.
- 0.01uF capacitors should be placed close to the LSI than other capacitors.

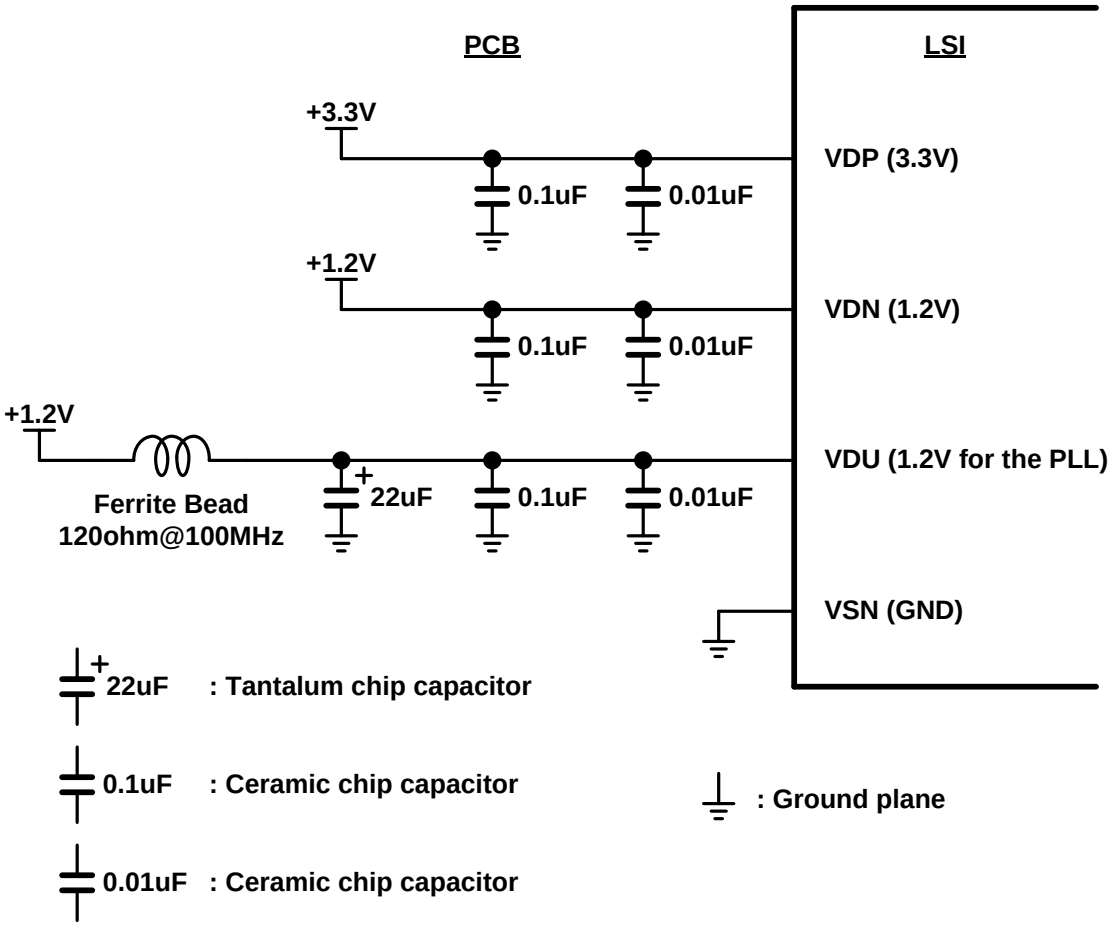


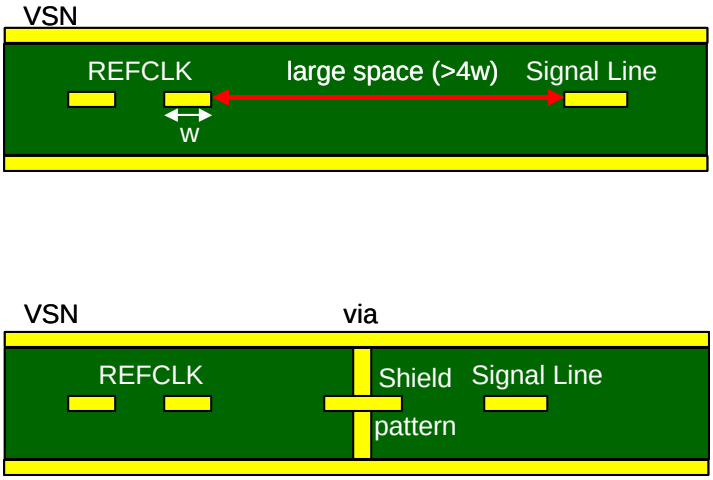
FIGURE 10. RECOMMENDED POWER SUPPLY PIN CONNECTIONS

Clocking Design

The reference clock supplied to high-speed I/O macros has a significant influence over the macro operations. To supply the clock having a waveform with less noise, pay attention to the rule in “General Description” on page 30.

REQUIREMENT OF JITTER

The influence of the noise from power supply and signal lines over the clock line to the chip causes an increase in the jitter of the macros. To avoid crosstalk from the lines in the peripheral area, provide a space of at least four times the clock line width (≥ 4w, w: clock line width) between the clock line and other lines, or take other measures such as providing shield patterning.



Specific requirements may be specified in specification documents. Please make sure to check them.

FIGURE 11. SEPARATION OF REFCLK AND OTHER SIGNALS

Additional Parts (for ESD and EMI)

ESD PROTECTOR

For higher ESD performance, ESD protection elements can be used. They must be limited to those parts specified for HDMI use. (Indispensable) Variations in the characteristics impedance of transmission lines caused by connection patterning, vias, parts mounting pads, and protection elements themselves cause degradation of signal integrity. In the design of printed circuit board, take measures such as making a sample board to check signal integrity. When using ESD protectors, place them near the HDMI connectors.

EMI FILTER

As an EMI countermeasure, EMI filters can be used.

The filters must be limited to those specified for HDMI use. (Indispensable) Variations in the characteristics impedance of transmission lines caused by connection patterning, vias, parts mounting pads, and filter elements themselves cause degradation of signal integrity. In the design of printed circuit board, take measures such as making a sample to check signal integrity. When using EMI filters, place them near the HDMI macros (package).

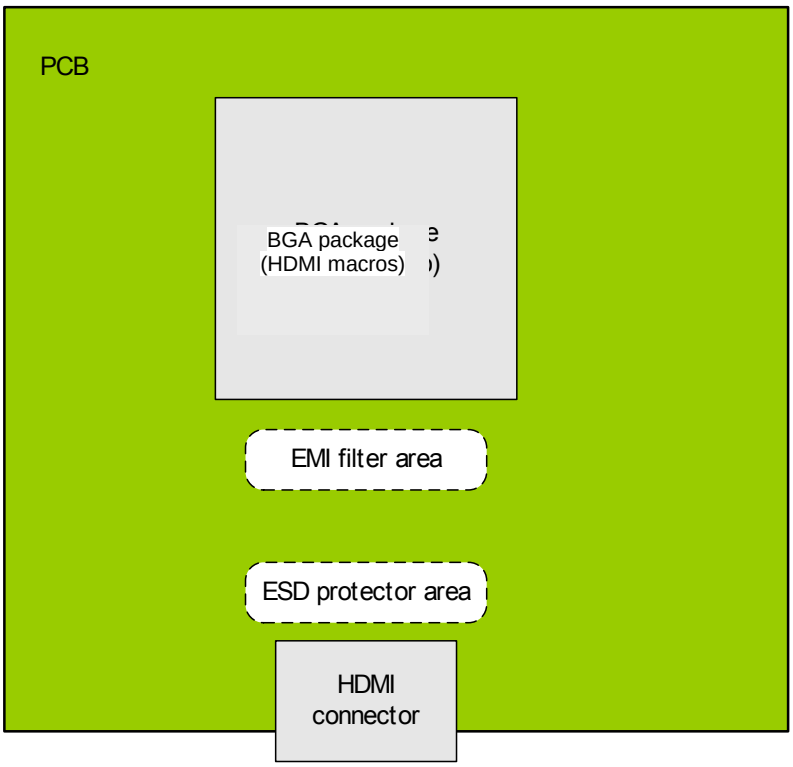


FIGURE 12. EXAMPLE OF PLACEMENT OF ESD PROTECTORS AND EMI FILTERS

Check List

NO.		CHECK ITEMS	RESULT OK/NG	COMMENT
1		GENERAL DESCRIPTION		
2		SIGNAL INTEGRITY		
	1	Characteristic Impedance		
		Micro Strip Line / Strip Line		
		Differential mode Impedance = 100 ohm +/- 10%		
		Common mode Impedance ≤ 35 ohm		
	2	Tracing of Bends (45 degree) or Curve		
		Differential Lines Space of 45 degree bends		
	3	Skew Adjustments		
		Channel to Channel Skew Adjustments		
		Differential Pair Skew Adjustments		
	4	Symmetrical design (shield)		
3		POWER AND GROUND		
	1	Plane Isolation		
	2	Separation of VDI(digital) and VDN(Analog)		
		Separation of VSS(digital GND) and VSN(Analog GND)		
	3	Power Supply and GND Layer Selection		
	4	Filtering for the power supply		
4		CLOCKING DESIGN		
	1	Cross Talk		
	2	Channel Spacing (ex: 4W -)		
		GND pattern between channels		
5		ADDITIONAL PARTS (FOR ESD AND EMI)		
	1	ESD Protector		
	2	EMI Filter		

Capacitance Reduced PADs

The portions of BGA and connectors mounted are prone to couple to the GND plane below them, causing lower impedance. To prevent the lowering of impedance, there is a method to reduce the capacitive coupling, where holes of the same size as the pads are created through the plane immediately under the pads.

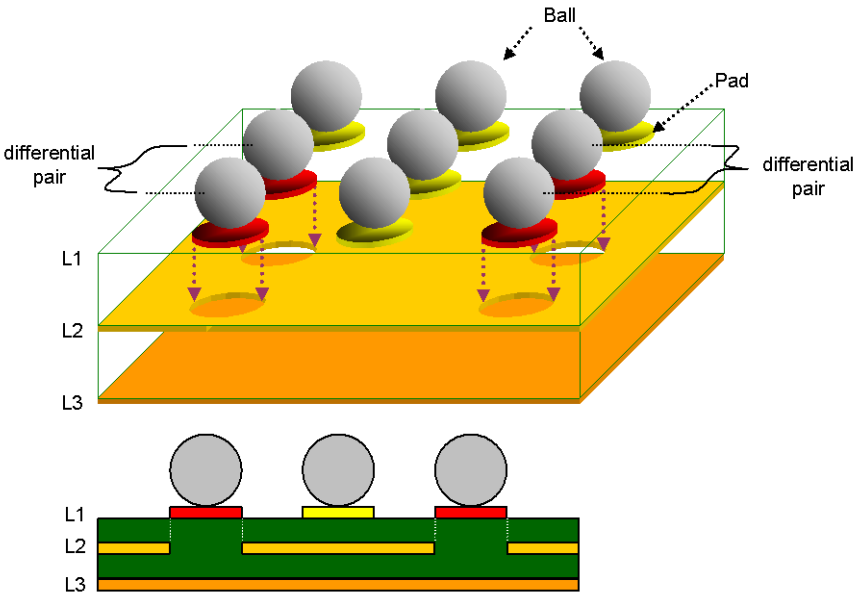


FIGURE 13. REDUCTION OF BALL PAD CAPACITANCE

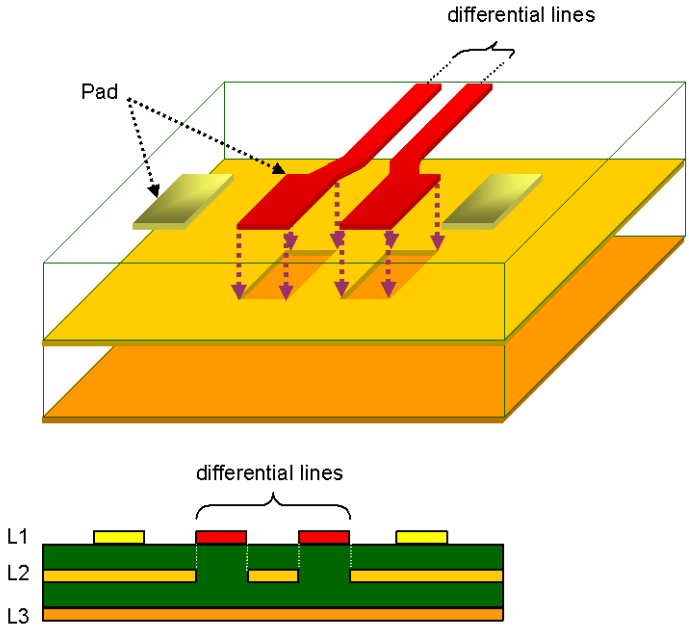


FIGURE 14. REDUCTION OF LEAD PAD CAPACITANCE

Section 3: PB Window and Channel ID Decoding

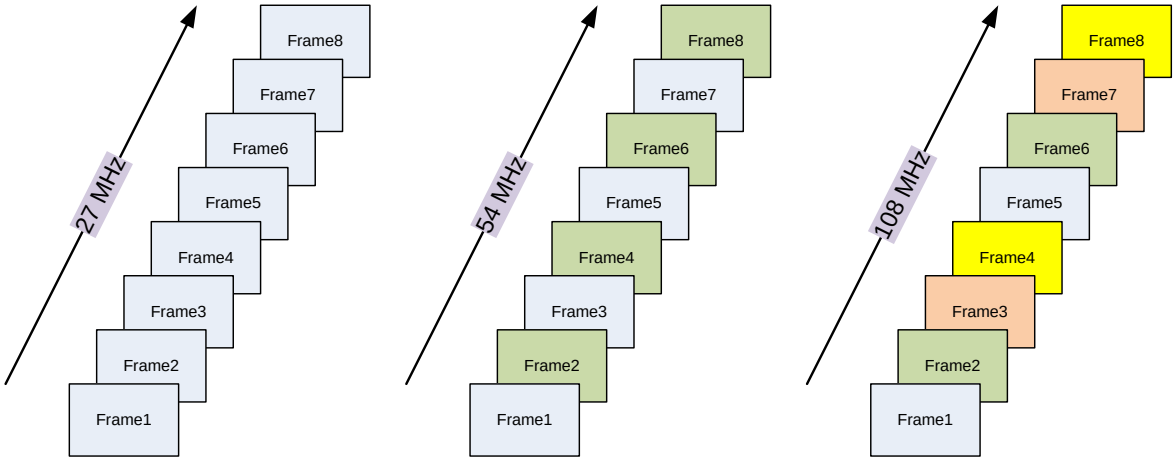
Introduction

TW2880 has four 8-bit play back ports. It can be used as four 8 bit interface input or two 16 bit interface input. It supports embedded sync video sequence coded in BT.656 or BT.1120 format. If multi-channel input is expected, it can only take sequence coded in frame interleaved, field interleaved and field switching format. There are two main playback operating modes in TW2880: normal mode and auto mode. In normal mode, the channel ID is not used and only single channel video data is assumed in the incoming stream. In auto mode, TW2880 is using channel ID information embedded in the video stream to decode the incoming stream so multiple channels are allowed.

The maximum number of channel allowed in one port is sixteen and each channel can appear in one of the four ports once. If a channel appears more than once in the port list then the port with a higher port number will have abnormal timing. Each channel has its own window control so it can be displayed in any non-overlapped fashion on any screen locations.

The following is an illustration of possible play back video streams. Remember that if you are sending multi-channel video stream but do not enable the CHID detection and auto mode register 0x6B4[3], you will have multiple channel data superimpose together, which renders the image useless. Also, remember the frame rate of each channel is calculated from the input clock and the number of channels. Two channel video stream with input clock rate at 54 MHz will be displayed in real time for each channel.

PB Mode Video Sequence Illustration



Normal / Auto mode **Auto mode (2CH)** **Auto mode (4CH)**

Features

- Accepts both interlaced and progressive format
- Resolution: up to 1080p, Data rate: up to 108 (or 74.25) MHz
- 8-bit (BT656) or 16-bit (BT1120, YC can be swapped)
- Supports frame interleaved mode (FMI) or field interleaved mode (FLI)
- Digital channel ID can be pushed to first active line
- Arbitrary ratio down scaler for each port
- Automatic channel ID insertion (For cascading purpose)
- Supports up to 16 channels / windows

Limitations

There are some limitations for play back ports.

- Cannot support byte interleave format
- In auto mode, any particular channel can only occur in one stream (port) and one position.
- Channel ID cannot support more than 4 channels mixing in one frame (QUAD)
- PB channel frame rate will affect live channel frame rate (non-real time)
- If channel cutting is used, the cutting function does not support more than one horizontal cut line

Normal Mode Registers Setting

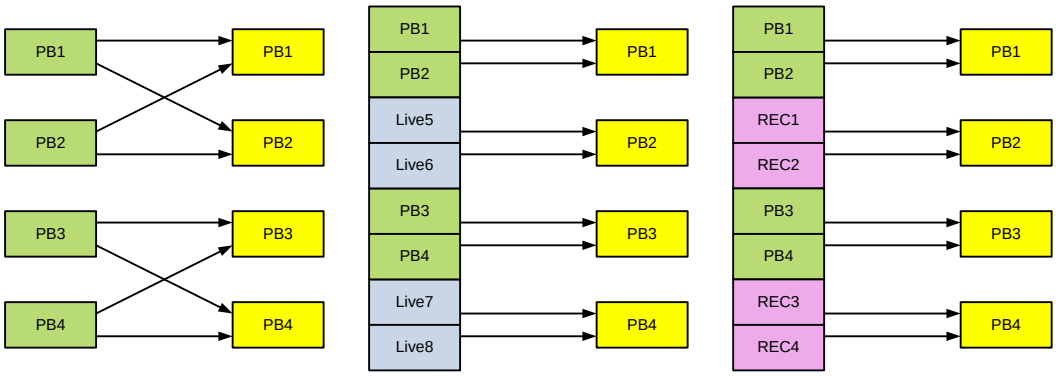
Here is register setting sequence to enable four PB port in 8-bit mode (BT656):

1. [0x3f2], bit 4, 5, 6 and 7 set to 1, this will enable manually set interlaced or progressive, not from channel ID. (default is from channel ID)
2. [0x3f2], bit 0, 1, 2 and 3 set to 1, this will force all ports to interlaced mode. You can set to 0 for progressive video.
3. [0x3fc], bit 0, 1, 2 and 3 set to 1, select odd/even field information is from SAV/EAV, not from channel ID. (default is from channel ID)
4. [0x3f3], bit 4, 5, 6 and 7 set to 1, this will enable manually set top field mode (top filed is 0 or 1, default is from channel ID)
5. [0x3f3], bit 0, 1, 2 and 3 set to 0, top field is 0. (only take effect when bit 4, 5, 6, 7 are '1')
6. program [0x358] to [0x36f], PB down scale ratio calculation from source size and target size.
7. [0x6b4], bit 3 set to 0, normal mode. (default is normal mode)
8. [0x610] to [0x613], 16 to 19 channel enable.
9. [0x684] to [0x68b], horizontal position
10. [0x634] to [0x63b], vertical position
11. [0x6ac] to [0x6b3], horizontal size
12. [0x65c] to [0x663], vertical size
13. [0x6d0] to [0x6d3] and [0x6e4] to [0x6e7] are the normal mode Hstart and Vstart registers. These registers are used in the traditional sense of cropping a single channel.

For 16-bit mode, the difference from 8-bit is:

1. [0x371], bit 0 and 1 set to 1, 16-bit mode
2. The input of the PB2 is the same as PB1 by default. PB4 is the same as PB3.
3. If using RGB interface you only need to set PB1 and PB3 register. PB2 and PB4 are using the same input.
4. [0x3ff] bit 5, 6 are the **PBX2_SEL** and **PBX4_SEL** bit, setting those bits to 1 will switch the input of PB2 and PB4 to Live5 – Live8
5. [0x3c6] bit 6, 7 are the **RECX2_SEL** and **RECX4_SEL** bit, setting those bits to 1 will switch the input of PB2 and PB4 to REC1 - REC4

Using these input pin sharing methods, it is possible to support 4 HD PB channels in TW2880-C1 chip in 1080i, 720p and 1080p format.



Auto Mode Registers Setting

Register Description

In auto mode, register setting is quite complex, user needs to pay attention to every steps. Even one register setting wrong can trash the display. Here are some important descriptions for registers.

1. [0x684] to [0x68b] horizontal position, [0x634] to [0x63b] vertical position, [0x6ac] to [0x6b3] horizontal size, [0x65c] to [0x663], vertical size for channel 16-19 **are not used in auto mode**

2. [0x610] to [0x613] rgb_wr_ctrl 16-19 are for PB port 1-4, they are port control, not channel control

3. The addresses of hpos0_pb to hpos15_pb, vpos0_pb to vpos15_pb, hsize0_pb to hsize15_pb, vsize0_pb to vsize15_pb, hstart0_pb to hstart15_pb, and vstart0_pb to vstart15_pb are **shared** with live channels. **Use [0x6b6] bit 0 to select live or PB channels**. {0} selects live channels and {1} selects PB channels.

4. [0x6c0] to [0x6cf] and [0x6d4] to [0x6e3] are the hstart / vstart registers used for PB channel 0 - 15 when [0x6b6] bit 0 is set to one. They are used for cutting operations in receiving multi-channel video frames. A more detailed explanation can be found in the next section. These registers need to be changed when PB down scale ratio is changed.

5. For example, in QUAD mode, you can set hstart0_pb=hstart2_pb=0, hstart1_pb=hstart3_pb=1/2 width after down scale, vstart0_pb=vstart1_pb=0, vstart2_pb=vstart3_pb=1/2 height after down scale to get the correct image.

6. [0x6fd] to [0x6fe] pb_ch_en. These bits can enable or disable separate channels. If one channel is not in PB port, you must disable this channel.

7. If digital channel ID cannot be inserted in VBI area, it can also be placed in first active line. In this case, analog channel ID must be disabled. To get correct image, you need to set register [0x3fc] bit[7:4] all to high.

Register Setting Sequence

1. [0x3f2], bit 4, 5, 6 and 7 set to 0, use channel ID to set progressive mode or interlaced mode. (default is from channel ID)

2. [0x3fc], bit 0, 1, 2 and 3 set to 0, odd/even field information is from channel ID, not from SAV/EAV. (default is from channel ID)

3. [0x3f3], bit 4, 5, 6 and 7 set to 0, field mode (top field is 0 or 1) is get from channel ID, not from register

4. [0x358] to [0x36f], PB down scale ratio. For example, if source is a 1920x1080 interlaced stream and the target size is 720x480, the source horizontal registers need to be set to 1920 and the vertical registers need to be set to 540. The target horizontal registers need to be set to 720 and the target vertical registers need to be set to 240.

5. Same resolutions but if the input is progressive, the source horizontal registers stay the same but the source / target vertical registers need to be double.

6. [0x370] bit 7 to 4 can be set to 0 or 1, **1 is for auto scale ratio selection**. It is used in the case when quad and D1 in one port. They need different down scale ratio. D1 frame use down scale ratio register setting. Quad use 1/2 of down scale ratio register setting.

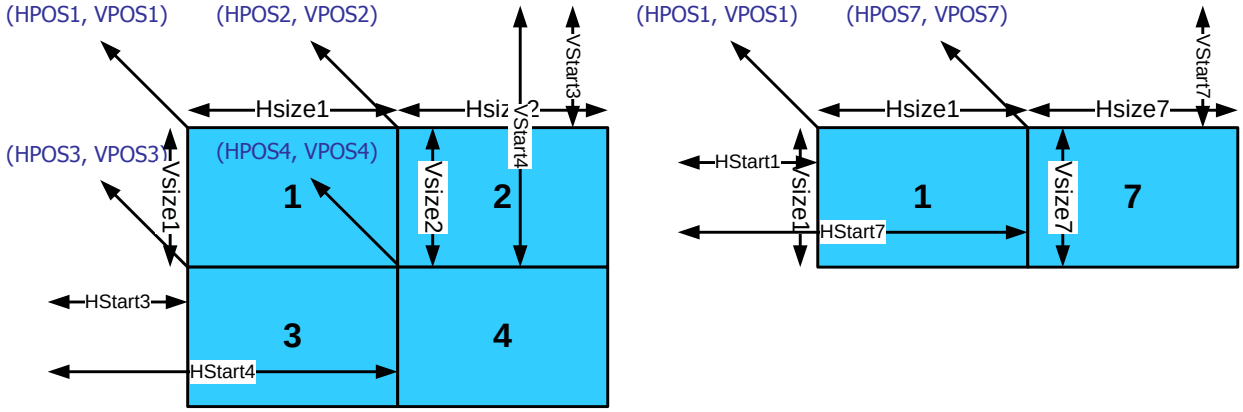
7. [0x6b4], bit 3 set to 1, auto mode. (default is normal mode)

8. [0x6b6], bit 0 set to 0 or 1, select PB channel or live channel registers

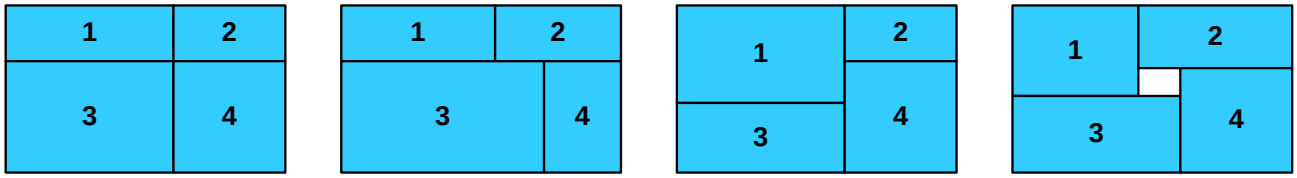
9. [0x610] to [0x613], PB port 1 to 4 enable:

10. [0x664] to [0x683], [0x614] to [0x633] horizontal and vertical position, set for both PB and live channels
11. [0x68c] to [0x6ab], horizontal size, set for both PB and live channels
12. [0x63c] to [0x65b], vertical size, set for both PB and live channels
13. [0x6c0] to [0x6cf], horizontal start for different position
14. [0x6d4] to [0x6e3], vertical start for different position
15. [0x6fd] to [0x6fe], PB channel enable. (very important)

Channel Cutting Using Hstart and Vstart

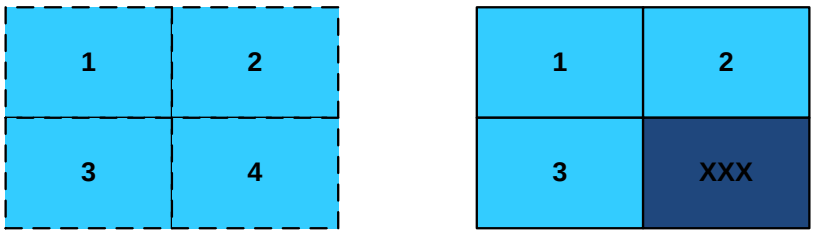


Register `[0x6c0] to [0x6cf]` and `[0x6d4] to [0x6e3]` hstart / vstart registers are used for channel cutting in a multi-channel video frame. In each port with multi-channel frames, user need to put correct values into the corresponding registers to let each receiving port knows how to divide the channels. In the above diagram, we illustrate some examples. The maximum cutting is four cuts where one frame is divided into four small frames. There are also two cuts and (together with ignored bits) three cuts. The channel arrangement is random with no pre-determined order in mind.



The horizontal and vertical cut lines of each channel do not need to have the same values. The above diagram is showing some possible combinations. Just remember when you are doing this kind of dividing you have to make sure each display window for each channel is in the correct sizes or wrong pixels are displayed in the screen.

Automatic CHID Insertion



A very useful feature in the PB port is the automatic CHID insertion and cutting. This feature enables the proper reception and dividing of video stream from any CODEC. No CHID is needed. However, one thing needs to be considered when using this feature: the reception frame implies an imaginary QUAD frame with fixed CHID order. We can use this frame as a degenerated one (less channels) but the QUAD frame concept always applies. We will discuss this concept in more detail later on.

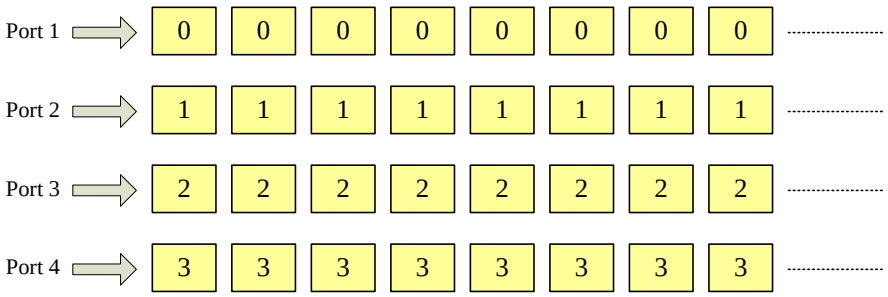
Channel Ignore Function

Using the above right diagram as an example, by using CHID, the user can ask the PB port to stop displaying a channel in the frame without affecting other channels. We will show the register setting in the next section.

Some Setting Examples

Channel Setting Example in Auto Mode

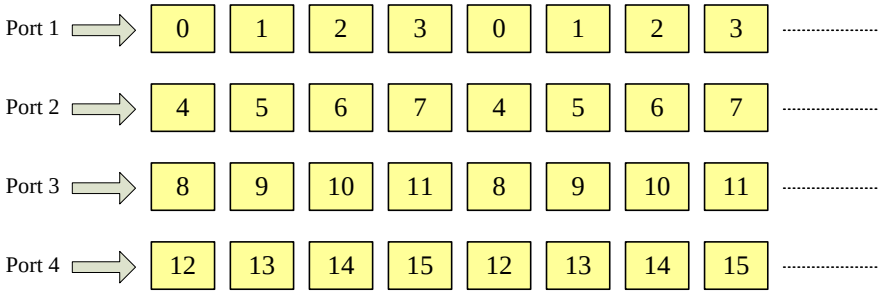
ONE PORT HAS ONE CHANNEL



The setting should be:

- pb_ch_en[15:0] = 0x000f

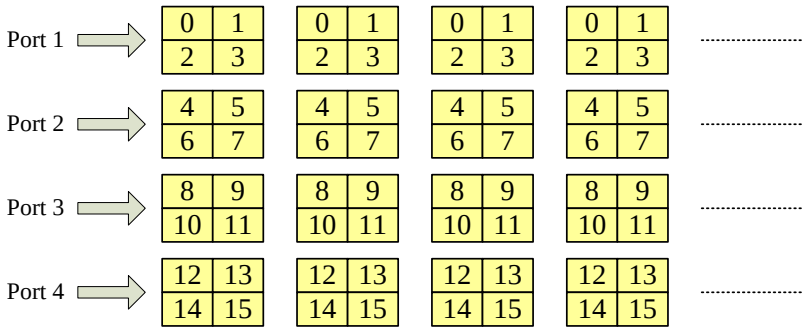
ONE PORT HAS FOUR CHANNELS, FRAME / FIELD INTERLEAVED



The setting should be:

- pb_ch_en[15:0] = 0xffff

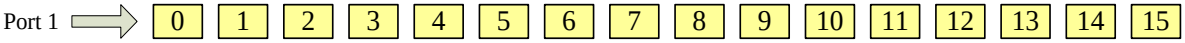
ONE PORT HAS FOUR CHANNELS, QUAD MODE



The setting should be:

- pb_ch_en[15:0] = 0xffff

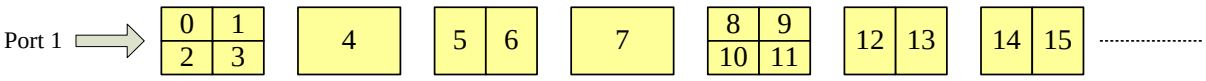
ONE PORT HAS 16 CHANNELS, CIF MODE



The setting should be:

- pb_ch_en[15:0] = 0xffff

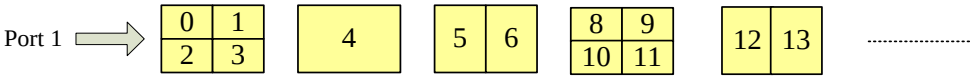
ONE PORT HAS 16 CHANNELS, MIXED MODE



The setting should be:

- pb_ch_en[15:0] = 0xffff
- [0x371] bit 4 can be set to 1, enable auto_scale

ONE PORT HAS 13 CHANNELS, MIXED MODE



The setting should be:

- pb_ch_en[15:0] = 0x3f7f
- [0x371] bit 4 can be set to 1 to enable auto_scale

HSTART AND VSTART SETTING EXAMPLE

In Quad mode, if input frame resolution is NTSC 720x480, then the setting should be:

```
ww 06c0 00;  rgb hstart0
ww 06c1 5a;  rgb hstart1 (0x5a = 0d90, 90x4 = 360)
ww 06c2 00;  rgb hstart2
ww 06c3 5a;  rgb hstart3
ww 06c4 00;  rgb hstart4
ww 06c5 5a;  rgb hstart5
ww 06c6 00;  rgb hstart6
ww 06c7 5a;  rgb hstart7
ww 06c8 00;  rgb hstart8
ww 06c9 5a;  rgb hstart9
ww 06ca 00;  rgb hstart10
ww 06cb 5a;  rgb hstart11
ww 06cc 00;  rgb hstart12
ww 06cd 5a;  rgb hstart13
ww 06ce 00;  rgb hstart14
ww 06cf 5a;  rgb hstart15
ww 06d0 00;  rgb hstart16
ww 06d1 00;  rgb hstart17
ww 06d2 00;  rgb hstart18
ww 06d3 00;  rgb_hstart19

ww 06d4 00;  rgb vstart0
ww 06d5 00;  rgb_vstart1
```

```
ww 06d6 78;  rgb vstart2
ww 06d7 78;  rgb vstart3
ww 06d8 00;  rgb vstart4
ww 06d9 00;  rgb vstart5
ww 06da 78;  rgb vstart6 (0x78 = 0d120, 120x2 = 240)
ww 06db 78;  rgb vstart7
ww 06dc 00;  rgb vstart8
ww 06dd 00;  rgb vstart9
ww 06de 78;  rgb vstart10
ww 06df 78;  rgb vstart11
ww 06e0 00;  rgb vstart12
ww 06e1 00;  rgb vstart13
ww 06e2 78;  rgb vstart14
ww 06e3 78;  rgb vstart15
ww 06e4 00;  rgb vstart16
ww 06e5 00;  rgb vstart17
ww 06e6 00;  rgb vstart18
ww 06e7 00;  rgb_vstart19
```

The unit for hstart is 4 pixels, and 2 lines for vstart. Vstart can also has 4 lines unit when register [0x6ff] bit 0 is set to '1'

ONE HD STREAM GETS DIVIDED INTO 16 CHANNEL EXAMPLE

Using TW2880C, it is possible to cut a 1080p HD input into 16 windows video data, the following the procedure:

```
ww 03c6 01;  Set to NTSC
ww 0370 0f;  Bypass 4 PBs, it is OK to use downscaler, but needs calculations
ww 03db 44;  PB path select, all PB port select data from PBIN1 and PBIN2

ww 0224 00;  Loop cnt1 selection
ww 0225 11;  Loop cnt2 selection
ww 0226 22;  Loop cnt3 selection
ww 0227 33;  Loop_cnt4 selection

ww 03fc 0f;  Enable FLD using SAV-EAV
ww 03f2 ff;  Enable manual mode, set FLI mode
ww 0371 03;  PB 1-2 16 bit
ww 06b4 08;  Enable PB auto mode
ww 06b6 03;  Set automatic CHID [1], set window control to PB [0]
ww 06fd ff;  Enable channel 1-8
ww 06fe ff;  Enable channel 9-16
ww 06fb 01;  Turn on vstart unit
ww 06fc 02;  Turn on hstart_unit
```

Enable 4 PB port (0x610 - 0x613) = 0x04.
Program vertical positions (0x614 - 0x61b) = 0x00.
Program vertical positions (0x61c - 0x623) = 0x10e.
Program vertical positions (0x624 - 0x62b) = 0x21c.
Program vertical positions (0x62c - 0x633) = 0x32a.

Program vertical sizes (0x63c - 0x65b) = 0x10e.

Program horizontal positions (0x664 - 0x683) = 0x00, 0x78, 0xf0, 0x168
Program horizontal sizes (0x68c - 0x6ab) = 0x78.
Program hstart positions (0x6c0 - 0x6cf) = 0x00, 0x3c, 0x78, 0xb4
Program vstart positions (0x6d4 - 0x6e3) = 0x00, 0x43, 0x86, 0xc9

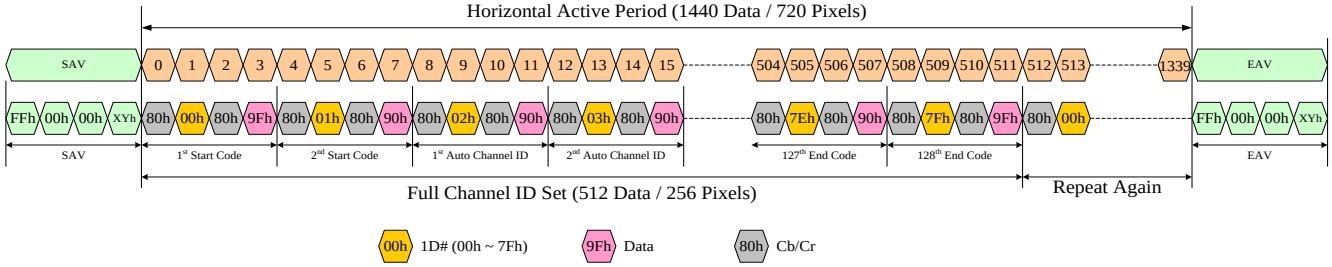
Digital Channel ID in First Active Line

ID STRUCTURE

TW2880 provides the digital type channel ID during VBI period. It is useful for DSP application because the channel ID can be inserted in just one line with special format. The digital channel ID is located before analog channel ID line. The digital channel ID can be enabled via the VIS_CODE_EN register.

The digital channel ID is inserted in Y data in BT656 or BT1120 stream and composed of ID # and channel information. The ID # indicates the index of digital type channel ID including the start code, auto/detection/user channel ID and end code. The ID # has 0 ~ 0x7F index and each channel information of one byte is divided into 2 bytes of 4 LSB that takes “0x90” offset against ID # for discrimination. The start code is located in ID# 0 ~1 and auto channel ID is situated in ID # 2 ~ 11. The detection channel ID is located in ID # 12 ~ 91 and the user channel ID is situated in ID # 92 ~ 121. The end codes occupy the others. The digital channel ID will be repeated during horizontal active period. There is no repeat auto channel ID in digital format. The following figure shows the illustration of digital channel ID.

In the TW2880 output sequence, the vertical offset of odd field is controlled by the VIS_LINE_VOS register with 1 line unit and even field is control by the VIS_LINE_FOS register with 1 line unit. Channel ID can be flexibly enabled by register VIS_ID_OEN and VIS_ID_EEN for odd field and even field respectively. These registers make it possible to insert channel ID in vertical active region if user required.



DIGITAL CHANNEL ID TIMING DIAGRAM

DIGITAL CHANNEL ID DATA FORMAT

1D#	DATA	DESCRIPTION
0 (00h)	9Fh	Start Code
1 (01h)	90h	
2 (02h)	{9, A0_MSB}	Auto Channel ID (5x2=10 bytes)
3 (03h)	{9, A0_LSB}	
...	...	
10 (0Ah)	{9, A4_MSB}	
11 (0Bh)	{9, A4_LSB}	
12 (0Ch)	{9, D0_MSB}	Detection Channel ID (40x2=80) bytes
13 (0Dh)	{9, D0_LSB}	
...	...	
90 (5Ah)	{9, D39_MSB}	
91 (5Bh)	{9, D39_LSB}	
92 (5Ch)	{9, U0_MSB}	User Channel ID (15x2=30 bytes)
93 (5Dh)	{9, U0_LSB}	
...	...	
120 (78h)	{9, D14_MSB}	
121 (79h)	{9, D14_LSB}	
122 (7Ah)	90h	End Code
123 (7Bh)	9Fh	
124 (7Ch)	90h	End Code
125 (7Dh)	9Fh	
126 (7Eh)	90h	End Code
127 (7Fh)	9Fh	

Digital channel ID data sequence is: Cb, ID#, Cr, 9Xh, Cb ID#, Cr, 9Xh, ... After one full set, the same data must be repeated in the active area.

REGISTER SETTING

If more than one channel exists in one PB port, channel ID must be inserted in video stream. If the codec chip can only insert channel ID in the active area, the digital channel ID must be inserted in the first active line, and no analog channel ID is needed. In default setting, digital channel ID and analog channel ID must be in vertical blank area. To support digital channel ID in first active line, some registers must be set correctly.

Here is an example for PB port 1

1. set [0x3DE] bit 0 to “0”, disable analog channel ID
2. set [0x3DE] bit 1 to “1”, enable digital channel ID
3. set [0x3DE] bit 2 to “1”, auto detect channel ID position enable
4. set [0x3FC] to 0xF0, delay one line for video

READ CHANNEL ID FROM REGISTERS

To check if TW2880 is receiving correct channel ID, you can read channel ID from registers. Here is sequence to read channel ID.

1. set [0x3DE] bit [7:4] to 0, set to auto channel ID.
2. set [0x372] to any value, write channel ID to registers
3. read register [0x372], [0x373], [0x374], [0x3C7] and [0x3C8]. In [0x373], the register shows:

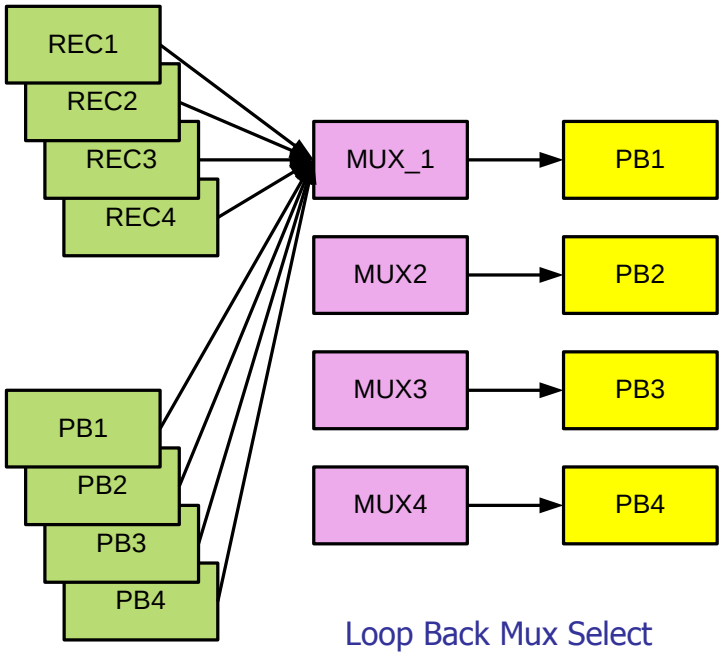
```
[7]: digital_chid_valid_pb1, 1: digital chid valid, 0: analog chid valid
[6]: auto_valid_pb1, 1: auto_chid valid, 0 not valid
[5:0] auto_chid_pb1[37:32]
```

Frame Interleaved Mode Setting

In frame interleaved mode, some registers must be set correctly to avoid frame dropping. For example, if input PB channels are channel 0, channel 1, channel 2, channel 3. Then you must set [0x6B9] bit 0 to "1", and set [0x6B7] bit [3:0] to 0. Besides this, user must set [0x4F6] bit [3:0] to 0x1 (default is 0xF).

PB Loop Back Control

There are four input data and clock multiplexers sitting between the play input pin groups and the real play back hardware unit. The multiplexes serve two purposes: One is to select play back input from TW2880's recording output for testing purpose; another is direct same play back input source to different play back units. The latter functions are very useful in cascade mode. We will talk about this in a minute. The registers for the ports are: 0x224-0x227.



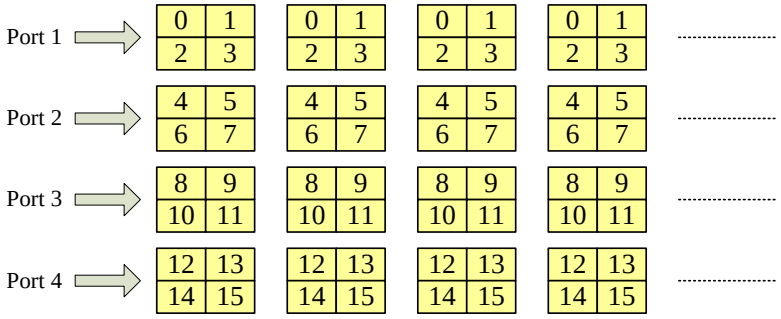
Loop Back Mux Select

PB INPUT CLOCK PHASE DELAY CONTROL

BIT	R/W	DEFAULT	DESCRIPTION
7		0	Reserve
6:4	R/W	001	Playback Port data is from: 111: rec4 port 110: rec3 port 101: rec2 port 100: rec1 port 011: PB4 port 010: PB3 port 001: PB2 port 000: PB1 port
3:0	R/W	0001	Playback port clock is coming from: 1111: rec4_clkn 1110: rec4_clkp 1101: rec3_clkn 1100: rec3_clkp 1011: rec2_clkn 1010: rec2_clkp 1001: rec1_clkn 1000: rec1_clkp 0111: pb4_clkn 0110: pb3_clkn 0101: pb2_clkn 0100: pb1_clkn 0011: pb4_clkp 0010: pb3_clkp 0001: pb2_clkp 0000: pb1_clkp

Automatic Channel ID Insertion

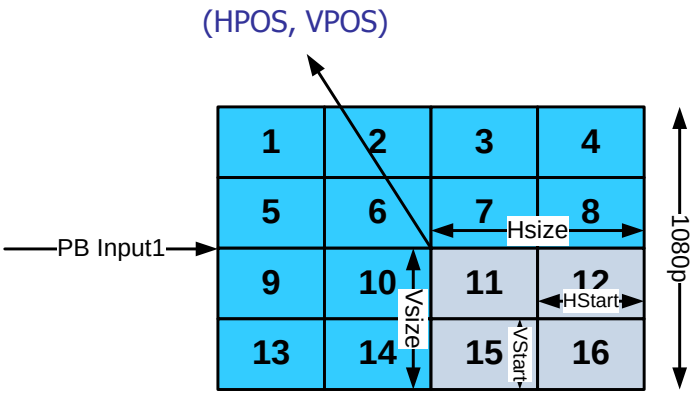
Register [0x6B6] bit 1 controls the automatic channel ID insertion function. If this bit is enabled (set to 1), TW2880 will automatically insert a pre-determined channel ID into the video streams received from the four PB ports. This function is very useful when the CODEC is having difficulties to insert channel ID in the vertical blanking area according to Techwell's format. The channel ID assignments for each port are fixed: channel 0 to 3 are for PB port 1 stream, channel 4 to 7 are for PB port 2stream, channel 8 to 11 are for PB port 3 stream, and channel 12 to 15 are for PB port 4 stream. Please see the next diagram.



The incoming video format is automatically set to QUAD mode so four windows are expected within a port but you can enable less windows. We will come back to this later. On the other hand, the most cutting you can do on the D1 stream is 16. This will requirement you direct all the PB inputs to one source. The following is the detailed register setting; you can use the EV board to do this test.

1. set [0x224]to [0x227] bit[6:4] to the same input, for example PB1 port "0"
2. In 8-bit PB mode, 16 channels can be supported, in 16-bit mode, 8 channels are supported.
3. Set hstart, vstart, hsize, vsize, hpos, vpos to get desired channels into the desired channel. Use the diagram below as an example, enable channel 11, 12, 15, 16 and have the parameters set up correctly. You can also enable channel 11 only with double size parameters.
4. set [0xD01] bit 2 to "0", disable digital CHID generation for port1
5. set [0xD13] bit 2 for port2, set [0xD25] bit 2 for port3 and set [0xD37] bit 2 for port4
6. set [0x3FC] bit 3..0 to "1", force PB port1-4 to use SAV-EAV signature to generate sync instead of using channel ID
7. set [0x3F2] bit 7..4, 3..0 to the correct output sequence (frame or field interleaved)

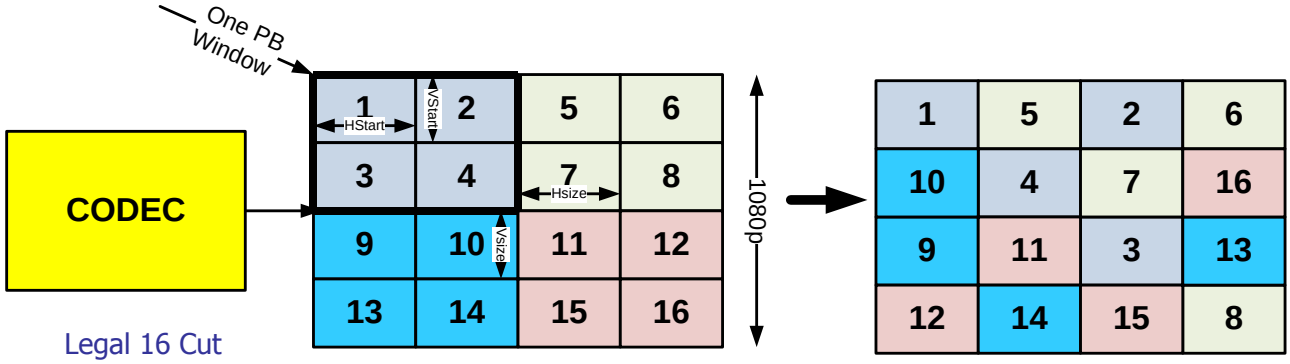
In this way, the TW2880 PB port can accept any output sequence from the CODEC, divide them to the proper sizes, and distribute into the desired PB windows.



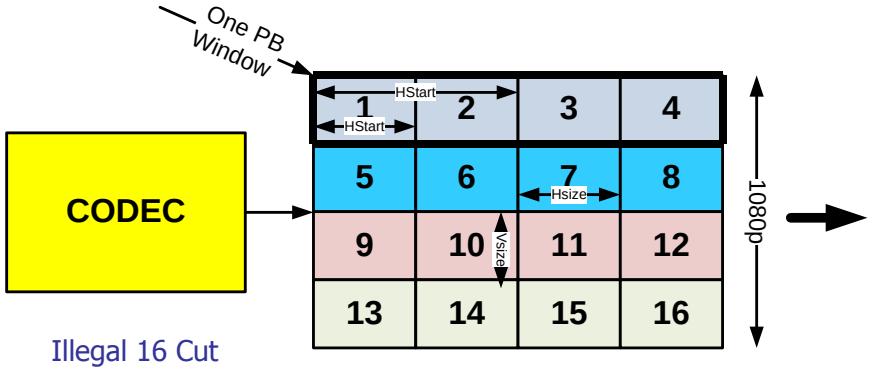
Use the hstart and vstart to determine the cutting point, using hpos and vpos to determine display position.

Repeat Cutting

The four PB ports can be adjusted to have the same input. Using this setting, the user can cut the incoming channel to more than four pieces. In the next example, the four PB have the same size and different Hstart and Vstart. This will cut the incoming stream into 16 smaller streams or channels. One thing to remember is: Although the channel location can be anywhere in the screen, the cutting setup has to be in QUAD mode, just as in the example illustrated below.



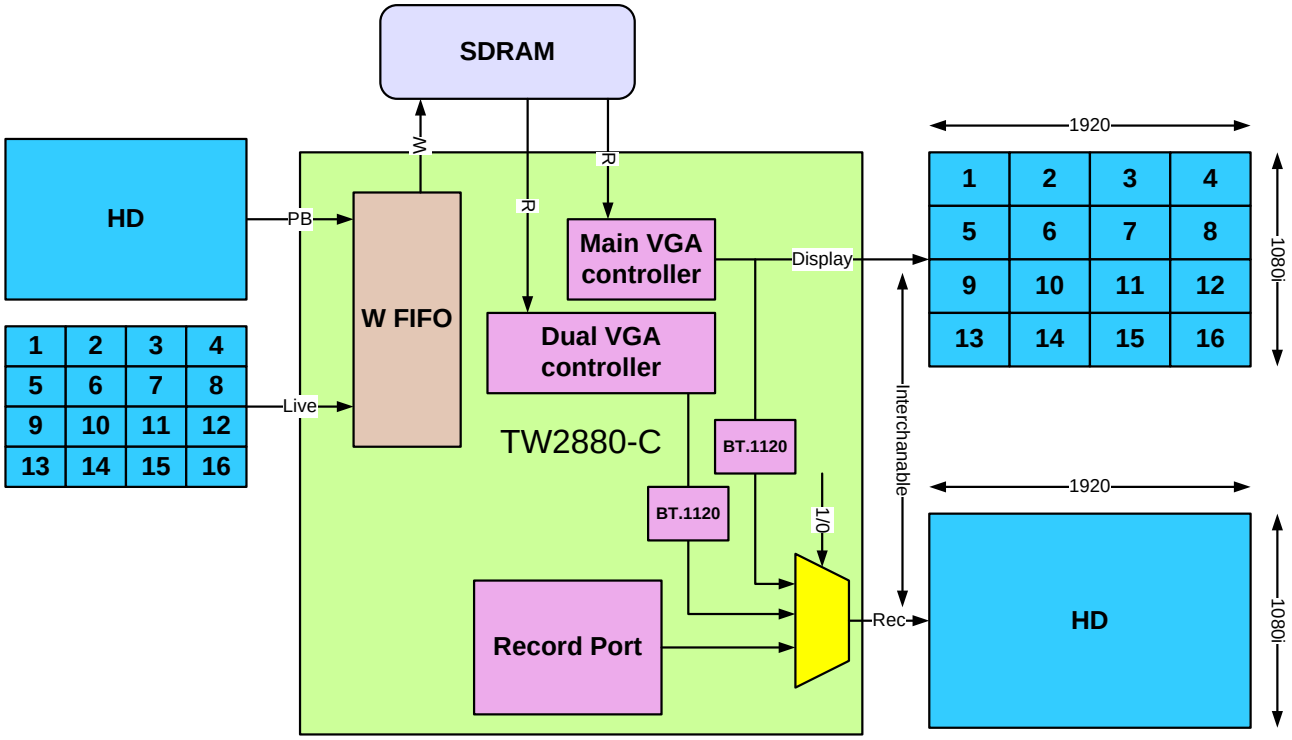
Legal 16 Cut



Illegal 16 Cut

Cascading Two TW2880Cs

Display Output Multiplexing



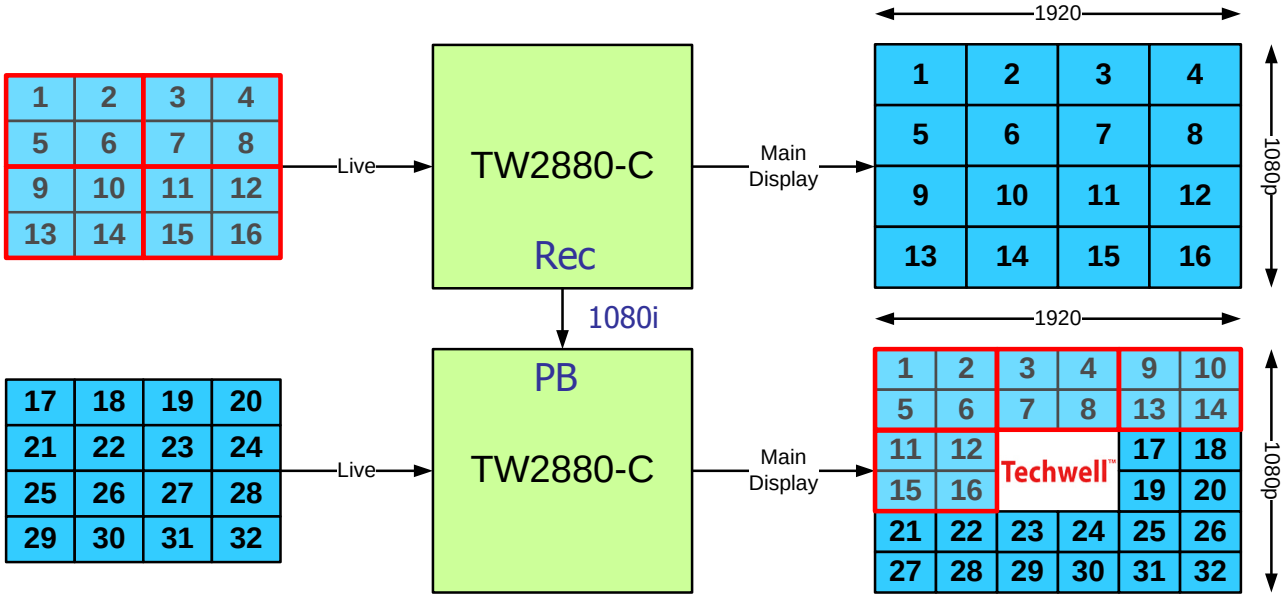
Multi-mode Application Example 1

The output of the display (both main and dual) can be redirect to the recording output of TW2880 if the resolution is interlaced HD. This means user of TW2880 does not need to enable the recording portion and install the SDRAM to drive CODEC chips. However, since no multiplexing functions are provided, the flexibilities are limited.

To enable this function, user need to do:

1. Set main display or dual display to 1080i resolution. No need to set both.
2. set [0x22A] bit 7 to “1” to let recording port 1, 2 has the BT.1120 output
3. set [0x22A] bit 6 to “1” to let recording port 3, 4 has the BT.1120 output
4. Set [0x22A] bit 5 to “0” to select main LCD in port 1, 2.
5. Set [0x22A] bit 4 to “0” to select main LCD in port 3, 4.

32 Live Channel Example



Multi-mode Application Example 2

Now we demonstrate an application that will require several advanced functions we described earlier. To do a 32-channel DVR system we need put two TW2880 chips cascade together. Each chip will capture 16 live channels in the system. To enable this function, user need to do:

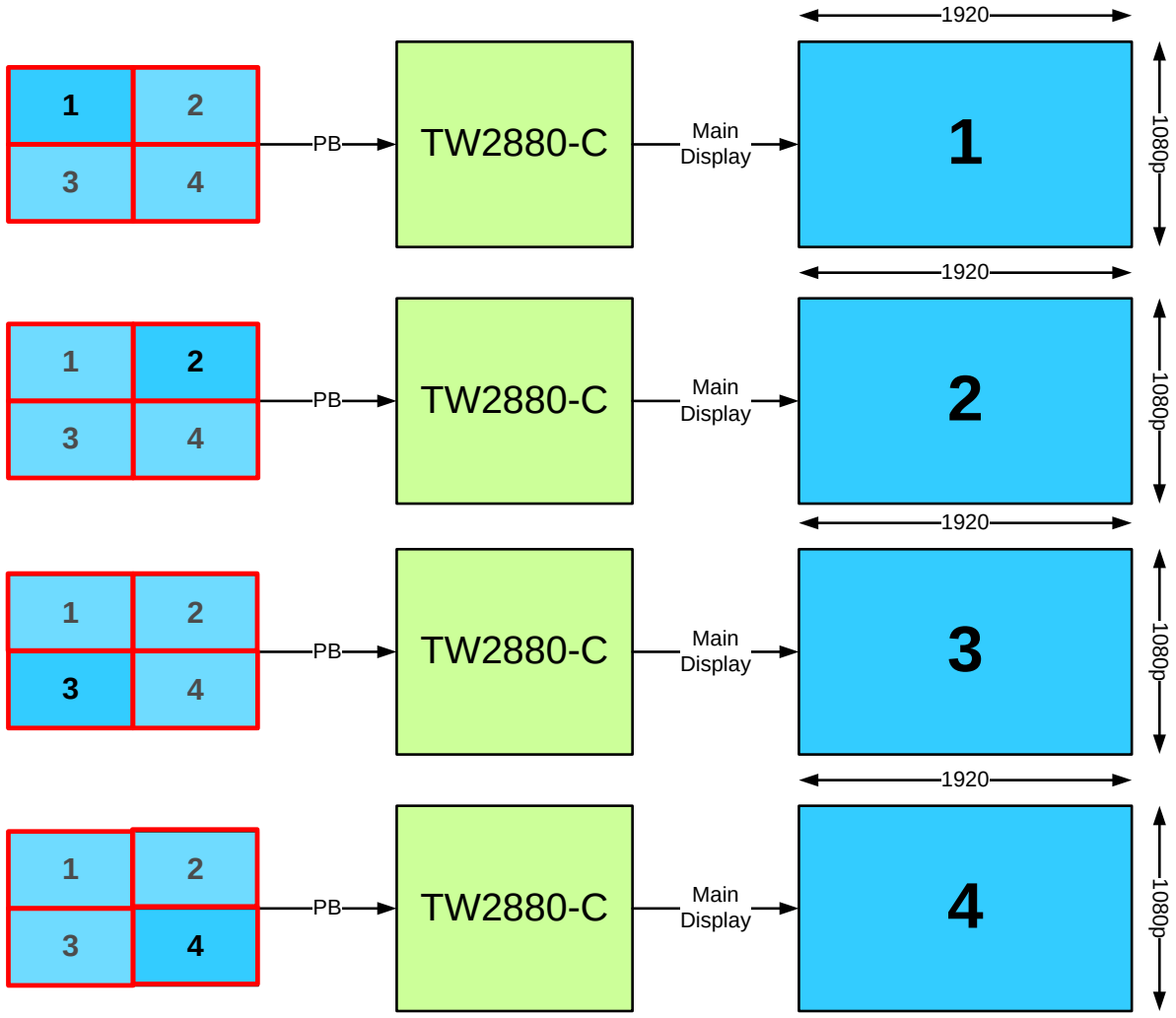
1. Set the first main display to 1080i.
2. Set [0x22A] bit 7 to "1" to let recording port 1, 2 has the display output.
3. Set [0x22A] bit 5 to "0" to select main LCD.
4. Connect the record port 1, 2 to PB port 1,2 of the second chip.
5. Set PB port of the second chip as automatic CHID insertion mode, cut the incoming picture as four window data or 16 window data.
6. Display the four "big" channel data with the remaining 16 live channels in the second display. 32-channel system is done.

Advanced Topics

TV Wall Example

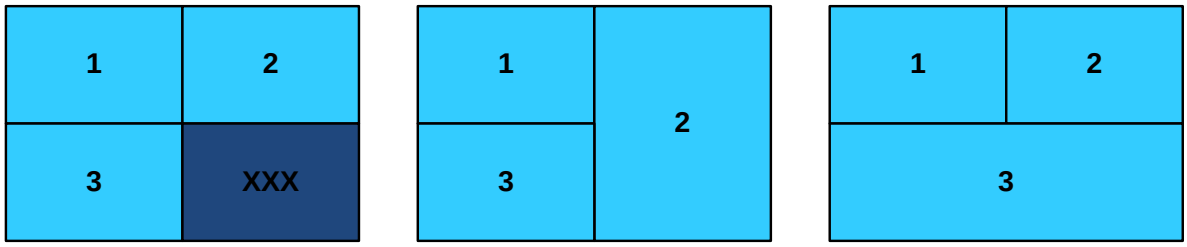
This application example is utilizing the cutting capability of TW2880. Assuming the incoming image is in HD format, we can use four TW2880 to receive the same image in BT.1120 format through PB input. Cut one quarter of the image and store into memory, then use up scaler to fir the final screen. Tile those screens together to form the final display.

As you can see, this method is very easy to propagate to 3x3 and 4x4 configuration. In addition, we can add OSG or overlapping windows into the final screen by using external OSG function.



2x2 TV Wall Example

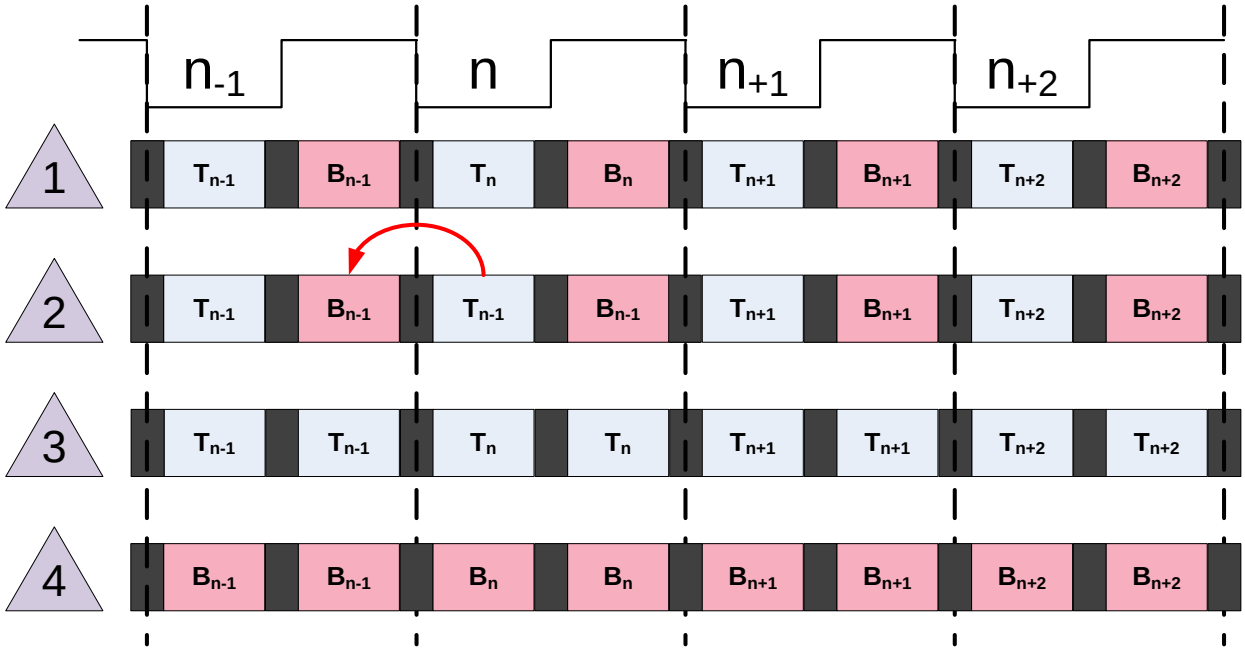
Ignore Bit



Ignore information is extracted from auto channel ID bit [7:4]. The original intent of these bits is used for reporting "no video status" in the incoming stream. We use the Auto CHID [8] = 1 to redefine these bits as ignored bits. When PB port received the ignored bits information and 0x6fa [7:4] is set to 1, the relevant channel in the incoming stream will be ignored by the write FIFO. That means no updating for this channel but the rest of the operation is just keep going on.

A by-product of this function is for PB to display non-conforming video stream. Using the above diagram as an example, if channel 4 is noted ignored, the CODEC programmer can extend either channel 2 in Y direction or extend channel 3 in X direction. All channels will still be displayed correctly.

One Field Mode



When TW2880C receives non real time video stream from CODEC to the PB port, although every channel of video can be displayed in the windows, the visual effect will not be great due to the non real slow down effect. This problem will get even worse if you are using 3D de-interlacing circuit. The problem is coming from two fold:

1. The 3D-DI circuit is not design for used in non-real time case.
2. If we repeat the frames in the stream like some CODEC can to maintain the frame rate, an inevitable situation will occur is the repeating top field is actually earlier than the previous bottom field. This will create a go back effect in the motion picture.

Both artifacts can be removed if you turn on the saving one field mode in TW2880C. In this mode, TW2880C will be saving one field for each channel so the annoying artifact will be gone. The duplicate field is selectable.

To use this feature, the user needs to:

1. Program [0x6F0] bit 7, 3 to enable PB2 and PB1. Program [0x6F1] bit 7, 3 to enable PB4 and PB3. [0x6F0] - [0x6F1] are sharing with line number registers.
2. Program [0x6F0] bit 6, 2 to 1 to select bottom field

Section 4: Recording and SPOT Unit

Overview

Programming Model

The recording unit consists of the following three parts

- Write buffer
- Read Port
- Pin output

Write buffer has 16 independent buffers. Each buffer selects channel and image size and stores that image into the SDRAM.

Read port has 9 read ports. Each port selects buffer and sends that image data to output pin.

There are four output pins. Each pin supports 2-port muxing with two clocks that have different phase and independent control.

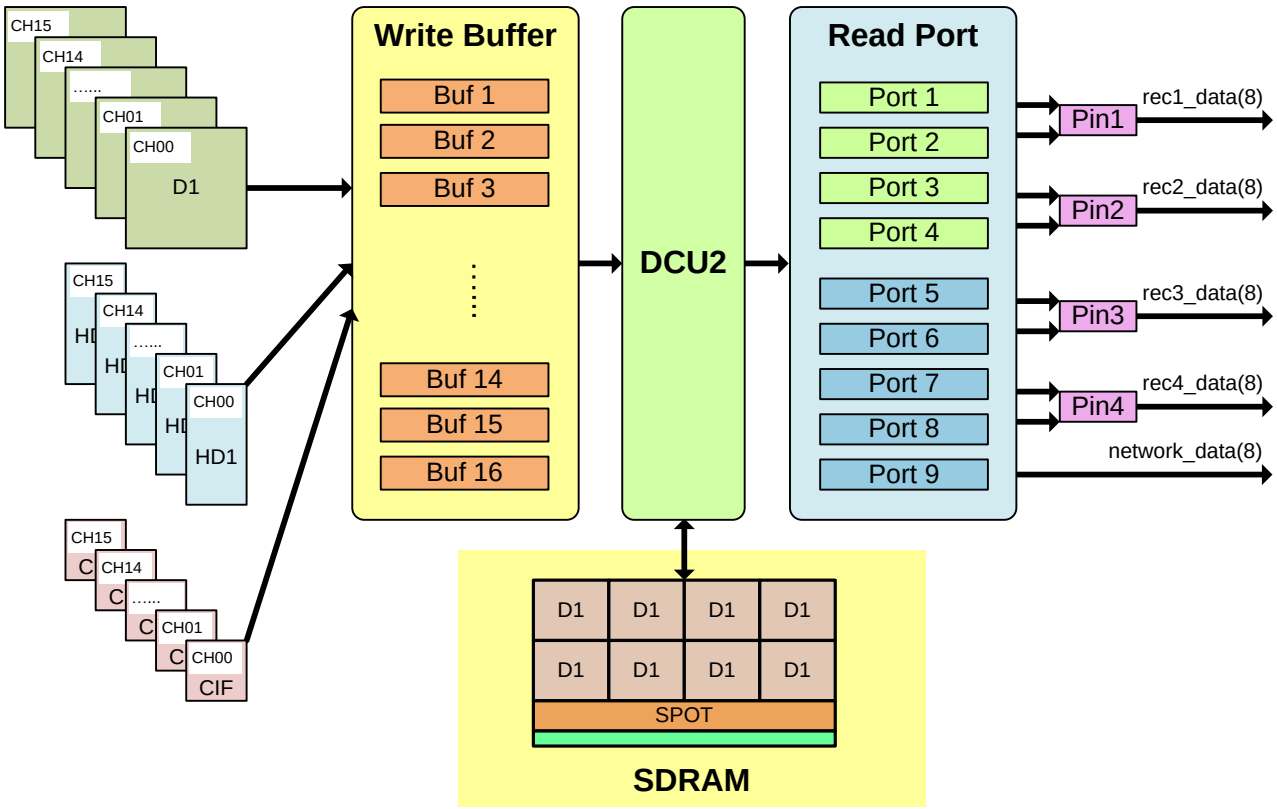


FIGURE 15. PROGRAMMING MODEL OF RECORDING PATH

For recording operation, there are three GUI setting window according three parts that consists of write buffer, read port and pin control (Refer to Figure 16 to Figure 18).

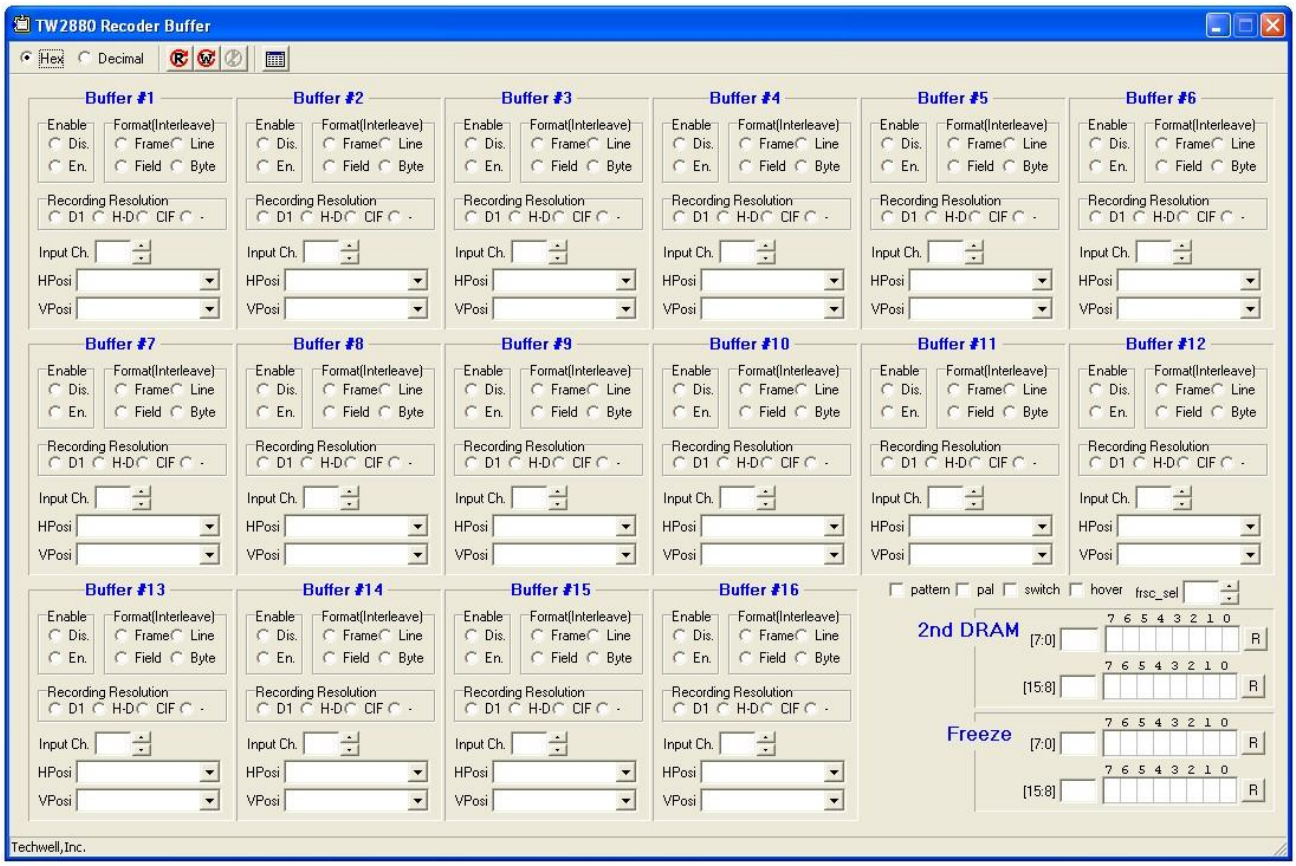


FIGURE 16. RECORD BUFFER CONTROL WINDOW

The record buffer control window is for setting write buffer control values. By using this window, you can set whole write buffer setting values, including buffer on/off, saving format (Frame or Field), image resolution (D1 or Half-D1 or CIF), channel number and position in the SDRAM canvas.

Freeze function can also set by using this window.

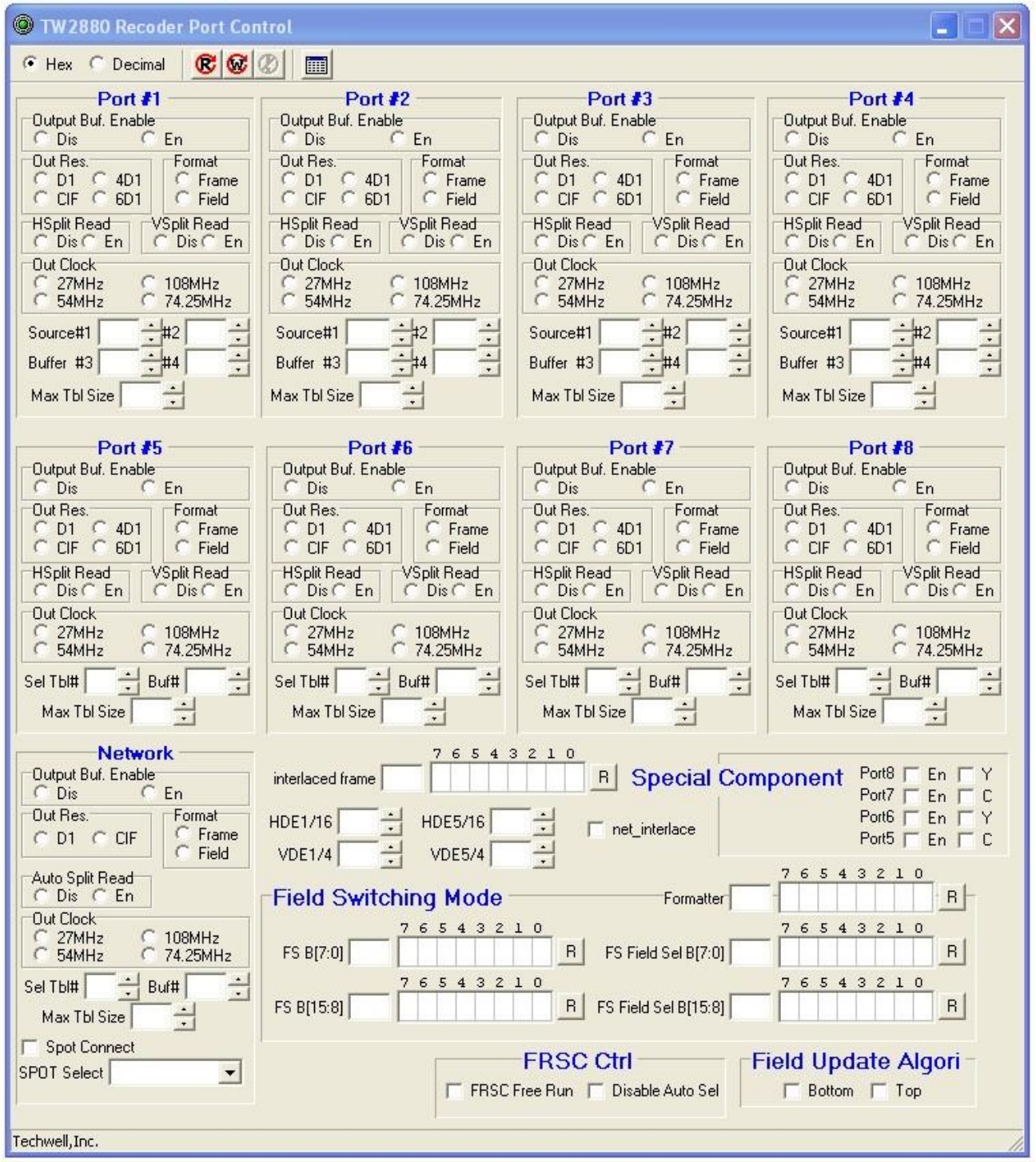


FIGURE 17. RECORD PORT CONTROL WINDOW

The record port control window is for setting 9 read ports control values including network port.

You can set whole port control values including port on/off control, output resolution (D1 or CIF or 4D1 or 6VGA), x and y split, output clock rate and source indexing control.

This window also support control button for field switching mode and frame rate control.

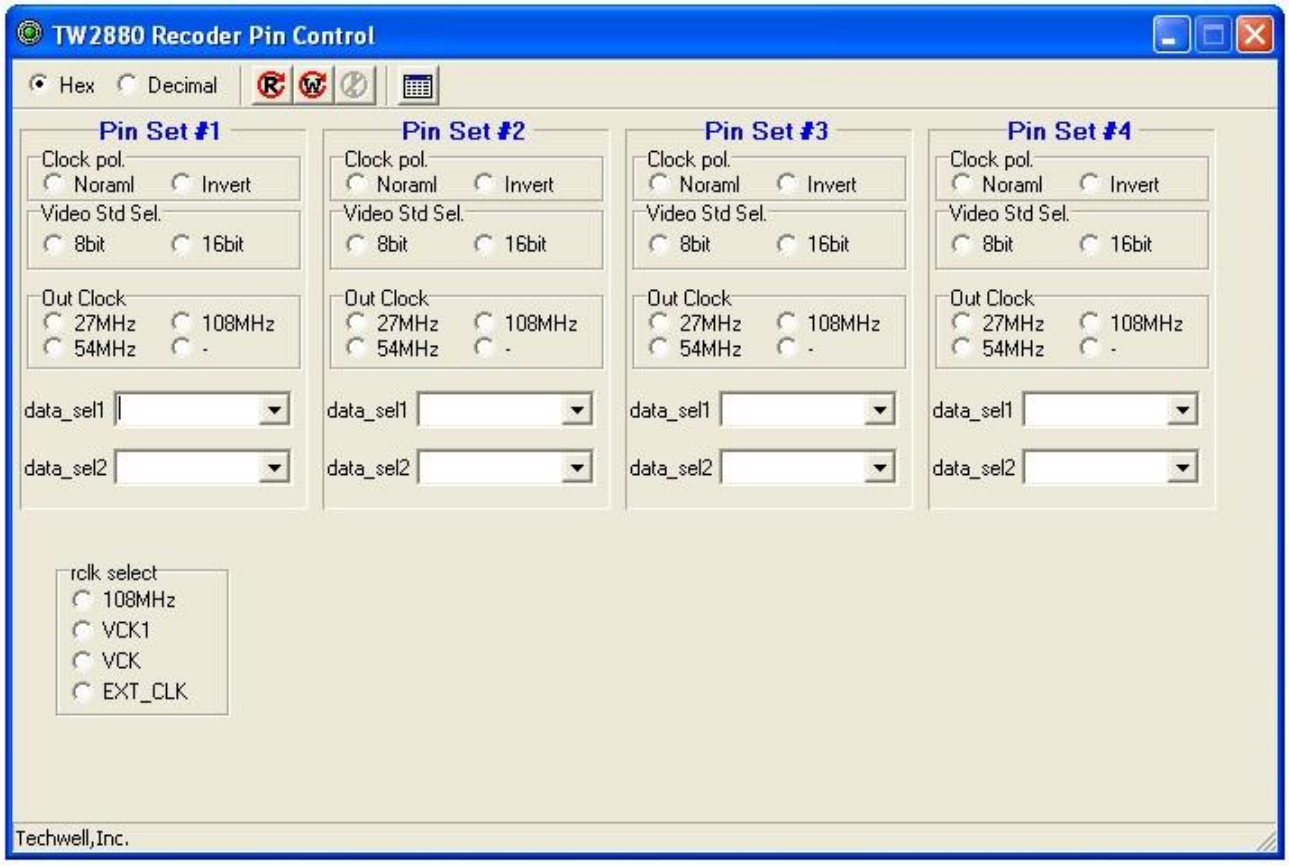


FIGURE 18. RECORD PIN CONTROL WINDOW

Record pin control window is for setting output pin control values.

You can set whole output pin control values including pin clock polarity, data width, clock rate and source port.

This window also supports control button for record clock source selection.

Programming Flow

Record control values are set with the sequence that is described in the following flow chart.

After register setting, we need to reset record part with software reset and then enable port.

If you need to update port table index without stopping operation, firstly you set new table values(ex. 0xC35 and 0xC36) and then set table update enable bit(ex. 0xC36[5]).

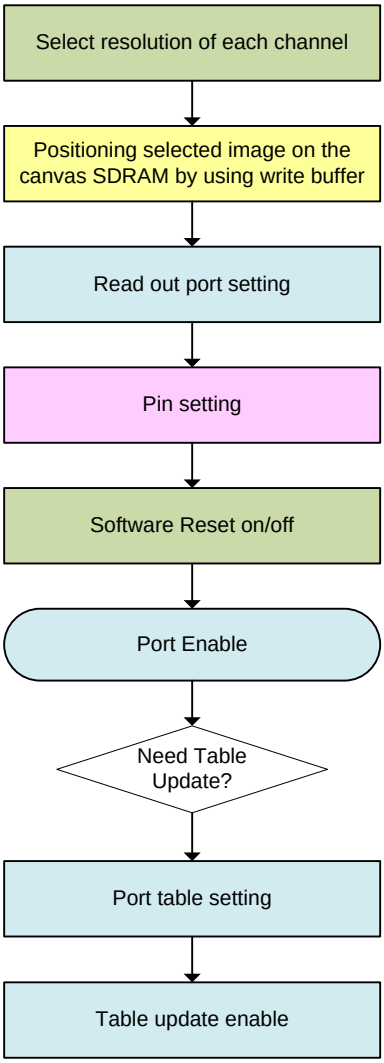


FIGURE 19. FLOW CHART FOR RECORD PROGRAMMING

Write Buffer Setting

256Mbit

This memory configuration can be used for FLI mode or FMI mode except full 16-D1 resolution.

Each write buffer can select channel number and buffer position by setting register 0xC00 ~ 0xC0F and 0xC10 ~ 0xC1F. Each write buffer can select image resolution and recording format.

Example codes are as the follows.

CASE 1: 16-D1, FLI AND NTSC (REFER TO FIGURE 20)

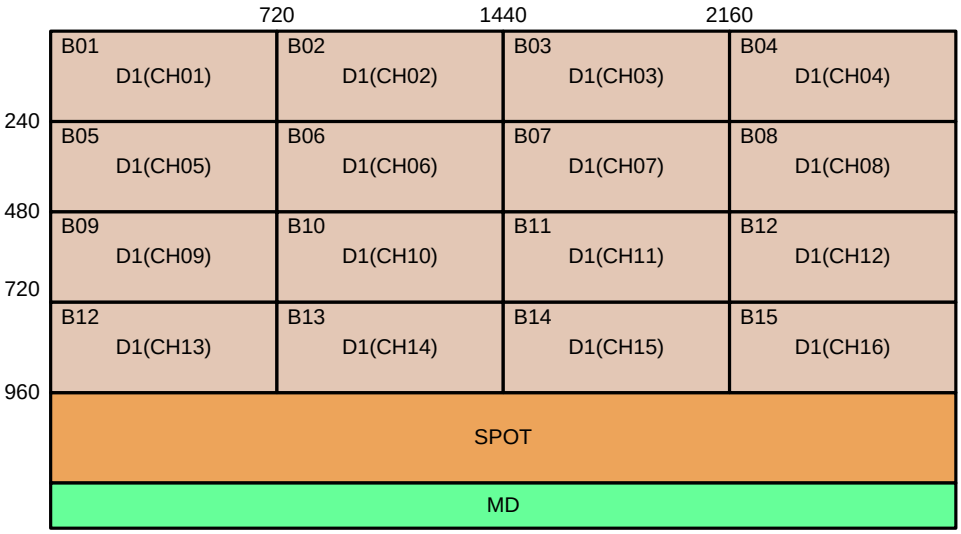


FIGURE 20. WRITE BUFFER SETTING EXAMPLE FOR 16-D1, FLI MODE AND NTSC

TABLE 1. WRITE BUFFER SETTING EXAMPLE CODE FOR 16-D1, FLI MODE AND NTSC

```
; Buffer control setting ( [6]Recoding format, [5:4]Resolution, [3:0] Channel number)
ww   0c00    40    ;   Buf 1 control (FLI mode, D1, CH num : 01)
ww   0c01    41    ;   Buf 2 control (FLI mode, D1, CH num : 02)
ww   0c02    42    ;   Buf 3 control (FLI mode, D1, CH num : 03)
ww   0c03    43    ;   Buf 4 control (FLI mode, D1, CH num : 04)
ww   0c04    44    ;   Buf 5 control (FLI mode, D1, CH num : 05)
ww   0c05    45    ;   Buf 6 control (FLI mode, D1, CH num : 06)
ww   0c06    46    ;   Buf 7 control (FLI mode, D1, CH num : 07)
ww   0c07    47    ;   Buf 8 control (FLI mode, D1, CH num : 08)
ww   0c08    48    ;   Buf 9 control (FLI mode, D1, CH num : 09)
ww   0c09    49    ;   Buf 10 control (FLI mode, D1, CH num : 10)
ww   0c0a    4a    ;   Buf 11 control (FLI mode, D1, CH num : 11)
ww   0c0b    4b    ;   Buf 12 control (FLI mode, D1, CH num : 12)
ww   0c0c    4c    ;   Buf 13 control (FLI mode, D1, CH num : 13)
ww   0c0d    4d    ;   Buf 14 control (FLI mode, D1, CH num : 14)
ww   0c0e    4e    ;   Buf 15 control (FLI mode, D1, CH num : 15)
ww   0c0f    4f    ;   Buf 16 control (FLI mode, D1, CH num : 16)

; Buffer position selection and on/off control setting( [7]on/off, [6:3]Hori. position, [2:0] Verti position)
ww   0c10    80    ;   Buf 1 position setting
ww   0c11    90    ;   Buf 2 position setting
ww   0c12    a0    ;   Buf 3 position setting
```

```
ww   0c13    b0    ;   Buf 4 position setting
ww   0c14    82    ;   Buf 5 position setting
ww   0c15    92    ;   Buf 6 position setting
ww   0c16    a2    ;   Buf 7 position setting
ww   0c17    b2    ;   Buf 8 position setting
ww   0c18    84    ;   Buf 9 position setting
ww   0c19    94    ;   Buf 10 position setting
ww   0c1a    a4    ;   Buf 11 position setting
ww   0c1b    b4    ;   Buf 12 position setting
ww   0c1c    86    ;   Buf 13 position setting
ww   0c1d    96    ;   Buf 14 position setting
ww   0c1e    a6    ;   Buf 15 position setting
ww   0c1f    b6    ;   Buf 16 position setting
```

CASE 2: 16-D1, FMI AND NTSC (REFER TO FIGURE 21)

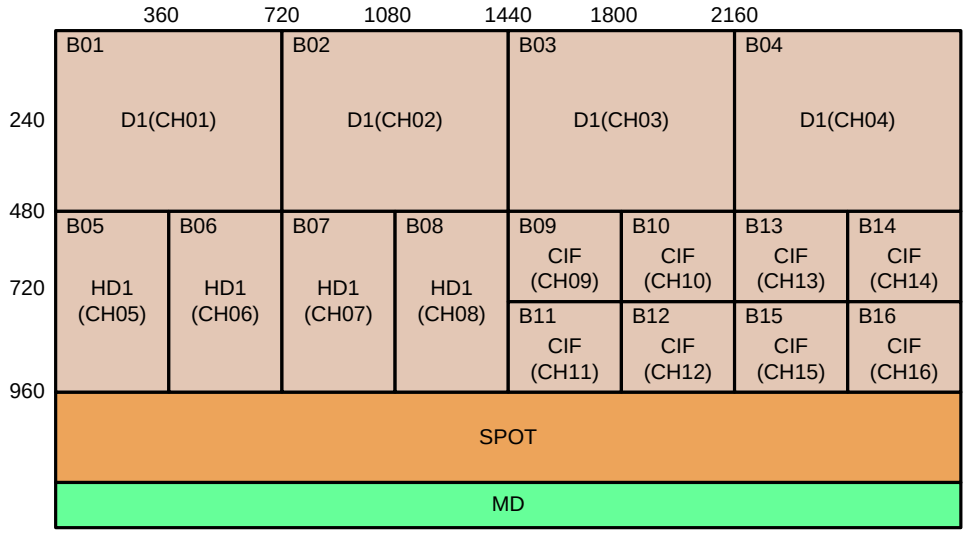


FIGURE 21. WRITE BUFFER SETTING EXAMPLE FOR MIXED RESOLUTION, FMI MODE AND NTSC

TABLE 2. WRITE BUFFER SETTING EXAMPLE CODE FOR MIXED RESOLUTION, FMI MODE AND NTSC

```
; Buffer control setting ( [6]Recoding format, [5:4]Resolution, [3:0] Channel number)
ww   0c00    40    ;   Buf 1 control (FLI mode, D1, CH num : 01)
ww   0c01    41    ;   Buf 2 control (FLI mode, D1, CH num : 02)
ww   0c02    42    ;   Buf 3 control (FLI mode, D1, CH num : 03)
ww   0c03    43    ;   Buf 4 control (FLI mode, D1, CH num : 04)
ww   0c04    54    ;   Buf 5 control (FLI mode, HD1, CH num : 05)
ww   0c05    55    ;   Buf 6 control (FLI mode, HD1, CH num : 06)
ww   0c06    56    ;   Buf 7 control (FLI mode, HD1, CH num : 07)
ww   0c07    57    ;   Buf 8 control (FLI mode, HD1, CH num : 08)
ww   0c08    68    ;   Buf 9 control (FLI mode, CIF, CH num : 09)
ww   0c09    69    ;   Buf 10 control (FLI mode, CIF, CH num : 10)
ww   0c0a    6a    ;   Buf 11 control (FLI mode, CIF, CH num : 11)
ww   0c0b    6b    ;   Buf 12 control (FLI mode, CIF, CH num : 12)
ww   0c0c    6c    ;   Buf 13 control (FLI mode, CIF, CH num : 13)
ww   0c0d    6d    ;   Buf 14 control (FLI mode, CIF, CH num : 14)
ww   0c0e    6e    ;   Buf 15 control (FLI mode, CIF, CH num : 15)
ww   0c0f    6f    ;   Buf 16 control (FLI mode, CIF, CH num : 16)
```

```
; Buffer position selection and on/off control setting( [7]on/off, [6:3]Hori. position, [2:0] Verti position)
ww   0c10    80   ;   Buf 1 position setting
ww   0c11    90   ;   Buf 2 position setting
ww   0c12    a0   ;   Buf 3 position setting
ww   0c13    b0   ;   Buf 4 position setting
ww   0c14    84   ;   Buf 5 position setting
ww   0c15    8c   ;   Buf 6 position setting
ww   0c16    94   ;   Buf 7 position setting
ww   0c17    9c   ;   Buf 8 position setting
ww   0c18    a4   ;   Buf 9 position setting
ww   0c19    ac   ;   Buf 10 position setting
ww   0c1a    a6   ;   Buf 11 position setting
ww   0c1b    ae   ;   Buf 12 position setting
ww   0c1c    b4   ;   Buf 13 position setting
ww   0c1d    bc   ;   Buf 14 position setting
ww   0c1e    b6   ;   Buf 15 position setting
ww   0c1f    be   ;   Buf 16 position setting
```

512Mbit

This memory configuration can be used for full 16-D1 resolution and stores 4 frames for each channel.

In this mode, we need to turn on the 2nd SDRAM by setting register 0xCCA and 0xCCB.

If you want to store buffer 9 ~ buffer 16 to 2nd SDRAM, you need to set register 0xCCB to '0xFF'.

Example codes are as follows.

CASE 1: 16-D1, FMI AND NTSC(REFER TO FIGURE 22)

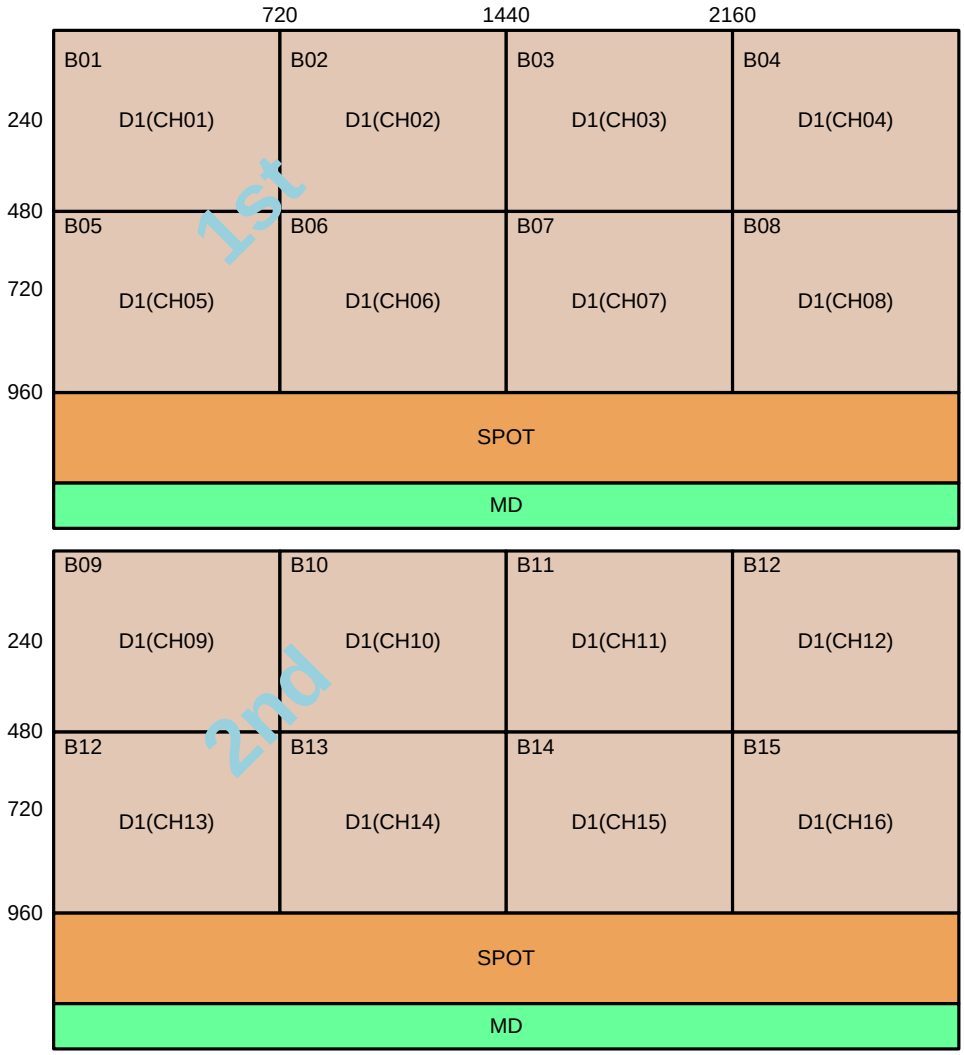


FIGURE 22. WRITE BUFFER SETTING EXAMPLE FOR 16-D1, FMI MODE AND NTSC

TABLE 3. WRITE BUFFER SETTING EXAMPLE CODE FOR 16-D1, FMI MODE AND NTSC

```
; Buffer control setting ( [6]Recoding format, [5:4]Resolution, [3:0] Channel number)
ww   0c00    00   ;   Buf 1 control (FMI mode, D1, CH num : 01)
ww   0c01    01   ;   Buf 2 control (FMI mode, D1, CH num : 02)
ww   0c02    02   ;   Buf 3 control (FMI mode, D1, CH num : 03)
ww   0c03    03   ;   Buf 4 control (FMI mode, D1, CH num : 04)
ww   0c04    04   ;   Buf 5 control (FMI mode, D1, CH num : 05)
```

```
ww   0c05    05    ;   Buf 6 control (FMI mode, D1, CH num : 06)
ww   0c06    06    ;   Buf 7 control (FMI mode, D1, CH num : 07)
ww   0c07    07    ;   Buf 8 control (FMI mode, D1, CH num : 08)
ww   0c08    08    ;   Buf 9 control (FMI mode, D1, CH num : 09)
ww   0c09    09    ;   Buf 10 control (FMI mode, D1, CH num : 10)
ww   0c0a    0a    ;   Buf 11 control (FMI mode, D1, CH num : 11)
ww   0c0b    0b    ;   Buf 12 control (FMI mode, D1, CH num : 12)
ww   0c0c    0c    ;   Buf 13 control (FMI mode, D1, CH num : 13)
ww   0c0d    0d    ;   Buf 14 control (FMI mode, D1, CH num : 14)
ww   0c0e    0e    ;   Buf 15 control (FMI mode, D1, CH num : 15)
ww   0c0f    0f    ;   Buf 16 control (FMI mode, D1, CH num : 16)

; Buffer position selection and on/off control setting( [7]on/off, [6:3]Hori. position, [2:0] Verti position)
ww   0c10    80    ;   Buf 1 position setting
ww   0c11    90    ;   Buf 2 position setting
ww   0c12    a0    ;   Buf 3 position setting
ww   0c13    b0    ;   Buf 4 position setting
ww   0c14    84    ;   Buf 5 position setting
ww   0c15    94    ;   Buf 6 position setting
ww   0c16    a4    ;   Buf 7 position setting
ww   0c17    b4    ;   Buf 8 position setting
; Turn on second SDRAM for buffer 9 ~ buffer 16 writing
ww   0ccb    ff    ;   Turn on second SDRAM for buffer 9 ~ buffer 16
ww   0c18    80    ;   Buf 9 position setting
ww   0c19    90    ;   Buf 10 position setting
ww   0c1a    a0    ;   Buf 11 position setting
ww   0c1b    b0    ;   Buf 12 position setting
ww   0c1c    84    ;   Buf 13 position setting
ww   0c1d    94    ;   Buf 14 position setting
ww   0c1e    a4    ;   Buf 15 position setting
ww   0c1f    b4    ;   Buf 16 position setting
```

SPOT Buffer

SPOT buffer uses the same SDRAM memory with record write buffers.

Each write buffer can select channel number and buffer position by setting register 0xC90 ~ 0xC9F and 0xCA0 ~ 0xCAF. Each write buffer can select image resolution and recording format.

Example codes are as follows.

CASE 1: 16-CIF (REFER TO FIGURE 23)

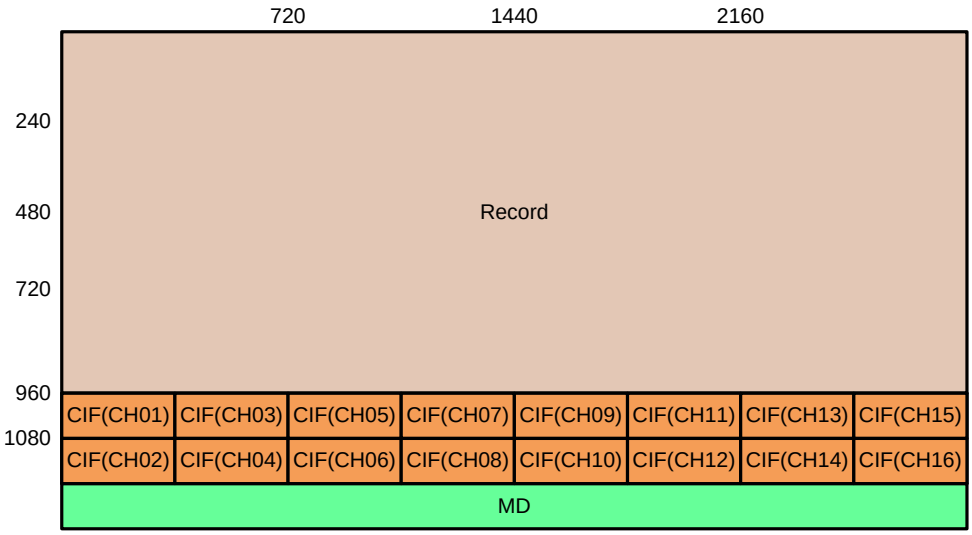


FIGURE 23. SPOT WRITE BUFFER SETTING EXAMPLE FOR 16-CIF, FLI MODE AND NTSC

TABLE 4. SPOT WRITE BUFFER SETTING EXAMPLE CODE FOR 16-CIF, FLI MODE AND NTSC

```
; Buffer control setting ( [6]Recoding format, [5:4]Resolution, [3:0] Channel number)
ww   0c90    60   ;   Buf 1 control (FLI mode, CIF, CH num : 01)
ww   0c91    61   ;   Buf 2 control (FLI mode, CIF, CH num : 02)
ww   0c92    62   ;   Buf 3 control (FLI mode, CIF, CH num : 03)
ww   0c93    63   ;   Buf 4 control (FLI mode, CIF, CH num : 04)
ww   0c94    64   ;   Buf 5 control (FLI mode, CIF, CH num : 05)
ww   0c95    65   ;   Buf 6 control (FLI mode, CIF, CH num : 06)
ww   0c96    66   ;   Buf 7 control (FLI mode, CIF, CH num : 07)
ww   0c97    67   ;   Buf 8 control (FLI mode, CIF, CH num : 08)
ww   0c98    68   ;   Buf 9 control (FLI mode, CIF, CH num : 09)
ww   0c99    69   ;   Buf 10 control (FLI mode, CIF, CH num : 10)
ww   0c9a    6a   ;   Buf 11 control (FLI mode, CIF, CH num : 11)
ww   0c9b    6b   ;   Buf 12 control (FLI mode, CIF, CH num : 12)
ww   0c9c    6c   ;   Buf 13 control (FLI mode, CIF, CH num : 13)
ww   0c9d    6d   ;   Buf 14 control (FLI mode, CIF, CH num : 14)
ww   0c9e    6e   ;   Buf 15 control (FLI mode, CIF, CH num : 15)
ww   0c9f    6f   ;   Buf 16 control (FLI mode, CIF, CH num : 16)

; Buffer position selection and on/off control setting( [7]on/off, [6:3]Hori. position, [2:0] Verti. position)
ww   0ca0    86   ;   Buf 1 position setting
ww   0ca1    87   ;   Buf 2 position setting
ww   0ca2    8e   ;   Buf 3 position setting
ww   0ca3    8f   ;   Buf 4 position setting
ww   0ca4    96   ;   Buf 5 position setting
ww   0ca5    97   ;   Buf 6 position setting
```

```
ww   0ca6    9e    ;   Buf 7 position setting
ww   0ca7    9f    ;   Buf 8 position setting
ww   0ca8    a6    ;   Buf 9 position setting
ww   0ca9    a7    ;   Buf 10 position setting
ww   0caa    ae    ;   Buf 11 position setting
ww   0cab    af    ;   Buf 12 position setting
ww   0cac    b6    ;   Buf 13 position setting
ww   0cad    b7    ;   Buf 14 position setting
ww   0cae    be    ;   Buf 15 position setting
ww   0caf    bf    ;   Buf 16 position setting
```

Record read port can also use SPOT buffers by setting register 0xCCE and 0xCCF and source buffer number register (ex.0xC36[4]).

Example codes are as follows.

CASE 2: PORT 5 USES RECORD BUFFER 12, SPOT BUFFER 1, SPOT BUFFER 2 AND SPOT BUFFER 3

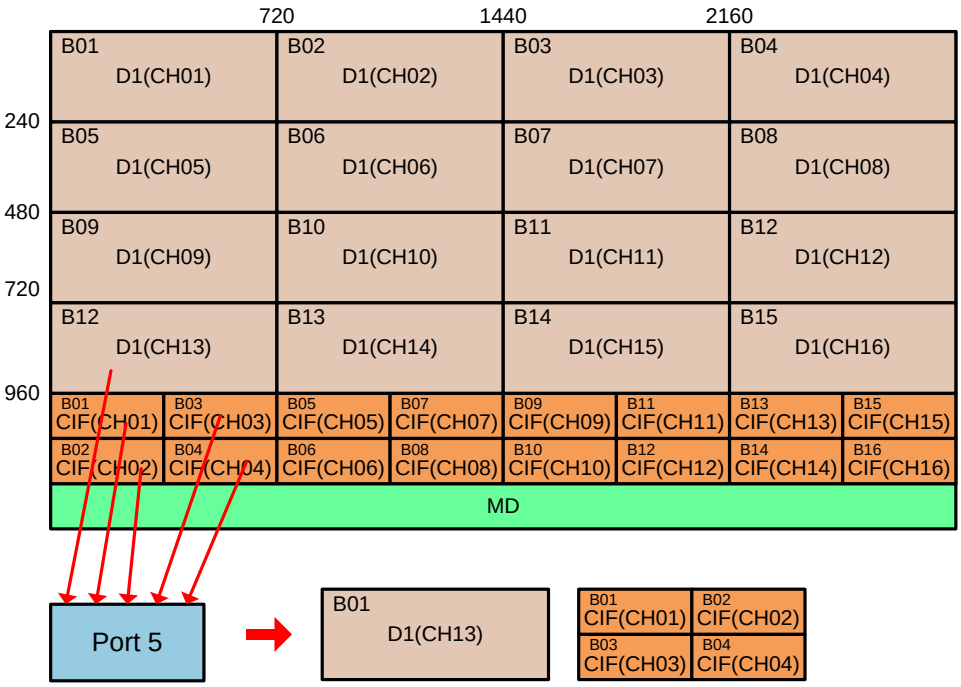


FIGURE 24. EXAMPLE FOR RECORD USING SPOT BUFFER

TABLE 5. EXAMPLE CODE FOR RECORD USING SPOT BUFFER

```
; SPOT buffer path change from SPOT to record
ww   0cce    0f    ;  SPOT buffer 1 ~ SPOT buffer 4 are used by record unit
; Source Mapping
ww   0c35    00    ;  Source table index value is '0'
ww   0c36    0c    ;  Source buffer number : 1st buffer number is '12'
ww   0c35    01    ;  Source table index value is '1'
ww   0c36    10    ;  Source buffer number : 2nd buffer number is '16' that is 1st SPOT buffer
ww   0c35    02    ;  Source table index value is '2'
ww   0c36    11    ;  Source buffer number : 3rd buffer number is '17' that is 2nd SPOT buffer
ww   0c35    03    ;  Source table index value is '2'
ww   0c36    12    ;  Source buffer number : 3rd buffer number is '18' that is 3rd SPOT buffer
ww   0c35    04    ;  Source table index value is '2'
ww   0c36    13    ;  Source buffer number : 3rd buffer number is '19' that is 4th SPOT buffer
```

Read Port Setting

TW2880 has two type ports. One is normal port and the other is multi port. Port 1 ~ Port 4 are normal ports and support only 4 channel index. Port 5 ~ Port 8 are multi ports and can support up to 128-channel index.

Normal Port(Port 1 ~ Port 4)

The following four registers control each port.

Port control register (ex. 0xC20)

Source selection register A (ex. 0xC21)

Source selection register B (ex. 0xC22)

Source number register (0xC4C)

Port control register controls output resolution, output format, split, output clock rate and port on/off control

Source selection register A controls 1st and 2nd source buffer index.

Source selection register B controls 3rd and 4th source buffer index.

Source number register controls number of sources that is need to used.

Example codes are as follows.

CASE 1: PORT 1, D1, FMI AND 27MHZ (REFER TO FIGURE 25)

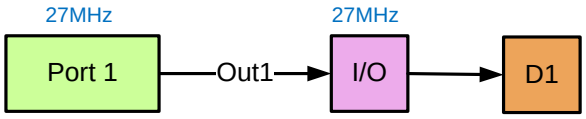
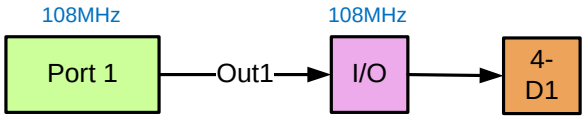


FIGURE 25. PORT SETTING EXAMPLE 1 : D1

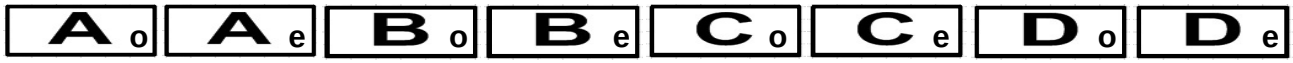
TABLE 6. PORT SETTING EXAMPLE CODE 1 : D1

```
ww   0c20    00   ;   Port 1 control register : D1, FMI, no split, 27MHz data rate and off
ww   0c21    10   ;   Port 1 Source selection register A : 1st source come from 2nd buffer
ww   0c4c    00   ;   Port 1 Source number register : Total number of sources is 1

;Other register setting and software reset

ww   0c20    01   ;   Port 1 control register : D1, FMI, no split, 27MHz data rate and on
```

CASE 2: PORT 1, 4-D1, FMI AND 108MHZ (REFER TO FIGURE 26)



(A) PORT CONFIGURATION



(B) IMAGE FLOW IN INTERLACED MODE

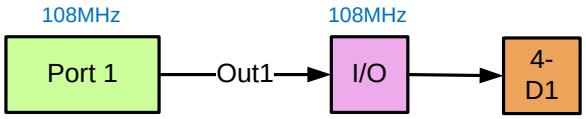
(C) IMAGE FLOW IN PROGRESSIVE MODE

FIGURE 26. PORT SETTING EXAMPLE 2 : 4-D1, FMI

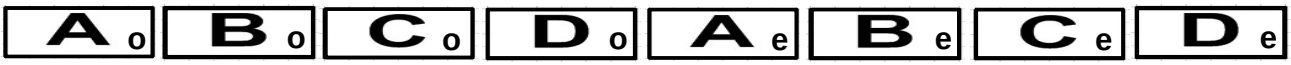
TABLE 7. PORT SETTING EXAMPLE CODE 2: 4-D1, FMI

```
;Interlaced or progressive frame selection
ww   0c69    ff    ;   This setting value is for 'Interlaced Frame mode' (Refer to the Figure 26 (A)). For
                       progressive mode, you need to set this register to '0xff'(Refer to Figure 26 (B))

ww   0c20    04    ;   Port 1 control register : D1, FMI, no split, 108MHz data rate and off
ww   0c21    01    ;   Port 1 Source selection register A : 1st source(1st buf), 2nd source(2nd buf)
ww   0c22    23    ;   Port 1 Source selection register B : 3rd source(3rd buf), 4th(4th buf)
ww   0c4c    03    ;   Port 1 Source number register : Total number of sources is 4

;Other register setting and software reset

ww   0c20    05    ;   Port 1 control register : D1, FMI, no split, 108MHz data rate and on
```

CASE 3: PORT 1, 4-D1, FLI AND 108MHZ (REFER TO FIGURE 27)



(A) PORT CONFIGURATION



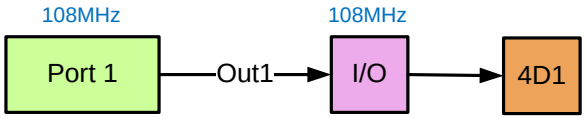
(B) IMAGE FLOW (FLI MODE SUPPORTS ONLY INTERLACED MODE)

FIGURE 27. PORT SETTING EXAMPLE 3 : 4-D1, FLI

TABLE 8. PORT SETTING EXAMPLE CODE 3 : 4-D1, FLI

```
ww   0c20    24   ;   Port 1 control register : D1, FLI, no split, 108MHz data rate and off
ww   0c21    01   ;   Port 1 Source selection register A : 1st source(1st buf), 2nd source(2nd buf)
ww   0c22    23   ;   Port 1 Source selection register B : 3rd source(3rd buf), 4th(4th buf)
ww   0c4c    03   ;   Port 1 Source number register : Total number of sources is 4

;Other register setting and software reset

ww   0c20    25   ;   Port 1 control register : D1, FLI, no split, 108MHz data rate and on
```

CASE 4: PORT 1, 4D1 MODE (SPECIAL), FLI AND 108MHZ (REFER TO FIGURE 28)



PORT CONFIGURATION

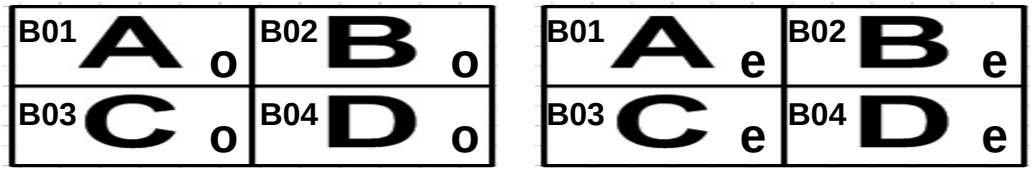


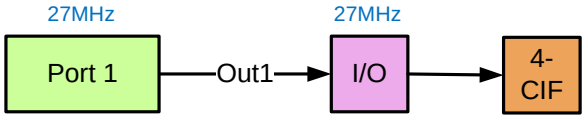
IMAGE FLOW

FIGURE 28. PORT SETTING EXAMPLE 4 : 4-D1, FLI

TABLE 9. PORT SETTING EXAMPLE CODE 4 : 4-D1, FLI

```
ww   0c20    ba    ;   Port1 control register : 4D1, FLI, x and y split, 108MHz data rate and off
ww   0c21    01    ;   Port 1 Source selection register A : 1st source(1st buf), 2nd source(2nd buf)
ww   0c22    23    ;   Port 1 Source selection register B : 3rd source(3rd buf), 4th(4th buf)
ww   0c4c    00    ;   Port 1 Source number register : Total number of sources is 4 but need to set '0'

;Other register setting and software reset

ww   0c20    bb    ;   Port 1 control register : 4D1, FLI, no split, 108MHz data rate and on
```

CASE 5: PORT 1, 4-CIF AND 27MHZ (REFER TO FIGURE 29)



(A) PORT CONFIGURATION

(B) IMAGE FLOW

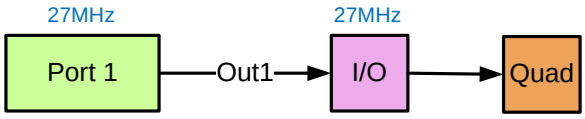
FIGURE 29. PORT SETTING EXAMPLE 5 : 4-CIF

TABLE 10. PORT SETTING EXAMPLE CODE 5 : 4-CIF

```
ww   0c20    40    ;   Port1 control register : CIF, no split, 27MHz data rate and off
ww   0c21    01    ;   Port1 Source selection register A : 1st source(1st buf), 2nd source(2nd buf)
ww   0c22    23    ;   Port1 Source selection register B : 3rd source(3rd buf), 4th(4th buf)
ww   0c4c    03    ;   Port1 Source number register : Total number of sources is 4

;Other register setting and software reset
```

```
ww   0c20    05    ;   Port control register : CIF, no split, 27MHz data rate and on
```

CASE 6: PORT 1, QUAD, FLI AND 27MHZ (REFER TO FIGURE 30)



(A) PORT CONFIGURATION

(B) IMAGE FLOW

FIGURE 30. PORT SETTING EXAMPLE 6 : QUAD

TABLE 11. PORT SETTING EXAMPLE CODE 6 : QUAD

```
ww   0c20    38    ;   Port 1 control register : CIF, no split, 108MHz data rate and off
ww   0c21    01    ;   Port 1 Source selection register A : 1st source(1st buf), 2nd source(2nd buf)
ww   0c22    23    ;   Port 1 Source selection register B : 3rd source(3rd buf), 4th(4th buf)
ww   0c4c    00    ;   Port 1 Source number register : Total number of sources is 4 but need to set '0'

;Other register setting and software reset

ww   0c20    39    ;   Port 1 control register : D1, FLI, x and y split, 27MHz data rate and on
```

Multi Port (Port 5 ~ Port 8)

Each port is controlled by the following four registers.

Port control register (ex. 0xC34)

Source table index number (ex. 0xC35)

Source buffer number (ex. 0xC36)

Source number register (0xC37)

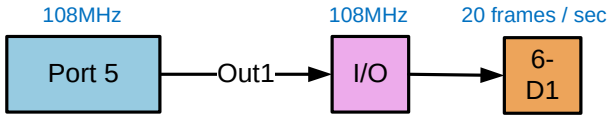
Port control register controls output resolution, output format, output clock rate and port on/off control.

Multi port can support up to 128-source sequence by using two registers; one is source table index number and the other is source buffer number.

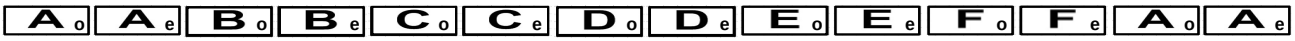
Source number register controls number of sources that is need to used.

Example codes are as follows.

CASE 1: PORT 5, 6-D1, FMI, 108MHZ (REFER TO FIGURE 31)



(A) PORT CONFIGURATION



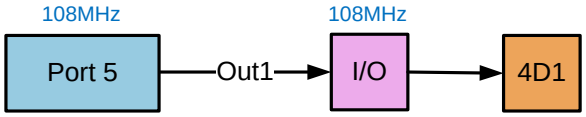
(B) IMAGE FLOW

FIGURE 31. PORT SETTING EXAMPLE 1 : 6-D1

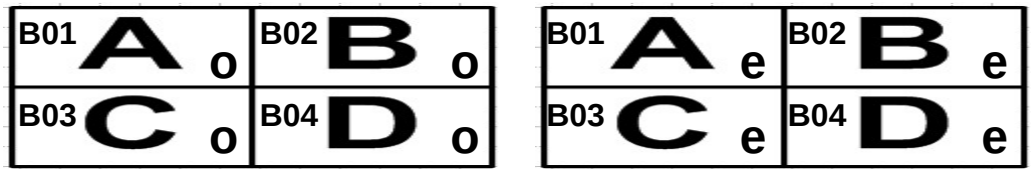
TABLE 12 PORT SETTING EXAMPLE CODE 1 : 6-D1

```
ww   0c34   04   ;   Port 5 control register : D1, FMI, 108MHz data rate and off
ww   0c35   00   ;   Port 5 Source table index value is '0'
ww   0c36   00   ;   Port 5 Source buffer number : 1st buffer number is '0'
ww   0c35   01   ;   Port 5 Source table index value is '1'
ww   0c36   02   ;   Port 5 Source buffer number : 2nd buffer number is '1'
ww   0c35   03   ;   Port 5 Source table index value is '2'
ww   0c36   03   ;   Port 5 Source buffer number : 3rd buffer number is '2'
ww   0c35   04   ;   Port 5 Source table index value is '3'
ww   0c36   04   ;   Port 5 Source buffer number : 4th buffer number is '3'
ww   0c35   05   ;   Port 5 Source table index value is '4'
ww   0c36   05   ;   Port 5 Source buffer number : 5th buffer number is '4'
ww   0c35   02   ;   Port 5 Source table index value is '5'
ww   0c36   01   ;   Port 5 Source buffer number : 6th buffer number is '5'
ww   0c37   05   ;   Port 5 Source number register : Total number of sources is 6(20 frame per sec)

;Other register setting and software reset

ww   0c34   05   ;   Port 5 control register : D1, FMI, 108MHz data rate and on
```

CASE 2: PORT 5, 4D1 MODE(SPECIAL), FLI AND 108MHZ (REFER TO FIGURE 32)



(A) PORT CONFIGURATION



(B) IMAGE FLOW

FIGURE 32. PORT SETTING EXAMPLE 2 : 4D1, FLI

TABLE 13 PORT SETTING EXAMPLE CODE 2 : 4D1, FLI

```
ww   0c34    a4    ;  Port 5 control register : 4D1 mode(Special), FLI, 108MHz data rate and off
ww   0c35    00    ;  Port 5 Source table index value is '0'
ww   0c36    00    ;  Port 5 Source buffer number : 1st buffer number is '0'
ww   0c35    01    ;  Port 5 Source table index value is '1'
ww   0c36    02    ;  Port 5 Source buffer number : 2nd buffer number is '1'
ww   0c35    03    ;  Port 5 Source table index value is '2'
ww   0c36    03    ;  Port 5 Source buffer number : 3rd buffer number is '2'
ww   0c35    04    ;  Port 5 Source table index value is '3'
ww   0c36    04    ;  Port 5 Source buffer number : 4th buffer number is '3'
ww   0c37    03    ;  Port 5 Source number register : Total number of sources is 4

;Other register setting and software reset

ww   0c34    a5    ;  Port 5 control register : 4D1 mode(Special), FLI, 108MHz data rate and on
```

CASE 3: TABLE LIVE UPDATE

When port is on, source table setting does not take affect until update enable bit (source buffer number register: ex. 0xC36[5]) is set to '1'. When port is off, source table setting affects immediately.

TABLE 14 TABLE LIVE UPDATE EXAMPLE CODE

```
;Change table index
ww   0c35    00    ;  Port 5 Source table index value is '0'
ww   0c36    00    ;  Port 5 Source buffer number : 1st buffer number is '0'
ww   0c35    01    ;  Port 5 Source table index value is '1'
ww   0c36    02    ;  Port 5 Source buffer number : 2nd buffer number is '1'
ww   0c35    03    ;  Port 5 Source table index value is '2'
ww   0c36    03    ;  Port 5 Source buffer number : 3rd buffer number is '2'
ww   0c35    04    ;  Port 5 Source table index value is '3'
ww   0c36    04    ;  Port 5 Source buffer number : 4th buffer number is '3'
ww   0c37    03    ;  Port 5 Source number register : Total number of sources is 4
```

```
;Table update enable
ww    0c36      24    ;    Port 5 table update enable
```

Output Pin Setting

Port Muxing

TW2880 record has 8 ports and 4 output pins.

Each port supports both of the 8-bit and 16-bit data transfer (refer to Figure 33).

Each port has 16-bit output. In case of 16-bit mode, lsb 8-bit (ex. out1[7:0]) is for Y data, msb 8-bit (ex. out1[15:8]) is for Cb or Cr data. In case of 8-bit mode, only lsb 8-bit is used and msb 8-bit is not used.

Each output pin has two clocks of which phase can be controlled individually and then 2-port output can be transferred by using 1 output pin.

Each output pin can select two 8-bit data(ex. out1_A and out1_B) from any port and any byte(lsb 8-bit or msb 8-bit) by setting 0xC4E ~ 0xC51 and 0xCF5 ~ 0xCF6 register. Register 0xC4E ~ 0xC51 select 1st phase data (ex. out1_A) and that data uses with positive clock (rec*_clkp). Register 0xCF5 and 0xCF6 select 2nd phase data(ex.out1_B) and that data uses with negative clock(rec*_clkn).

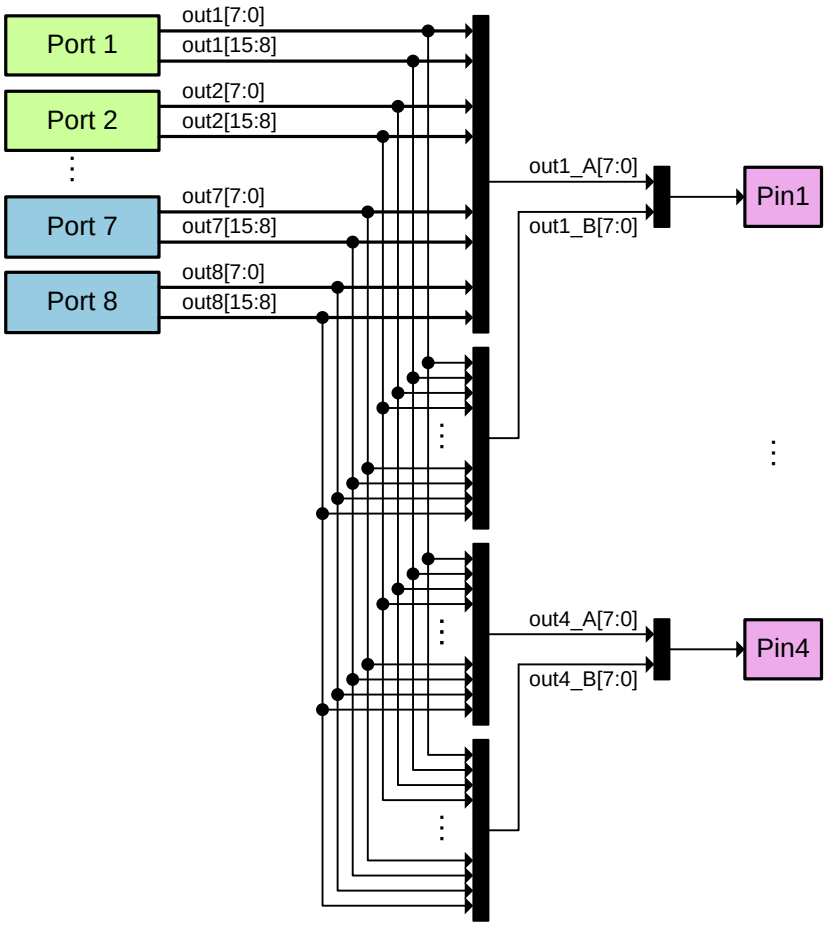


FIGURE 33. OUTPUT PIN MUXING

CASE 1: OUTPUT PIN 1, 8-BIT, 1 CODEC (REFER TO FIGURE 34)

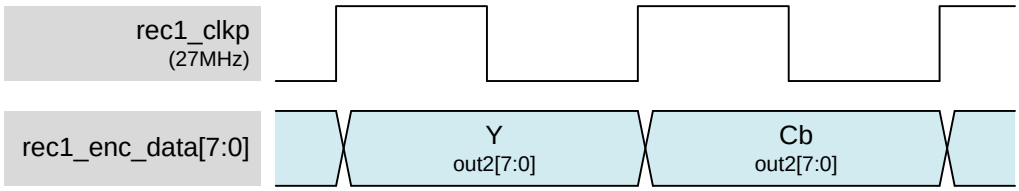


FIGURE 34. OUTPUT PIN SETTING EXAMPLE 1 : 8-BIT, 1-CODEC

TABLE 15. OUTPUT PIN SETTING EXAMPLE CODE 1 : 8-BIT, 1-CODEC

```
ww   0c4e   20   ;   Output pin 1 control : assign port2 lsb to pin 1 source 1, 8-bit, 27MHz

ww   0cf5   02   ;   Output pin set source 2 control : assign port2 lsb to pin 1 source 2
```

CASE 2: OUTPUT PIN 1, 8-BIT, 2 CODEC (REFER TO FIGURE 35)

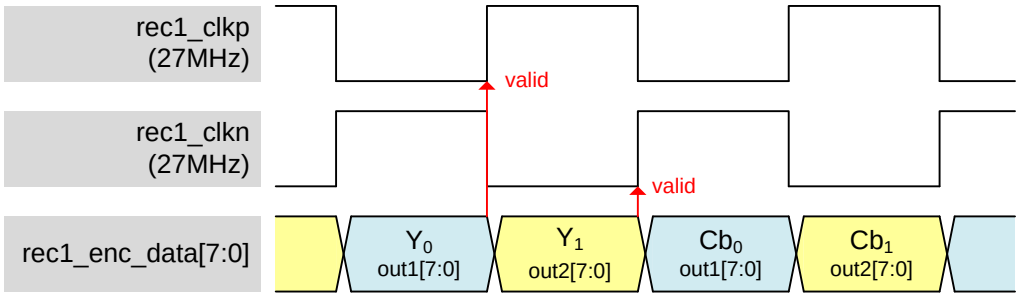


FIGURE 35. OUTPUT PIN SETTING EXAMPLE 2 : 8-BIT, 2-CODEC

TABLE 16 OUTPUT PIN SETTING EXAMPLE CODE 2 : 8-BIT, 2-CODEC

```
ww   0c4e   00   ;   Output pin 1 control : assign port1 lsb to pin 1 source 1, 8-bit, 27MHz

ww   0cf5   02   ;   Output pin set source 2 control : assign port2 lsb to pin 1 source 2
```

CASE 3: 16-BIT, 1 CODEC, 54MHZ (REFER TO FIGURE 36)

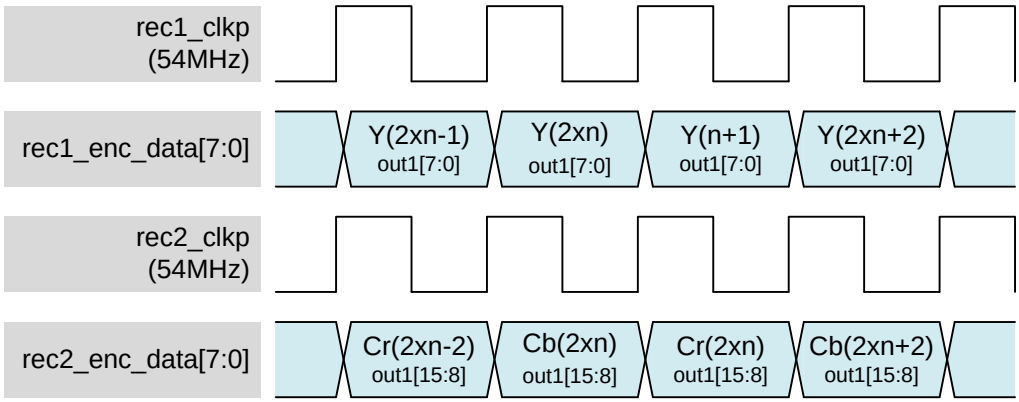


FIGURE 36. OUTPUT PIN SETTING EXAMPLE 3 : 16-BIT, 1-CODEC\

TABLE 17 OUTPUT PIN SETTING EXAMPLE CODE 3 : 16-BIT, 1-CODEC

ww	0c4e	05	;	Output pin 1 control : assign port1 lsb to pin 1 source 1, 16-bit, 54MHz
ww	0c4f	15	;	Output pin 2 control : assign port1 msb to pin 2 source 1, 16-bit, 54MHz
ww	0cf5	10	;	Output pin set source 2 control : pin 2 source 2(port1 msb), pin 1 source 2(port1 lsb)

Output Clock Selection

Record clock source can be selected among internal system clock (108MHz, sclk), internal video clock (variable frequency, vclk) and external clock source by setting register 0xC68[1:0].

Usually, internal system clock is used except BT.1120 mode. In case of BT.1120 mode, when video resolution is 1080i, the internal video clock can be used, otherwise, the external clock source needs to be used.

Output Clock Phase Control

Recording output clock phase needs to be controlled because there is some board delay and I/O delay in the system. TW2880 supports clock phase control for each record port with 4-phase shift value that has 0°, 90°, 180° and 270° value by setting register 0x219.

ETC

OSD

Refer to “Section 6: OSG and Simple OSD” starting on page 147.

Privacy Window

Live, record and SPOT have independent privacy windows and share address of shadow registers.

Shadow register can be selected by setting register 0xE4F[1:0].

Each channel has 4 independent privacy windows and control.

Users need to set start position of privacy window by using register 0xE50 ~ 0x E5F and 0xE60 ~ 0xE6F for horizontal and vertical start position. Users also need to set size of window by setting register 0xE70 ~ 0x7F.

Using 4 privacy windows, privacy windows cover up to 640x512 size area.

Privacy window has 8-type content and this contents can be selected by setting register 0xE70 ~ 0xE7F[7:5]

Case 1: Position(H: 288 pixels, V: 224 lines), Content(32x32 mosaic), Size(H:160 pixels, V:128 lines); refer to Figure 37.

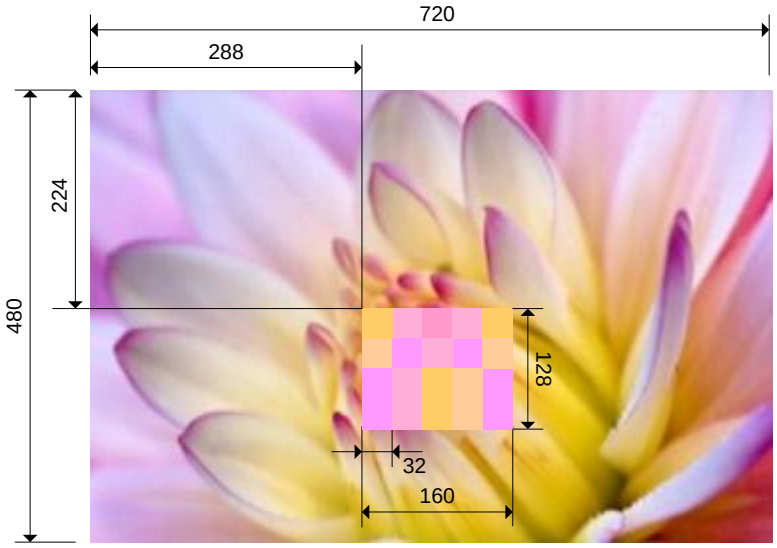


FIGURE 37. PRIVACY WINDOW SETTING EXAMPLE 1 : 16-BIT, 1-CODEC

TABLE 18 PRIVACY WINDOW SETTING EXAMPLE CODE 1 : 16-BIT, 1-CODEC

```
ww   0e4f    00   ;   Shadow register control : Record privacy control register on

ww   0e50    90   ;   Horizontal start position : 288 pixels(2x144)

ww   0e60    f0   ;   Privacy window enable and Vertical start position : 224 lines(2x112)

ww   0e70    f3   ;   Mosaic(32x32), Hori. size(160 pixels), Verti. size(128 lines)
```

You can use GUI control window for setting control register value (refer to Figure 38).

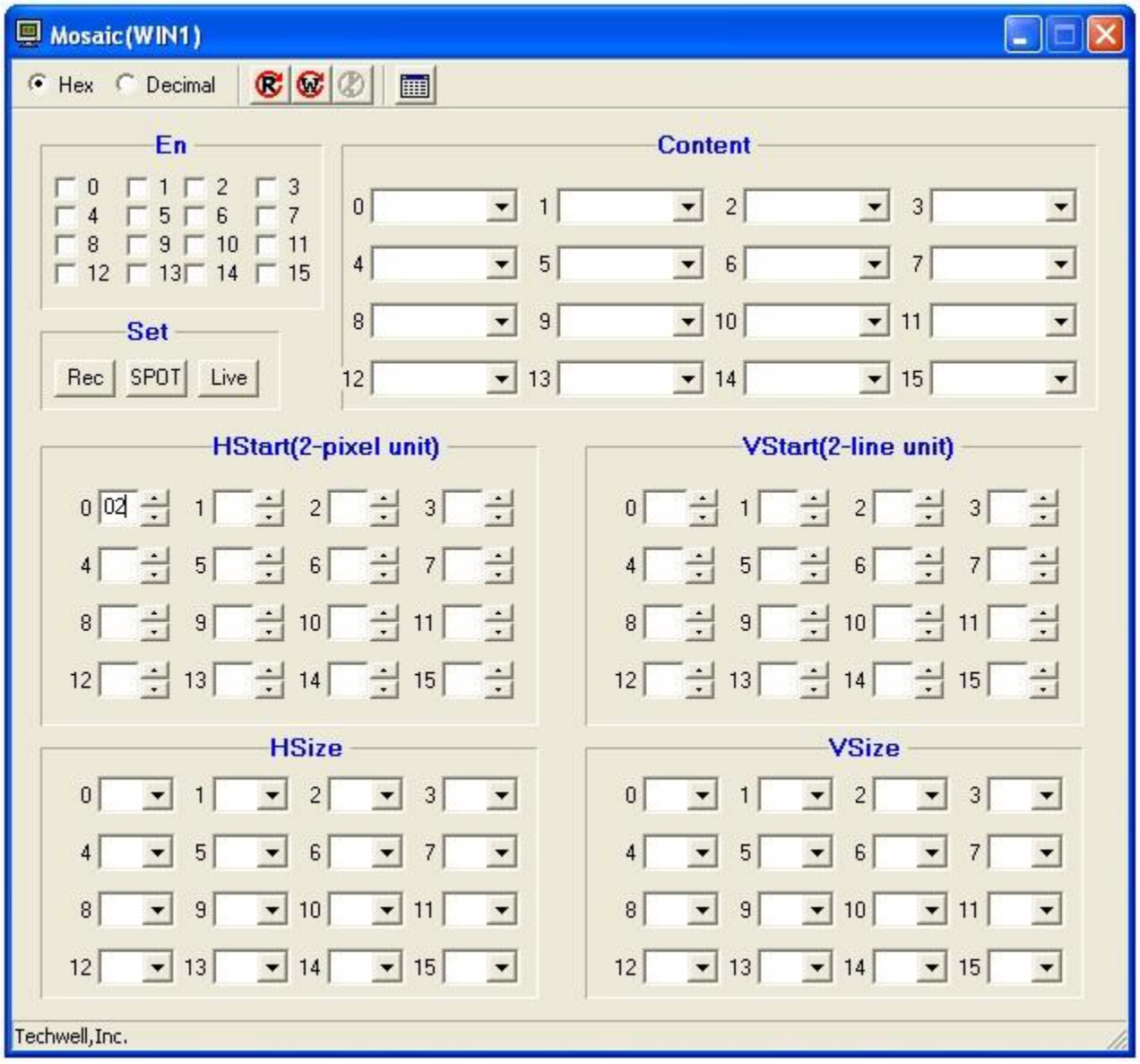


FIGURE 38. PRIVACY WINDOW' CONTROL WIDOW

Freeze

TW2880 can freeze every buffer with independent control register 0xC64 and 0xC65.

Register 0xC64 is for buffer 0 ~ buffer 7 and register 0xC65 is for buffer 8 ~ buffer 15.

BT.1120

In the 6 VGA modes, 6 cropped D1 images (640x240) make 1080i image. Each D1 image is horizontally cropped by setting horizontal offset control register, 0xCD0 ~ 0xCDF (Refer to Figure 39).

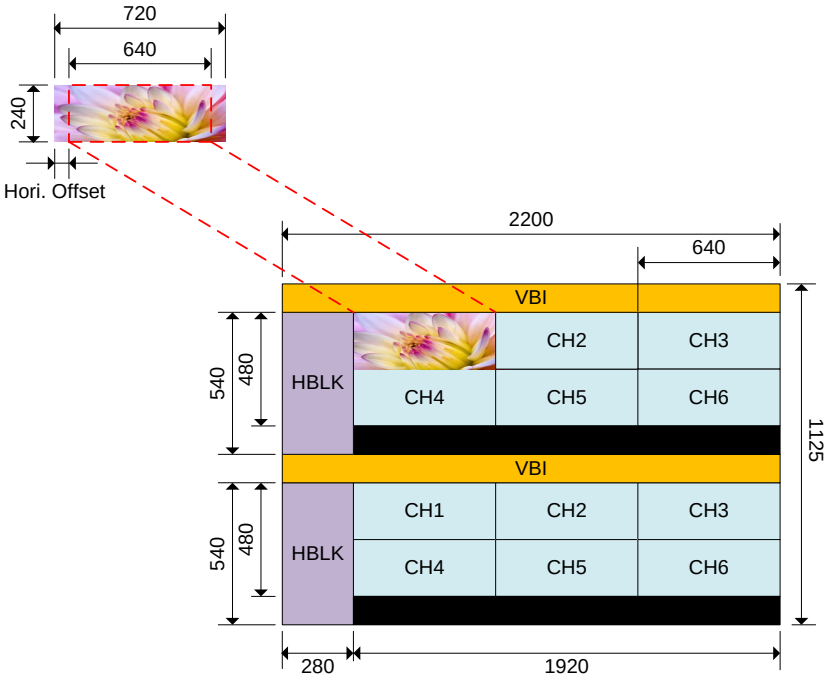


FIGURE 39. 6VGA(BT.1120) IMAGE MAPPING BY HORIZONTAL CROPPING

For BT.1120 mode, several register settings are need. When video resolution is 1080i, the internal video clock can be used (148.5MHz), otherwise, the external clock source needs to be used. Output pin clock need to set according to the port bit width. In the 16-bit mode, output pin clock need to set ½ internal operation clock (74.25MHz). In the 8-bit mode, output pin clock need to be same as the internal operation clock (148.5MHz). Even though 6 D1 images are used, only one 1st source setting is needed and number of source is needed to set '1'.

SPOT Connection

All record ports can be connected to any SPOT port by setting the following registers.

- 0xF9C and 0xFCC : Select record source included network port
- 0xF0E[7], 0xF6E[7], 0xF9E[7] and 0xFCE[7] : Enable control for connection record port to SPOT
- {0xF1B[1:0], 0xF1A[7:0]}, {0xF7B[1:0], 0xF7A[7:0]}, {0xFAB[1:0], 0xFAA[7:0]} and {0xFDB[1:0], 0xFDA[7:0]} : SPOT TV encoder active pixel delay control. This value is variable according to the display.

Frame Rate Control

In the multi port (port 5 ~ port 8 and network port), frame rate of each channel can be controlled by setting source table (ex. 0xC35 and 0xC36). For example, if you want to send 4 channel images with the following frame rate

CH1: 30 frames per sec, CH2: 15 frames per sec, CH3: 15 frames per sec

You need to set the source table as the following sequence

CH1, CH2, CH1, CH3

and the number of source is '4', port clock is '54MHz'.

Programming Example

Eight 2-D1, FLI

This setting records whole live input with D1 resolution using 8 ports (Refer to Figure 40).

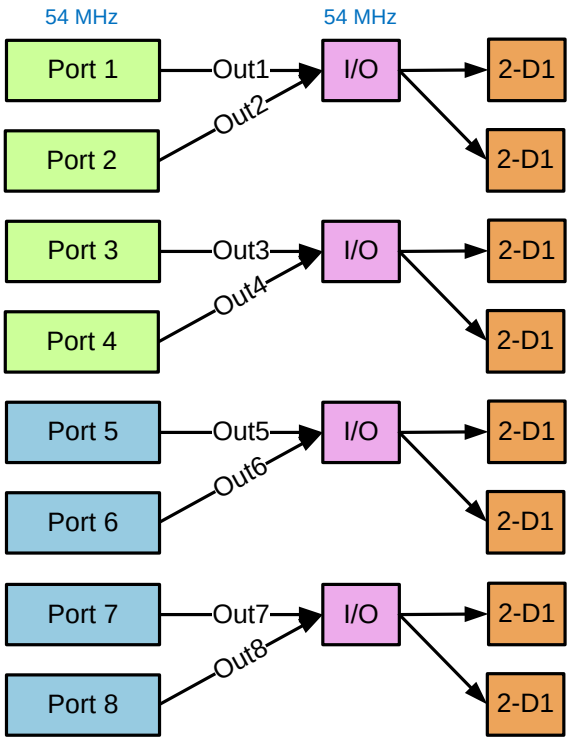


FIGURE 40. PROGRAMMING EXAMPLE 1 : EIGHT 2-D1, FLI

TABLE 19 PROGRAMMING EXAMPLE CODE 1 : EIGHT 2-D1, FLI

```
; Buffer control setting ( [6]Recoding format, [5:4]Resolution, [3:0] Channel number)
ww    0c00      40    ;    Buf 1 control (FLI mode, D1, CH num : 01)
ww    0c01      41    ;    Buf 2 control (FLI mode, D1, CH num : 02)
ww    0c02      42    ;    Buf 3 control (FLI mode, D1, CH num : 03)
ww    0c03      43    ;    Buf 4 control (FLI mode, D1, CH num : 04)
ww    0c04      44    ;    Buf 5 control (FLI mode, D1, CH num : 05)
ww    0c05      45    ;    Buf 6 control (FLI mode, D1, CH num : 06)
ww    0c06      46    ;    Buf 7 control (FLI mode, D1, CH num : 07)
ww    0c07      47    ;    Buf 8 control (FLI mode, D1, CH num : 08)
ww    0c08      48    ;    Buf 9 control (FLI mode, D1, CH num : 09)
ww    0c09      49    ;    Buf 10 control (FLI mode, D1, CH num : 10)
ww    0c0a      4a    ;    Buf 11 control (FLI mode, D1, CH num : 11)
ww    0c0b      4b    ;    Buf 12 control (FLI mode, D1, CH num : 12)
ww    0c0c      4c    ;    Buf 13 control (FLI mode, D1, CH num : 13)
ww    0c0d      4d    ;    Buf 14 control (FLI mode, D1, CH num : 14)
ww    0c0e      4e    ;    Buf 15 control (FLI mode, D1, CH num : 15)
ww    0c0f      4f    ;    Buf 16 control (FLI mode, D1, CH num : 16)
; Buffer position selection and on/off control setting( [7]on/off, [6:3]Hori. position, [2:0] Verti position)
ww    0c10      80    ;    Buf 1 position setting
ww    0c11      90    ;    Buf 2 position setting
ww    0c12      a0    ;    Buf 3 position setting
```

```
ww   0c13    b0    ;   Buf 4 position setting
ww   0c14    82    ;   Buf 5 position setting
ww   0c15    92    ;   Buf 6 position setting
ww   0c16    a2    ;   Buf 7 position setting
ww   0c17    b2    ;   Buf 8 position setting
ww   0c18    84    ;   Buf 9 position setting
ww   0c19    94    ;   Buf 10 position setting
ww   0c1a    a4    ;   Buf 11 position setting
ww   0c1b    b4    ;   Buf 12 position setting
ww   0c1c    86    ;   Buf 13 position setting
ww   0c1d    96    ;   Buf 14 position setting
ww   0c1e    a6    ;   Buf 15 position setting
ww   0c1f    b6    ;   Buf 16 position setting
;Port 1 ~ Port   4 setting : D1, FLI, 54 MHz
ww   0c20    22    ;   Port1 control register : D1, FMI, no split, 54MHz data rate and off
ww   0c21    01    ;   Port1 Source selection register A : 1st source(1st buf), 2nd source(2nd buf)
ww   0c25    22    ;   Port2 control register : D1, FMI, no split, 54MHz data rate and off
ww   0c26    23    ;   Port2 Source selection register A : 1st source(3rd buf), 2nd source(4th buf)
ww   0c2a    22    ;   Port3 control register : D1, FMI, no split, 54MHz data rate and off
ww   0c2b    45    ;   Port3 Source selection register A : 1st source(5th buf), 2nd source(6th buf)
ww   0c2f    22    ;   Port4 control register : D1, FMI, no split, 54MHz data rate and off
ww   0c30    67    ;   Port4 Source selection register A : 1st source(7th buf), 2nd source(8th buf)
;Number of active channel of port 1 ~ port 4
ww   0c4c    55    ;   Each port has 2 active channels
;Port 5 ~ Port 8 setting : D1, FLI, 54MHz
ww   0c34    22    ;   Port6 control register : D1, FMI, 54MHz data rate and off
ww   0c35    00    ;   Source table index value is '0'
ww   0c36    08    ;   Source buffer number : 1st buffer number is '8'
ww   0c35    01    ;   Source table index value is '1'
ww   0c36    09    ;   Source buffer number : 2nd buffer number is '9'
ww   0c37    01    ;   Source number register : Total number of sources is 2
ww   0c3a    22    ;   Port6 control register : D1, FMI, 54MHz data rate and off
ww   0c3b    00    ;   Source table index value is '0'
ww   0c3c    0a    ;   Source buffer number : 1st buffer number is '10'
ww   0c3b    01    ;   Source table index value is '1'
ww   0c3c    0b    ;   Source buffer number : 2nd buffer number is '11'
ww   0c3d    01    ;   Source number register : Total number of sources is 2
ww   0c40    22    ;   Port7 control register : D1, FMI, 54MHz data rate and off
ww   0c41    00    ;   Source table index value is '0'
ww   0c42    0c    ;   Source buffer number : 1st buffer number is '12'
ww   0c41    01    ;   Source table index value is '1'
ww   0c42    0d    ;   Source buffer number : 2nd buffer number is '13'
ww   0c43    01    ;   Source number register : Total number of sources is 2
ww   0c46    22    ;   Port8 control register : D1, FMI, 54MHz data rate and off
ww   0c47    00    ;   Source table index value is '0'
ww   0c48    0e    ;   Source buffer number : 1st buffer number is '14'
ww   0c47    01    ;   Source table index value is '1'
ww   0c48    0f    ;   Source buffer number : 2nd buffer number is '15'
ww   0c49    01    ;   Source number register : Total number of sources is 2
;Output Pin setting
ww   0c4e    01    ;   Output pin 1 control : assign port1 lsb to pin 1 source 1, 8-bit, 54MHz
ww   0c4f    41    ;   Output pin 2 control : assign port3 lsb to pin 1 source 1, 8-bit, 54MHz
ww   0c50    81    ;   Output pin 3 control : assign port5 lsb to pin 1 source 1, 8-bit, 54MHz
ww   0c51    c1    ;   Output pin 3 control : assign port7 lsb to pin 1 source 1, 8-bit, 54MHz
ww   0cf5    62        Output pin set source 2 control : Pin2 src 2(Port 4 lsb), Pin1 src 2(Port2 lsb)
ww   0cf6    ea    ;   Output pin set source 2 control : Pin4 src 2(Port 8 lsb), Pin3 src 2(Port6 lsb)
; Software reset
```

```
ww    020e    0f    ;    Active software reset for record port
ww    020e    00    ;    Release software reset for record port
; Enable port
ww    0c20    23    ;    Port1 control register : D1, FLI, no split, 54MHz data rate and on
ww    0c25    23    ;    Port2 control register : D1, FLI, no split, 54MHz data rate and on
ww    0c2a    23    ;    Port3 control register : D1, FLI, no split, 54MHz data rate and on
ww    0c2f    23    ;    Port4 control register : D1, FLI, no split, 54MHz data rate and on
ww    0c34    23    ;    Port6 control register : D1, FLI, 54MHz data rate and on
ww    0c3a    23    ;    Port6 control register : D1, FLI, 54MHz data rate and on
ww    0c40    23    ;    Port7 control register : D1, FLI, 54MHz data rate and on
ww    0c46    23    ;    Port8 control register : D1, FLI, 54MHz data rate and on
```

Four 4D1, FMI

This setting records whole live input with 4D1 resolution using 4 ports (Refer to Figure 41).

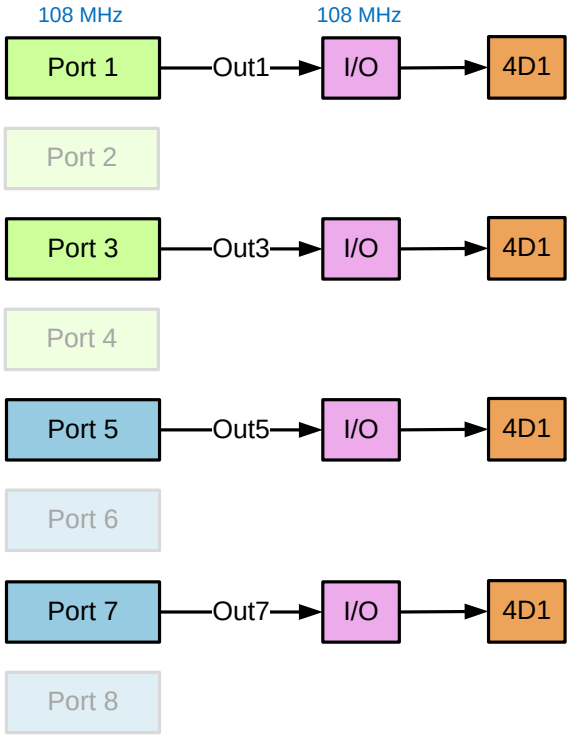


FIGURE 41. PROGRAMMING EXAMPLE 2 : FOUR 4D1, FMI

TABLE 20 PROGRAMMING EXAMPLE CODE 2 : FOUR 4D1, FMI

```
; Buffer control setting ( [6]Recoding format, [5:4]Resolution, [3:0] Channel number)
ww    0c00    00    ;    Buf 1 control (FMI mode, D1, CH num : 01)
ww    0c01    01    ;    Buf 2 control (FMI mode, D1, CH num : 02)
ww    0c02    02    ;    Buf 3 control (FMI mode, D1, CH num : 03)
ww    0c03    03    ;    Buf 4 control (FMI mode, D1, CH num : 04)
ww    0c04    04    ;    Buf 5 control (FMI mode, D1, CH num : 05)
ww    0c05    05    ;    Buf 6 control (FMI mode, D1, CH num : 06)
ww    0c06    06    ;    Buf 7 control (FMI mode, D1, CH num : 07)
ww    0c07    07    ;    Buf 8 control (FMI mode, D1, CH num : 08)
ww    0c08    08    ;    Buf 9 control (FMI mode, D1, CH num : 09)
ww    0c09    09    ;    Buf 10 control (FMI mode, D1, CH num : 10)
```

```
ww   0c0a    0a    ;   Buf 11 control (FMI mode, D1, CH num : 11)
ww   0c0b    0b    ;   Buf 12 control (FMI mode, D1, CH num : 12)
ww   0c0c    0c    ;   Buf 13 control (FMI mode, D1, CH num : 13)
ww   0c0d    0d    ;   Buf 14 control (FMI mode, D1, CH num : 14)
ww   0c0e    0e    ;   Buf 15 control (FMI mode, D1, CH num : 15)
ww   0c0f    0f    ;   Buf 16 control (FMI mode, D1, CH num : 16)
; Buffer position selection and on/off control setting( [7]on/off, [6:3]Hori. position, [2:0] Verti position)
ww   0c10    80    ;   Buf 1 position setting
ww   0c11    90    ;   Buf 2 position setting
ww   0c12    a0    ;   Buf 3 position setting
ww   0c13    b0    ;   Buf 4 position setting
ww   0c14    84    ;   Buf 5 position setting
ww   0c15    94    ;   Buf 6 position setting
ww   0c16    a4    ;   Buf 7 position setting
ww   0c17    b4    ;   Buf 8 position setting
; Turn on second SDRAM for buffer 9 ~ buffer 16 writing
ww   0ccb    ff    ;   Turn on second SDRAM for buffer 9 ~ buffer 16
ww   0c18    80    ;   Buf 9 position setting
ww   0c19    90    ;   Buf 10 position setting
ww   0c1a    a0    ;   Buf 11 position setting
ww   0c1b    b0    ;   Buf 12 position setting
ww   0c1c    84    ;   Buf 13 position setting
ww   0c1d    94    ;   Buf 14 position setting
ww   0c1e    a4    ;   Buf 15 position setting
ww   0c1f    b4    ;   Buf 16 position setting
;Port 1 and Port 4 setting : 4D1 mode(Special), FMI, 108 MHz
ww   0c20    22    ;   Port1 control register : 4D1, FMI, x and y split, 108MHz data rate and off
ww   0c21    01    ;   Port1 Source selection register A : 1st source(1st buf), 2nd source(2nd buf)
ww   0c22    23    ;   Port1 Source selection register B : 3rd source(3rd buf), 4th source(4th buf)
ww   0c2a    22    ;   Port3 control register : 4D1, FMI, x and y split, 108MHz data rate and off
ww   0c2b    45    ;   Port3 Source selection register A : 1st source(5th buf), 2nd source(6th buf)
ww   0c2c    67    ;   Port3 Source selection register B : 3rd source(7th buf), 4th source(8th buf)

;Number of active channel of port 1 ~ port 4
ww   0c4c    00    ;   Each port has 4 active channels but this register is need to set '0'
;Port 5 and Port 7 setting : 4D1 mode(Special), FMI, 108MHz
ww   0c34    84    ;   Port5 control register : 4D1, FMI, 108MHz data rate and off
ww   0c35    00    ;   Source table index value is '0'
ww   0c36    08    ;   Source buffer number : 1st buffer number is '8'
ww   0c35    01    ;   Source table index value is '1'
ww   0c36    09    ;   Source buffer number : 2nd buffer number is '9'
ww   0c35    02    ;   Source table index value is '2'
ww   0c36    0a    ;   Source buffer number : 1st buffer number is '10'
ww   0c35    03    ;   Source table index value is '3'
ww   0c36    0b    ;   Source buffer number : 2nd buffer number is '11'
ww   0c37    03    ;   Source number register : Total number of sources is 4

ww   0c40    84    ;   Port7 control register : 4D1, FMI, 108MHz data rate and off
ww   0c41    00    ;   Source table index value is '0'
ww   0c42    0c    ;   Source buffer number : 1st buffer number is '12'
ww   0c41    01    ;   Source table index value is '0'
ww   0c42    0d    ;   Source buffer number : 2nd buffer number is '13'
ww   0c41    02    ;   Source table index value is '0'
ww   0c42    0e    ;   Source buffer number : 1st buffer number is '12'
ww   0c41    03    ;   Source table index value is '0'
ww   0c42    0f    ;   Source buffer number : 2nd buffer number is '13'
ww   0c43    03    ;   Source number register : Total number of sources is 4
```

```
;Output Pin setting
ww    0c4e      01    ;   Output pin 1 control : assign port1 lsb to pin 1 source 1, 8-bit, 54MHz
ww    0c4f      41    ;   Output pin 2 control : assign port3 lsb to pin 1 source 1, 8-bit, 54MHz
ww    0c50      81    ;   Output pin 3 control : assign port5 lsb to pin 1 source 1, 8-bit, 54MHz
ww    0c51      c1    ;   Output pin 3 control : assign port7 lsb to pin 1 source 1, 8-bit, 54MHz
ww    0cf5      40        Output pin set source 2 control : Pin2 src 2(Port 3 lsb), Pin1 src 2(Port1 lsb)
ww    0cf6      c8    ;   Output pin set source 2 control : Pin4 src 2(Port 7 lsb), Pin3 src 2(Port5 lsb)
; Software reset
ww    020e      0f    ;   Active software reset for record port
ww    020e      00    ;   Release software reset for record port
; Enable port
ww    0c20      85    ;   Port1 control register : 4D1, FMI, x and y split, 108MHz data rate and on
ww    0c2a      85    ;   Port3 control register : 4D1, FMI, x and y split, 108MHz data rate and on
ww    0c34      85    ;   Port6 control register : 4D1, FMI, 108MHz data rate and on
ww    0c40      85    ;   Port7 control register : 4D1, FMI, 108MHz data rate and on
```

6VGA

This setting records 6-D1 image with 6VGA size resolution and supports BT.1120 format (Refer to Figure 42).

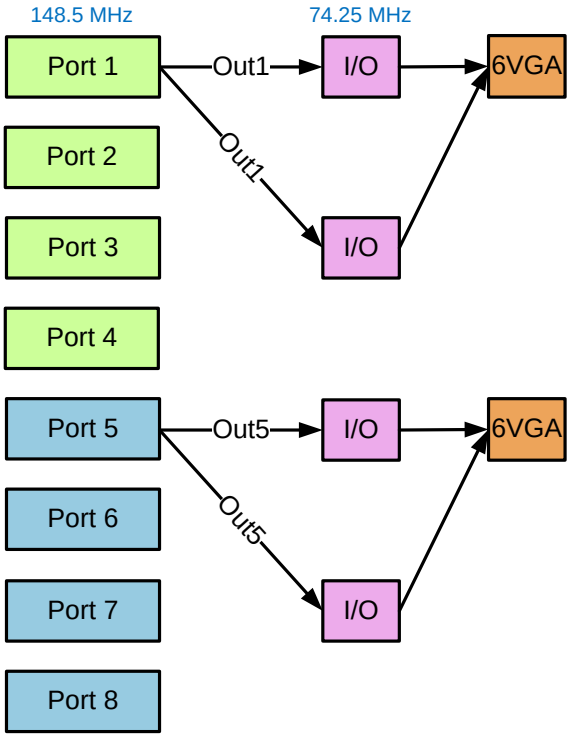


FIGURE 42. PROGRAMMING EXAMPLE 3 : 6VGA, FMI

TABLE 21 PROGRAMMING EXAMPLE CODE 3 : 6VGA, FMI

```
; Record clock setting
ww    0c68      02    ;   If internal video clock frequency is 148.5MHz, this value can be used but If this
                          clock frequency is not 148.5MHz, external clock need to connect TW2880 chip
                          and register value is ‘0x03).
; Buffer control setting ( [6]Recoding format, [5:4]Resolution, [3:0] Channel number)
ww    0c00      00    ;   Buf 1 control (FMI mode, D1, CH num : 01)
```

```
ww   0c01    01    ;   Buf 2 control (FMI mode, D1, CH num : 02)
ww   0c02    02    ;   Buf 3 control (FMI mode, D1, CH num : 03)
ww   0c03    03    ;   Buf 4 control (FMI mode, D1, CH num : 04)
ww   0c04    04    ;   Buf 5 control (FMI mode, D1, CH num : 05)
ww   0c05    05    ;   Buf 6 control (FMI mode, D1, CH num : 06)
ww   0c06    06    ;   Buf 7 control (FMI mode, D1, CH num : 07)
ww   0c07    07    ;   Buf 8 control (FMI mode, D1, CH num : 08)
ww   0c08    08    ;   Buf 9 control (FMI mode, D1, CH num : 09)
ww   0c09    09    ;   Buf 10 control (FMI mode, D1, CH num : 10)
ww   0c0a    0a    ;   Buf 11 control (FMI mode, D1, CH num : 11)
ww   0c0b    0b    ;   Buf 12 control (FMI mode, D1, CH num : 12)
ww   0c0c    0c    ;   Buf 13 control (FMI mode, D1, CH num : 13)
ww   0c0d    0d    ;   Buf 14 control (FMI mode, D1, CH num : 14)
ww   0c0e    0e    ;   Buf 15 control (FMI mode, D1, CH num : 15)
ww   0c0f    0f    ;   Buf 16 control (FMI mode, D1, CH num : 16)
; Buffer position selection and on/off control setting( [7]on/off, [6:3]Hori. position, [2:0] Verti position)
ww   0c10    80    ;   Buf 1 position setting
ww   0c11    90    ;   Buf 2 position setting
ww   0c12    a0    ;   Buf 3 position setting
ww   0c13    b0    ;   Buf 4 position setting
ww   0c14    84    ;   Buf 5 position setting
ww   0c15    94    ;   Buf 6 position setting
ww   0c16    a4    ;   Buf 7 position setting
ww   0c17    b4    ;   Buf 8 position setting
; Turn on second SDRAM for buffer 9 ~ buffer 16 writing
ww   0ccb    ff    ;   Turn on second SDRAM for buffer 9 ~ buffer 16
ww   0c18    80    ;   Buf 9 position setting
ww   0c19    90    ;   Buf 10 position setting
ww   0c1a    a0    ;   Buf 11 position setting
ww   0c1b    b0    ;   Buf 12 position setting
ww   0c1c    84    ;   Buf 13 position setting
ww   0c1d    94    ;   Buf 14 position setting
ww   0c1e    a4    ;   Buf 15 position setting
ww   0c1f    b4    ;   Buf 16 position setting
;Horizontal Offset for each channel
ww   0cd0    03    ;   shift 12 pixels
ww   0cd1    03    ;   shift 12 pixels
ww   0cd2    03    ;   shift 12 pixels
ww   0cd3    03    ;   shift 12 pixels
ww   0cd4    03    ;   shift 12 pixels
ww   0cd5    03    ;   shift 12 pixels
ww   0cd6    03    ;   shift 12 pixels
ww   0cd7    03    ;   shift 12 pixels
ww   0cd8    03    ;   shift 12 pixels
ww   0cd9    03    ;   shift 12 pixels
ww   0cda    03    ;   shift 12 pixels
ww   0cdb    03    ;   shift 12 pixels
ww   0cdc    03    ;   shift 12 pixels
ww   0cdd    03    ;   shift 12 pixels
ww   0cde    03    ;   shift 12 pixels
ww   0cdf    03    ;   shift 12 pixels
;Port 1 setting : 6VGA mode(Special), FMI, 74.25MHz
ww   0c20    c6    ;   Port1 control register : 6VGA, FMI, no split, 74.25 MHz data rate and off
ww   0c21    01    ;   Port1 Source selection register A : 1st source(1st buf), 2nd source setting does not
                      need
ww   0c23    78    ;   Custom HDE, 10'd1920 / 16
ww   0c24    87    ;   Custom VDE, 10'd540 / 4
```

```
;Number of active channel of port 1
ww   0c4c    00    ;   Port 1 has 6 active channels but this register is need to set ‘0’
;Port 5 setting : 6VGA mode(Special), FMI, 74.25MHz
ww   0c34    c6    ;   Port5 control register : 6VGA, FMI, 74.25 MHz data rate and off
ww   0c35    00    ;   Source table index value is ‘0’
ww   0c36    08    ;   Source buffer number : 1st buffer number is ‘8’
ww   0c37    00    ;   Source number register : Total number of sources is 6 but this register need to
                        set ‘0’
ww   0c38    78    ;   Custom HDE, 10’d1920 / 16
ww   0c39    87    ;   Custom VDE, 10’d540 / 4

;Output Pin setting
ww   0c4e    05    ;   Output pin 1 control : assign port1 lsb to pin 1 source 1, 16-bit, 74.25MHz
ww   0c4f    15    ;   Output pin 2 control : assign port1 msb to pin 1 source 1, 16-bit, 74.25MHz
ww   0c50    85    ;   Output pin 3 control : assign port5 lsb to pin 1 source 1, 16-bit, 74.25MHz
ww   0c51    95    ;   Output pin 3 control : assign port5 msb to pin 1 source 1, 16-bit, 74.25MHz
ww   0cf5    10        Output pin set source 2 control : Pin2 src 2(Port 1 msb), Pin1 src 2(Port1 lsb)
ww   0cf6    98    ;   Output pin set source 2 control : Pin4 src 2(Port 5 msb), Pin3 src 2(Port5 lsb)
; Software reset
ww   020e    0f    ;   Active software reset for record port
ww   020e    00    ;   Release software reset for record port
; Enable port
ww   0c20    c7    ;   Port1 control register : 6VGA, FMI, 74.25 MHz data rate and on
ww   0c34    c7    ;   Port6 control register : 6VGA, FMI, 74.25 MHz data rate and on
```

8-D1 and Two 4D1

This setting records 8-D1 live input by using only one port with real time frame rate (Refer to Figure 43).

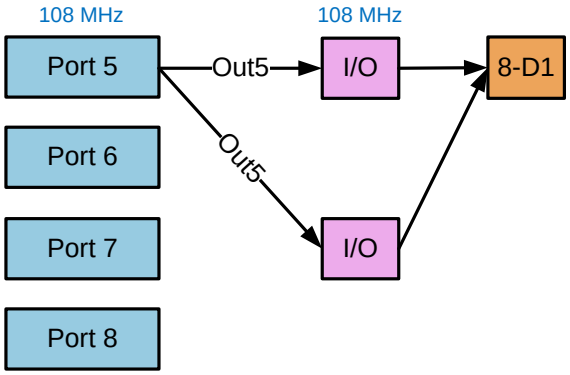


FIGURE 43. PROGRAMMING EXAMPLE 4 : 8-D1, FMI

TABLE 22 PROGRAMMING EXAMPLE CODE 4 : 8-D1, FMI

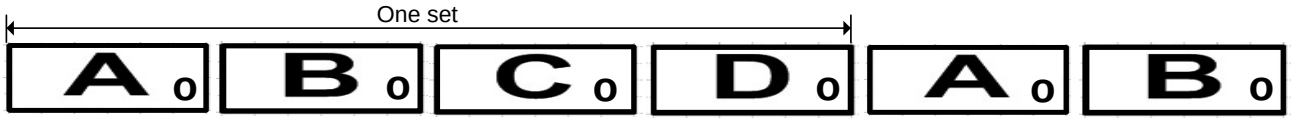
```
; Buffer control setting ( [6]Recoding format, [5:4]Resolution, [3:0] Channel number)
ww   0c00    40    ;   Buf 1 control (FLI mode, D1, CH num : 01)
ww   0c01    41    ;   Buf 2 control (FLI mode, D1, CH num : 02)
ww   0c02    42    ;   Buf 3 control (FLI mode, D1, CH num : 03)
ww   0c03    43    ;   Buf 4 control (FLI mode, D1, CH num : 04)
ww   0c04    44    ;   Buf 5 control (FLI mode, D1, CH num : 05)
ww   0c05    45    ;   Buf 6 control (FLI mode, D1, CH num : 06)
ww   0c06    46    ;   Buf 7 control (FLI mode, D1, CH num : 07)
ww   0c07    47    ;   Buf 8 control (FLI mode, D1, CH num : 08)
```

```
; Buffer position selection and on/off control setting( [7]on/off, [6:3]Hori. position, [2:0] Verti position)
ww   0c10    80   ;  Buf 1 position setting
ww   0c11    90   ;  Buf 2 position setting
ww   0c12    a0   ;  Buf 3 position setting
ww   0c13    b0   ;  Buf 4 position setting
ww   0c14    82   ;  Buf 5 position setting
ww   0c15    92   ;  Buf 6 position setting
ww   0c16    a2   ;  Buf 7 position setting
ww   0c17    b2   ;  Buf 8 position setting
;Change frsc reference source if port 0 is not used
ww   0c56    04   ;  Change frsc source from port 1 to port 5
;Port 5 setting : 8-D1, FMI, 108MHz
ww   0c34    84   ;  Port6 control register : D1, FMI, 108MHz data rate and off
ww   0c35    00   ;  Source table index value is '0'
ww   0c36    00   ;  Source buffer number : 1st buffer number is '0'
ww   0c35    01   ;  Source table index value is '1'
ww   0c36    01   ;  Source buffer number : 2nd buffer number is '1'
ww   0c35    02   ;  Source table index value is '2'
ww   0c36    02   ;  Source buffer number : 3rd buffer number is '2'
ww   0c35    03   ;  Source table index value is '3'
ww   0c36    03   ;  Source buffer number : 4th buffer number is '3'
ww   0c35    04   ;  Source table index value is '4'
ww   0c36    04   ;  Source buffer number : 5th buffer number is '4'
ww   0c35    05   ;  Source table index value is '5'
ww   0c36    05   ;  Source buffer number : 6th buffer number is '5'
ww   0c35    06   ;  Source table index value is '6'
ww   0c36    06   ;  Source buffer number : 7th buffer number is '6'
ww   0c35    07   ;  Source table index value is '7'
ww   0c36    07   ;  Source buffer number : 8th buffer number is '7'
ww   0c37    07   ;  Source number register : Total number of sources is 2
Single element fetching mode setting
ww   0cf0    02   ;  Port 5 can support 8-D1
;Output Pin setting
ww   0c4e    82   ;  Output pin 1 control : assign port5 lsb to pin 1 source 1, 8-bit, 108MHz
ww   0c4f    92   ;  Output pin 2 control : assign port5 msb to pin 2 source 1, 8-bit, 108MHz
ww   0cf5    98       Output pin set source 2 control : Pin2 src 2(Port 5 msb), Pin1 src 2(Port5 lsb)
; Software reset
ww   020e    0f   ;  Active software reset for record port
ww   020e    00   ;  Release software reset for record port
; Enable port
ww   0c34    85   ;  Port6 control register : D1, FMI, 108MHz data rate and on
```

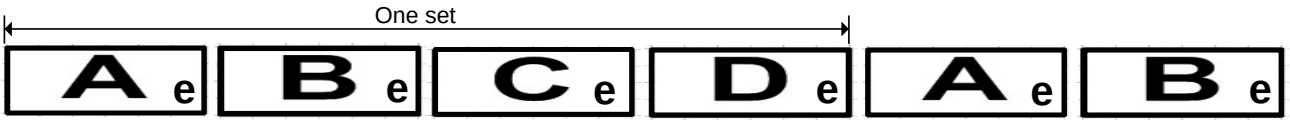
Field Switching Mode

Field switching mode has only odd field or only even field output (Refer to Figure 44 A and B).

Field switching mode turn on or off each buffer by setting register 0xCF1 and 0xCF2 and even and odd field are selected by setting register 0xCF3 and 0xCF4.



(A) FIELD SELECT REGISTER(0XCF3 AND 0XCF4) IS '0'



(B) FIELD SELECT REGISTER(0XCF3 AND 0XCF4) IS '1'

FIGURE 44. IMAGE FLOW OF FIELD SWITCHING MODE

CASE 1: 4-D1 AND FIELD SWITCHING MODE, ONLY EVEN FIELD OUT (REFER TO FIGURE 45)

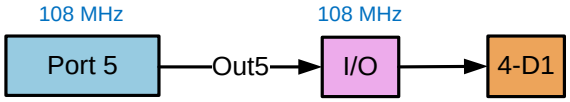


FIGURE 45. PROGRAMMING EXAMPLE 6 : 8-D1, FMI

TABLE 23 PROGRAMMING EXAMPLE CODE 6 : 8-D1, FMI

```
; Buffer control setting ( [6]Recoding format, [5:4]Resolution, [3:0] Channel number)
ww   0c00    40   ;   Buf 1 control (FLI mode, D1, CH num : 01)
ww   0c01    41   ;   Buf 2 control (FLI mode, D1, CH num : 02)
ww   0c02    42   ;   Buf 3 control (FLI mode, D1, CH num : 03)
ww   0c03    43   ;   Buf 4 control (FLI mode, D1, CH num : 04)
; Buffer position selection and on/off control setting( [7]on/off, [6:3]Hori. position, [2:0] Verti position)
ww   0c10    80   ;   Buf 1 position setting
ww   0c11    90   ;   Buf 2 position setting
ww   0c12    a0   ;   Buf 3 position setting
ww   0c13    b0   ;   Buf 4 position setting
;Change frsc reference source if port 0 is not used
ww   0c56    04   ;   Change frsc source from port 1 to port 5
;Port 5 setting : 4-D1, FLI, 108MHz
ww   0c34    24   ;   Port5control register : D1, FLI, 108MHz data rate and off
ww   0c35    00   ;   Source table index value is '0'
ww   0c36    00   ;   Source buffer number : 1st buffer number is '0'
ww   0c35    01   ;   Source table index value is '1'
ww   0c36    01   ;   Source buffer number : 2nd buffer number is '1'
ww   0c35    02   ;   Source table index value is '2'
ww   0c36    02   ;   Source buffer number : 3rd buffer number is '2'
```

```
ww   0c35    03    ;   Source table index value is '3'
ww   0c36    03    ;   Source buffer number : 4th buffer number is '3'
ww   0c37    03    ;   Source number register : Total number of sources is 4
Field switching mode
ww   0cf1    0f    ;   buffer 0 ~ buffer 3 are set to field switching mode
ww   0cf3    0f    ;   buffer 0 ~ buffer 3 save only even field
;Output Pin setting
ww   0c4e    82    ;   Output pin 1 control : assign port5 lsb to pin 1 source 1, 8-bit, 108MHz
ww   0cf5    x8        Output pin set source 2 control : Pin1 src 2(Port5 lsb)
; Software reset
ww   020e    0f    ;   Active software reset for record port
ww   020e    00    ;   Release software reset for record port
; Enable port
ww   0c34    25    ;   Port65control register : D1, FL, 108MHz data rate and on
```

Priority & Frame Rate Control

Multi port supports up to 128 buffer indexes and we can assign different frame rate to each channel.

Case 1: buffer 1(15 frames / sec), buffer 2(5 frames / sec), buffer 2(5 frames /sec), buffer 3(5 frames /sec), Refer to Figure 46.

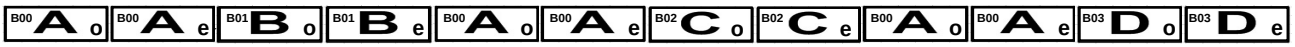


FIGURE 46. PRIORITY AND FRAME RATE CONTROL EXAMPLE

TABLE 24 PRIORITY AND FRAME RATE CONTROL EXAMPLE CODE

```
ww   0c35    00    ;   Source table index value is '0'
ww   0c36    00    ;   Source buffer number : 1st buffer number is '0'
ww   0c35    01    ;   Source table index value is '1'
ww   0c36    01    ;   Source buffer number : 2nd buffer number is '1'
ww   0c35    02    ;   Source table index value is '2'
ww   0c36    00    ;   Source buffer number : 3rd buffer number is '0
ww   0c35    03    ;   Source table index value is '3'
ww   0c36    02    ;   Source buffer number : 4th buffer number is '2
ww   0c35    04    ;   Source table index value is '4
ww   0c36    00    ;   Source buffer number : 3rd buffer number is '0
ww   0c35    05    ;   Source table index value is '5
ww   0c36    03    ;   Source buffer number : 4th buffer number is '3'
ww   0c37    05    ;   Source number register : Total number of sources is 6
```

Using SPOT Buffer for Recording

Record port can use SPOT buffers by setting register 0xCCE and 0xCCF. SPOT buffer that is used by record buffer cannot be used by SPOT port. Record port can select SPOT buffer by setting '1' to the port source control register (ex. 0xc36[4]). For example, 1st SPOT buffer is used by record port 5

Case 1: Port 5 uses the following write buffer

2nd record buffer, 5th record buffer, 2nd SPOT buffer, 3rd SPOT buffer

TABLE 25 PROGRAMMING EXAMPLE CODE 1 : USING SPOT BUFFER FOR RECORDING

```
; Change SPOT buffer 2 and 3 to record buffer
ww    0cce      06    ;   Use 2nd and 3rd SPOT buffer as record buffer
; Buffer position selection and on/off control setting( [7]on/off, [6:3]Hori. position, [2:0] Verti position)
ww    0c10      80    ;   Buf 1 position setting
ww    0c11      90    ;   Buf 2 position setting
ww    0c12      a0    ;   Buf 3 position setting
ww    0c13      b0    ;   Buf 4 position setting
; Port 5 Table Setting
ww    0c35      00    ;   1st Table index for port 5
ww    0c36      01    ;   1st Table data(buffer number) for port 5(2nd record buffer)
ww    0c35      01    ;   2nd Table index for port 5
ww    0c36      04    ;   2nd Table data(buffer number) for port 5(5th record buffer)
ww    0c35      02    ;   3rd Table index for port 5
ww    0c36      11    ;   3rd Table data(buffer number) for port 5(2nd SPOT buffer)
ww    0c35      02    ;   4th Table index for port 5
ww    0c36      12    ;   4th Table data(buffer number) for port 5(3rd SPOT buffer)
```

Network Port

Network port is same to the multi port except SPOT connection. SPOT CVBS output can be connected to the network port by setting the following registers

0xF6A[5:4]: Select SPOT CVBSs out to network port (0: SPOT1, 1: SPOT2, 2: SPOT3, 3: SPOT4)

0x21E[1:0]: Select Network port data source (0: Original network data, 1: SPOT vdout, 2: DM vdout, 3: REC2Netork)

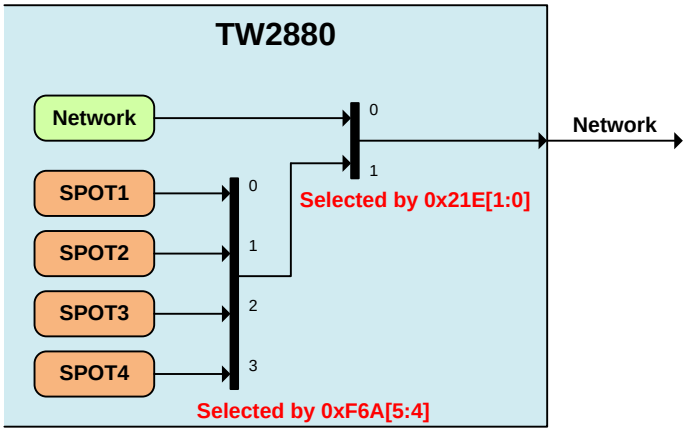


FIGURE 47. SPOT CONNECTION TO THE NETWORK PORT

PB Loopback Test

For the purpose of test, record port output can be connect to the PB input port by setting the following registers

0x224[6:4]: Playback port 1 data source selection (0: PB1 port, 1: PB2 port, 2: PB3 Port, 3: PB4 port, 4: rec1 port, 5: rec2 port, 6: rec3 port, 7: rec4 port)

0x224[3:0]: Playback port 1 clock source selection (0: pb1_clkp, 1: pb2_clkp, 2: pb3_clkp, 3: pb4_clkp, 4: pb1_clkn, 5: pb2_clkn, 6: pb3_clkn, 7: pb4_clkn, 8: rec1_clkp, 9: rec2_clkp, 10: rec3_clkp, 11: rec4_clkp, 12: rec1_clkn, 13: rec2_clkn, 14: rec3_clkn, 15: rec4_clkn)

0x225 ~ 0x227: PB 2 ~ PB 4 input port data and clock source selection (same to the above two registers)

If you want to test with loopback connection, you need to set 0x224[6:4] by 4 ~ 7. (Refer to Figure 48).

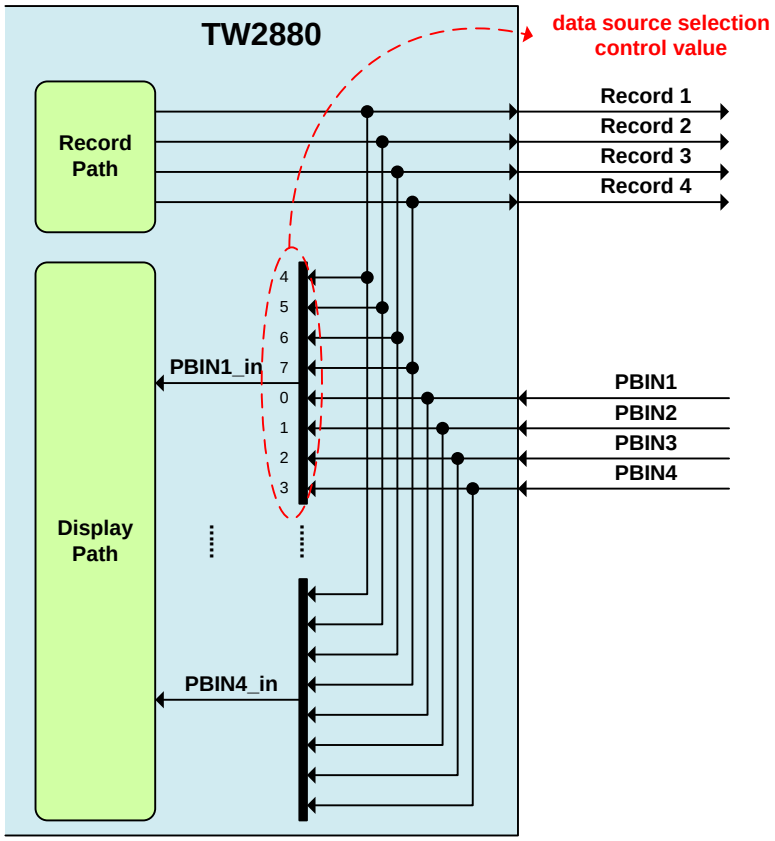


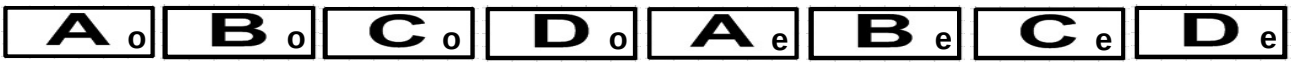
FIGURE 48. PB LOOPBACK CONNECTION FOR TEST

Q & A

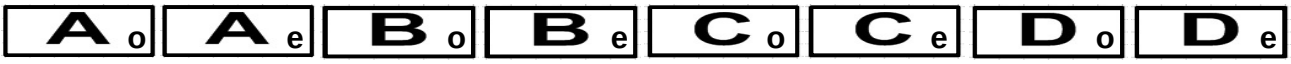
Q001: WHAT IS DIFFERENCE BETWEEN FLI MODE AND FMI MODE IN BUFFER CONTROL?

A001 : There are two differences. One is field image output sequence and the other is memory utilization. For example, when 4-D1 image send

In the FLI mode, image flow is like the following figure.



In the FMI mode, image flow is like the following figure.



In the FLI mode, only 4 fields are saved into the SDRAM.

In the FMI mode, 4 frames(8 fields) are saved into the SRAM.

Q002 : DOES TW2880 SUPPORT PROGRESSIVE FRAME INTERLEAVED RECORD OUTPUT

A002 : Record output capability is describe in the chapter 13.8 of TW2880 spec. TW2880 can support progressive frame interleaved record output for only D1 size image.

Section 5: How to Setup a TW2880C-Based Display

Introduction

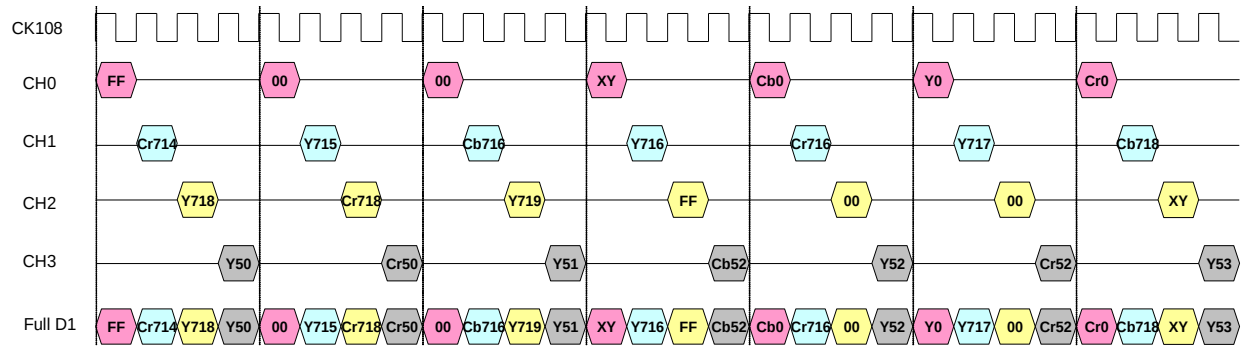
TW2880C is a multi channel multiplex chip equipped with a glue less VGA display interface which can drive LCD TV and PC display directly. Because so many functions are packed inside the TW2880C, it is not easy to understand, calculate and utilize all the features and display capabilities that TW2880C provides. This section serves as a guide to use the display portion of the chip.

This guide is largely divided into three parts: windows setup, main display setup and dual monitor setup. Because the similarity between the main and dual display, many of parameter registers are described only once in detail. The other settings can be deduced from the same set of rules.

Because TW2880C's display sub-system is a very powerful and complicated circuit, this note certainly cannot cover everything that the user might want to know. If you still have questions about the setting, please contact Techwell FAEs in your region.

Input Arrangement

LIVE INPUT



TW2880C's live input only support embedded video sequence coded in BT.656 format. On top of this, TW2880C also support multi-channel video sequence coded in byte interleaved BT.656 format. The above diagram illustrated the format. The data rate and input clock will need to run higher according to the channel number. For instance, 4-channel byte interleaved format will need to run at 108 MHz. The input rate select register is 0x20A[1:0].

There are eight live input so if couple with TW2864 running at 54 MHz data rate, all input ports will be needed. However if running at 108 MHz, only 4 ports are needed. The rest of the port (port 4, 5, 6, 7) can be turn into other usage. More on this later. The clock sent to TW2864 is selectable through 0x21A[0]. TW2880C has 4 live clock input so each clock input is shared between two data ports.

PLAYBACK INPUT

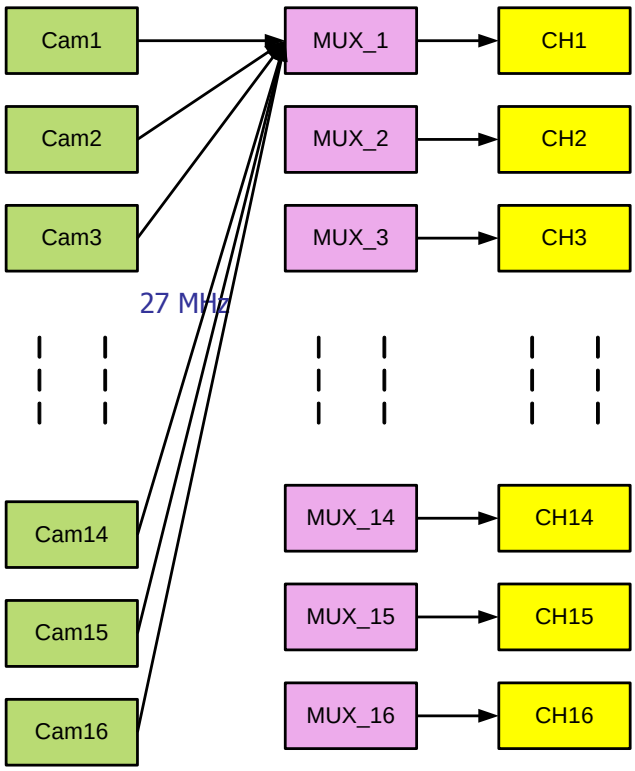
TW2880C has four 8-bit playback ports. It can be treated as four 8 bit interface input or two 16 bit interface input. The playback port only supports embedded sync video sequence coded in BT.656 or BT.1120 format. If multi-channel input is expected, it can only take sequence coded in frame interleaved or field interleaved format. It does not support byte-interleaved format. The setting of the playback port is quite complex and is covered in a separate section.

INPUT AND CHANNEL MAPPING

The inputs coming from 16 live cameras first travel through a multiplexer before going into the down scaler of each channel. This is designed for the ultimate user freedom.

Please refer to the diagram on the right. The video plug on the customer's box does not determine the channel number for this camera input. It is determined by the live / record channel select register 0x3D2 – 0x3D9. So maintaining the same connection, user can rotate the camera to different channels. Also, note that it is possible to assign same camera input to two different input channels. Recording channel input control is based on these registers but user need to set register 0x3DA bit 7 to 1.

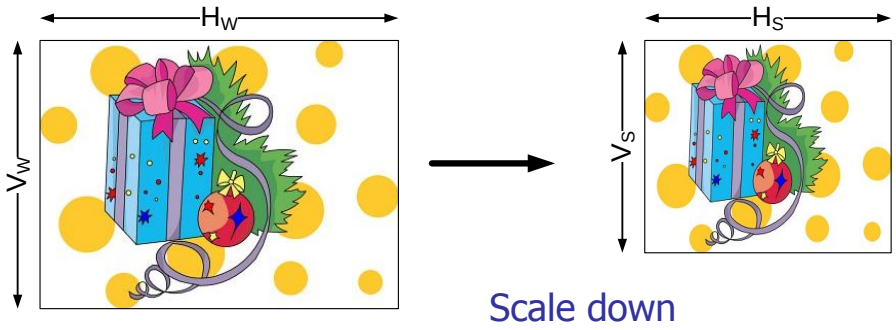
Depends on the video decoder used, several register bits need to be set properly. These bits are mainly involved in the channel ID. For a 108MHz input, the width of the channel ID needs to set as two bits. There is a bit which determines the receiving channel format needs to be set.




```
(1) Live/Record Channel Select Register 1     0x3D2[3:0], 0x0 mean ch1, 0xf means ch16
(2) Live/Record Channel Select Register 1     0x3D2[7:4], 0x0 mean ch1, 0xf means ch16
.....
(3) Live/Record Channel Select Register 8     0x3D9[3:0], 0x0 mean ch1, 0xf means ch16
(4) Live/Record Channel Select Register 8     0x3D9[7:4], 0x0 mean ch1, 0xf means ch16
(5) Live/Record Set Select Register      :    0x3DA[7], 0 = live and 1 = record
(6) Format select                             0x3C6[6], 0 = NTSC, 1= PAL
```

DOWN SCALER

For each live channel, two sixteen-bit registers control the final video stream size. One is for horizontal ratio and the other one is for vertical ratio. Take window 1 for example, 0x301 and 0x300 is the horizontal down scale ratio register and 0x321 and 0x320 are the vertical down scale register. The formula is:



Ratio = 65535 * target size / source size.

From this formula, we can see if the When down scaler is set to 65535 (0xFFFF), the down scaler is disabled.

If the original stream is D1 stream and the result size is

Therefore, the ratio is: 65535 x 600 / 720

TEST PATTERN

TW2880 has built-in test pattern generator in the input section. This way, even without connecting to a live camera, user can do some system setup testing and debugging. The pattern is a set of different color bars with a big square traveling dot in the lower half of the window. The control and definitions are as follows:

```
(1) Test pattern enable:          0x3C5[4], 1= enable
(2) Still pattern enable          0x3C5[6], 1= enable, no moving image
(3) Format select                 0x3C5[5], 0 = NTSC, 1= PAL
(4) Channel ID select             0x3C5[2], 1= 2 bits, (for four channels)
(5) No Channel ID select          0x3C5[1], 1= no CHID information
(6) Channel ID location           0x3C5[0], 1= protection bits, 0= In HB
```

Main Display

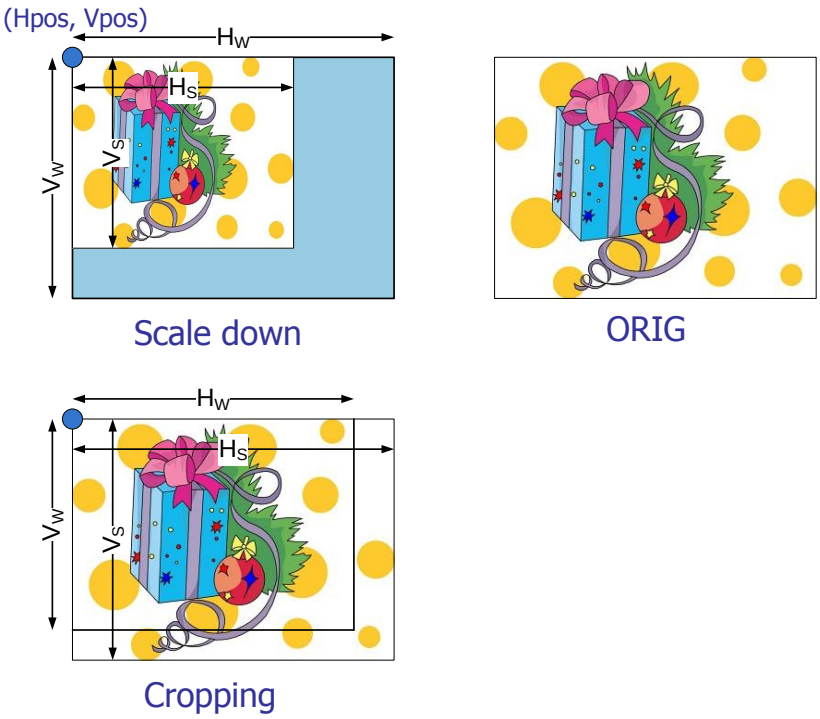
Introduction

After camera inputs and channels are linked together, the next thing user should do is to determine window parameters in a TW2880 display. TW2880 can display all 16 live channels and all 16 playback channels on a single display in a **non-overlapping** fashion. The window can appear anywhere in the display. Because TW2880 supports display with many sizes, a 33rd window is also created to assist tile arrangement for odd size display. This window can be used as logo or advertisement usage.

Live and PB Window Register Arrangement

Four parameters determine the size and position of a window: **Horizontal position**, **Vertical position**, **Horizontal size** and **Vertical size**. These registers must have an even value. If the user set the size register equal to the original image size, this channel will display in its native form. If the user set the size register smaller than the original size, a cropped image will be displayed. If the user set the size register larger than the original size, a native image together with some background information will be displayed. The mode used most often is down scale the video stream in the input section and display the channel in a same size window.

Auto mode playback windows are setup using the same set of registers. In other word, they are shadowed by 0x6B6 bit 0. Default is set as live window. Normal mode playback windows are controlled by channel 16 – 20 registers. There are other visual effects available like horizontal flipping vertical flipping and freeze. The following diagram shows the resulting effect. A list of registers is also provided.




```
(1) H position register:    0x665[0], 0x664[7:0], unit is 4 pixel, this for ch1
(2) H size register:        0x68D[0], 0x68C[7:0], unit is 4 pixel, this is for ch1
(3) V position register:    0x615[3:0], 0x614[7:0], unit is line, this is for ch1
(4) V size register:        0x63D[2:0], 0x63C[7:0], unit is line, this is for ch1
```

Please consult the datasheet for the complete register listing.

Window Write Process Protection

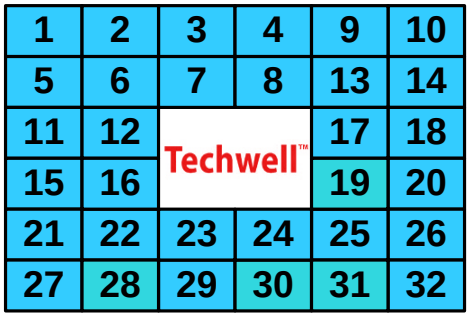
The write process of the individual channel can cause severe damage to the content stored in the DRAM, for example, the OSG bitmap for display or OSD font data. The reason for this to happen is if a video input is really weak or coupled with other signals such that the embedded sync signal does not receive by TW2880 input section causing the write process to malfunction.

To prevent this from happening we have setup a protection mechanism. Based on the input video size, user setup a address register no bigger than its image size, this way, when write buffer attempts a write with a range large than its size, we know something wrong has happen and can stop it.

```
(1) PB side enable register:    0x6B6[7], 1 = enable
(2) Live side enable register:  0x6B6[6], 1 = enable
(3) Protection register:        0x6BF[5], 0x6BE[7:0], 0x6BD[7:0], linear address
```

33rd Window

In addition to the regular 32 video windows, TW2880 also provides a 33rd static window. This window can be very useful in adjusting the final presentation of the main display. For example, it can be used in adjusting aspect ratio, showing company logo or present some important real time messages to the viewer. Unlike other TW2880 windows, the content is managed by host.



To use this feature, several registers need to be programmed:

```
(1) H position register:    0x476[5:4], 0x473[7:0], unit is 4 pixel
(2) H size register:        0x477[4], 0x475[7:0], unit is pixel
(3) V position register:    0x476[3:0], 0x472[7:0], unit is line
(4) V size register:        0x477[2:0], 0x474[7:0], unit is line
(5) Enable register:        0x47E[0], 1=enable
(6) Boundary Ena register:  0x47E[1], 1=enable
```

Test Pattern

In addition to the test pattern generator in input section, TW2880 has built-in test pattern generator in the buffer update section as well. This pattern generator, along with the other three pattern generators, forms a complete self-test and debugging system tool. This pattern is used to test the interactions between write buffers and the SDRAM controller. The patterns are almost the same with the first one. The control and definitions are as follows:

```
(7) Live channels test pattern enable:   0x6B4[0], 1= enable
(8) PB channels test pattern enable:     0x6B4[2], 1= enable
(9) Format select                        0x6B4[1], 0 = NTSC, 1= PAL
```

CRTC Parameters

INTRODUCTION

CRT controller is a general term for display data fetching unit. In TW2880C, we also adopted the same terminologies in the display industry. In raster scan technology, the data is fetched and displayed from left to right, top to bottom. To create these actions, ten important parameters need to be programmed in order to get a correct and stable video output in TW2880C. Five parameters are in the horizontal group and five are in vertical group. The horizontal parameters are:

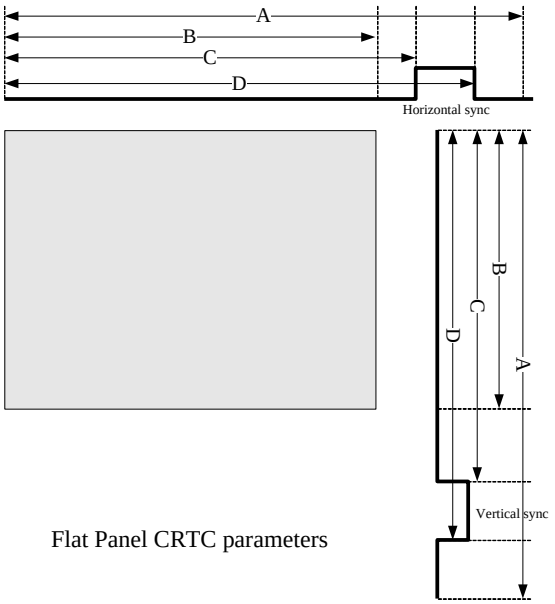
```
(1) H. total register:          0x501[3:0], 0x500[7:0], unit is pixel, the value put in is real
    value -1
(2) H. display end register:    0x505[3:0], 0x504[7:0], unit is pixel, the value put in is real
    value -1
(3) H. sync start register:     0x508[7:0], unit is double pixel
(4) H. sync width register:     0x50A[7:0], unit is pixel
(5) H. sync polarity register:  0x480[1]
```

Similarly, we have vertical parameters:

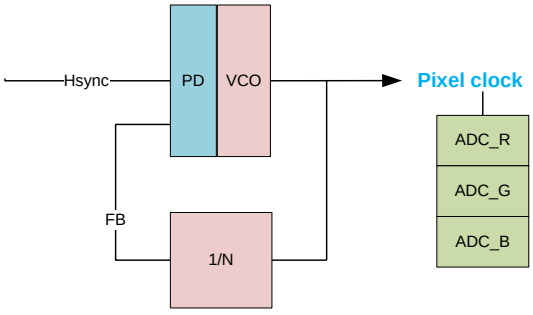
```
(1) V. total register:          0x503[3:0], 0x502[7:0], unit is line, the value put in is value
    -1
(2) V. display end register:    0x507[3:0], 0x506[7:0], unit is line, the value put in is value
    -1
(3) V. sync start register:     0x509[7:0], unit is line
(4) V. sync width register:     0x50A[7:0], unit is line
(5) V. sync polarity register:  0x480[0]
```

There is no particular sequence to program the TW2880C CRT controller, however, to prevent garbage data shown on the screen, a good practice is disable the output (VGA, 0xxxx or HDMI, oxxxx) before you completely program the ten parameters.

The Image fetched from the DRAM is defaulted to start at location (0, 0) unless you program the offset register 0x4A1[1:0], 0x4A0[7:0] for horizontal and 0x4A3[3:0], ox4A2[7:0] for vertical. The unit for horizontal is 4 pixels and the unit for vertical is line.



Flat Panel CRTC parameters



HORIZOTAL SYNCHRONIZATION AND REFRESH RATE ADJUSTMENT

Horizontal sync frequency plays a very important part in monitor compatibility. This is because modern analog monitor interface (VGA, component) is using horizontal sync signal to generate pixel clock and use this clock to drive ADC and sample the incoming data. Thus depends on the monitor IC used, each monitor will have a range for the internal line locked PLL to function correctly. The calculation of the line frequency will be based on dot clock and the horizontal total.

	Resolution	FPS	PCLK	HTT	VTT	HS	HSW	VS	VSW	HPOL	VPOL	N	DIV	PCLK2	diff	diff(%)
4:3	640x480	50Hz	19.75	800	495	16	64	1	4	N	P	23	28	22.18	2.43	12.30%
	640x480	60Hz	23.88	800	497	16	64	1	4	N	P	25	28	24.11	0.23	0.97%
	800x600	50Hz	31.13	1008	618	24	80	1	4	N	P	32	28	30.86	-0.27	-0.86%
	800x600	60Hz	38.13	1024	622	32	80	1	4	N	P	40	28	38.57	0.45	1.17%
	1024x768	50Hz	51.75	1312	791	40	104	1	4	N	P	23	12	51.75	0.00	0.00%
	1024x768	60Hz	64.13	1344	795	56	104	1	4	N	P	38	16	64.13	0.00	0.00%
	1280x960	50Hz	83.00	1680	988	64	136	1	4	N	P	37	12	83.25	0.25	0.30%
	1280x960	60Hz	102.00	1712	994	80	136	1	4	N	P	38	10	102.60	0.60	0.59%
	1400x1050	50Hz	99.75	1848	1081	80	144	1	4	N	P	37	10	99.90	0.15	0.15%
	1400x1050	60Hz	122.50	1880	1087	88	152	1	4	N	P	32	7	123.43	0.93	0.76%
	1600x1200	50Hz	132.38	2144	1235	104	168	1	4	N	P	39	8	131.63	-0.75	-0.57%
	1600x1200	60Hz	160.88	2160	1242	104	176	1	4	N	P	24	4	162.00	1.13	0.70%
	1600x1200r	60Hz	130.38	1760	1235	48	32	2	4	P	N	29	6	130.50	0.13	0.10%
16:9	848x480	50Hz	26.00	1056	495	88	104	1	5	N	P	27	28	26.04	0.04	0.14%
	848x480	60Hz	31.50	1056	497	88	104	1	5	N	P	33	28	31.82	0.32	1.02%
	1064x600	50Hz	41.25	1336	618	32	104	1	5	N	P	43	28	41.46	0.21	0.52%
	1064x600	60Hz	51.00	1368	622	40	112	1	5	N	P	30	16	50.63	-0.38	-0.74%
	1280x720	50Hz	60.38	1632	741	48	128	1	5	N	P	27	12	60.75	0.38	0.62%
	1280x720	60Hz	74.38	1664	746	56	136	1	5	N	P	33	12	74.25	-0.13	-0.17%
	1360x768	50Hz	69.50	1760	791	56	144	1	5	N	P	31	12	69.75	0.25	0.36%
	1360x768	60Hz	84.63	1776	795	64	144	1	5	N	P	25	8	84.38	-0.25	-0.30%
	1704x960	50Hz	110.25	2232	988	88	176	1	5	N	P	41	10	110.70	0.45	0.41%
	1704x960	60Hz	134.88	2264	994	96	184	1	5	N	P	30	6	135.00	0.13	0.09%
	1864x1050	50Hz	133.50	2472	1081	104	200	1	5	N	P	30	6	135.00	1.50	1.12%
	1864x1050	60Hz	163.25	2504	1087	120	200	1	5	N	P	24	4	162.00	-1.25	-0.77%
	1864x1050r	60Hz	131.13	2024	1080	48	32	2	5	P	N	34	7	131.14	0.02	0.01%
	1920x1080	50Hz	141.38	2544	1112	112	200	1	5	N	P	42	8	141.75	0.38	0.27%
	1920x1080	60Hz	172.73	2576	1118	120	208	1	5	N	P	38	6	171.00	-1.72	-1.00%
	1920x1080r	60Hz	138.63	2080	1111	48	32	2	5	P	N	36	7	138.86	0.23	0.17%
16:10	768x480	50Hz	23.63	960	495	16	80	1	6	N	P	24	28	23.14	-0.48	-2.04%
	768x480	60Hz	28.63	960	497	16	80	1	6	N	P	30	28	28.93	0.30	1.06%
	960x600	50Hz	37.00	1200	618	24	96	1	6	N	P	38	28	36.64	-0.36	-0.97%
	960x600	60Hz	45.88	1232	622	40	96	1	6	N	P	27	16	45.56	-0.31	-0.68%
	1152x720	50Hz	54.50	1472	741	40	120	1	6	N	P	24	12	54.00	-0.50	-0.92%
	1152x720	60Hz	67.25	1504	746	56	120	1	6	N	P	25	10	67.50	0.25	0.37%
	1224x768	50Hz	62.25	1576	791	48	128	1	6	N	P	23	10	62.10	-0.15	-0.24%
	1224x768	60Hz	76.63	1608	795	64	128	1	6	N	P	34	12	76.50	-0.13	-0.16%
	1440x900	60Hz	106.50	1904	934	80	152	3	6	P	P	32	8	108.00	1.50	1.41%
	1536x960	50Hz	99.50	2016	988	80	160	1	6	N	P	37	10	99.90	0.40	0.40%
	1536x960	60Hz	122.00	2048	994	96	160	1	6	N	P	27	6	121.50	-0.50	-0.41%
	1680x1050	50Hz	120.13	2224	1081	96	176	1	6	N	P	31	7	119.57	-0.55	-0.46%
	1680x1050	60Hz	147.00	2256	1087	104	184	1	6	N	P	38	7	146.57	-0.43	-0.29%
	1680x1050r	60Hz	119.13	1840	1080	48	32	2	6	P	N	44	10	118.80	-0.33	-0.27%
	1728x1080	50Hz	127.13	2288	1112	96	184	1	6	N	P	33	7	127.29	0.16	0.13%
	1728x1080	60Hz	155.50	2320	1118	112	184	1	6	N	P	23	4	155.25	-0.25	-0.16%
	1728x1080r	60Hz	125.75	1888	1111	48	32	2	6	P	N	28	6	126.00	0.25	0.20%

	1920x1200	50Hz	158.00	2560	1235	112	208	1	6	N	P	41	7	158.14	0.14	0.09%
	1920x1200	60Hz	193.13	2592	1242	128	208	1	6	N	P	43	6	193.50	0.38	0.19%
	1920x1200r	60Hz	154.13	2080	1235	48	32	2	6	P	N	40	7	154.29	0.16	0.10%
5:4	1280x1024	50Hz	89.38	1696	1054	72	136	1	7	N	P	33	10	89.10	-0.28	-0.31%
	1280x1024	60Hz	108.88	1712	1080	80	136	1	7	N	P	24	6	108.00	-0.88	-0.80%
15:9	1280x768	50Hz	65.13	1648	791	56	128	1	7	N	P	29	12	65.25	0.13	0.19%
	1280x768	60Hz	80.13	1680	795	64	136	1	7	N	P	24	8	81.00	0.88	1.09%
TV	1920x1080i	25Hz	74.25	2640	562	528	44	2	5	P	P	44	16	74.25	0.00	0.00%
	1920x1080i	30Hz	74.25	2200	562	88	44	2	5	P	P	44	16	74.25	0.00	0.00%
	1920x1080p	50Hz	148.50	2640	1125	528	44	2	5	P	P	44	8	148.50	0.00	0.00%
	1920x1080p	60Hz	148.50	2200	1125	88	44	2	5	P	P	44	8	148.50	0.00	0.00%
	1280x720p	50Hz	74.25	1980	750	440	40	5	5	P	P	44	16	74.25	0.00	0.00%
	1280x720p	60Hz	74.25	1650	750	110	40	5	5	P	P	44	16	74.25	0.00	0.00%
	720x480p	60Hz	27.00	858	525	16	62	9	6	N	N			27.00	0.00	0.00%
	720x480i	30Hz	13.50	1716	262	38	124	4	3	N	N			13.50	0.00	0.00%
	720x576p	50Hz	27.00	864	625	12	64	5	5	N	N			27.00	0.00	0.00%
	720x576i	25Hz	13.50	1728	312	24	126	2	3	N	N			13.50	0.00	0.00%

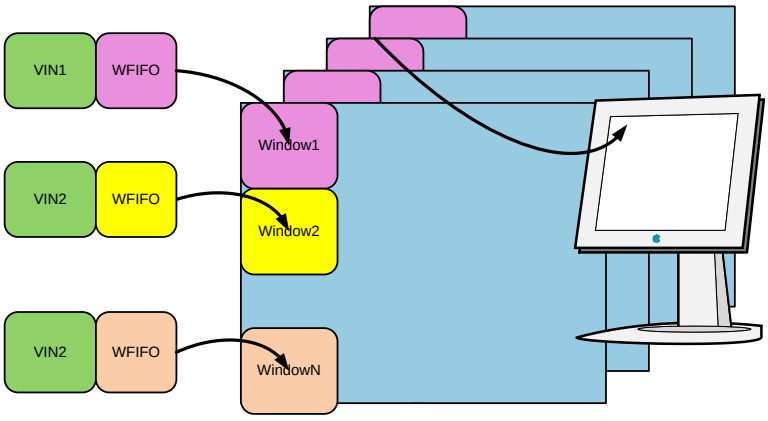
The above table is listing of all the popular VESA modes that TW2880C supports. Notice PCLK is the VESA standard frequency and PCLK2 is the frequency TW2880C can generate. Although not perfect but every mode have its closest approximation. The result will be determined by field test.

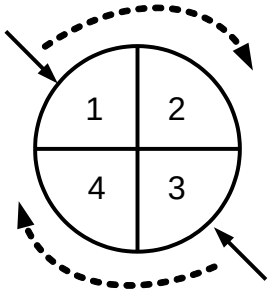
Another thing affecting the pixel clock and the refresh rate is the **over scan ratio**. Since the raster scan technology has to over scan, the percentage of overscan, in horizontal sense this will be:

$(H_{total} - H_{DE}) / H_{DE}$

This number plays a very important role in determine refresh rate and pixel clock. For example, if a VESA mode is needed but the pixel clock TW2880C generated is not close, the user can adjust the H_{total} to meet the refresh rate but still manage to get a lock from the monitor. This means, the above table is for reference, user still adjust the CRTC parameters to fit their environment.

TW2880C FRAME SYNCHRONIZATION





The multiple live and playback windows in the TW2880C system will go through a frame buffer and some calculation hardware to prevent video tearing during the normal display. There are four buffers allocated for each window. The hardware will try its best to maintain enough spacing between the write and read unit. Take the above right diagram for example. If the write side is in buffer 1, the read side needs to be in buffer 3. The write side hardware will try to advance to buffer 2 if every input channel has finished updating the data and read side is not on buffer 2. If it

cannot jump to buffer 2 and start updating, it will stay at the current buffer and repeat the write process again. This means, an incoming video frame is dropped. Similarly, the read side hardware will try to advance to buffer 4 if the reading process has finished. If it cannot jump to buffer 4, it will repeat the read process again so we say a frame has been repeated.

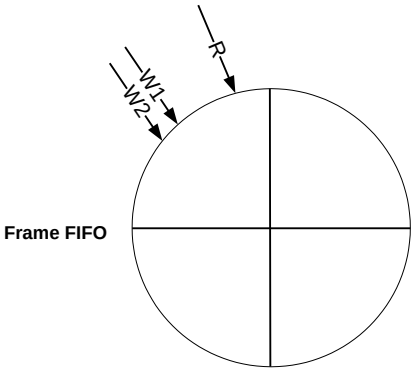
To get smooth animated result on the display, we should try to optimize the CRTC parameters to let the frame rates of the two processes as close as possible. To further facilitate on this subject, two import values are captured through hardware during the normal operation. They are:

```
(1) Live input vertical total register:  0x4A7[3:0], 0x4A6[7:0], unit is line,  the value is for
    two field
(2) Live input selection register:       0x6B4[7:4]: select which channel go through hardware
    counter
(3) Main output vertical number:         0x4A9[10:8], 0x4A8[7:0], unit is line,  the value is for
    one field
```

The use can select any one of the sixteen live channels to go through a hardware counter to count its vertical total value. The user then uses this value to compare with the value he put in the main CRT controller. If the value differs too much, either the write side frame dropping or read side frame repeating phenomenon described above will occur. To prevent this from happening, user can put a section of auto calibration routine inside their firmware to adjust the CRTC parameters and optimize the display quality.

WRITE BUFFER UPDATE AND CORRECTION CIRCUIT (NEW FOR TW2880C)

In TW2880C the write FIFO of the live channels and PB channels will also take suggesting value from frame rate controller when update the write pointer. The situation is a little different in Rev. C as we have new write buffer update method options for user to select. First, we need to familiar with the option registers:



(1) Live channels forced compatible mode register: 0x6fc[5], set one to this bit will force the live channel write FIFO using the old method of advancing. This mode is very helpful if the windows are used to display non-real time video sequence.

(2) Live input correction mode register: from 0x6e8 to 0x6ef for all 16 channels. The default value is 0xff (all on). The control is separated for each channel so that user can tune the circuit to have the correction they want.

There are four correction options for each live channel. They are (from the highest priority):

1. Channel Vsync is substituted by a pre-determined source if non-standard video is detected.

2. Current write FIFO pointer gets an educated update if non-standard video sequence is detected.

3. External loop timer has reach its predetermined value (period controlled by 0x6f9) and start the educated update.

4. Internal loop timer has reach its predetermined value (period controlled by 0x6ff) and start the educated update.

(3) PB channel has its own correction circuit. It is controlled by the following registers:

a. PB free running register 0x6fa[3:0]. 1 = even (always plus 1).

b. PB control register: 0x6f1, 0x6f0. Each four bit control one PB port.

60HZ DISPLAY AND CORRECTION TABLE

On the right hand side is the correction table for 60 Hz or higher monitor. It specifies which combinations are supported and which are not supported. It also specifies the register setting for the supported one. In the table "Both" means you can use the new update method and revert back to old method. "New" means you have to use new update method.

PB \ Live		Not Present	NTSC Real Time	PAL Real Time
NTSC Real Time	M	Both	Both	New
	S	✓	✓	✓
PAL Real Time	M	New	New	New
	S	✓	✓	✓
NTSC Non-real time	M	New	New	
	S	✓	✓	✗
PAL Non-real time	M	Old		Old
	S	✓	✗	✓
Not Present	M		Both	New
	S		✓	✓

PAL MODE INTERPOLATION AND CORRECTION

For systems that support PAL video input, there are two ways to display the image: (1) using a 50 Hz LCD monitor or (2) still using 60 Hz LCD monitor to display the image and let TW2880 do a 50-to-60 Hz frame rate interpolation on the final image to bridge the gap. Either way user does not need to adjust any register setting. It is an automatic process. One more thing to remember is if the user chooses to use 50 Hz display as main display, you cannot support any NTSC video inputs.

CORRECTION IN DETAILS

As you see from the frame FIFO pointer diagram, the write pointer is slower than the read pointer so the read pointer is always chasing the write pointer. If the time comes when we need to advance the read pointer but the FIFO is not ready, the read pointer will repeat itself.

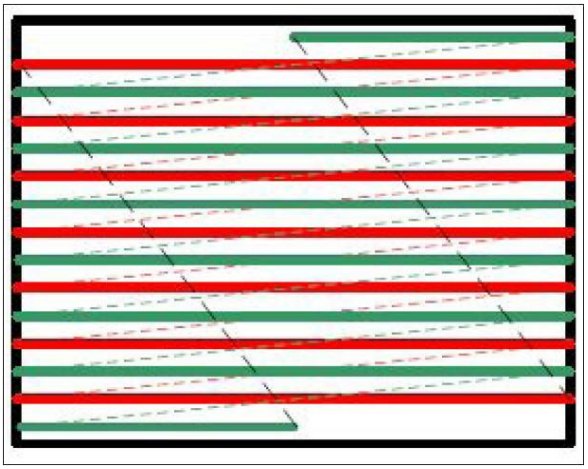
Because the one read pointer limitation in TW2880, once a channel's signal quality has gone bad and cannot generate valid vertical sync to maintain to total read / write relationship, you will see jumping image or so call the stop-n-go phenomenon in the display. In TW2880C, we have several levels of hardware correction to prevent these from happening.

All those conditions will force current FIFO into a write page correction phase, so the image of current channel may become less desired if the condition of the signal is very bad. However, the total channel image will be saved.

The detection of the non-standard video in TW2880 is using a vertical line counter located in the input section. If the incoming video stream has vertical line number smaller than a certain number or larger than a certain number, a non-standard signal for this particular channel is asserted. These number registers (0x3f8 – 0x3fb) are user programmable. The non-standard operation is controlled by 0x6b9[7:6] as it can be turn off. In addition, there is an interrupt associated with the non-standard video signal. It is default to be turned off. User can use non-standard interrupt to do auto adjust on the correction items.

BEAT FREQUENCY

In a video capture system, it is impossible to keep the frame rate of the incoming video stream and display video stream exactly the same. This minute difference between the two frame rates is called “beat frequency”. In an easier term, when the system reach the beat frequency and the input side is slightly faster, a frame is skipped, but if the output side is slight faster, a frame will be repeated.



Skipping a frame is easier to detect than repeat a frame especially horizontal line pattern is involved. To overcome this and get optimal visual quality, try to set the display to run just a tad faster than the input then this phenomenon will not be too obvious.

INTERLACED MODE SETTING

The diagram on the right illustrated the detailed action behind an interlaced display: it is displayed every other line and start of the odd field is in the middle of the horizontal scan line. To achieve this in a TW2880 system, five parameters need to be set correctly:

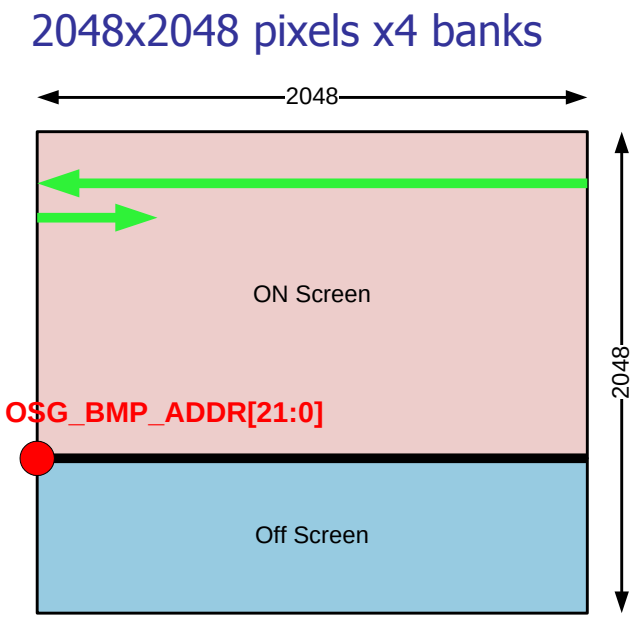
(1) VCLK need to be half of the progressive mode counterpart.
(2) Vertical total register 0x501[3:0], 0x500[7:0], need to be programmed as Vertical / 2 – 1
(3) Vertical sync delay register 0x481[3:0] need to put some value, unit is pixel. For HDMI mode, this has to be zero.
(4) Turn on interlaced mode, register 0x4C3, bit 0

Display Memory and Buffer Management

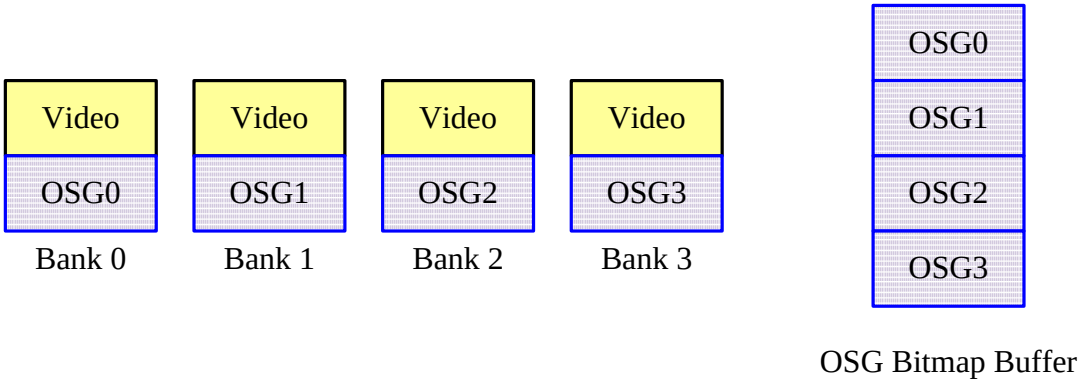
TW2880 display memory address management is automatically handled by hardware. Take the most commonly used 128Mbit x2 configuration (64 bit) for example, the memory can be structured as a 2048x2048 pixel x4 memory array. For the main display or dual display to run properly, user does not need to specify anything other than the starting addresses. The CRT controller of both units will calculate the next address and initiate the buffer read process. In the diagram, remember it actually refers to a single bank or page as TW2880 is using 4 banks to do channel synchronization and buffering. The page advancement is handling by Frame Rate Control Unit.

The memory in the pink portion is often referred to as "on-screen memory" since you will see the content in your display. The rest is called "off-screen" memory as you should never see that content on display and it is used as storage for many things. The split point between the two buffers is determined by OSG bitmap starting address.

Another important term is called "display pitch" and is defined by register 0x210. The unit is 16 pixels. This register defined a virtual width for the memory buffer. Because of the fixed horizontal size of the physical memory, it will create memory holes when mapping a particular size display to the memory that does not has a similar width. Therefore, if use this register, the size of memory buffer will become flexible to the user so no memory will be wasted.

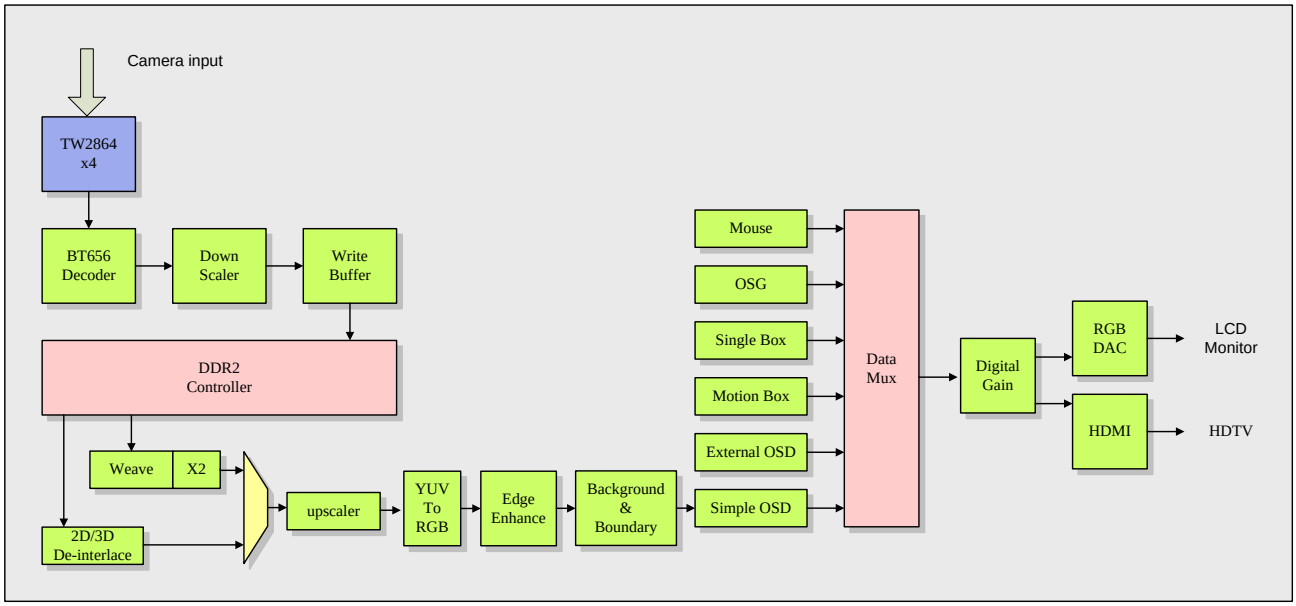


DISPLAY SDRAM Mapping (128Mbitx2)



The off-screen memory scattered in the four pages will be remapped and linked into a contiguous array where its starting address is defined by OSG_BMP_ADDR, 0x13D, 0x13C, 0x13B. More of this subject can be found in "Section 6: OSG and Simple OSD" beginning on page 147.

Display Pipe



DE-INTERLACING EFFECT SELECT AND UP SCALER

Based on the above diagram, three de-interlacing effect can be selected. Program 0x54B bit 0 is to select Weave mode or 2D-3D mode. "0" will select Weave mode. Please note if select Weave mode there will be no upscale support and you don't need to setup more registers. If select "2D-3D' mode, user need further program 0x400 bit 0 to select 2D or 3D mode. Program "0" will select 3D mode. There are many options in 2D-3D unit. Please look at TW2880 data book.

(1) For 2D-3D mode, need to program 0x405[2:0], 0x404[7:0] for vertical size in one field. The unit is line.
(2) Program 0x407[2:0], 0x406[7:0] for horizontal size, unit is pixel. Item1 and item2 will determine the 2D/3D source area.
(3) Now if the video stream goes into display pipe has the same size, Program HSCALE registers 0x418[4:0], 0x417[7:0] to 0x1000.
(4) Program VSCALE registers 0x41A[4:0], 0x419[7:0] to 0x400. These values mean no scale.
(5) Program Final Horizontal Width registers 0x420[2:0], 0x41F[7:0] to be the same as the screen width.
(6) Program Final Vertical Height registers 0x422[2:0], 0x421[7:0] to be the same as screen height.
(7) If scale up is needed, user needs to program 0x41C[2:0], 0x41B[7:0] to determine Horizontal start position. Program 0x41E[2:0], 0x41D[7:0] to determine Vertical start position.
(8) As in Item5, Item6, Program Final Horizontal Width and Height registers if the screen is different than the source.
(9) Program 0x418[4:0], 0x417[7:0] for Horizontal Scale Factor.
(10) Program 0x41A[4:0], 0x419[7:0] for Vertical Scale Factor.

If the original size times the scale factor is larger than the final output size, the result is cropping. On the other hand, if the original size times the scale factor is smaller than the final output size, a smaller image with the rest fill with background color will show.

The main usage for up scaler is bandwidth saving. Because 3D de-interlacing function consumes a lot of bandwidth, we often start the processing at a smaller screen size and use the scaler to increase the image to the final size. However, by doing this, you will need to upscale the boundaries of each window, otherwise the overlapping result will look funny. To use the position update function, you need to:

(1) Program 0x486[7:0], 0x485[7:0] for Horizontal Scale Factor.
(2) Program 0x488[7:0], 0x487[7:0] for Vertical Scale Factor.
(3) Program 0x489[0], 1 = enable, the scaling will go into effect when the next vertical sync pulse comes.

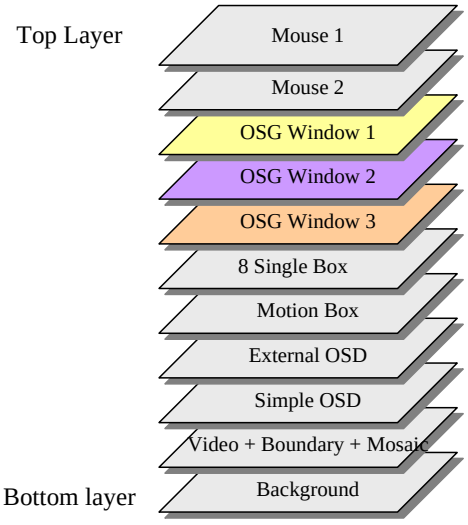
3D MODE ADRESS CALCULATION

Just like the TW2880's Weave style display memory management, the 3D display memory management is also automatic. User does not need to calculate and put addresses in linear form, they only need to specify the starting screen address, native image size, upscale coefficients and the size of the display. The only information needed to provide in linear address form is the location of the 3D error information buffer. Here is the register setting sequences:

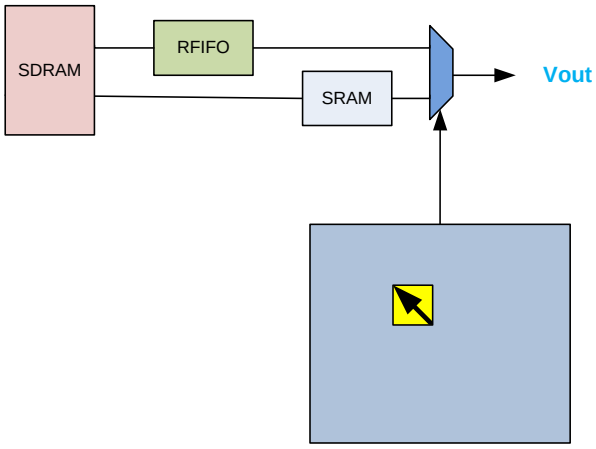
1. Calculate space needed. SIZE = REC_PITCH x H/2 x 2. double buffer.
2. Counting from behind, get the starting address. We usually put the error buffer at the end of the display memory.
3. Put into register 0x403, 0x402, 0x401 as addr[23:0], set bit [23;22] to "2'b11", the last bank.

Display Layers

There are 11 display layers in the main display of the TW2880. See the diagram on the right for each one. The precedence is from top to bottom. OSG and Simple OSD contain too much information to be explained in this section, so it will be covered in a separate section. We will describe each layer in detailed from the top.



MOUSE POINTER



Mouse pointer is sometimes called Hardware cursor. There are two independent units in TW2880 so many special effect can be made. The components of a hardware cursor are a hardware position comparator and a video data multiplexer. The host from the outside will program these position registers and it is in the range the display pixel will switch to content stored in the SRAM. The position update should only happen if ox54E[0] is set to one. If the user wants to update the positions of the cursor, they need to program this bit as "0" to keep a smooth overlay effect.

There are two ways to update the content in SRAM, by host or by an updating agent pulling result from SDRAM. Here is the procedure:

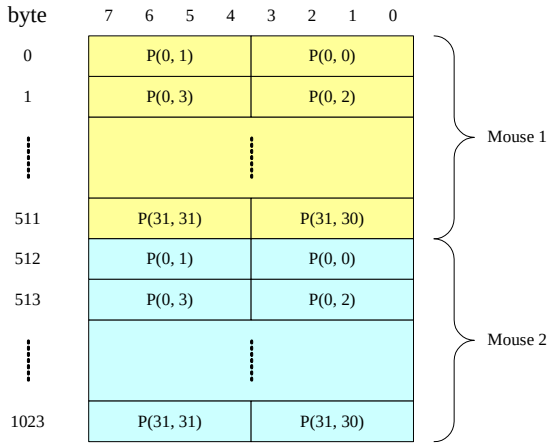
Put content into SRAM:

(1) Program 0x547[7:0], Mouse data location.
(2) Write mouse data 0x54d[7:0] four times, the SRAM is configured as 256x32.
(3) Write 0x54f with any value, 4 byte mouse data will be loaded into SRAM.
(4) Repeat step(1)-(2) 256 times or any sub-set for local update.

Put content into SDRAM:

(1) Program 0x46e[7:0],0x46d[7:0],0x46c[7:0] as the Mouse Base Address. This is a linear address.

```
   Mouse circuit will pull data from here.
(2) Program 0x003[7:0] = 0xe0,enable write and with burst length = 1.
(3) Program 0x002[7:0],0x001[7:0],0x000[7:0] to have content the same as the Mouse Base Address.
(4) Program 0x004[7:0] as the mouse data and repeat 256 times.  A 32x64 SDRAM write has been requested.
(5) Read status from 0x044[0]. If high then it is done.
(6) Repeat (3) – (5) but in step (3) address need to be 32 more.
(7) Repeat (2) – (6) 16 times to cover 16 locations. (i.e. 16 mouse shapes)
(8) Program 0x470 to select which mouse and which shape is to be loaded from SDRAM.
(9) Write 0x46f[0] = 1, enable the update process. Check 0 = done
```



Bit	Description
3	**Blinking Control** 1: Blink 0: Static
2	**Mixing Control** 1: Based on mouse mix control 0: mouse color
1:0	**Mouse Type** 00: Transparent 01: Filled with white color 10: Foreground color 11: Background color

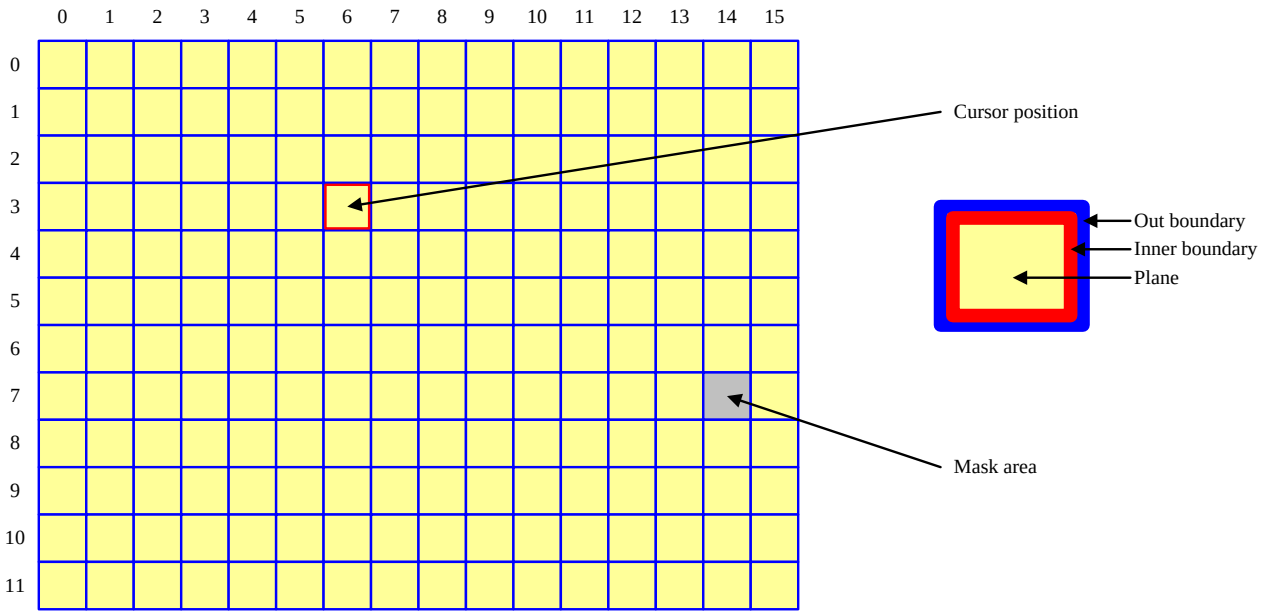
SINGLE BOX

TW2880 provides 8 single boxes that can be used for highlighting portion of the display. The effects include a single box or box cursor, a masking box and a box blending with a plane color. Each box has programmable location and sizes and controlled by BOX_HL (0x513 - 0x51A), BOX_HW (0x51B - 0x522), BOX_VT (0x523 – 0x52A) and BOX_VW (0x52B – 0x532) registers. The BOX_HL is the horizontal location of box with 2 pixel units and the BOX_HW is the horizontal size of box with 2 pixel units. The BOX_VT is the vertical location of box with 1 line unit and the BOX_VW is the vertical size of box with 1 line unit.

The display option is controlled by Control registers (0x50C – 0x510). BOX_PLNEN bit in these register enables each plane color and its color is defined by the BOX_PLNCOL (0x221, 0x227, 0x22C, 0x233) register. Mixing is also controlled by these registers. The color of box boundary is enabled via the BOX_BNDEN bit in the control registers and its color is defined by the BOX_BNDCOL (0x220, 0x226, 0x22B, 0x232) registers.

In cases where several boxes have same region specified, there would be a conflict of what to display for that region. Generally, the TW2880 defines that box 0 has priority over box 7. So if a conflict happens between more than 2 boxes, box 0 will be displayed first as top layer and box 1 to box 7 are hidden beneath that are not supported for pop-up attribute unlike channel display.

MOTION BOX



TW2880 supports an array boxes layer that has a programmable cell size up to 16x16. This box array can be used to make table menu or display motion detection information to the user. When motion detection mode is enabled, user must set horizontal cell number to 15 and vertical cell number to 11. This layer is available to all live channels so most of the time, users need to program 16 set of registers. The subsequent explanation of the function only talks about the first channel. However, it is easy to duplicate on other channels as well. To use it first you need to determine the mode and enable it.

```
(1) Program 0x550[6] to determine display mode. 1 = motion display mode.
(2) Program 0x550[7] to enable MD box.
```

Cell Composition

A motion cell is composed by four elements: out boundary, inner boundary, mask and plane. Out boundary and plane color make up the usual overlay color. Inner boundary and mask is used to show special event like cursor and motion. To determine the color of these elements, user need to program:

```
(1) Program 0x493,0x494,0x495 to determine out boundary R, G, B color.
(2) Program 0x496,0x497,0x498 to determine inner boundary R, G, B color.
(3) Program 0x499,0x49A,0x49B to determine mask R, G, B color
(4) Program 0x49C,0x49D,0x49E to determine plane R, G, B color
(5) The boundary can be enabled by programming 0x500[4], MDBOX_BNDEN
```

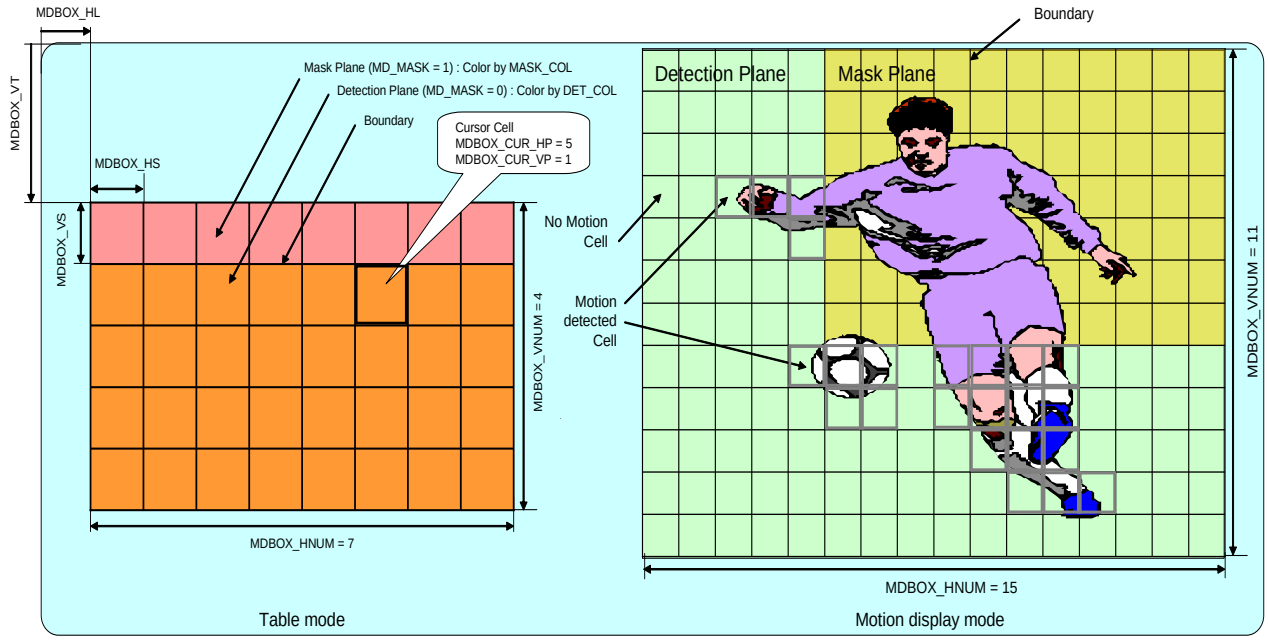
The cursor cell is enabled by the MDBOX_CUREN 0x550[5] register and the displayed location is defined by the MDBOX_CURHP 0x5F8 and MDBOX_CURVP 0x48B registers. Its color is a reverse color of cell boundary. It is useful function to control motion mask region.

Motion box positions and sizes are controlled by registers. To overlay mask information and motion result on video data properly, the scaling ratio of video should be matched with motion box size.

For each MD array, the number of row and column cells is defined via the MDBOX_HNUM(0x5E8[3:0] ~ 0x5EF[7:4]) and MDBOX_VNUM (0x5F0[3:0] ~ 0x5F7[7:4]) registers. The horizontal and vertical location of left top is controlled by the MDBOX_HL (0x568 ~ 0x587) register and the MDBOX_VT (0x558 ~ 0x5A7) registers. The horizontal and vertical size of each cell is defined by the MDBOX_VS (0x5C8 ~ 0x5E7) registers and the MDBOX_HS (0x5A8 ~ 0x5C7) registers. Therefore, the whole size of MD arrayed box is same as the sum of cells in row and column.

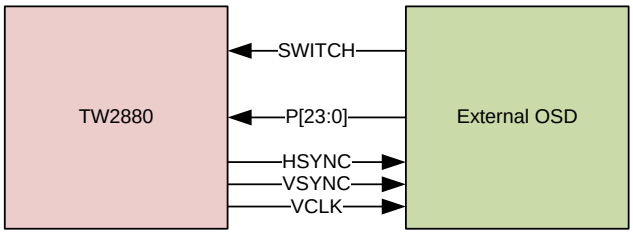
The plane of MD arrayed box is separated into mask plane and detection plane. The mask plane represents the cell defined by MD_MASK (0x800 ~ 0xBD7) register. The detection plane represents the motion-detected cell excluding

the mask cells among whole cells. The mask plane of MD arrayed box is enabled by the MDBOX_MSKEN (0x550[3] ~ 0x55F[3]) register and the detection plane is enabled by the MDBOX_DETEN (0x550[2] ~ 0x55F[2]) register. The color of mask plane is controlled by the MASK_COL register and the color of detection plane is defined by the DET_COL register. The mask plane of the MD arrayed box shows the mask information according to the MD_MASK registers automatically, the additional narrow boundary of each cell is provided to display motion detection via the MDBOX_DETEN register, and its color is a reverse cell boundary color. The plane can be mixed with video data by the MDBOX_MIX (0x550[1:0] ~ 0x55F[1:0]) register. Even in the horizontal / vertical mirroring mode, the video data and motion detection result can be matched via the MDBOX_HINV and MDBOX_VINV registers.

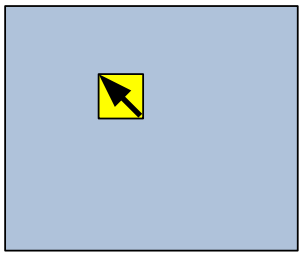


EXTERNAL OSD

TW2880 supports a master mode external OSD function. In this setup, TW2880 will send out pixel clock and Hsync and Vsync information to the external chip. The external chip will pick up the pixel clock and sync up with TW2880's main display, then the external chip will also provide strobe signal and the OSD data to TW2880 to display.



The pins of this function are shared with live video port. TW2880's live video port 5, 6, 7, 8 have other definition when running at 108 MHz mode. Please refer to control register 0x201 bit 2. When this bit equal to 0, the inputs are used for live video channel running at 54 MHz. When this bit equal to 1, the port 5. 6. 7. 8 are defined as output pins and are used to output digital R, G, and B of the main display. However, if this bit equals 1 and the external OSD option is selected, these extra pins are used as input pins to accept OSD data from an external chip in master mode. See the following chart as reference.



	0x201 Bit [2] = 0	0x201 Bit [2] = 1	
INPUT	LV clock12	LV clock12	
IN1	BT656[7:0]	BT656[7:0]	
IN2	BT656[7:0]	BT656[7:0]	
INPUT	LV clock34	LV clock34	
IN3	BT656[7:0]	BT656[7:0]	
IN4	BT656[7:0]	BT656[7:0]	
IN/OUT	LV clock56	VCLK out	Pixel clock
IN/OUT 5	BT656[7:0]	R[7:0]	
IN/OUT 6	BT656[7:0]	G[7:0]	
IN/OUT	LV clock78	DEN / SWITCH	Display enable
IN/OUT 7	BT656[7:0]	B[7:0]	
IN/OUT 8	BT656[7:0]		

To use this function, following this procedure:

```
(1) Program 0x4BB[4] to select which you want pixel from outside or using internal color.
(2) Program 0x4BD,0x4BE,0x4BF to determine internal R, G, B color if needed.
(3) Program 0x4BB[2:1] to select enable line options
(4) Program 0x4BB[0] to enable alpha blending
(5) Program 0x4BC[3:0] to enter alpha ratio,
(6) Program 0x4BB[3] to enable external OSD layer.
```

PRIVACY WINDOWS

This feature is used to provide privacies for the monitored objects. In many occasions, we need to provide options to block certain private information from being monitored. There are actually three sets of window in the TW2880. One for live video (dual monitor is controlled by the same set), one for SPOT and one for record. To simplify and reduce the number of control registers, we have shadowed the three register sets under the control of register 0xE4F bit 1 and 0. 0x00 will control the reading and writing of the recording privacy windows, 0x01 will control the SPOT window set and 0x10 will control live window set. To read / write the respective register sets you need to put proper values to these two bits first.

Assume we want to use privacy window 1 in live display set, to use this function, following this procedure:

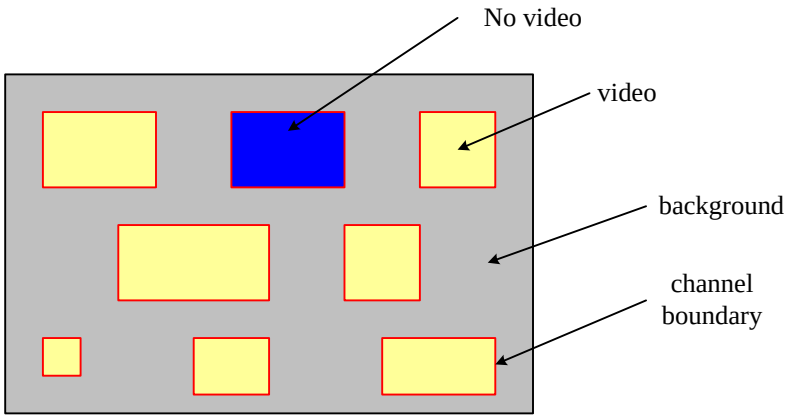
```
(1) Program 0xE4F[1:0] to "10" to select live windows.
(2) Program 0xE50[7:0], this is horizontal start position.  Unit is double pixel.
(3) Program 0xE60[6:0], this is Vertical start position.  Unit is double line
(4) Program 0xE70[4:2] to determine Hsize, 0xE70[1:0] to determine Vsize.
(5) Program 0xE70[7:5] to determine effect, black out or Mosaic.
(6) Program 0xE60[7] = 1 to enable the window
```

BACKGROUND AND CHANNEL BOUNDARY

The area without video will show background. Background color is a 24-bit color, which can be set by programming registers: 0x539, 0x53A and 0x53B for RGB respectively. Please note that background color is not “no video” color. “No video” color is sent by TW2864/TW2865.

Each channel has its own boundary. TW2880C supports 32 channel boundaries. Channel position and size information are retrieved from rgb_interface related registers. If upscale is enable, user must turn on pos_ups_en and set correct pos_hscale and pos_vscale registers. In some cases, if video is not turn on but user still wanted to show boundaries for this channel. User must set the following registers:

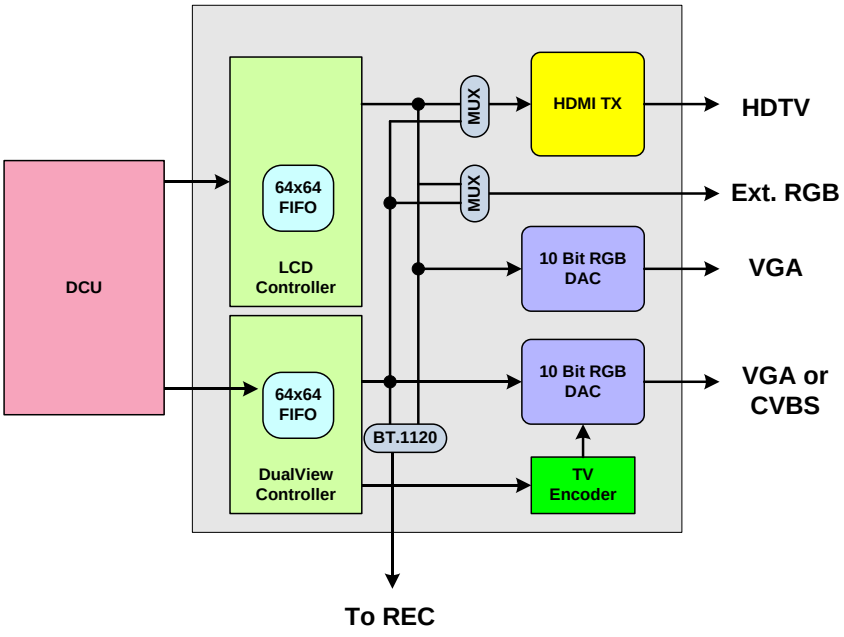
```
[0x4F0] to [0x4F3]: BND CH EN
[0x4CE] bit 4: BND CH EN SEL, this bit must be set to “1”
[0x4F8] to [0x4FA]: NOVID_R/G/B
```



Display Background and Boundary

Flexible Output

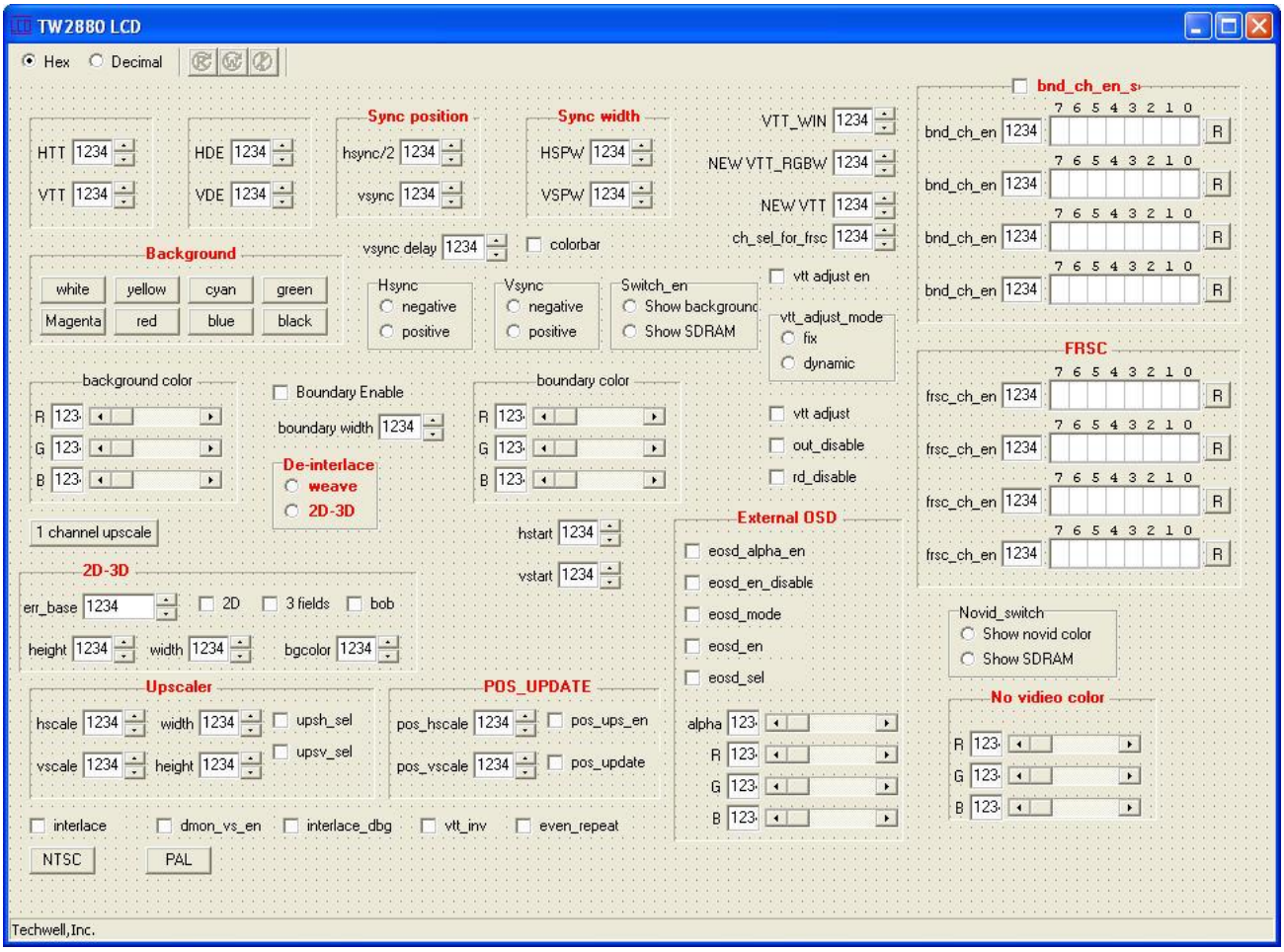
The display output of TW2880 is very powerful and flexible. It supports HDMI, external RGB, VGA, dual monitor VGA and dual monitor TV output. Some of the outputs are shared by the two units.

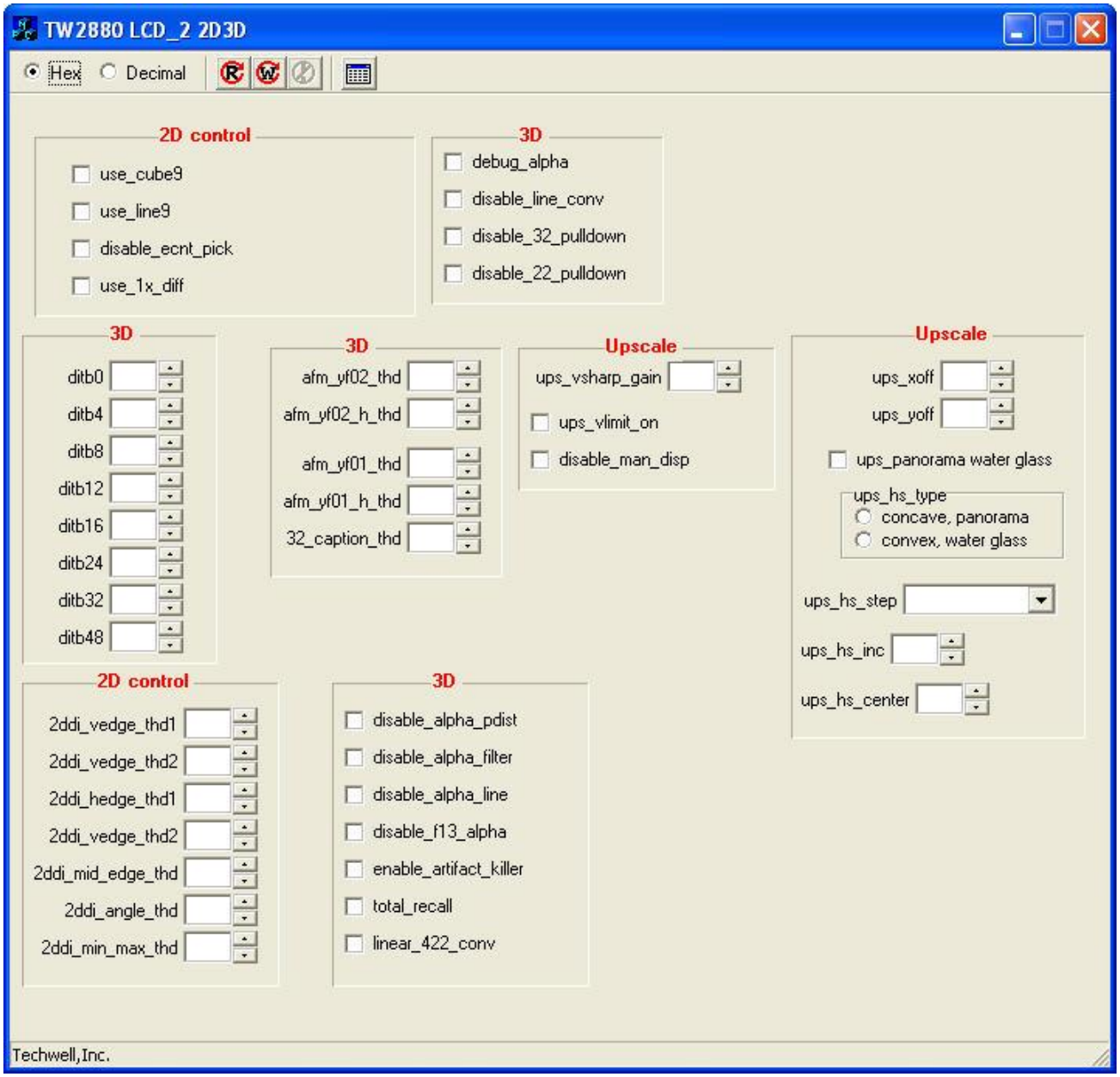


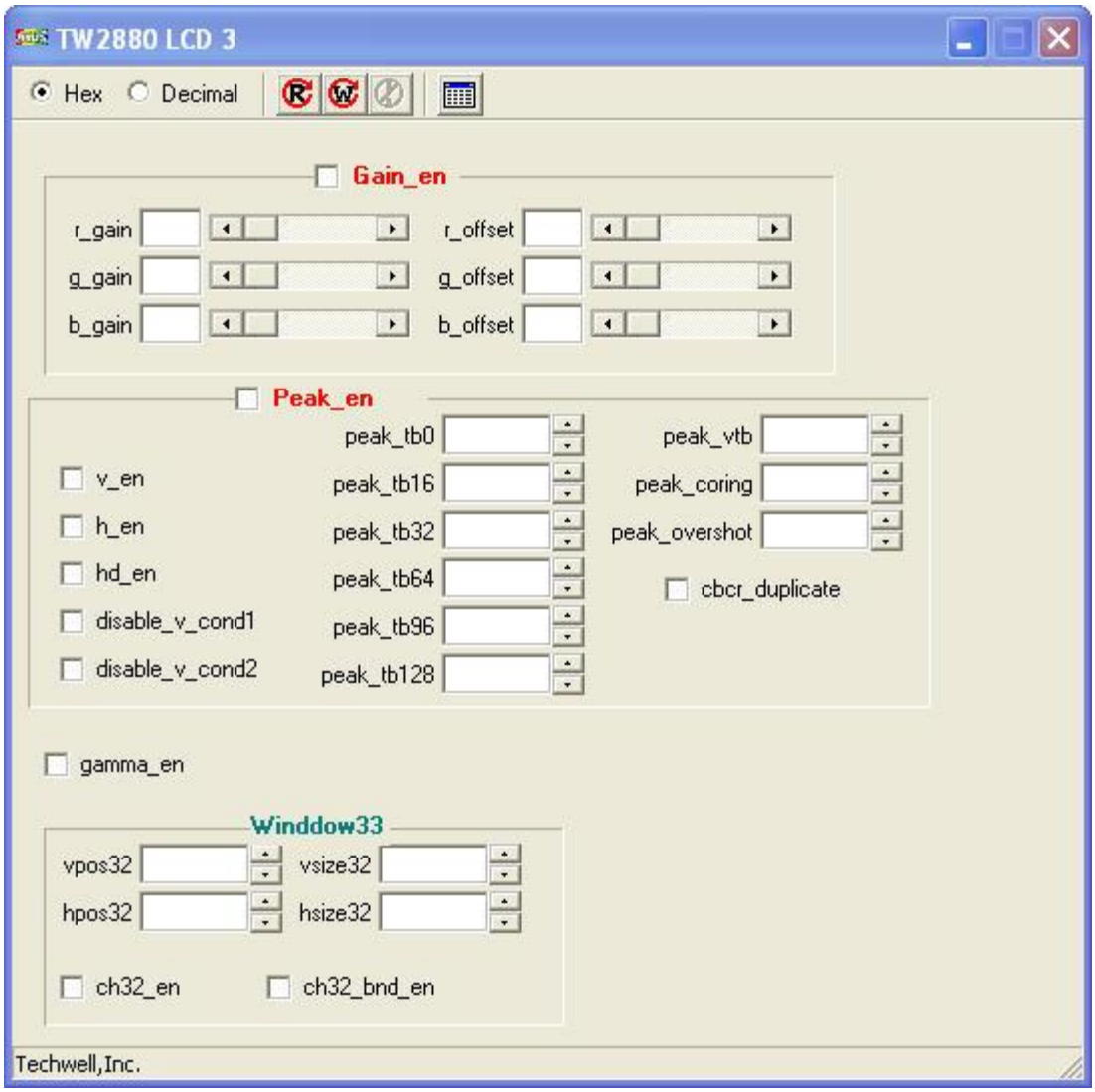
(1) Program 0x21E[7] to select which source go to HDMI. "0" select main display.
(2) Program 0x201[7] to select which source go to external RGB output. "0" select main display.
(3) Program 0x712[2] to select dual monitor output to VGA or TV. "0" select VGA.

The external RGB interface can be very useful as it can drive an external 3D de-interlated chip or an external HDMI transmitter chip. The output from both units can be redirected to a BT.1120 encoder. The output will go out through recording port. This is very useful link for cascading the TW2880. This subject is covered in detail in "Section 3: PB Window and Channel ID Decoding" beginning on page 43.

Terminal Tool




TW2880 LCD_2 2D3D
Hex
Decimal
2D control
use_cube9
use_line9
disable_ecnt_pick
use_1x_diff
3D
debug_alpha
disable_line_conv
disable_32_pulldown
disable_22_pulldown
3D
ditb0
ditb4
ditb8
ditb12
ditb16
ditb24
ditb32
ditb48
3D
afm_yf02_thd
afm_yf02_h_thd
afm_yf01_thd
afm_yf01_h_thd
32_caption_thd
Upscale
ups_vsharp_gain
ups_vlimit_on
disable_man_disp
Upscale
ups_xoff
ups_yoff
ups_panorama water glass
ups_hs_type
concave, panorama
convex, water glass
ups_hs_step
ups_hs_inc
ups_hs_center
2D control
2ddi_vedge_thd1
2ddi_vedge_thd2
2ddi_hedge_thd1
2ddi_vedge_thd2
2ddi_mid_edge_thd
2ddi_angle_thd
2ddi_min_max_thd
3D
disable_alpha_pdist
disable_alpha_filter
disable_alpha_line
disable_f13_alpha
enable_artifact_killer
total_recall
linear_422_conv
Techwell,Inc.


TW2880 LCD 3
Hex
Decimal
Gain_en
r_gain
r_offset
g_gain
g_offset
b_gain
b_offset
Peak_en
peak_tb0
peak_vtb
v_en
peak_tb16
peak_coring
h_en
peak_tb32
peak_overshot
hd_en
peak_tb64
cbcr_duplicate
disable_v_cond1
peak_tb96
disable_v_cond2
peak_tb128
gamma_en
Winddow33
vpos32
vsize32
hpos32
hsize32
ch32_en
ch32_bnd_en
Techwell,Inc.

Dual Monitor

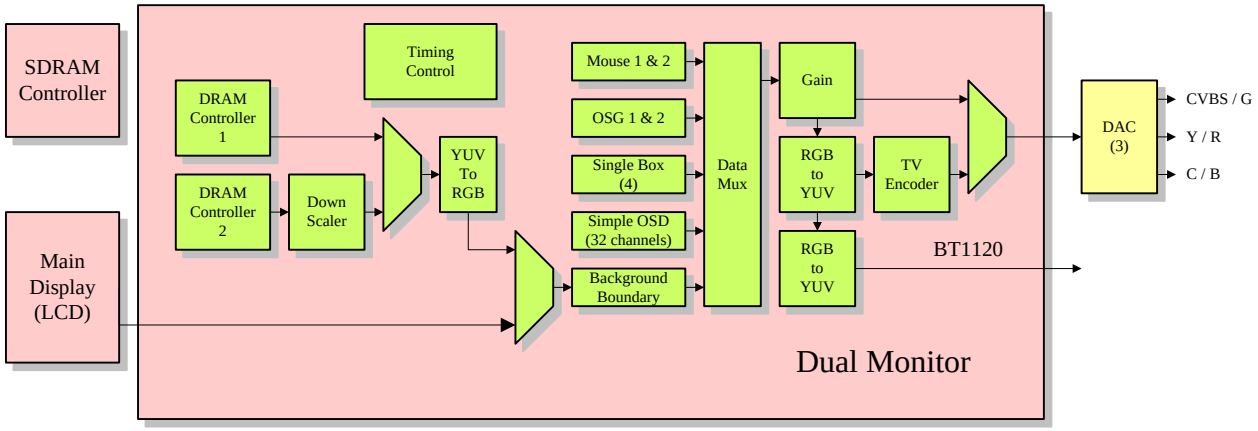
Introduction

In addition to the main display controller, TW2880 has a secondary display controller, which can support display devices with interlaced or progressive timing. Using the integrated TV encoder and the DAC, this controller can drive traditional TV with CVBS or S-Video output or a progressive LCD display with VGA socket. With properly setup software registers and display memory planning between the two display controllers, the sixteen input channels and sixteen playback channels can be displayed in many different resolutions and combinations between the two different monitors.

Features

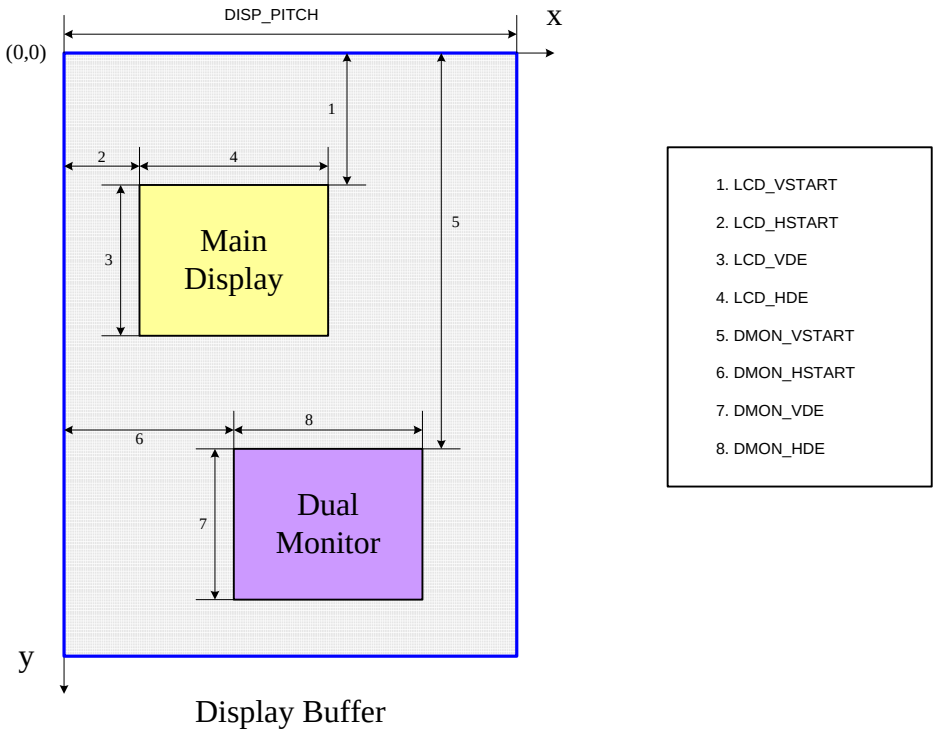
- Supports NTSC / PAL standard TV monitor with integrated 10 bit DAC and TV encoder
- Supports LCD monitor (up to 1080p)
- Down scaling from 1920x1080 resolution
- Supports same or different video content with main display
- Two OSG layers, each layer has four sub-window
- Two OSG layers support different upscale ratio
- Supports same or different OSG content with main display
- Two mouse layers on screen
- Supports 16 mouse shape in SDRAM
- Four single boxes
- Supports main display to CVBS output
- Supports 16 live channels and 16 play back channels
- Supports simple OSG for 32 channels

Dual Monitor Controller Block Diagram



Memory Diagram

The following picture shows the LCD and Dual monitor memory in SDRAM and some related registers. For the dual monitor start address, it can start anywhere by program the DM_HSTART and DM_VSTART address.



Display Buffer

[0x704, 0x705] : Dual monitor horizontal active data width. This register must be set to read value minus 1.

[0x706, 0x707] : Dual monitor vertical display height. This register must be set to real value minus 1.

In interlaced mode, it is set to height of one field minus 1.

[0x713, 0x715] : Dual monitor DRAM horizontal start register (four pixels unit).

[0x714, 0x715] : Dual monitor DRAM vertical start register (one line unit).

CRTC setting

The horizontal parameters are:

(1) Horizontal total register: 0x701[3:0], 0x700[7:0], unit is pixel, the value put in is real value -1
(2) Horizontal display end register: 0x705[3:0], 0x704[7:0], unit is pixel, the value put in is real value -1
(3) Horizontal sync start register: 0x709[0],0x708[7:0], unit is double pixel
(4) Horizontal sync width register: 0x70B[7:0], unit is pixel
(5) Horizontal sync polarity register: 0x712[0]

Similarly, we have vertical parameters:

(1) Vertical total register: 0x703[3:0], 0x702[7:0], unit is line, the value put in is value -1
(2) Vertical display end register: 0x707[3:0], 0x706[7:0], unit is line, the value put in is value -1
(3) Vertical sync start register: 0x70A[7:0], unit is line

(4) Vertical sync width register: 0x70C[7:0], unit is line
(5) Vertical sync polarity register: 0x712[1]

DM_HTT, DM_HDE, DM_VTT and DM_VDE are based on pixel clock.

In NTSC mode, dm_vclk must be set to 13.5MHz.

[0x700, 0x701] : DM_HTT = 857

[0x704, 0x705] : DM_HDE = 719

[0x702, 0x703] : DM_VTT = 261

[0x706, 0x707] : DM_VDE = 239

In 1280x1024 VGA mode, dm_vclk must be set to 108MHz.

[0x700, 0x701] : DM_HTT = 1687

[0x704, 0x705] : DM_HDE = 1276

[0x702, 0x703] : DM_VTT = 1065

[0x706, 0x707] : DM_VDE = 1023

In 1920x1080 VGA mode, dm_vclk must be set to 148.5MHz.

[0x700, 0x701] : DM_HTT = 2199

[0x704, 0x705] : DM_HDE = 1920

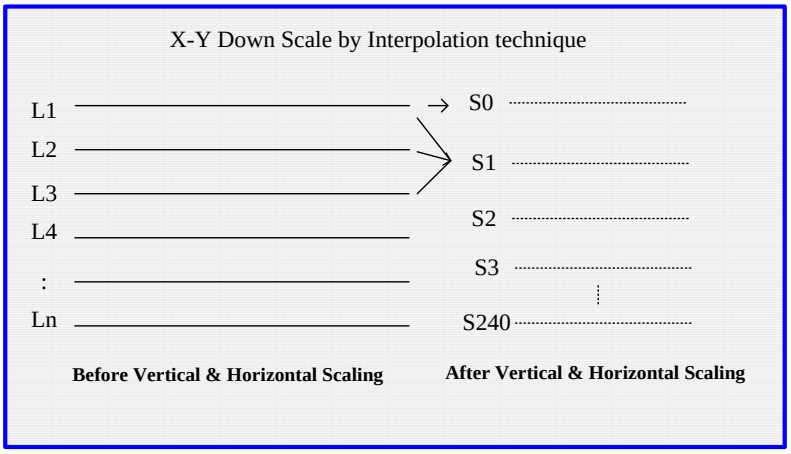
[0x702, 0x703] : DM_VTT = 1124

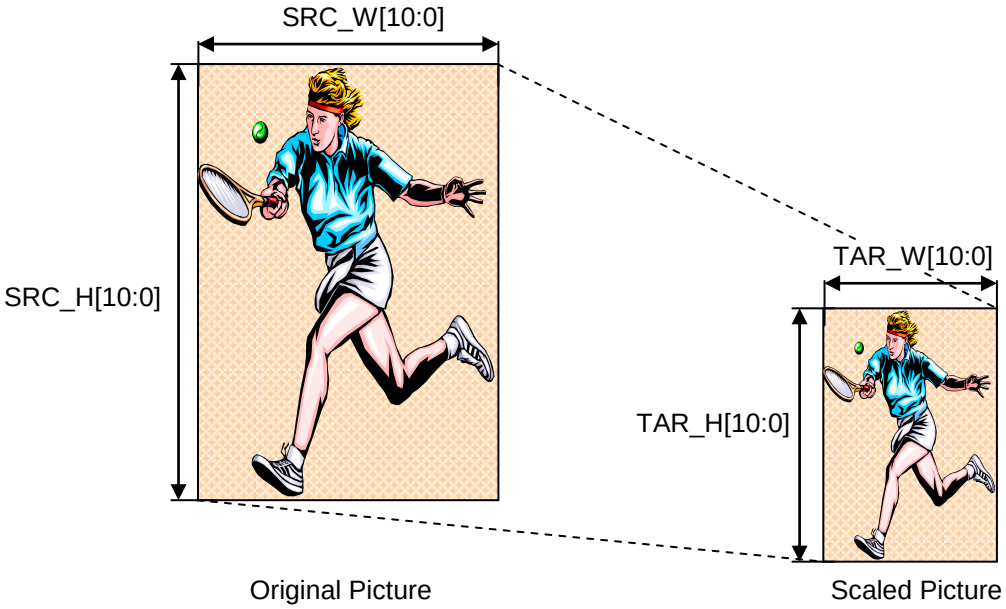
[0x706, 0x707] : DM_VDE = 1079

Down Scalar

Down scalar is a module uses to scale down more channels on the display to fit the TV standard timing. So on the second monitor, we can pack as many channels as seen on the LCD and display them on a regular TV.

By using the linear interpolation technique, with independent setting of horizontal and vertical scale factors and others registers. We can scale the number of channels, the size and the location of the channels to the display's native resolution.





[0x716, 0x717] : Dual monitor down scaler target image width((1 pixel unit).

[0x718, 0x719] : Dual monitor down scaler target image height(1 line unit).

[0x70E, 0x70F] : Dual monitor down scaler source image width(1 pixel unit).

[0x710, 0x711] : Dual monitor down scaler source image height(1 line unit).

Ex) D1 → CIF

[0x716, 0x717] : SRC_W = 720

[0x718, 0x719] : SRC_H = 240

[0x70E, 0x70F] : TAR_W = 360

[0x710, 0x711] : TAR_H = 240

Ex) D1 → CIF

[0x700, 0x701] : DM_HTT = 720

[0x704, 0x705] : DM_HDE = 240

[0x702, 0x703] : DM_VTT = 360

[0x706, 0x707] : DM_VDE = 120

OSD Control

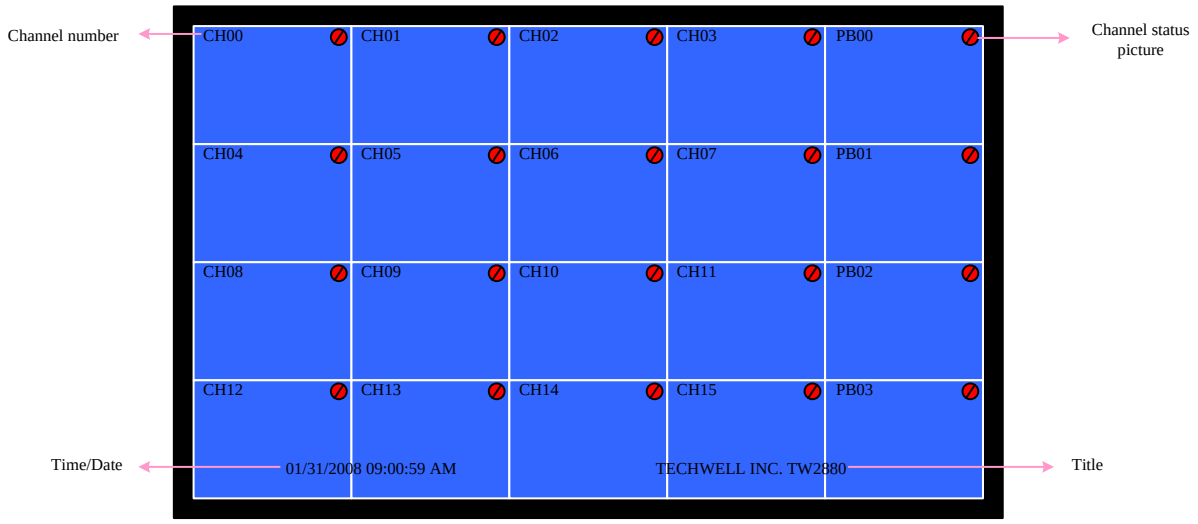
Dual monitor use the same architecture and design as the OSD in the LCD display path.

With all DM_OSD control registers in the Dual monitor page, Firmware can set up the DM_OSD display as the same source as the LCD Main Display Port if the Dual monitor displays the same channels.

If the LCD Main Display Port and Dual monitor show different channels on two displays, each OSD will display the channel correspondent to its own cameras. In other words, different channels will be mapped to different OSD source accordingly.

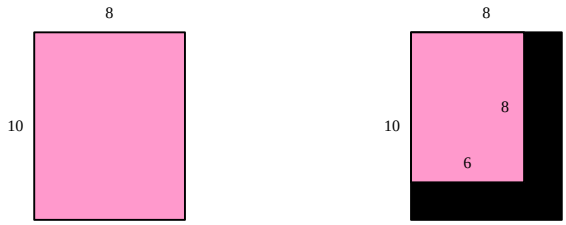
All the operation and programming of the DM_OSD registers. Please refer to the section where the detailed explanation and operation are fully documented.

- 64 fonts table saved in SRAM
- Channel information table saved in SRAM
- Three lines channel information
- 32 characters date/time
- 16 characters channel title for each channel
- 16 characters channel status for each channel
- font size can be changed in 6x8, 8x10, 12x16, 16x20


Channel number
CH00
CH01
CH02
CH03
PB00
Channel status picture
CH04
CH05
CH06
CH07
PB01
CH08
CH09
CH10
CH11
PB02
CH12
CH13
CH14
CH15
PB03
Time/Date
01/31/2008 09:00:59 AM
TECHWELL INC. TW2880
Title

FONT & PICTURE

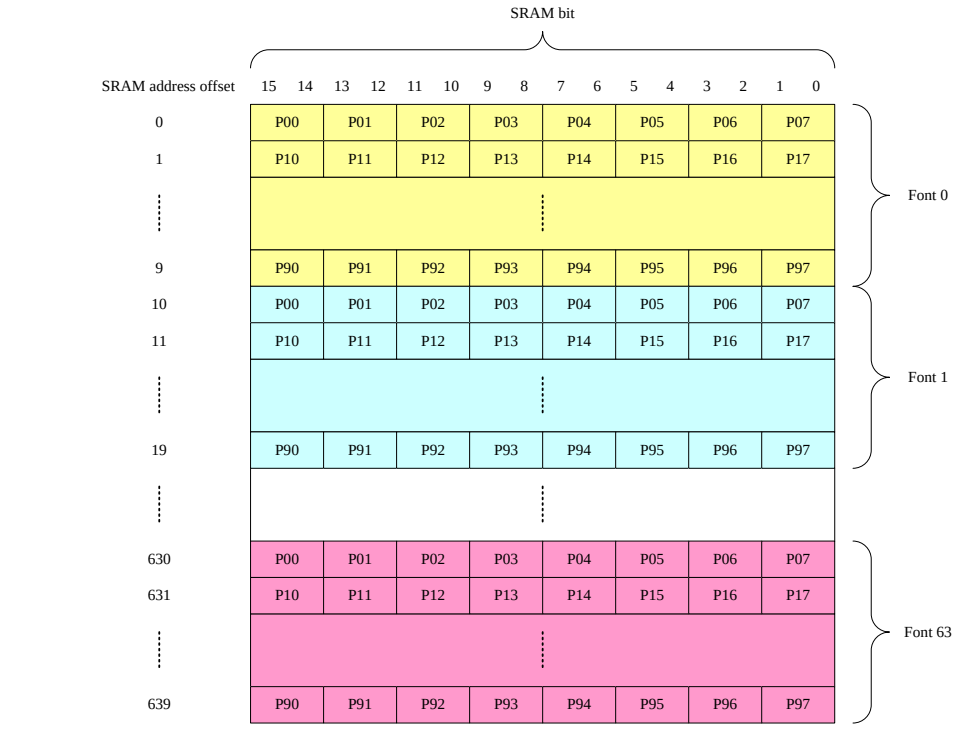
There are totally 64 fonts, which can be saved in SRAM. The font size saved in SRAM is fixed to 8x10. However, displayed font size can be changed. Horizontal can display four sizes: 6, 8, 12 or 16. Size 12 or 16 are doubled from size 6 or 8. If 6 or 12 are selected, fonts saved in SRAM must have small size. However, an additional two pixels must be saved in SRAM. Vertical can display four sizes: 8, 10, 16 or 20. Size 16 or 20 are doubled from size 8 or 10. If 8 or 16 are selected, fonts saved in SRAM must have small size. However, an additional two lines must be saved in SRAM. The following picture shows SRAM data. For 8x10 font, all SRAM data is valid. For 6x8 font, only 6x8 area is valid. The black area is for dummy data.

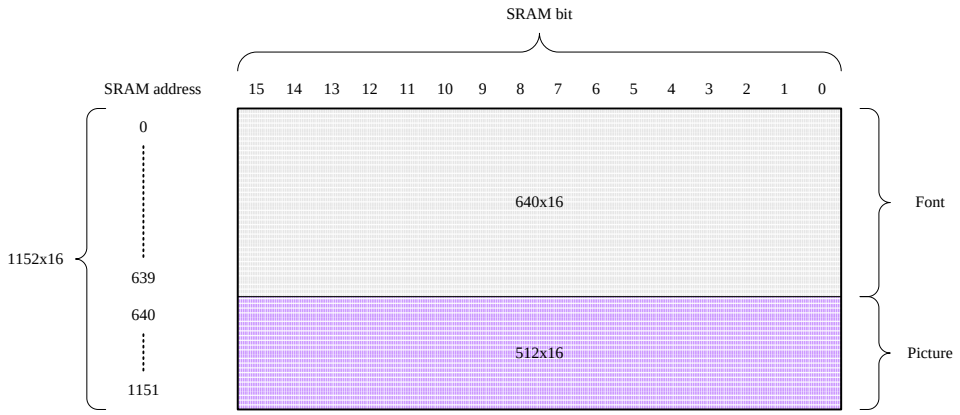


There are 2 bits for each pixel color. 00 means transparent, color 01, 10 and 11 can be set by registers: OSD_FONT_R1 (G1, B1), OSD_FONT_R2 (G2, B2) and OSD_FONT_R3 (G3, B3).

Data saved in SRAM is shown below. There are totally 64x8x10*2=640x16 bits in SRAM. Data are saved font by font. For each font, data is saved line by line. Pixel data in each line is in big Endean.

The picture is used for channel status. Each channel has picture display. The picture size in SRAM is fixed to 32x32. The display picture size is also fixed to 32x32. Same as font, it uses 2 bits for picture color. Therefore, four colors can be set by registers: OSD_PIC_R0 (G0, B0), OSD_PIC_R1 (G1, B1), OSD_PIC_R2 (G2, B2) and OSD_PIC_R3 (G3, B3). There are four pictures saved in SRAM. Data in SRAM are same as fonts. Data are saved picture by picture. In addition, in one picture, data are line by line. In one-byte data, pixel data is stored in big Endean. There are totally 32x32x4x2 = 512x16 bits in SRAM.





During system initialization, host need to write font and picture data in this SRAM. The write sequence is:

[0x758, 0x759] : Font and Picture SRAM address.

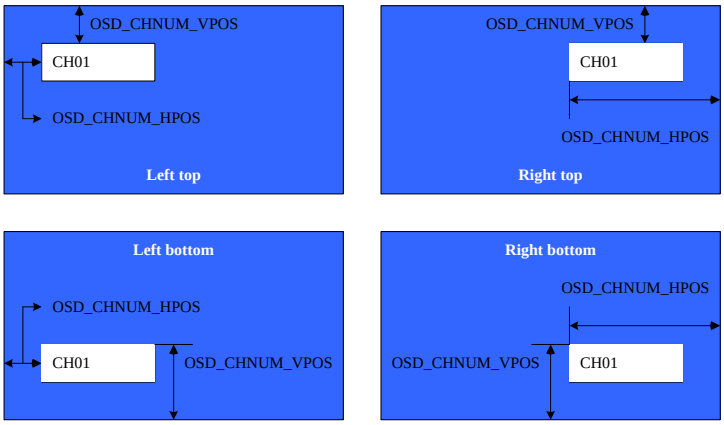
[0x75A, 0x75B] : Font and Picture SRAM data.

OSD_FRAM_DATA[0x75B] must be the last one.

CHANNEL NUMBER

For each channel, there is an 8-font channel number information. Each font is selected from 64-font table. Therefore, index for each font is 6-bit. Channel number can be enabled by setting register OSD_CHNUM_EN and OSD_EN to high. Channel number can be mixed with video data by setting register OSD_CHNUM_MIX to high. Mix percentage is 50% video plus 50% channel number. The positions for each channel are same. They can be in four positions: left top, right top, left bottom and right bottom. It is set by register OSD_CHNUM_POS. For each position, horizontal offset and

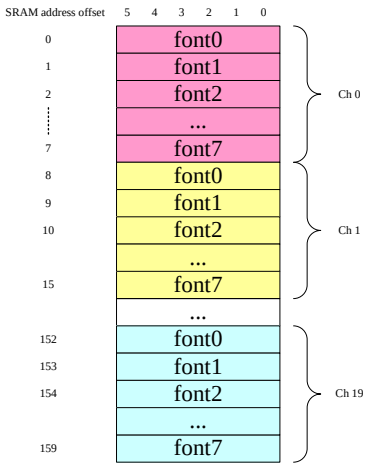
vertical offset can be set by OSD_CHNUM_HPOS and OSD_CHNUM_VPOS. For each position, the meaning for HPOS and VPOS is different because each channel size may be different.



If channel number information has less than 8 fonts, you can set the remaining font to space. Therefore, you need to put space font in the 64-font table.

The font size can be changed according to register OSD_FONT_HSIZE and OSD_FONT_VSIZE. However, remember the fonts saved in memory are always 8x10. If double size is selected, just repeat every pixel twice.

Channel information for display is saved in display SRAM. It contains 32x8x6=256x6. The sequence is channel by channel. In each channel, the sequence is font by font. SRAM is shown below:



[0x730[1]] : Channel number enable.

[0x731[0]] : Channel number mix enable.

[0x732[5:4]] : Channel number corner position.

00: left top
01: right top
10: left bottom
11: right bottom

[0x733, 0x734] : Channel number information horizontal position offset to each channel

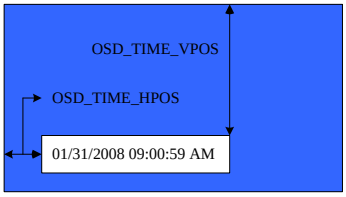
Horizontal start position. It is one pixel unit.

[0x735, 0x736] : Channel number information vertical position offset to each channel

Vertical start position. It is one line unit.

DATE AND TIME

Date and time are only display once on whole screen, not channel by channel. There are a total of 32 fonts that can be displayed, including space. Like other display information, the position for date and time can be programmed and can also be disabled and be mixed with video. The font index is saved in display SRAM. It needs 32x6 bits.



[0x730[3]] : Display time and date enable.

[0x731[2]] : Display time and date mix enable bit.

[0x73B, 0x73C] : Channel number information horizontal position offset to each channel

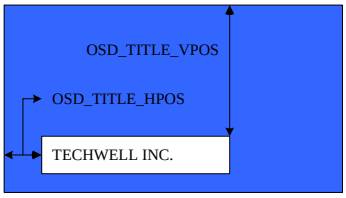
Horizontal start position. It is one pixel unit.

[0x73D, 0x73E] : Channel number information vertical position offset to each channel

Vertical start position. It is one line unit.

TITLE

Title is same as date and time. It has 32 fonts. It needs 32x6 SRAM size.



[0x730[4]] : Display time and date enable.

[0x731[3]] : Display time and date mix enable bit.

[0x73B, 0x73C] : Channel number information horizontal position offset to each channel

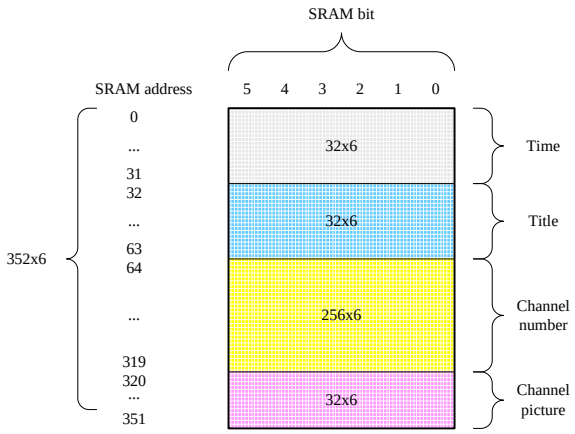
Horizontal start position. It is one pixel unit.

[0x73D, 0x73E] : Channel number information vertical position offset to each channel

Vertical start position. It is one line unit.

DISPLAY DRAM

Display SRAM includes channel number, channel picture, date/time and title information. Channel number needs 32x8x6=256 bits, channel picture needs 32x6 bits, date/time needs 32x6 bits and title needs 32x6 bits. Therefore, the total SRAM size is 352x6 bits.



During display, host need to write index data in this SRAM. The write sequence is:

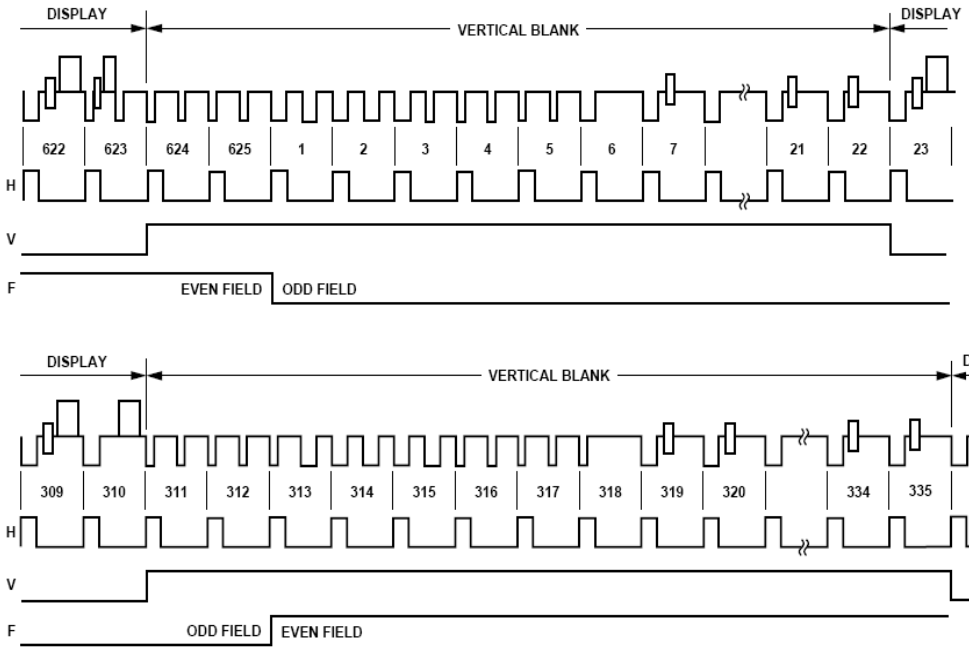
Set OSD_DRAM_ADDR, and then set OSD_DRAM_DATA.

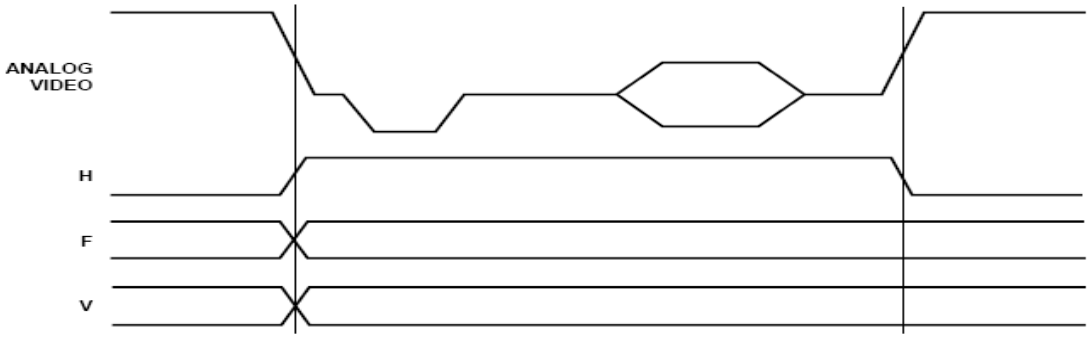
[0x75C, 0x75D] : Display SRAM address. When host write data to this SRAM, address is written first and then data. SRAM size is 244x6. Channel number size is 160x6, time/date size is 32x6, and title size is 32x6 and picture index size is 20x6.

[0x75E] : Display SRAM Data. When host write data to this SRAM, address is written first and then data. SRAM size is 244x6. Channel number size is 160x6, time/date size is 32x6, title size is 32x6 and picture index size is 20x6.

TV Encoder

TV Encoder is the module that converts all component data from scalar into a standard analog baseband television signal (CVBS) or S-Video signal, which is compatible with worldwide standards. Follow is the PAL timing.





In NTSC mode, the user needs to set:

[0x71A] = 0x00

[0x71B] = 0x01

[0x71C] = 0x08

In PAL mode, the user needs to set:

[0x71A] = 0x05

[0x71B] = 0x41

[0x71C] = 0x08

Mouse

The mouse layer in TW2880 dual monitor block is similar to the mouse layer in main display. Please consult the explanation in the main section to learn the details. The register however is different so please look it up in the data book.

[0x7B4] : mouse0, mouse1 control register.

[6]/[2] : mouse0, mouse1 enable

[5:4]/[1:0] : Mixing control

00 : 75% original pixel value / 25% mouse color mix

01 : 50% original pixel value / 50% mouse color mix

10 : 25% original pixel value / 75% mouse color mix

11 : mouse color

[0x7AC ~ 0x7AD] : Mouse0 Horizontal position

[0x7AE ~ 0x7AF] : Mouse0 Vertical position

[0x7B0 ~ 0x7B1] : Mouse1 Horizontal position

[0x7B2 ~ 0x7B3] : Mouse1 Vertical position

[0x7B5 ~ 0x7B7] : Mouse background R/G/B color

[0x7B8 ~ 0x7BA] : Mouse foreground R/G/B color

OSG

INTRODUCTION

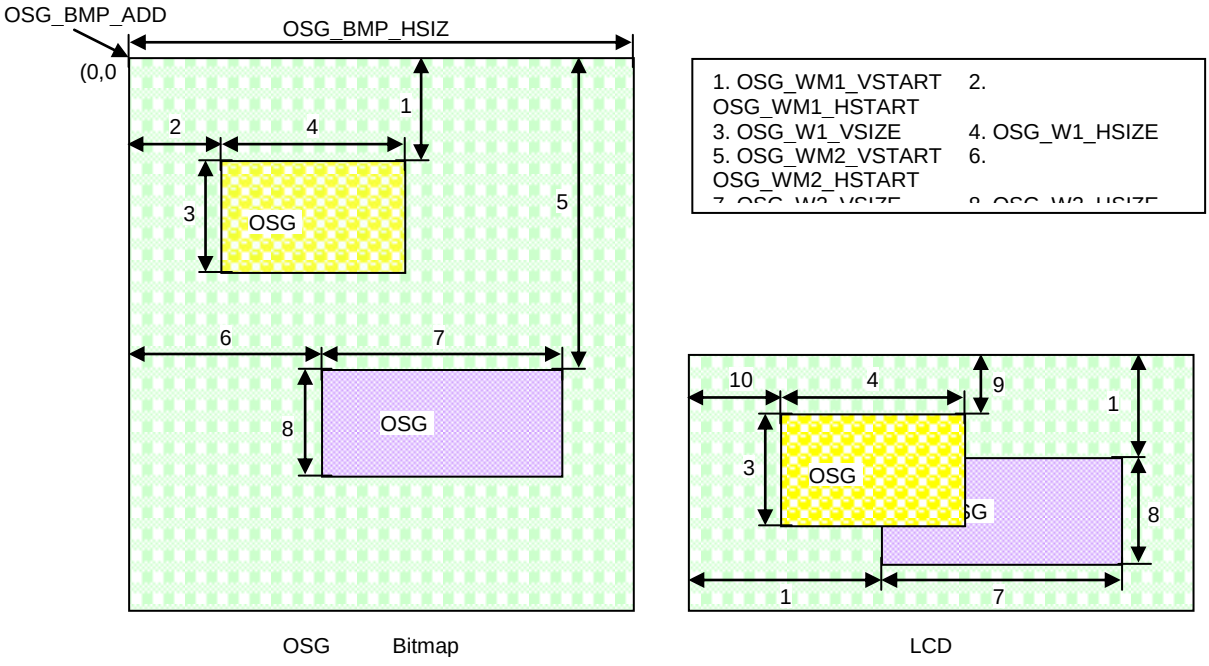
TW2880 OSG controller supports triple bitmap windows with 16 bit-per-pixel mode. Each OSG display window can support 4 sub-windows. OSG display engine supports upscale function. 16-bit color uses RGB 565 format, and it does not need color look-up-table. The input graphics from CPU can be 2 or 16 bit per pixel to reduce the amount of data writing by the CPU. The OSG writing engine automatically extends 2-bit pixel to 16-bit pixel format before writing into OSG graphic buffer in SDRAM. This is used for objects only with two colors such as fonts. Graphic data are saved in external SDRAM. The maximum size can be 8192x8192 pixels, depending on the SDRAM size.

FEATURES

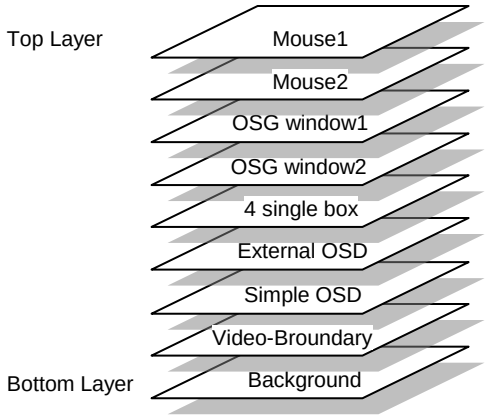
- Two windows bitmap OSG
- Each OSG window has 4 sub-windows
- Window upscale
- Bit extension from 2bit to 16 bit
- 4 Color conversion
- Blinking, transparent, alpha blending control when displaying on screen

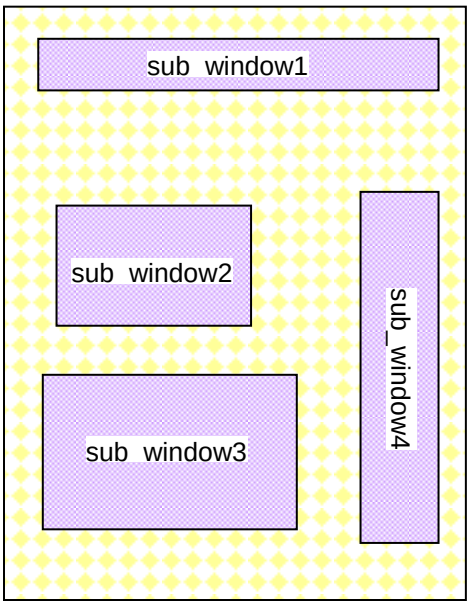
BITMAP BUFFER DISPLAY

TW2880 supports Two OSG windows. The contents showed on screen are based on Bitmap Buffer. The following picture shows the bitmap buffer and related display registers.



When displayed, OSG window 2 will overlay lower video, OSG window 1 will overlay OSG window 2. The sequence of display is:





Display modes of three windows are separate for transparent, alpha blending, blinking.

In each OSG window, there are 4 sub-windows that can select from 4 different SDRAM contents. All four windows can be different size and position, but cannot overlap.

Each window has its own position and size. To set different register, user must set register OSG_W1_SEL for window 1, OSG_W2_SEL or window 2. Each sub-window can be turn on or off by register OSG_W1_ENn or OSG_W2_ENn. Here n is sub-window number. OSG_WM1_VSTARTn, OSG_WM1_HSTARTn, OSG_W1_VSIZEn, OSG_W1_HSIZEn, OSG_WS1_VSTARTn and OSG_WS1_HSTARTn share same address.

Ex) sub_window1 for window1

[0x7EC[0]] = 1'd1 : Sub-window1 enable bit for window 1

[0x7ED[1:0]] = 2'd0 : Register selection for window 1

[0x7D4, 0x7D5] : OSG first window vertical start in bitmap memory. Unit is 1 pixel / 2 bytes.

The maximum is 8191.

[0x7D6, 0x7D7] : OSG first window horizontal start in bitmap memory. Unit is 1 pixel / 2 bytes.

The maximum is 8191.

[0x7D8, 0x7D9] : OSG first window vertical size. Unit is 1 pixel / 2 bytes. The maximum is 2047.

[0x7DA, 0x7DB] : OSG first window horizontal size. Unit is 1 pixel / 2 bytes. The maximum is 2047.

[0x7DC, 0x7DD] : OSG first window vertical start on screen. Unit is 1 pixel / 2 bytes. The maximum is 2047.

[0x7DE, 0x7DF] : OSG first window horizontal start on screen. Unit is 1 pixel / 2 bytes. The maximum is 2047.

Ex) sub_window4 for window2

[0x7EC[7]] = 1'd1 : Sub-window4 enable bit for window 2

[0x7ED[3:2]] = 2'd3 : Register selection for window 2

[0x7E0, 0x7E1] : OSG second window vertical start in bitmap memory. Unit is 1 pixel / 2 bytes.

The maximum is 8191.

[0x7E2, 0x7E3] : OSG second window horizontal start in bitmap memory. Unit is 1 pixel / 2 bytes.

The maximum is 8191.

[0x7E4, 0x7E5] : OSG second window vertical size. Unit is 1 pixel / 2 bytes. The maximum is 2047.

[0x7E6, 0x7E7] : OSG second window horizontal size. Unit is 1 pixel / 2 bytes. The maximum is 2047.

[0x7E8, 0x7E9] : OSG second window vertical start on screen. Unit is 1 pixel / 2 bytes. The maximum is 2047.

[0x7EA, 0x7EB] : OSG second window horizontal start on screen. Unit is 1 pixel / 2 bytes. The maximum is 2047.

ALPHA BLENDING

Register OSG_BLEND_MODE controls the alpha blending function. Mode "00" will disable blending. The other three modes enable alpha blending. When mode "01" enabled, pixels with color OSG_BLEND_COLOR will be mixed with lower layer image, and the other pixels will overwrite lower layer image. When mode "10" enabled, pixels with color OSG_BLEND_COLOR will overwrite lower layer image, and the other pixels will mixed with lower layer image. When mode "11" is set, all the pixels will mixed with lower layer image. The blending function will be: Video_data * alpha + osg_data * (1 – alpha). There are four big registers to program alpha. Each window has two blending colors.

[0x7C4[3:2], 0x7C5[3:2]] : Alpha blending mode window 1/window 2.

00 : alpha blending disable

01 : alpha blending in pixels with color OSG_BLEND_COLOR

10 : alpha blending in pixels with color not equals to OSG_BLEND_COLOR

11 : alpha blending in all pixels

[0x7C6[7:4]] : Alpha blending alpha number for OSG window 2.

The output image will be: Video_data * alpha + osg_data * (1 – alpha).

Here alpha = OSG_ALPHA / 16. Maximum is 15.

[0x7C6[3:0]] : Alpha blending alpha number for OSG window 1.

The output image will be: Video_data * alpha + osg_data * (1 – alpha).

Here alpha = OSG_ALPHA / 16. Maximum is 15.

[0x7C8, 0x7C9] : Alpha blending color for OSG window 1.

[0x7CA, 0x7CB] : Alpha blending color for OSG window 1.

[0x7CC, 0x7CD] : Alpha blending color for OSG window 2.

[0x7CE, 0x7CF] : Alpha blending color for OSG window 2.

BLINKING

When 0x105 bit [5:4] is other than 0x0, the pixel blinking feature is on. This feature allows the pixel to switch back and forth between a foreground color and a background color. Both colors can be programmed. If bit[5:4] equals 0x1, the pixels with color OSG_BLINK_COLOR will blink. If bit[5:4] equals 0x2, the pixels with colors not equals to OSG_BLINK_COLOR will blink. If bit[5:4] equals 0x3, all pixels will blink. The blink speed is determined by programming the OSG_BLINK_FRAME's value.

[0x7C4[5:4], 0x7C5[5:4]] : Blink enable. Blink frequency is determined by OSG_BLINK_FRAME

00 : blink disable

01 : blink in pixels with color OSG_BLINK_COLOR

10 : blink in pixels with color not equals to OSG_BLINK_COLOR

11 : blink in all pixels

[0x7C7[1:0]] : OSG blinking frequency control

00 : blinking on each 32 frames

01 : blinking on each 16 frames

10 : blinking on each 8 frames

11 : blinking on each 4 frames

[0x7D0, 0x7D1] : Blinking foreground color for OSG window 1.

[0x7D2, 0x7D3] : Blinking foreground color for OSG window 2.

TRANSPARENT

TW2880 has a fixed transparent color: 0xFFFF. To enable transparent, set bit OSG_TRANS_EN to high. The pixels with color 0xFFFF will not be display on screen. The lower layer image will be displayed. If OSG_TRANS_EN is set to low, all the pixels not in OSG window will display white.

[0x7C4[1], 0x7C5[1]] : OSG transparent enable window 1/window 2.

RGB FORMAT

TW2880 Support for 16 bit RGB Format.

Mode	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	G						R					B				
1	B	G					R					B				
2	A	G					R					B				
3	B	A	G				R					B				

In mode 0, blinking and blending can be done by color. In mode 1 to 3, blinking and blending can be done by pixel.

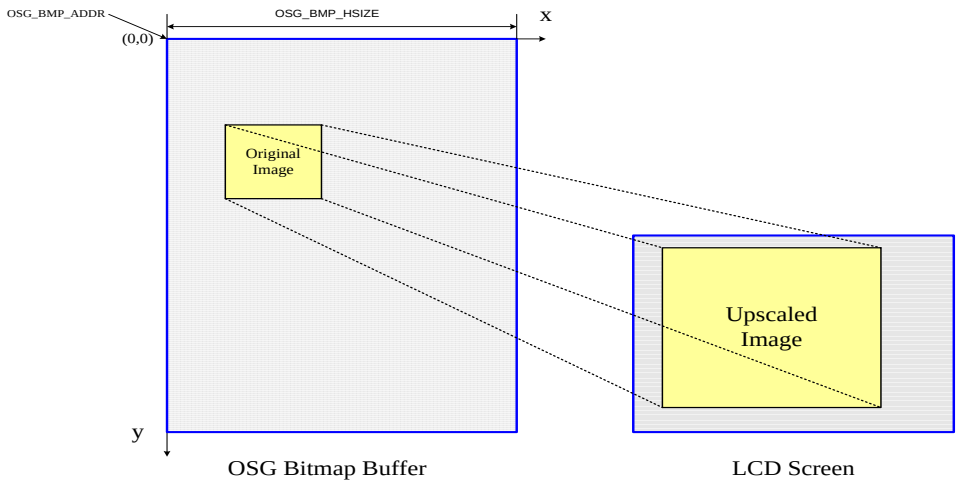
If OSG_FORMAT_RG is set to high, the data in 16-bit mode is as follows:

Mode	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	R					G						B				
1	B	R					G					B				
2	A	R					G					B				
3	B	A	R					G					B			

[0x7C7[3:2]] : OSG 16 bit data format
00 : RGB 565 format
01 : Blinking bit + RGB 555 format
10 : Alpha bit + RGB 555 format
11 : Blinking and alpha bit + RGB 554 format

UPSCALE

OSG can display upscaled bitmap image to screen. That means bitmap in SDRAM can be a small image, and display image can be upscaled image. Two windows can have different upscale ratios. Upscale ratio is determined by original image size to target image size. Original image size is determined by register [0x7EE] to [0x7F5]. Target image size is determined by LCD screen size. Upscale function can be turned on or off by register [0x7C4], [0x7C5] bit 6. OSG_WM_HSTART, OSG_WM_VSTART, OSG_WS_HSTART, OSG_WS_VSTART, OSG_W_HSIZE and OSG_W_VSIZE are setting for original size, not for target size.



Upscale function is used in bandwidth limitation case. User can use small size bitmap but display it in bigger LCD monitor. For example, original bitmap is 640x480, and LCD monitor is 1920x1080. Then OSG_VSIZE_DN should be 479, and OSG_HSIZE_DN should 639.

[0x7C4[6], 0x7C5[6]] : Upscale enable window1/ window 2.

[0x7EE, 0x7EF] : OSG original vertical size minus 1 for window1.

[0x7F0, 0x7F1] : OSG original horizontal size minus 1 for window1.

OSG_HSIZE_DN1+1 must be times of 4.

[0x7F2, 0x7F3] : OSG original vertical size minus 1 for window2.

[0x7F4, 0x7F5] : OSG original horizontal size minus 1 for window3.

OSG_HSIZE_DN1+1 must be times of 4.

Single Box

TW2880 provides 4 single boxes that can be used for highlighting portion of the display. The effects include a single box or box cursor, a masking box and a box blending with a plane color. Each box has programmable location and sizes and controlled by BOX_HL (0x786 - 0x78D), BOX_HW (0x78E - 0x795), BOX_VT (0x796 – 0x79D) and BOX_VW (0x79E – 0x7A5) registers. The BOX_HL is the horizontal location of box with 2-pixel unit and the BOX_HW is the horizontal size of box with 2-pixel unit. The BOX_VT is the vertical location of box with 1 line unit and the BOX_VW is the vertical size of box with 1 line unit.

The display option is controlled by Control registers (0x780 – 0x783). BOX_PLNEN bit in these register enables each plane color and its color is defined by the BOX_PLNCOL (0x7A9 – 0x7AB) register. Mixing is also controlled by these

registers. The color of box boundary is enabled via the BOX_BNDEN bit in the control registers and its color is defined by the BOX_BNDCOL (0x7A6 – 0x7A8) registers.

In cases where several boxes have the same region specified, there will be a conflict of what to display for that region. Generally, the TW2880 defines that box 0 has priority over box 3. So if a conflict happens between more than 2 boxes, box 0 will be displayed first as top layer and box 1 to box 3 are hidden beneath that are not supported for pop-up attribute unlike channel display.

[0x780 ~ 0x783] : Sbox0 ~ Sbox3 control register.

[4] : Boundary line enable
[3] : Box plane enable
[2] : Blinking enable

[1:0] : Mixing control

00 : 75% original pixel value / 25% plane (boundary) color mix

01 : 50% original pixel value / 50% plane (boundary) color mix

10 : 25% original pixel value / 75% plane (boundary) color mix

11 : plane (boundary) color

[0x784, 0x785] : Sbox0 ~ Sbox3 horizontal & vertical line control register.

00 : 1 line

01 : 2 line

10 : 3 line

11 : 4 line

[0x786 ~ 0x78D] : Sbox0~Sbox3 Left Horizontal point

[0x78E ~ 0x795] : Sbox0~Sbox3 Right Horizontal point

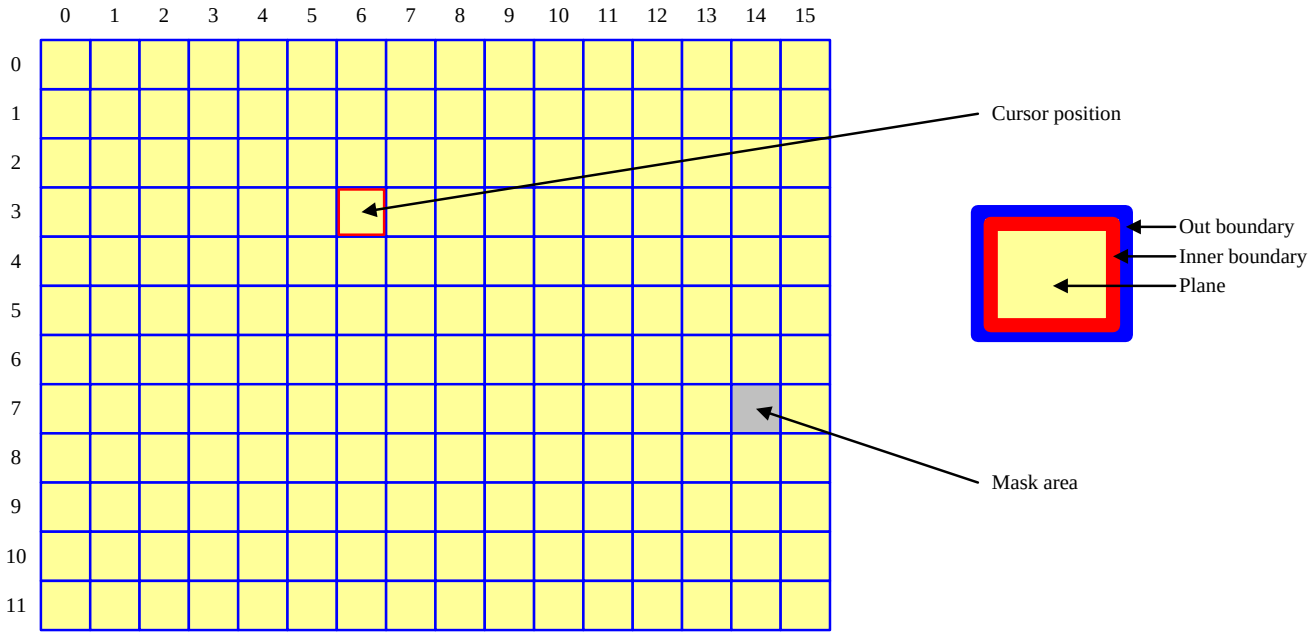
[0x796 ~ 0x79D] : Sbox0~Sbox3 Top Vertical point

[0x79E ~ 0x7A5] : Sbox0~Sbox3 Bottom Vertical point

[0x7A6 ~ 0x7A8] : Boundary R/G/B color

[0x7A9 ~ 0x7AB] : Plane R/G/B color

Motion Box



TW2880 supports an array boxes layer that has a programmable cell size up to 16x12. This box array can be used to make table menu or display motion detection information available to the user. When motion detection mode is enabled, user must set horizontal cell number to 15 and vertical cell number to 11. This layer is available to all live channels so most of the time, users need to program 16 set of registers. The subsequent explanation of the function only talks about the first channel. However, it is easy to duplicate on other channels as well. To use it first you need to determine the mode and enable it.

Because Main & DMON use the same motion box control register address,

So need setting 0X54B[7] register ("1" : DMON setting, "0" : Main setting).

(1) Program 0x54B[7] to determine main & DMON setting . 1 = DMON setting 0 = Main setting.

(2) Program 0x550[6] to determine display mode. 1 = motion display mode.

(3) Program 0x550[7] to enable MD box.

(4) Program 0x70D[7] to enable DMON MD box.

Cell Composition

A motion cell is composed by four elements: out boundary, inner boundary, mask and plane. Out boundary and plane color make up the usual overlay color. Inner boundary and mask is used to show special event like cursor and motion. To determine the color of these elements, user need to program:

(1) Program 0x493,0x494,0x495 to determine out boundary R, G, B color.

(2) Program 0x496,0x497,0x498 to determine inner boundary R, G, B color.

(3) Program 0x499,0x49A,0x49B to determine mask R, G, B color

(4) Program 0x49C,0x49D,0x49E to determine plane R, G, B color

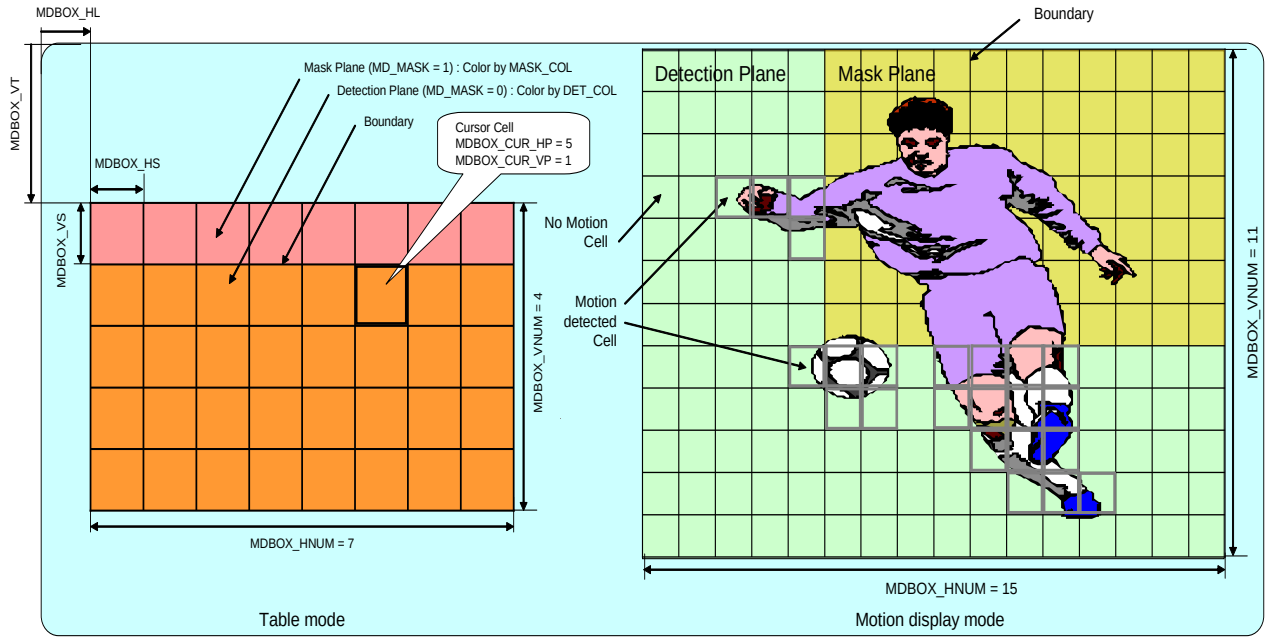
(5) The boundary can be enabled by programming 0x500[4], MDBOX_BNDEN

The cursor cell is enabled by the MDBOX_CUREN 0x550[5] register and the displayed location is defined by the MDBOX_CURHP 0x5F8 and MDBOX_CURVP 0x48B registers. Its color is a reverse color of cell boundary. It is useful function to control motion mask region.

Motion box positions and sizes are controlled by registers. To overlay mask information and motion result on video data properly, the scaling ratio of video should be matched with motion box size.

For each MD array, the number of row and column cells is defined via the MDBOX_HNUM(0x5E8[3:0] ~ 0x5EF[7:4]) and MDBOX_VNUM (0x5F0[3:0] ~ 0x5F7[7:4]) registers. The horizontal and vertical location of left top is controlled by the MDBOX_HL (0x568 ~ 0x587) register and the MDBOX_VT (0x558 ~ 0x5A7) registers. The horizontal and vertical size of each cell is defined by the MDBOX_VS (0x5C8 ~ 0x5E7) registers and the MDBOX_HS (0x5A8 ~ 0x5C7) registers. Therefore, the whole size of MD arrayed box is same as the sum of cells in row and column.

The plane of MD arrayed box is separated into mask plane and detection plane. The mask plane represents the cell defined by MD_MASK (0x800 ~ 0xBD7) register. The detection plane represents the motion detected cell excluding the mask cells among whole cells. The mask plane of MD arrayed box is enabled by the MDBOX_MSKEN (0x550[3] ~ 0x55F[3]) register and the detection plane is enabled by the MDBOX_DETEN (0x550[2] ~ 0x55F[2]) register. The color of mask plane is controlled by the MASK_COL register and the color of detection plane is defined by the DET_COL register. The mask plane of MD arrayed box shows the mask information according to the MD_MASK registers automatically and the additional narrow boundary of each cell is provided to display motion detection via the MDBOX_DETEN register and its color is a reverse cell boundary color. The plane can be mixed with video data by the MDBOX_MIX (0x550[1:0] ~ 0x55F[1:0]) register. Even in the horizontal / vertical mirroring mode, the video data and motion detection result can be matched via the MDBOX_HINV and MDBOX_VINV registers.

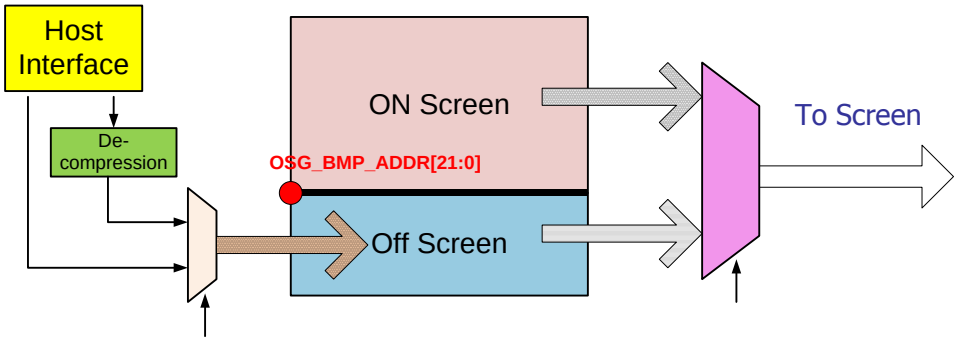


Section 6: OSG and Simple OSD

Introduction

TW2880 provides a very powerful graphics / video overlaying tools to let users create desired visual effect. The OSG controller supports three bitmapped-based windows with 16 bit-per-pixel mode, pixel manipulation tools like color expansion and BitBilt functions. TW2880's OSG can particular useful in preparing animated menu and map oriented operating index window. The following is a detailed explanation of the whole unit.

Programming Model



The basic operating structure is illustrated in the above diagram. The display memory is divided into two parts: on-screen memory and off-screen memory. On-screen memory is managed and used by the video windows. The off-screen memory is shared by many units like mouse, OSG and bitmapped OSD. The OSG unit will only care for the bitmap data, which is used to form menu and graph later on. The bitmap date is handled by Host and it is stored into off-screen buffer by host write. The bitmap data can be preprocessed into a compressed format at the host side to save storage space and transfer time. Therefore, after receiving by the host interface it will go through a decompression unit before it gets stored into the SDRAM.

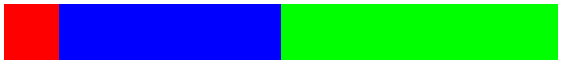
The screen data is composed from two or more memory read agents on a pixel-by-pixel basis. The final multiplexer unit will determined which layer is used based on user input.

Compression Format

TW2880 OSG unit supports compression using Run Length Encoding (RLE) format for the bitmap data. The compression format starts with command bit (1/0) which indicates whether following value is data or counter. If the command bit is '0', then the coming value is new data; if the command bit is '1', then the following value is a count value which represents how many times the data will repeat itself. The command must be occupied 1 bit. The data is 2bit or 16bit depends on the pixel expansion used. The count is changeable from 2bit to 16bit.

0	Data	1	Count	0	Data	0	Data ...

For example, if GRB color mode is used as RGB format, and an image is 10 pixels x 1 line like below,

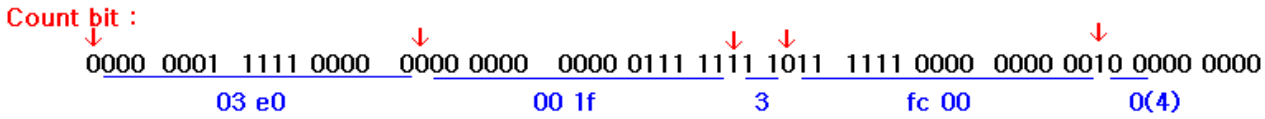


COLOR	DATA VALUE
Red	0x03e0
Blue	0x001f
Green	0xfc00

```
Original_Bitmap_data[ 10 pixels x 1line ] = {
                                    0x03e0,    //1 red pixel
                                    0x001f, 0x001f, 0x001f, 0x001f, // 4 Blue pixels
                                    0xfc00, 0xfc00, 0xfc00, 0xfc00, 0xfc00 //5 Green pixels
                                    };
```

The data is encoding below when count bit is 2.

```
Rlc_data[] = { // {command bit, data bit or count bit }
               { 0, 03e0 } // New data 0x03e0
               { 0, 001f } // New data 0x001f
               { 1, 3 }    // repeat 3 times of 0x001f
               { 0, fc00 } // New data 0xfc00
               { 1, 4 } : // repeat 4 times of 0xfc00
             } ;
```

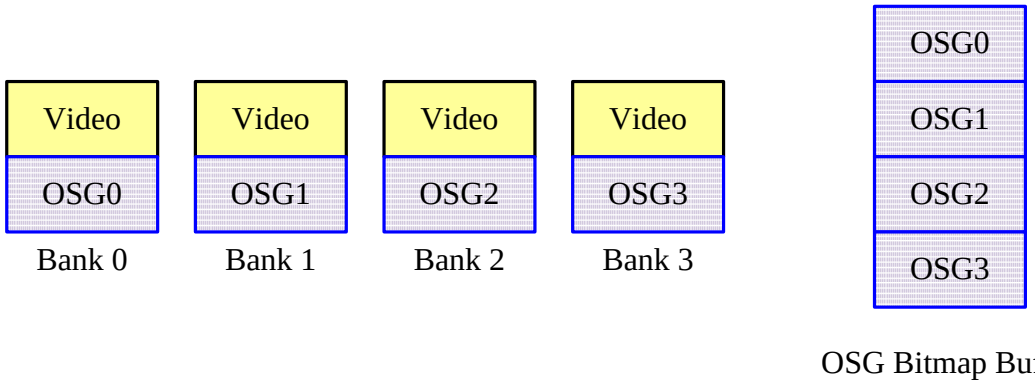


So, Rle_data[] = { 0x01, 0xf0, 0x00, 0x07, 0xfb, 0xf0, 0x02, 0x00 };

If count value is 0, then the count value is its complement plus one. For example, the count value "0" will be stored as 4 when count bit is 2bit. Number of count bit will affect the overall size of compressed image. Techwell provides to customer TW2880 User Builder tool for Windows to make compressed bitmap from the original. The tool can search and determine the minimum count bit of each data and generate compressed image in binary or text file from bitmap icons or fonts. Check the user's manual for more detailed information about the tool.

OSG Bitmap Buffer Start Address Calculation

TW2880's display memory management is automatically handled by hardware. For the main display or dual display to run properly, user does not need to specify anything other than the starting addresses. Based on the next diagram, the memory in yellow portion is often referred to as "on-screen memory" since your will see the content in your display. The rest is called "off-screen" memory as you should never see that content on display and it is used as storage for many things. The split point between the two buffers is determined by OSG bitmap buffer starting address.



The off-screen memory is linked by hardware and can be viewed as a continuous block. Take the most commonly used 128Mbit x2 configuration (64 bit) for example, let's calculate the off-screen memory size if the main display is in HD (1920x1080). Since the memory can be structured as a 2048x2048 pixel x4 memory array, the memory left will be:

Memory left will be: (2K x 2K x2 - 1920x1080x2) x 4 banks = 16965632 bytes

OSG bitmap buffer start address is the start bitmap writing position in the off-screen memory. Following is an example OSG bitmap data should be assigned to the rest of the area after video image (Figure 49).

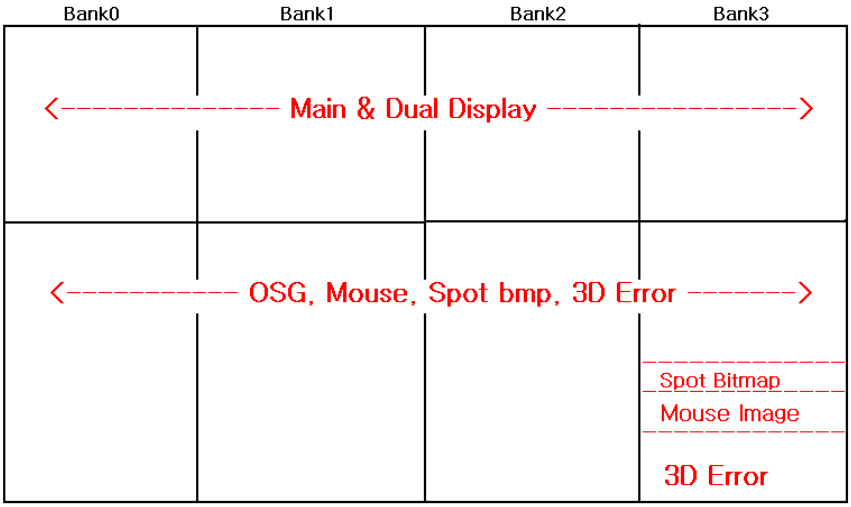


FIGURE 49. DISPLAY MEMORY MAP

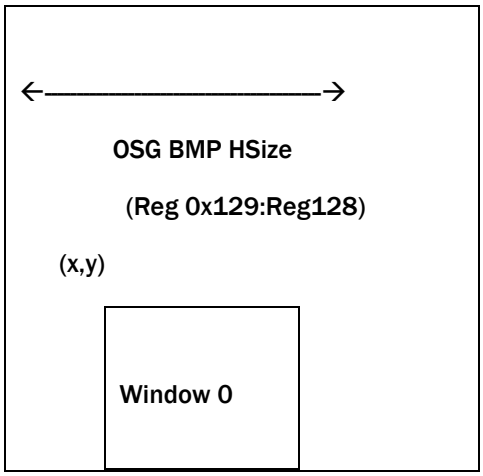
For example, the main display resolution is 1080p, and the dual monitor display resolution is NTSC (CVBS) as Figure 50.

This is recommended memory map arrangement for the display. 3D Error area must be located at the Bank3. OSG memory area must be started at Bank0. OSG module links all of the OSG area in each bank to virtual continuous address space.

OSG bitmap buffer starting address = (Main display hsize + Dual display hsize) * (main display vertical resolution)

OSG bitmap buffer starting address Register Value (4 pixel unit) = (1920+1024)x1080 /4

(0,0) –> OSG bitmap buffer starting address



< OSG Bitmap Memory >

* OSG BMP Hsize must be times of 128.

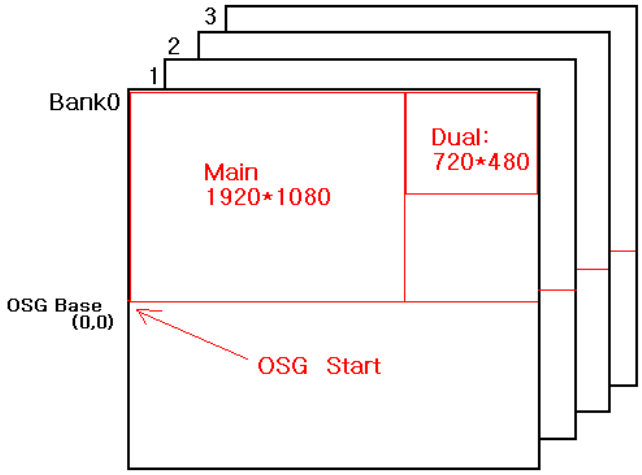


FIGURE 50. OSG BITMAP BUFFER STARTING ADDRESS

Writing Bitmap Data

Basically, the host sends bitmap data to TW2880 by the parallel interface, and the OSG module writes the data into the memory. The OSG display windows in either main or dual monitor can be overlaid on top of the video.

The writing bitmap data can be 16bit or 2bit. The 2-bit data mode is useful for using less then 4 colors such as simple characters. The 2-bit data must expand to 16-bit data by 2-bit color table while writing to SDRAM. The 2bit color look-up table registers are OSG_CON_TAR1 to OSG_CON_TAR4, and OSG_PIXEL_BIT Reg 0x102[0] must be set to 1 in the 2bit data mode case.

"00" -> OSG_CON_TAR1 (Reg 0x11A:Reg 0x11B)

"01" -> OSG_CON_TAR2 (Reg 0x11E:Reg 0x11F)

"10" -> OSG_CON_TAR3 (Reg 0x122:Reg 0x123)

"11" -> OSG_CON_TAR4 (Reg 0x126:Reg 0x127)

There are four ways of writing the data to the memory inside the chip. Each method can be found in our reference code source in “tw2880/osg.c.”

1. OSG module directly writes the data to the memory:

   void OsgLoadBmp2(U32 addr, U16 dx, U16 dy)

2. Host DMA module directly writes the data to the memory:

   void OsgLoadBmpByDMA(U32 saddr, U16 dx, U16 dy, U32 hand, U32 burst)

3. Host DMA module passes the data to the OSG module, then the OSG module write the data to the memory. Reg 0x17E[1] OSG_FROM_DMA, and Reg 0x17E[2] OSG_FROM_DMA_OSG need proper setting:

   void OsgLoadBmpByDMA2OSG(U32 addr, U16 dx, U16 dy, U32 hand, U32 burst)

4. DRAM Access writes the data:

   void WriteOsgDataToMemory(char *pTbl)

Basic process of the OSG bitmap data writing (# 1 case):

Reset OSG writing module (Reg 0x20C[4])

Set control register (including count bit, data bit, Big Endian/Little Endian, RGB format, etc.)

Reg 0x102 OSG_Mode 1

Reg 0x103 OSG_Mode 2

Reg 0x104 OSG_Mode 3

Set the image width and height

Set the color lookup table if 2-bit data mode

Set destination position(dx, dy)

Enable write start (Reg 0x100[0])

Write chunk of the image data through the data port (OSG_HOST_DATA Reg 0x13a).

Wait until writing is done. (Reg 0x101[0] OSG_WR_BUSY)

Done

Host can write bitmap data in 8-bit data bus or 16-bit data bus along the H16B pin(AB25 p_h16b_en). When H16B pin is high, Host Interface operates in 16-bit bus mode (p_hdata[15-0]). However, all the TW2880 registers are basically in the structure of 8-bit data bus except the OSG_HOST_DATA register. Be aware that cpu side also has bus selection.

TW2880 supports big-endian or little-endian. Our reference source code that provided is little endian as the default in OSG part.

In case of using OSG module for the bitmap writing, host should wait when OSG_WR_BUSY(Reg 0x101[0]) is enabled, otherwise image will be displayed broken. Alternatively, the host can use WAIT signal(AA24 p_wait_st) instead of monitoring OSG_WR_BUSY register. In this case, OSG_WAIT_PINEN (Reg 0x162[4]) must be set to 1.

When host uses Host DMA module, it requires DMAACK, DMAREQ signals as well as all of the host interface signals. TW2880 Host DMA interface module is verified with ARM CPU (Samsung s3c2410).

Visual Effect Walk Through

TW2880 OSG has block fill, block transfer, color conversion, bitblit, and selective overwrite for managing bitmap data. OSG Base Address is position (0,0) for these functions.

BLOCK FILL

This function is that OSG module draws a rectangular block in the OSG buffer with a single color.

Assign the destination position (dx, dy) : (Reg 0x135[4:0]:Reg 0x134, Reg 0x133[4:0]:Reg 0x132), horizontal

length(w),and vertical length(h). Set the block fill color (Reg 0x115:Reg 0x114).

U32 OsgBlockFill(U16 dx, U16 dy, U16 w, U16 h, U16 color)

BLOCK TRANSFER

The module copies a rectangular area from one location to another location in the bitmap buffer. When the font table and icons has been downloaded in the memory, this function can be used to copy some char or icon from the source position (sx, sy), area (w, h) to the destination area(dx, dy).

U32 OsgBlockTransfer(U16 sx, U16 sy, U16 w, U16 h, U16 dx, U16 dy)

COLOR CONVERSION

Changing color during block transfer or bitmap data writing.

TW2880 OSG module has 4-source color table and 4-target color conversion table.

BITBLIT AND SELECTIVE OVERWRITE

Bitblit function is useful if it need color bit operation such as ADD, OR, XOR between source area and destination area for block transfer. There are 256 operations (OPCODE Reg 0x104, Default:0xCA)

The Bitblit function needs three objects: source, destination and mask.

Sometimes, in the application, you don't want to write specific colors among the source color during the block transfer. This is called selective overwrite. TW2880 has 4 Selective Overwrite color table for this kind of operation.

OSG_OVERWRITE_COLOR1	Reg 0x117 : Reg 0x116
OSG_OVERWRITE_COLOR2	Reg 0x15d : Reg 0x15c
OSG_OVERWRITE_COLOR3	Reg 0x15f : Reg 0x15e
OSG_OVERWRITE_COLOR4	Reg 0x161 : Reg 0x160

If OSG_OVWR_MODE register is "00" and OPCODE is "0xCA", then selective overwrite function can work.

Set the colors according to whatever you don't want to transfer in, up to 4 colors. Then, the source pixel color is the same color as one of the selective overwrite color table, then it does not copy source color, leaving the original destination color.

Here in below table, destination 1 is destination color before writing, and destination 2 is destination color after writing as expected.

BIT NO	MASK	DESTINATION 1	SOURCE	DESTINATION 2
0	0	0	0	0
1	0	0	1	1
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	0
6	1	1	0	1
7	1	1	1	1

destination2 = opcode (mask, destination1, source)

For example, if you want to get: destination2 = mask & source & destination1.

The look up table should like this:

BIT NO	MASK	DESTINATION1	SOURCE	DESTINATION2
0	0	0	0	0
1	0	0	1	0
2	0	1	0	0
3	0	1	1	0
4	1	0	0	0
5	1	0	1	0
6	1	1	0	0
7	1	1	1	1

From the table, BitBlit operation code should be 0x80.

In the equation, mask is determined by register OSG_OVWR_MODE (Reg 0x102 bit [2:1].

OSG_OVWR_MODE	MASK BIT	FUNCTION
00	Mask is decided by Selective overwrite table: Mask=1 : Source color is the same as one of Selective overwrite table color. Mask=0 : Source color is not the same as any of the Selective Overwrite Table color.	Selective overwrite determined by OPCODE Register
01	Mask is decided by 0xFFFF	Selective overwrite determined by

OSG_OVWR_MODE	MASK BIT	FUNCTION
	(Transparency color) Mask=1 : Source color equals to 0xFFFF. Mask=0 : Source color does not equal to 0xFFFF.	OPCODE Register
10	Mask always 0	Selective overwrite determined by OPCODE Register
11	0	No Bitblit function, Destination equals to source (Bypass)

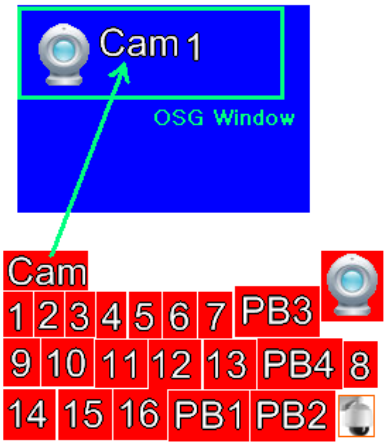
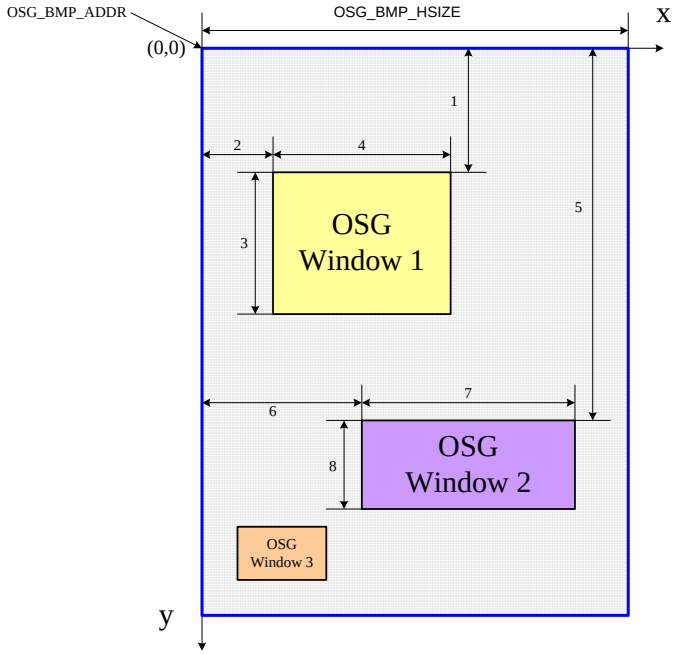


FIGURE 51. OSG BITMAP BUFFER

Figure 51 shows that a background character color is changed using the selective overwrite while the block transfer bitmap images to an OSG display window area as an example.

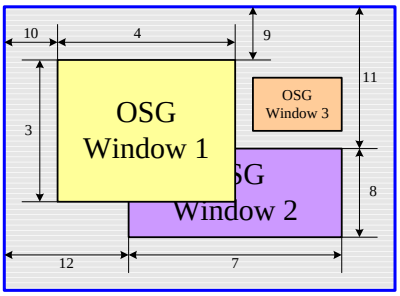
OSG Window Display

There are 3 OSG Window layers in the Main display OSG. Each window layer has 8 sub windows that cannot overlay each other in the same layer.



OSG Bitmap Buffer

1. OSG_WM1_VSTART	5. OSG_WM2_VSTART
2. OSG_WM1_HSTART	6. OSG_WM2_HSTART
9. OSG_WS1_VSTART	11. OSG_WS2_VSTART
10. OSG_WS1_HSTART	12. OSG_WS2_HSTART
3. OSG_W1_VSIZE	7. OSG_W2_VSIZE
4. OSG_W1_HSIZE	8. OSG_W2_HSIZE



LCD Screen

You can find the reference source code in "/tw2880/osg.c."

Set OSG window memory position:

void SetOsgDispMemoryAddr(U8 winno, U8 subno, U16 x, U16 y)

winno : OSG window layer no (0-2)
subno : Sub window no (0-7)
x : memory horizontal position (1 pixel unit)
y : memory vertical position
* Memory position (x,y) is based on OSG bitmap buffer in the memory.

Set OSG window display position and size:

void OsgWindowInit(U8 winno, U8 subno,U16 x, U16 y, U16 w, U16 h)

x : display horizontal position (1 pixel unit)
y : display vertical position
w : display width
h : display height
* Display position (x,y) is based on actual display screen.

Enable or disable OSG window:

void ShowOsgWindow(int winno, int subno, int on)

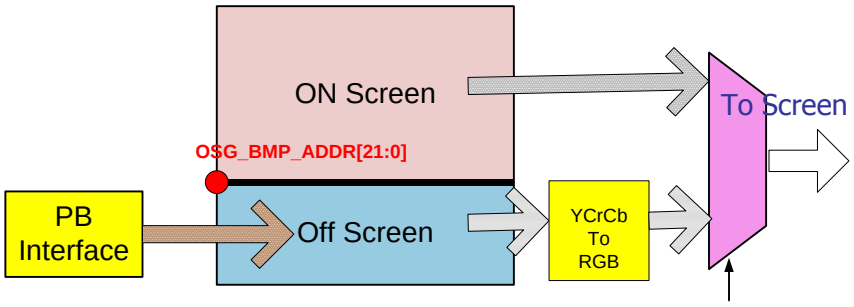
on : 1:OSG window enable, 0:Disable

External OSG Mater mode

External OSG Slave mode

TW2880 can use Playback ports to write OSG menu to off-screen memory instead of using host to write OSG menu to off-screen memory.

PROGRAMMING MODEL



The basic operating structure is illustrated in the above diagram. OSG menu can be loaded to the off-screen memory using the playback ports. User can program registers EXT SLAVE(6BDh~6BFh) to setup memory write address. EXT SLAVE[23] enables external OSG off screen memory write. EXT SLAVE[22:21] indicate the memory pages. EXT SLAVE[20:0] indicates the write start address in the memory page.

For EXT OSG display, user need to program registers OSG_BMP_ADDR to setup memory read address. OSG_BMP_ADDR[23] enable external OSG off screen memory read. OSG_BMP_ADDR[21:20] indicate the memory pages. OSG_BMP_ADDR[20:0] indicate the read start address in the memory page.

YCRCB TO RGB

If input data format is BT656 then data need to be converted to RGB format for OSG to display. User can set register 17E[4] to enable YCrCb to RGB converter. TW2880 only has OSG window 1 support YCrCb to RGB conversion. OSG window 2 and 3 are only for RGB format.

ON SCREEN MEMORY DISPLAY

User can also use OSG windows as pop up windows to display on screen memory content. To display on screen memory content, user can program OSG_BMP_ADDR point to the on screen memory address and enable YCrCb to RGB converter. To sync with the display video frames, user need to set register 17E[3] to one.

Simple OSD

Introduction

When OSG doesn't have enough memory bandwidth to display in the main display, Simple OSD is useful because all font tables and display RAM are stored in local SRAM of the chip. However, font size, position control and the number of characters to display are very limited. This document will describe the OSD controller architecture and its usage. Since the OSD controller is used in main LCD display, Dual monitor, SPOT, and REC port, the method used to calculate the size of the SRAM and the SRAM address is about the same.

Architecture

The OSD controller consists of four main functions as shown in Figure 52:

- put the unique channel number for each video input to show its identity
- put the picture for each channel to show the camera's working condition
- Put 32 characters to show the time/date on the display for all channels
- Put 32 characters to show the title on the display for all channels

For bandwidth saving, OSD uses SRAMs to store its display content and the fonts for characters mapping. All information can be selected from 64 fonts and 4 pictures saved in SRAM. Font width can be selected from 6, 8, 12 or 16. Font height can be selected from 8, 10, 16 or 20. Picture size is fixed to 32x32.

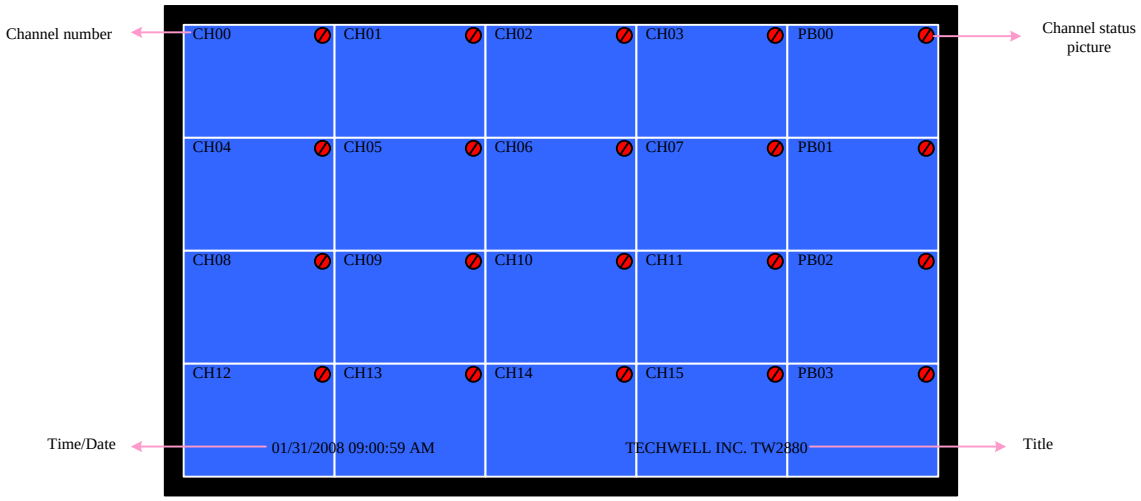
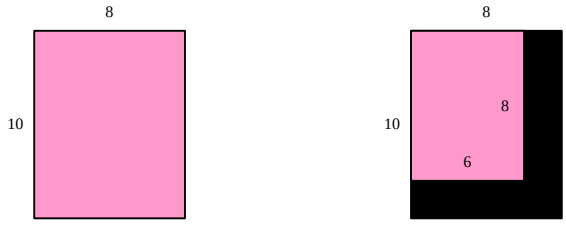


FIGURE 52. OSD FUNCTIONS AS SHOWN ON THE DISPLAY

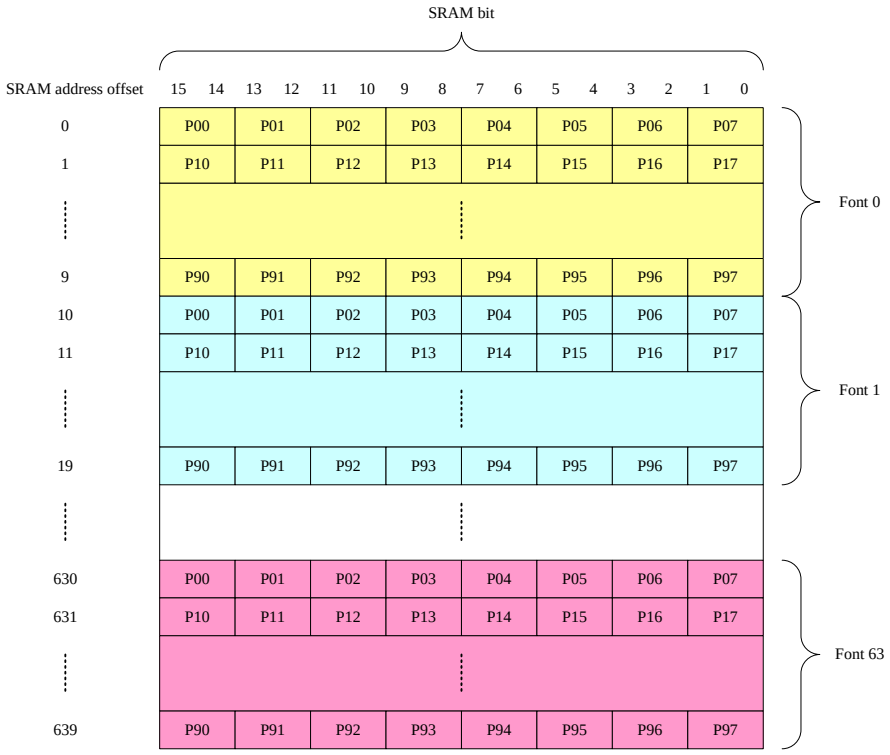
Fonts and SRAM Memory Size Requirement

There are a total of 64 fonts that can be saved in SRAM. The font size saved in SRAM is fixed to 8x10. However, displayed font size can be changed. Horizontal can display four sizes: 6, 8, 12 or 16. Size 12 or 16 are doubled from size 6 or 8. If 6 or 12 are selected, fonts saved in SRAM must have small size. However, additional two pixels must be saved in SRAM. Vertical can display four sizes: 8, 10, 16 or 20. Size 16 or 20 are doubled from size 8 or 10. If 8 or 16 are selected, fonts saved in SRAM must have small size. However, additional two lines must be saved in SRAM. The following picture shows SRAM data. For 8x10 font, all SRAM data is valid. For 6x8 font, only 6x8 area is valid. Black area is for dummy data.



There are 2 bits for each pixel color. 00 means transparent, color 01, 10 and 11 can be set by registers: OSD_FONT_R1 (G1, B1), OSD_FONT_R2 (G2, B2) and OSD_FONT_R3 (G3, B3).

Data saved in SRAM is shown below. There are totally 64x8x10*2=640x16 bits in SRAM. Data are saved font by font. For each font, data is saved line by line. Pixel data in each line is in big endian.



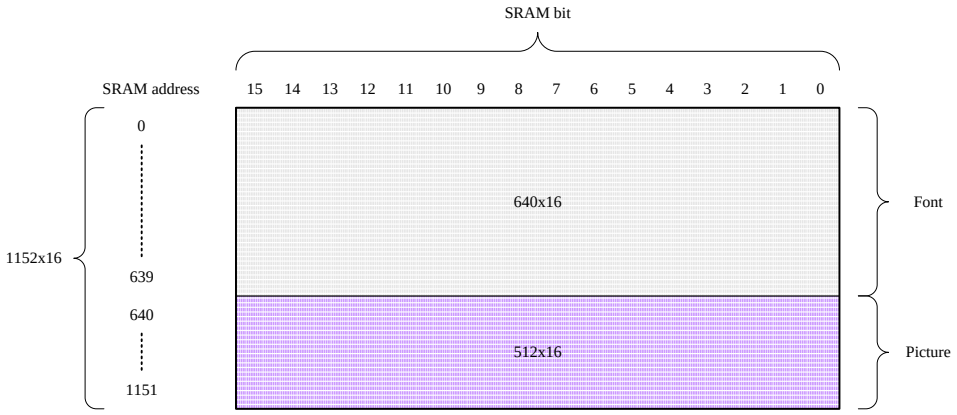
Pictues and SRAM Memory Requirements

The picture is used for channel status. Each channel has picture display. The picture size in SRAM is fixed to 32x32. The display picture size is also fixed to 32x32. Same as font, it uses 2 bits for picture color. Therefore, there are four colors that can be set by registers: OSD_PIC_R0 (G0, B0), OSD_PIC_R1 (G1, B1), OSD_PIC_R2 (G2, B2) and OSD_PIC_R3 (G3, B3). There are four pictures saved in SRAM. Data in SRAM are same as fonts. Data is saved picture by picture and in one picture, data is line by line. In one byte data, pixel data is stored in big endian. There are a total of 32x32x4x2 = 512x16 bits in SRAM.

Fonts and Pictues in SRAM memory allocation

The total Font and picture SRAM size is 640x16 + 512x16 = 1152x16

The allocation of the SRAM is as follows:



During system initialization, host need to write font and picture data in this SRAM. The write sequence is:

1. OSD_FRAM_ADDR[7:0]

2. OSD_FRAM_ADDR[10:0]

3. OSD_FRAM_DATA[7:0]

4. OSD_FRAM_DATA[15:8]

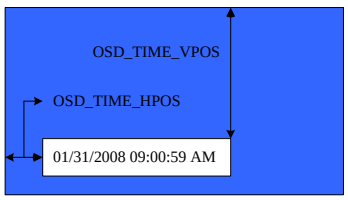
OSD_FRAM_DATA[15:8] must be the last one.

Display Information

Display information includes all data and pictures generated by the OSD modules and showed on the screen.

DISPLAY DATE AND TIME

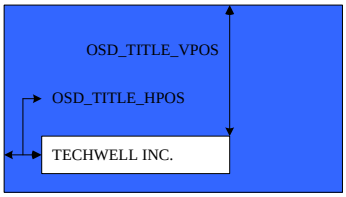
Date and time are only display once on whole screen, not channel by channel. There are a total of 32 fonts that can be displayed, including space. Like other display information, the position for date and time can be programmed. It can also be disabled and mixed with video. The font index is saved in display SRAM. It needs 32x6 bits.



Date/Time_SRAM_ADDRESS = Font_index

DISPLAY TITLE

Title is same as date and time. It has 32 fonts. It needs 32x6 SRAM size.



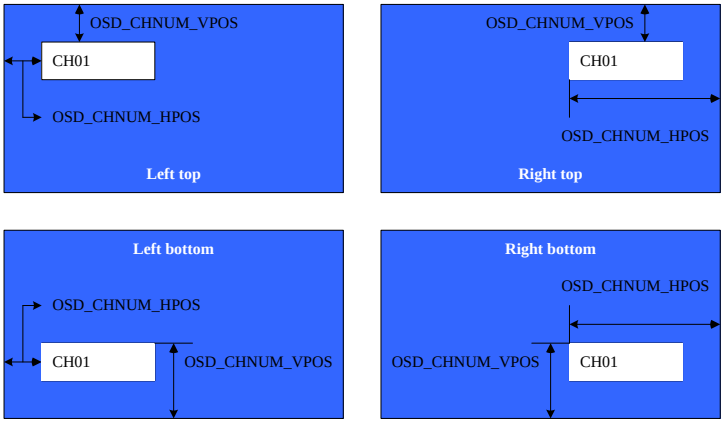
Tittle_SRAM_ADDRESS = ‘d32 + Font_index

DISPLAY CHANNEL NUMBERS

For each channel, there is max 8 characters channel number information.

Each character is selected from the 64-font table. If channel number information has less than 8 fonts, you can set the remaining font to space. Therefore, you need to put space font in the 64-font table.

The font size can be changed according to register OSD_FONT_HSIZE and OSD_FONT_VSIZE. However, remember the fonts saved in memory are always 8x10. If double size is selected, just repeat every pixel twice.



Ch_num_SRAM_ADDRESS = ‘d64 + Ch_num x ‘d8 + Font_index

DISPLAY CHANNEL PICTURES

For each channel on the display, there is a 32x32 channel picture for the channel status, which shows that the camera is in working condition. The picture size in SRAM is fixed to 32x32. The display picture size is also fixed to 32x32. Same as font, it uses 2 bits for picture color. Therefore, there are four colors that can be set by registers: OSD_PIC_R0 (G0, B0), OSD_PIC_R1 (G1, B1), OSD_PIC_R2 (G2,B2) and OSD_PIC_R3 (G3, B3). There are four pictures saved in SRAM. Data in SRAM are same as fonts.

Data is saved picture by picture and in one picture, data is line by line. In one byte data, pixel data is stored

in big endian. There are a total of 32x32x4x2 = 512x16 bits in SRAM.

Picture index information for each channel will be saved in SRAM. It needs a total of 32x2=64 bits. However, for ease of design, we use 32x6 bits. Each 6 bit is for each channel, but only LSB 2 bits are used.

Picture position can also be programmed same as channel number. The register setting is shown in the following.

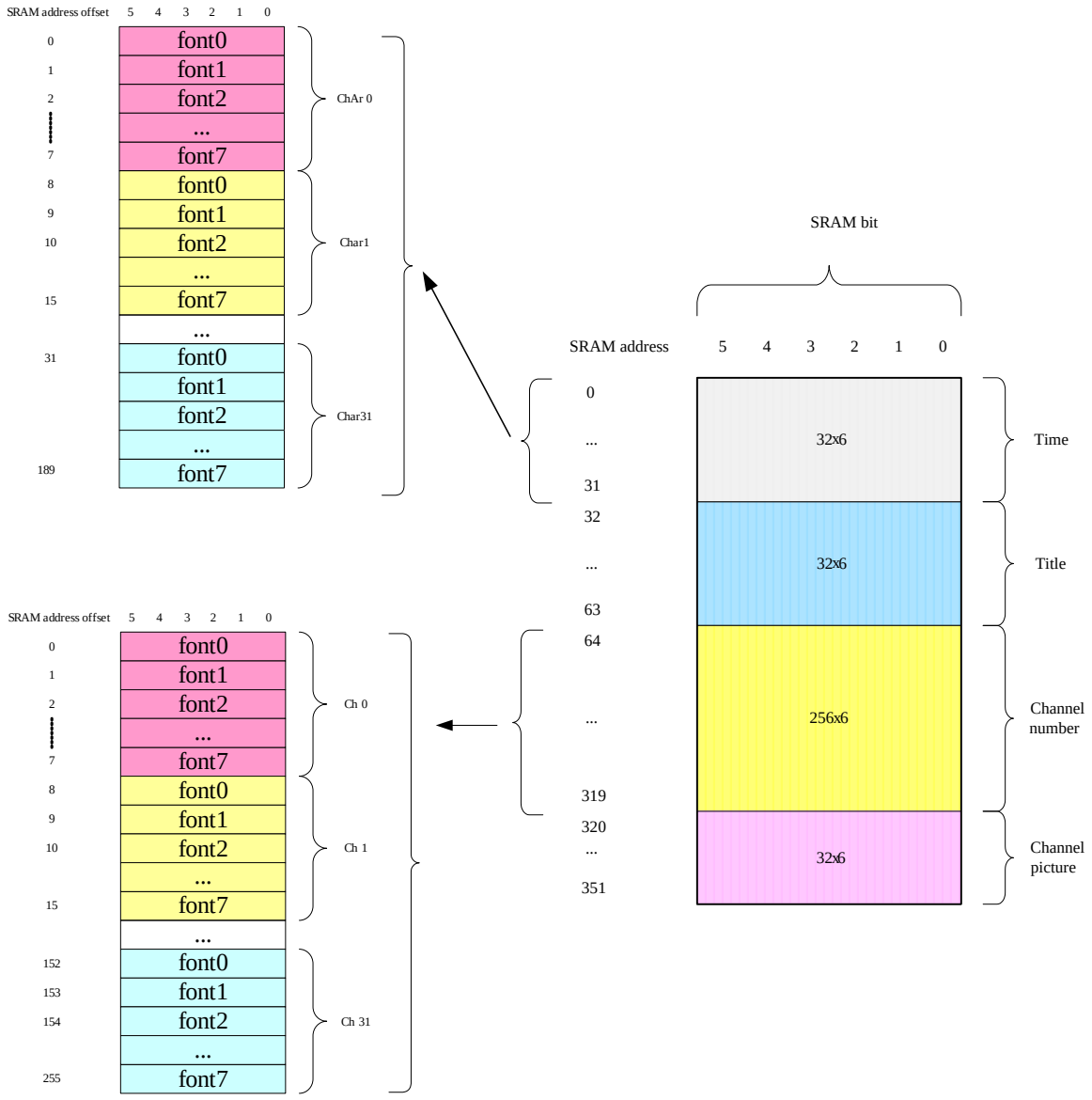
Ch_pic_SRAM_ADDRESS = ‘d320 + Ch_num

DISPLAY MEMORY

Display SRAM includes channel number, channel picture, date/time and title information. Channel number needs 32x8x6=256 bits, channel picture needs 32x6 bits, date/time needs 32x6 bits and title needs 32x6 bits. Therefore, the total SRAM size is 352x6 bits.

During display, host need to write index data in this SRAM. The write sequence is:

Set OSD_DRAM_ADDR, and then set OSD_DRAM_DATA.



Note: the SRAM size depends on the numbers of channels from the display modules, in LCD main display, the size is 252x6. on the SPOT size, it is 208x16. In REC port, it is 544x6

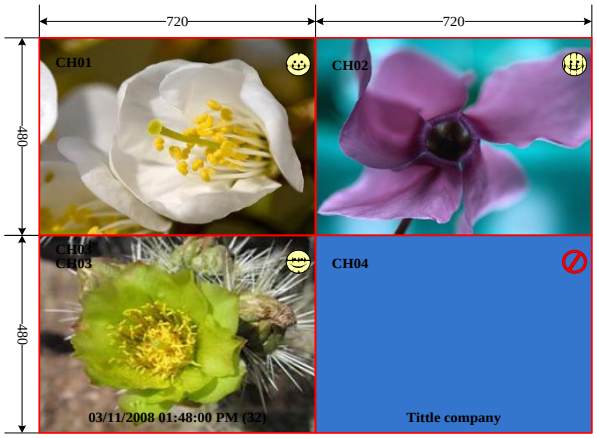
Example

Assume we want to show the display as the following pictures:

First, the TW2880 CRTC control registers need to be programmed accordingly. For instructions on how to program these registers, please refer to “Section 5: How to Setup a TW2880C-Based Display” on page 102.

Second, we need to load the Font and picture SRAM with all fonts will be used. Then we need to load all display information to the Display SRAM.

After all SRAM is written, we can program the OSD registers as the example values. These values are for illustration only, users can set any other options as they prefer. The Channel numbers display will be 33 pixels from the start of the display in horizontal and Vertical position. The Channel picture is chose from the top right start at the 33 pixels from the Vertical display start and 33 pixels from the end of horizontal line. The Time and title starts at Line 900 with Time/date begins at pixel 129 and Title begins at pixels 961. All mixing options Does not turn on. The OSD_Font, OSD_Pic colors register setting is for demonstration only. User can try to set all options and colors as they prefer.



4D1

Writing Simple OSD

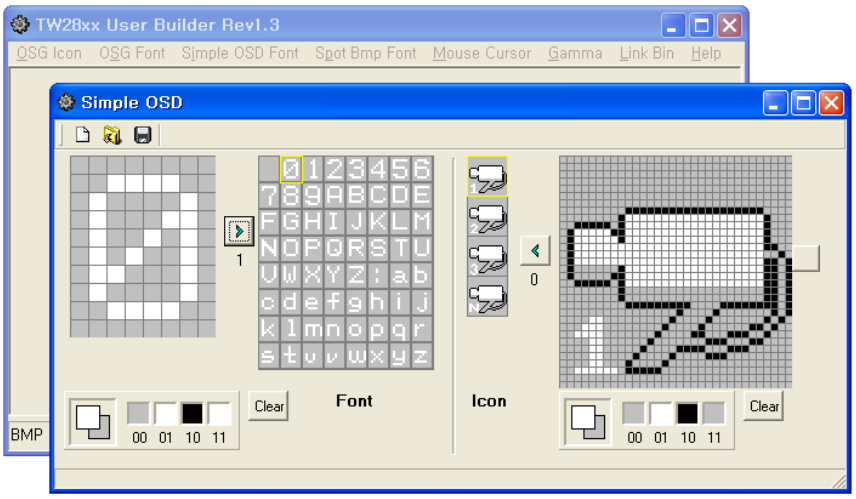
Simple OSD has 64 fonts table and 4 pictures table.

The size of each font is originally 8x10, but it has several various size controls to display character that is truncated to 6x8, truncated and doubled to 12x16, and doubled to 16x20 from the original size. Picture size is fixed to 32x32.

Font color is one and 2bit. Data “00” is dedicated to transparency color, other “01”, “10”, “11” can be changeable by OSD_FONT_Rx, OSD_FONT_Gx, and OSD_FONT_Bx registers.

Each Picture color can be assigned separately.

TW28xx User Builder tool can generate and edit Simple OSD data code.



Font needs 640x16 bits, and picture needs 512x16 bits. Therefore, the total SRAM size is 1152x16 bits.

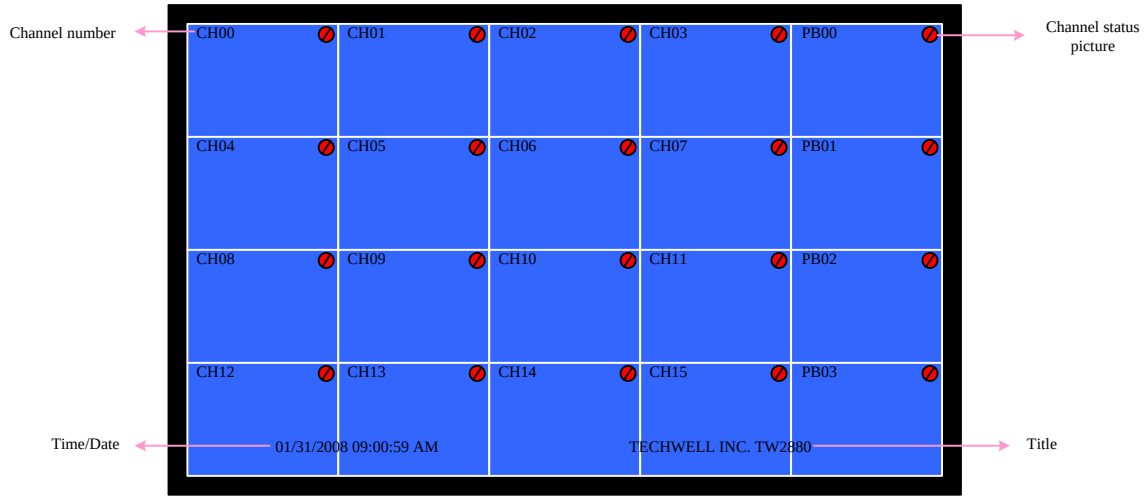
Below is the example of the writing Sample OSD after generating font and picture data in the file "/tw2880/osd.c."

```
#define FONTBYTE    640         // 8pixel * 10 line * 2 bit * 64 font / 16 (2byte)
#define PICBYTE 512         // 32 pixel * 32 line * 2 bit * 4 picture / 16 (2byte)

#define OSD_RAM_ADDR_L      0x1a8   // Address and Data Register
#define OSD_RAM_ADDR_H      0x1a9
#define OSD_RAM_DATA_L      0x1aa
#define OSD_RAM_DATA_H      0x1ab
#define MAXCOLOR                21     //  (R + G + B) * 3 color for font  +  (R + G +
B) * 4 picture

typedef struct OSD_HEADER {
    U16 *osdData;
     U8 *colortb;
} _OSD_HEADER;

void OsdLoadOSDData(_OSD_HEADER ptr)
{
    int i;

    if(ptr.osdData) // Load OSD data table
    {
        for(i=0; i<(FONTBYTE+PICBYTE); i++) {
            WriteP(OSD_RAM_ADDR_L, (U8)(i));
            WriteP(OSD_RAM_ADDR_H, (U8)(i>>8));
            WriteP(OSD_RAM_DATA_L, (U8)ptr.osdData[i]);
            WriteP(OSD_RAM_DATA_H, (U8)(ptr.osdData[i]>>8));

        }
    }

    if(ptr.colortb) // Load OSD lookup table
    {
        for(i=0; i<MAXCOLOR; i++)
            WriteP(OSD_FONT_COLOR_R1+i, (U8)ptr.colortb[i+3]);
    }
    }
```

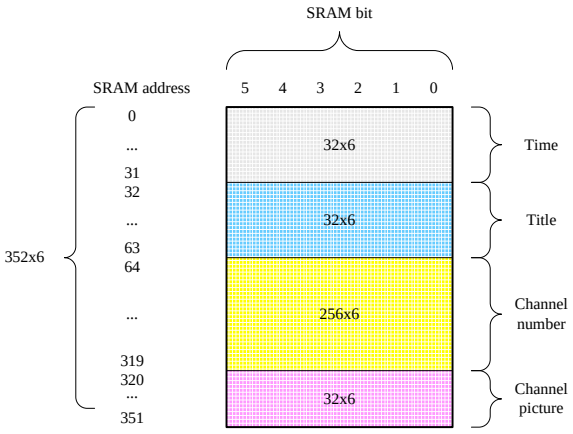
Display Simple OSD

It can display channel information for each 16 live channel and 16 playback. Each channel information display includes 8 characters and 1 picture. Simple OSD can also display two lines of 32-character strings for the whole main display, which can be used to display time, title, or status.

All of character can be selected from 64 font tables, and picture can be selected among the 4 picture tables saved in SRAM.



Channel number needs 32x8x6=256 bits, channel picture needs 32x6 bits, date/time needs 32x6 bits and title needs 32x6 bits. Therefore, the total SRAM size is 352x6 bits.



ADDRESS	R/W	DEFAULT	REGISTERS	SET VALUE TO
0x180	R/W	0	[4]: OSD_TITLE_EN [3]: OSD_TIME_EN [2]: OSD_CHPIC_EN [1]: OSD_CHNUM_EN [0]: OSD_EN	‘X1F
0x181	R/W	0	[5]: OSD_CHPIC_BLINK [4]: OSD_CHPIC_TRANS [3]: OSD_TITLE_MIX [2]: OSD_TIME_MIX [1]: OSD_CHPIC_MIX [0]: OSD_CHNUM_MIX	`X00
0x182	R/W	0x4F	[7:6]: OSD_CHPIC_POS [5:4]: OSD_CHNUM_POS [3:2]: OSD_FONT_VSIZE [1:0]: OSD_FONT_HSIZE	‘X4F
0x183	R/W	0	OSD_CHNUM_HPOS[7:0]	‘X20
0x184	R/W	0	OSD_CHNUM_HPOS[10:8]	‘X00
0x185	R/W	0	OSD_CHNUM_VPOS[7:0]	‘X20
0x186	R/W	0	OSD_CHNUM_VPOS[10:8]	‘X00
0x187	R/W	0	OSD_CHPIC_HPOS[7:0]	‘X20
0x188	R/W	0	OSD_CHPIC_HPOS[10:8]	‘X00
0x189	R/W	0	OSD_CHPIC_VPOS[7:0]	‘X20
0x18A	R/W	0	OSD_CHPIC_VPOS[10:8]	‘X00
0x18B	R/W	0	OSD_TIME_HPOS[7:0]	‘X80
0x18C	R/W	0	OSD_TIME_HPOS[10:8]	‘X00
0x18D	R/W	0	OSD_TIME_VPOS[7:0]	‘X84
0x18E	R/W	0	OSD_TIME_VPOS[10:8]	‘X03
0x18F	R/W	0	OSD_TITLE_HPOS[7:0]	‘XC0
0x190	R/W	0	OSD_TITLE_HPOS[10:8]	‘X03
0x191	R/W	0	OSD_TITLE_VPOS[7:0]	‘X84
0x192	R/W	0	OSD_TITLE_VPOS[10:8]	‘X03
0x193	R/W	0	OSD_FONT_R1[7:0]	‘X80
0x194	R/W	0	OSD_FONT_G1[7:0]	‘X10
0x195	R/W	0	OSD_FONT_B1[7:0]	‘X80
0x196	R/W	0	OSD_FONT_R2[7:0]	‘X00
0x197	R/W	0	OSD_FONT_G2[7:0]	‘X00
0x198	R/W	0	OSD_FONT_B2[7:0]	‘X00
0x199	R/W	0	OSD_FONT_R3[7:0]	‘X00
0x19A	R/W	0	OSD_FONT_G3[7:0]	‘X00
0x19B	R/W	0	OSD_FONT_B3[7:0]	‘X00
0x19C	R/W	0	OSD_PIC_R0[7:0]	‘X80
0x19D	R/W	0	OSD_PIC_G0[7:0]	‘X80
0x19E	R/W	0	OSD_PIC_B0[7:0]	‘XEB
0x19F	R/W	0	OSD_PIC_R1[7:0]	‘X80
0x1A0	R/W	0	OSD_PIC_G1[7:0]	‘XEF
0x1A1	R/W	0	OSD_PIC_B1[7:0]	‘X5C
0x1A2	R/W	0	OSD_PIC_R2[7:0]	‘X23
0x1A3	R/W	0	OSD_PIC_G2[7:0]	‘X8F
0x1A4	R/W	0	OSD_PIC_B2[7:0]	‘X37
0x1A5	R/W	0	OSD_PIC_R3[7:0]	‘X6F
0x1A6	R/W	0	OSD_PIC_G3[7:0]	‘X28

ADDRESS	R/W	DEFAULT	REGISTERS	SET VALUE TO
0x1A7	R/W	0	OSD_PIC_B3[7:0]	‘XEF
0x1A8	W	0	OSD_FRAM_ADDR[7:0]	
0x1A9	W	0	OSD_FRAM_ADDR[10:8]	
0x1AA	W	0	OSD_FRAM_DATA[7:0]	
0x1AB	W	0	OSD_FRAM_DATA[15:8]	
0x1AC	W	0	OSD_DRAM_ADDR[7:0]	
0x1AD	W	0	OSD_DRAM_ADDR[8]	
0x1AE	W	0	OSD_DRAM_DATA[5:0]	

Section 7: Motion Detection and Interrupt

Introduction

TW2880 has motion detection circuitry for each incoming video channels (all together the number is 16). The source of MD circuit is the 16 D1 stream coming from the input. To do the job, TW2880 divided the first field of each stream into a 16x12 cell array. From here, a unique signature for each cell is extracted and saved into DRAM buffer for later use. The second field of each frame is discarded for simplicity and cost issue. The motion detection algorithm compares the difference of luminance value between current field and reference field to determine whether a motion has occurred.

Uses the same detection engine, TW2880 also supports blind and night detection. The motion detector is operated with the second memory controller module.

Mask and Detection Region Selection

704 Pixels (44 Pixels/Cell)

240 Lines for 60Hz (20 Lines/Cell), 288 Lines for 50Hz (24 Lines/Cell)

MD_MASK0[0]	MD_MASK0[1]	MD_MASK0[2]	MD_MASK0[3]	MD_MASK0[4]	MD_MASK0[5]	MD_MASK0[6]	MD_MASK0[7]	MD_MASK0[8]	MD_MASK0[9]	MD_MASK0[10]	MD_MASK0[11]	MD_MASK0[12]	MD_MASK0[13]	MD_MASK0[14]	MD_MASK0[15]
MD_MASK1[0]	MD_MASK1[1]	MD_MASK1[2]	MD_MASK1[3]	MD_MASK1[4]	MD_MASK1[5]	MD_MASK1[6]	MD_MASK1[7]	MD_MASK1[8]	MD_MASK1[9]	MD_MASK1[10]	MD_MASK1[11]	MD_MASK1[12]	MD_MASK1[13]	MD_MASK1[14]	MD_MASK1[15]
MD_MASK2[0]	MD_MASK2[1]	MD_MASK2[2]	MD_MASK2[3]	MD_MASK2[4]	MD_MASK2[5]	MD_MASK2[6]	MD_MASK2[7]	MD_MASK2[8]	MD_MASK2[9]	MD_MASK2[10]	MD_MASK2[11]	MD_MASK2[12]	MD_MASK2[13]	MD_MASK2[14]	MD_MASK2[15]
MD_MASK3[0]	MD_MASK3[1]	MD_MASK3[2]	MD_MASK3[3]	MD_MASK3[4]	MD_MASK3[5]	MD_MASK3[6]	MD_MASK3[7]	MD_MASK3[8]	MD_MASK3[9]	MD_MASK3[10]	MD_MASK3[11]	MD_MASK3[12]	MD_MASK3[13]	MD_MASK3[14]	MD_MASK3[15]
MD_MASK4[0]	MD_MASK4[1]	MD_MASK4[2]	MD_MASK4[3]	MD_MASK4[4]	MD_MASK4[5]	MD_MASK4[6]	MD_MASK4[7]	MD_MASK4[8]	MD_MASK4[9]	MD_MASK4[10]	MD_MASK4[11]	MD_MASK4[12]	MD_MASK4[13]	MD_MASK4[14]	MD_MASK4[15]
MD_MASK5[0]	MD_MASK5[1]	MD_MASK5[2]	MD_MASK5[3]	MD_MASK5[4]	MD_MASK5[5]	MD_MASK5[6]	MD_MASK5[7]	MD_MASK5[8]	MD_MASK5[9]	MD_MASK5[10]	MD_MASK5[11]	MD_MASK5[12]	MD_MASK5[13]	MD_MASK5[14]	MD_MASK5[15]
MD_MASK6[0]	MD_MASK6[1]	MD_MASK6[2]	MD_MASK6[3]	MD_MASK6[4]	MD_MASK6[5]	MD_MASK6[6]	MD_MASK6[7]	MD_MASK6[8]	MD_MASK6[9]	MD_MASK6[10]	MD_MASK6[11]	MD_MASK6[12]	MD_MASK6[13]	MD_MASK6[14]	MD_MASK6[15]
MD_MASK7[0]	MD_MASK7[1]	MD_MASK7[2]	MD_MASK7[3]	MD_MASK7[4]	MD_MASK7[5]	MD_MASK7[6]	MD_MASK7[7]	MD_MASK7[8]	MD_MASK7[9]	MD_MASK7[10]	MD_MASK7[11]	MD_MASK7[12]	MD_MASK7[13]	MD_MASK7[14]	MD_MASK7[15]
MD_MASK8[0]	MD_MASK8[1]	MD_MASK8[2]	MD_MASK8[3]	MD_MASK8[4]	MD_MASK8[5]	MD_MASK8[6]	MD_MASK8[7]	MD_MASK8[8]	MD_MASK8[9]	MD_MASK8[10]	MD_MASK8[11]	MD_MASK8[12]	MD_MASK8[13]	MD_MASK8[14]	MD_MASK8[15]
MD_MASK9[0]	MD_MASK9[1]	MD_MASK9[2]	MD_MASK9[3]	MD_MASK9[4]	MD_MASK9[5]	MD_MASK9[6]	MD_MASK9[7]	MD_MASK9[8]	MD_MASK9[9]	MD_MASK9[10]	MD_MASK9[11]	MD_MASK9[12]	MD_MASK9[13]	MD_MASK9[14]	MD_MASK9[15]
MD_MASK10[0]	MD_MASK10[1]	MD_MASK10[2]	MD_MASK10[3]	MD_MASK10[4]	MD_MASK10[5]	MD_MASK10[6]	MD_MASK10[7]	MD_MASK10[8]	MD_MASK10[9]	MD_MASK10[10]	MD_MASK10[11]	MD_MASK10[12]	MD_MASK10[13]	MD_MASK10[14]	MD_MASK10[15]
MD_MASK11[0]	MD_MASK11[1]	MD_MASK11[2]	MD_MASK11[3]	MD_MASK11[4]	MD_MASK11[5]	MD_MASK11[6]	MD_MASK11[7]	MD_MASK11[8]	MD_MASK11[9]	MD_MASK11[10]	MD_MASK11[11]	MD_MASK11[12]	MD_MASK11[13]	MD_MASK11[14]	MD_MASK11[15]

MOTION DETECTION MASK AND CELL DEFINITION

The motion detection algorithm utilizes the full screen video data and detects individual motion of 16x12 cells. Like the extraction process, this full screen for motion detection consists of 704 pixels and 240 lines for NTSC and 288 lines for PAL.

Each cell can be masked via the MD_MASK (0x800 ~ 0x817, 0x840 ~ 0x857, 0x880 ~ 0x897, 0x8C0 ~ 0x8D7, page 175, page A and pageB) register, as illustrated in the above figure. If the mask bit in specific cell is programmed to high, the related cell is ignored for motion detection.

The motion detection result is stored in registers (0x820 ~ 0x837, 0x840 ~ 0x857, 0x8A0 ~ 0x8B7, 0x8E0 ~ 0x8F7, page 175, page A and page B) and a "1" indicates detecting motion and "0" denotes no motion detection in the cell.

To detect motion properly according to situation, TW2880 provides several sensitivity and velocity control parameters for each motion detector. TW2880 supports manual strobe function to update motion detection so that it is more appropriate for user-defined motion sensitivity control.

The motion detector has 3 sensitivity parameters to control threshold of motion detection such as the level sensitivity via the MD_LVSENS registers, the spatial sensitivity via the MD_SPSENS registers and the temporal sensitivity parameter via the MD_TMPSENS registers.

REGISTER SETTINGS

Channel 1

MASK Setting : "0" : Motion detection, "1" : Motion ignore

1st line MASK Setting: 0x800~0x801

2nd line MASK Setting: 0x802~0x803

3rd line MASK Setting: 0x804~0x805

4th line MASK Setting: 0x806~0x807

5th line MASK Setting: 0x808~0x809

6th line MASK Setting: 0x80A~0x80B

7th line MASK Setting: 0x80C~0x80D

8th line MASK Setting: 0x80E~0x80F

9th line MASK Setting: 0x810~0x811

10th line MASK Setting: 0x812~0x813

11th line MASK Setting: 0x814~0x815

12th line MASK Setting: 0x816~0x817

Channels 2~4

MASK Setting : (0x840 ~ 0x857, 0x880 ~ 0x897, 0x8C0 ~ 0x8D7)

Channels 5~8

MASK Setting : (0x900 ~ 0x917, 0x940 ~ 0x957, 0x980 ~ 0x997, 0x9C0 ~ 0x9D7)

Channels 9~12

MASK Setting : (0xA00 ~ 0xA17, 0xA40 ~ 0xA57, 0xA80 ~ 0xA97, 0xAC0 ~ 0xAD7)

Channels 13~16

MASK Setting : (0xB00 ~ 0xB17, 0xB40 ~ 0xB57, 0xB80 ~ 0xB97, 0xBC0 ~ 0xBD7)

Sensitivity Control

The motion detector has 3 sensitivity parameters to control threshold of motion detection such as the level sensitivity via the MD_LVSENS registers, the spatial sensitivity via the MD_SPSENS registers and the temporal sensitivity parameter via the MD_TMPSENS registers.

In built-in motion detection algorithm, the motion is detected when luminance level difference between current and reference field is greater than MD_LVSENS value. Motion detector is more sensitive for the smaller MD_LVSENS value and less sensitive for the larger. When the MD_LVSENS is too small, the motion detector may be weak in noise.

TW2880 uses 192 (16x12) detection cells in full screen for motion detection. Each detection cell is composed of 44 pixels and 20 lines for NTSC and 24 lines for PAL. Motion detection from only luminance level difference between two fields is very weak in spatial random noise. To remove the fake motion detection from the random noise, the TW2880 supports a spatial filter via the MD_SPSENS register, which defines the number of detected cell to decide motion detection in full size image. The large MD_SPSENS value increases the immunity of spatial random noise.

Similarly, temporal filter is used to remove the fake motion detection from the temporal random noise. The MD_TMPSENS regulates the number of taps in the temporal filter to control the temporal sensitivity so that the large MD_TMPSENS value increases the immunity of temporal random noise.

REGISTER SETTINGS

Channel 1

0x81A[4:0] : motion detection level sensitivity adjust (Recommend value : “3”)

0x81E[3:0] : motion detection temporal sensitivity adjust (Recommend value : “0”)

0x81E[7:4] : motion spatial sensitivity adjust (Recommend value : “0”)

Channels 2~4

Sensitivity Setting : (0x85A ~ 0x85E, 0x89A ~ 0x89E, 0x8DA ~ 0x8DE)

Channels 5~8

Sensitivity Setting : (0x91A ~ 0x91E, 0x95A ~ 0x95E, 0x99A ~ 0x99E, 0x9DA ~ 0x9DE)

Channels 9~12

Sensitivity Setting : (0xA1A ~ 0xA1E, 0xA5A ~ 0xA5E, 0xA9A ~ 0xA9E, 0xADA ~ 0xADE)

Channels 13~16

Sensitivity Setting : (0xB1A ~ 0xB1E, 0xB5A ~ 0xB5E, 0xB9A ~ 0xB9E, 0xBDA ~ 0xBDE)

Velocity Control

The motion has various velocities. That is, in a fast motion an object appears and disappears rapidly between the adjacent fields while in a slow motion it is to the contrary. As the built-in motion detection algorithm uses the only luminance level difference between two adjacent fields, a slow motion is inferior in detection rate to a fast motion. To compensate this weakness, MD_SPEED parameter is used, which is controllable up to 64 fields. MD_SPEED parameter adjusts the field interval in which the luminance level is compared. Thus, for detection of fast motion, a small value is needed, and for slow motion, a large value is required. The parameter MD_SPEED value should be greater than MD_TMPSENS value.

Additionally, TW2880 has 2 more parameters to control the selection of reference field. The MD_FIELD[1:0] bit is a field selection parameter such as odd, even, any field or frame.

The MD_REFFLD bit is designed to control the updating period of reference field. If MD_REFFLD = “0”, the interval from current field to reference field is always same as the MD_SPEED. It means that the reference field is always

updated every field. Figure 53 shows the relationship between current and reference field for motion detection when MD_REFFLD is set to 0.

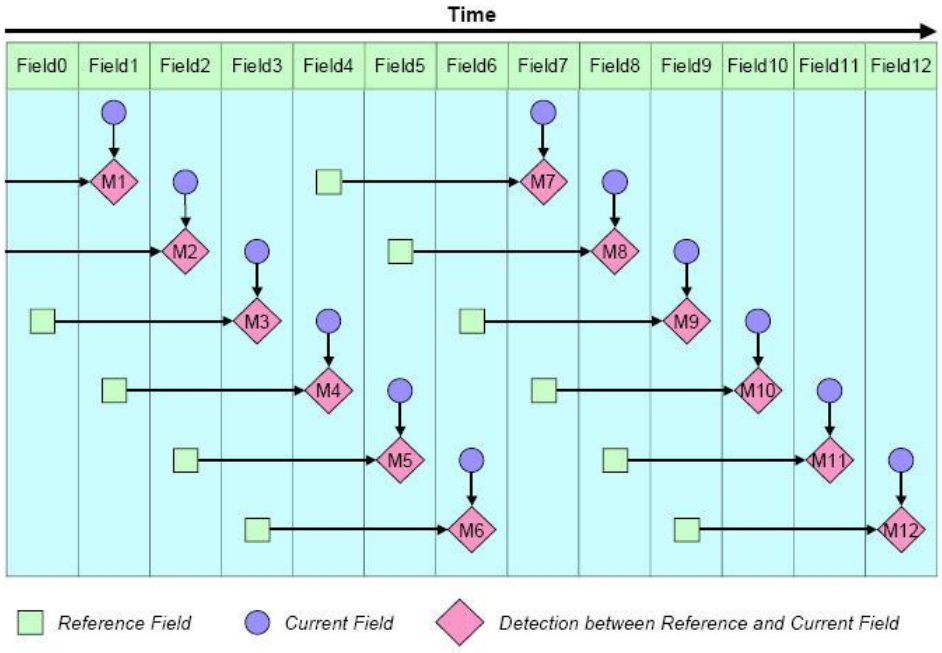


FIGURE 53. THE RELATIONSHIP BETWEEN CURRENT AND REFERENCE FIELD WHEN ND_REFFLD = “0”

TW2880 can update the reference field only at the period of MD_SPEED when the MD_REFFLD is high. For this case, the TW2880 can detect a motion with sense of a various velocity.

Figure 54 shows the relationship between current and reference field for motion detection when the MD_REFFLD equals to 1.

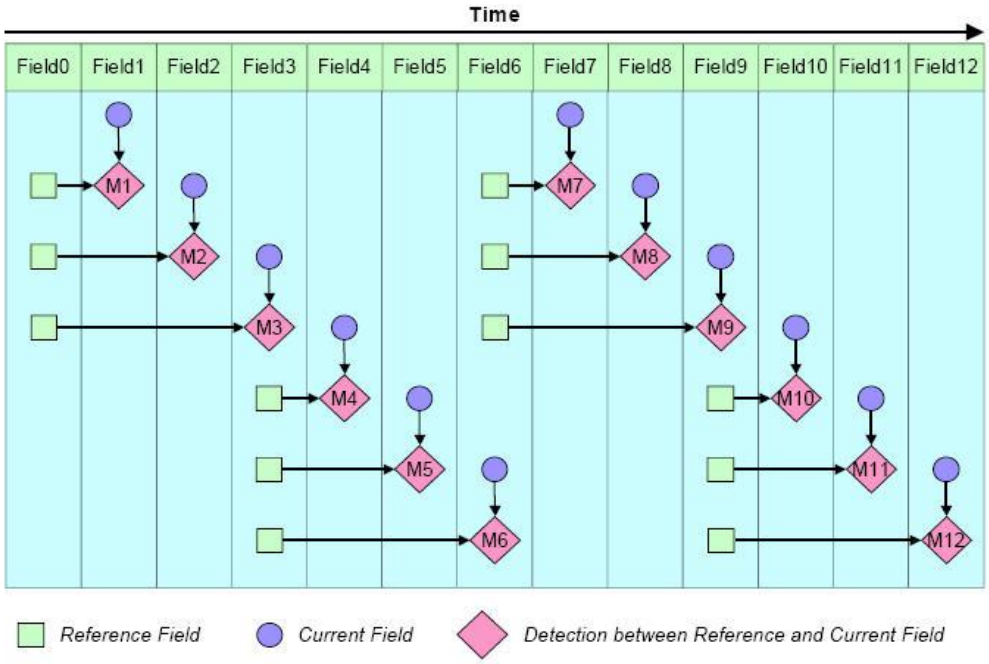


FIGURE 54. THE RELATIONSHIP BETWEEN CURRENT AND REFERENCE FIELD WHEN ND_REFFLD = "1

TW2880 also supports the manual detection timing control of the reference field/frame via the MD_STRB_EN and MD_STRB register bits in the MD control registers. If MD_STRB_EN is set to 0, the reference field/frame is automatically updated and reserved on every reference field/frame. If MD_STRB_EN is set to1, the reference field/frame is updated and reserved only when MD_STRB bit is set to 1. If an external strobing signal is used, one can set the mode to 1x and select it. The strobe signal is coming from outside via trigger_in pin. In these two modes, the interval between current and reference field/frame is controlled by user's strobe timing and are very useful for some specific purpose like non-periodical velocity control and very slow motion detection.

REGISTER SETTINGS

Channel 1

0x81B[6] : Reference Field Update Mode ("0" : update every field, "1" : update every MD_SPEED period)

(Recommend value : "0")

0x81B[5:0] : compare field interval adjust (Recommend value : "0")

Channels 2~4

Velocity Setting : (0x85B, 0x89B, 0x8DB)

Channels 5~8

Velocity Setting : (0x91B, 0x95B, 9x9B, 9xDB)

Channels 9~12

Velocity Setting : (0xA1B, 0xA5B, 0xA9B, 0xADB)

Channels 13~16

Velocity Setting : (0xB1B, 0xB5B, 0xB9B, 0xBDB)

Blind Detection

If the luminous level of a video input in every corner area is almost equal to the average luminous level of this frame due to the camera being covered by something, this input is defined as blind input. TW2880 supports blind detection individually for all 16 video inputs and generates an interrupt to host CPU.

TW2880 uses two sensitivity parameters to detect blind input. One is the level sensitivity via the BD_LVSENS register and the other is spatial sensitivity via the BD_CELSENS register. The TW2880 uses total 192 (16x12) cells in full screen for blind detection. The BD_LVSENS parameter controls the threshold of level between cell and field average. The BD_CELSENS parameter defines the number of cells to detect blind. For BD_CELSENS = "0", the number of cell whose level is same as average of field should be over than 60% to detect blind, 70% for BD_CELSENS = "1", 80% for BD_CELSENS = "2", and 90% for BD_CELSENS = "3". That is, the large value of BD_LVSENS and BD_CELSENS makes blind detector less sensitive.

REGISTER SETTINGS

Channel 1

0x818[3:0] : Blind level sensitivity adjust (Recommend value : "8")

0x818[5:4] : blind cell sensitivity adjust ("0" : 60%, "1" : 70%, "2" : 80%, "3" : 90%) (Recommend value : "0")

Channels 2~4

Blind Setting : (0x858, 0x898, 0x8D8)

Channels 5~8

Blind Setting : (0x918, 0x958, 0x998, 0x9D8)

Channels 9~12

Blind Setting : (0xA18, 0xA58, 0xA98, 0xAD8)

Channels 13~16

Blind Setting : (0xB18, 0xB58, 0xB98, 0xBD8)

Night Detection

TW2880 uses a user defined, fixed value to determine whether a video input is in a broad day light or at night situations. If the average of luminous level is lower than this fixed value, this input is defined as night input. Likewise, the opposite is defined as day input. The TW2880 supports night detection for all 16 video inputs and will generated interrupts to host CPU if triggered.

Two parameters are used to detect night input. One is the level sensitivity via the ND_LVSENS register and the other is temporal sensitivity via the ND_TMPSENS register. The ND_LVSENS parameter controls threshold level of day and night. The ND_TMPSENS parameter regulates the number of taps in the temporal low pass filter to control the temporal sensitivity. The large value of ND_LVSENS and ND_TMPSENS makes night detector less sensitive.

REGISTER SETTINGS

Channel 1

0x819[7:4] : night detection level sensitivity adjust (Recommend value : "3")

0x819[3:0] : night detection temporal sensitivity adjust (Recommend value : "3")

Channels 2~4

Night Setting : (0x859, 0x899, 0x8D9)

Channels 5~8

Night Setting : (0x919, 0x959, 0x999, 0x9D9)

Channels 9~12

Night Setting : (0xA19, 0xA59, 0xA99, 0xAD9)

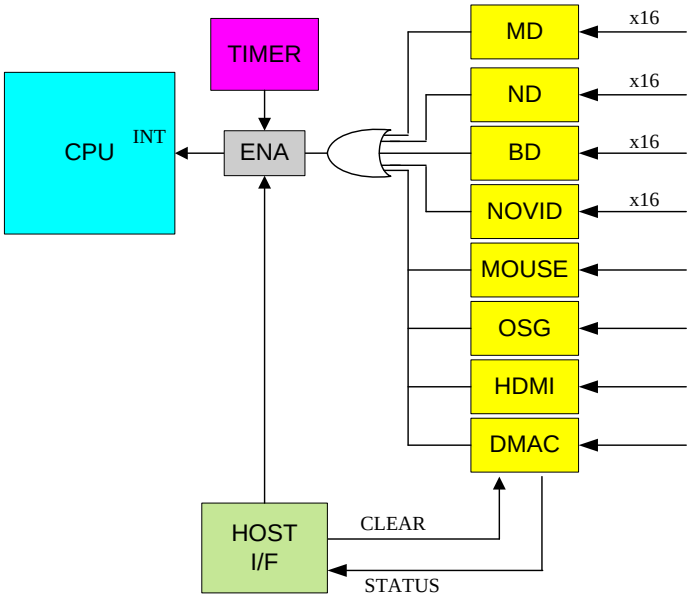
Channels 13~16

Night Setting : (0xB19, 0xB59, 0xB99, 0xBD9)

Interrupt Interface

Interrupt Interface

The TW2880 provides a very sophisticated interrupt request function for user to inter-react with the host CPU. Any video loss, motion, blind, or night detection in every channel will generate an interrupt request to the host CPU. The polarity of the interrupt is selectable by the user. The user can disable the one single interrupt function for each channel and category. After receiving the interrupt, the host can distinguish which functional unit generated the interrupt by writing to the interrupt status registers. The user can choose to read the real time detection status of the all functional units by changing a bit. The interrupt can be cleared by writing "1" the interrupt clearing registers.



A set of idle and resend counters is incorporated in the interrupt generation process to help easing the burden of the CPU is responding and switching between different interrupt service routines. Once an interrupt is raised and does not get attention from the CPU for a certain period of time, the interrupt of TW2880 will become inactive for a while and become active again. This process will go on indefinitely until the unit is reset. This function can be disabled by user.

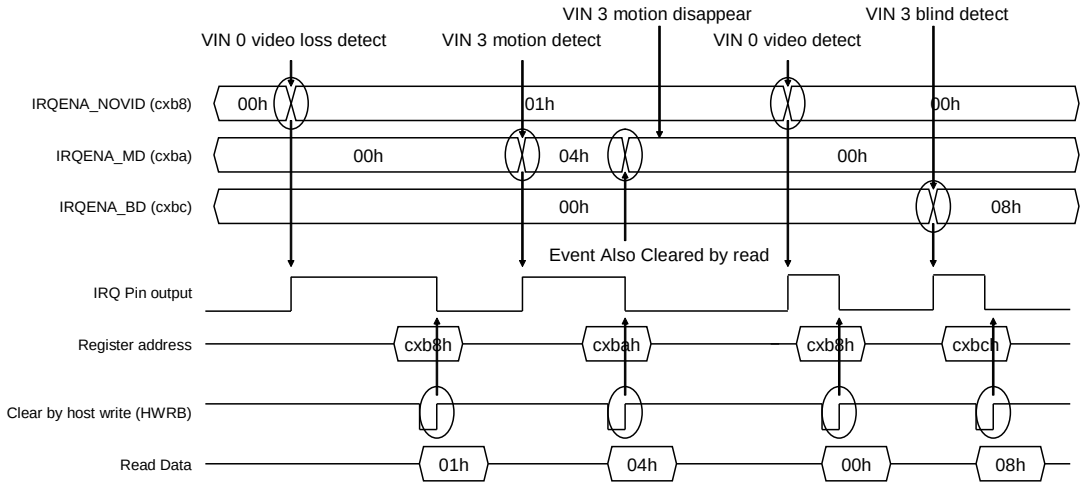


ILLUSTRATION OF THE INTERRUPT GENERATED AND CLEARED SEQUENCE

Register Settings

[0xCB0, 0xCB1] : Enable the interrupt for video loss detection for 16 channel.

[0xCB2, 0xCB3] : Enable the interrupt for motion detection for 16 channel.

[0xCB4, 0xCB5] : Enable the interrupt for blind detection for 16 channel.

[0xCB6, 0xCB7] : Enable the interrupt for night detection for 16 channel.

[0xCB8, 0xCB9] : No video IRQ status for 16 channel.

In case 0xCC0[7] == “1”

Read data is Enable the interrupt for video loss detection for 16 channel.

In case 0xCC0[7] == “0”, 0xCC0[0] == “1”

Read data is irq event status.

In case 0xCC0 [7] == “0”, 0xCC0[0] == “0”

Read data is real detection value.

[0xCBA, 0xCBB] : Motion detection IRQ status for 16 channel.

In case 0xCC0[7] == “1”

Read data is Enable the interrupt for motion detection for 16 channel.

In case 0xCC0 [7] == “0”, 0xCC0[1] == “1”

Read data is irq event status.

In case 0xCC0[7] == “0”, 0xCC0[1] == “0”

Read data is real detection value.

[0xCBC, 0xCBD] : Blind detection IRQ status for 16 channel.

In case 0xCC0[7] == “1”

Read data is Enable the interrupt for blind detection for 16 channel.

In case 0xCC0[7] == “0”, 0xCC0[2] == “1”

Read data is irq event status.

In case 0xCC0[7] == “0”, 0xCC0[2] == “0”

Read data is real detection value.

[0xCBE, 0xCBF] : Night detection IRQ status for 16 channel.

In case 0xCC0[7] == “1”

Read data is Enable the interrupt for night detection for 16 channel.

In case 0xCC0[7] == “0”, 0xCC0[3] == “1”

Read data is irq event status.

In case 0xCC0[7] == “0”, 0xCC0[3] == “0”

Read data is real detection value.

Detection Status Clear Method

[0xCB8, 0xCB9] : No video IRQ status clear by writing “1” and 0xCC0[7] == “0” for 16 channel.

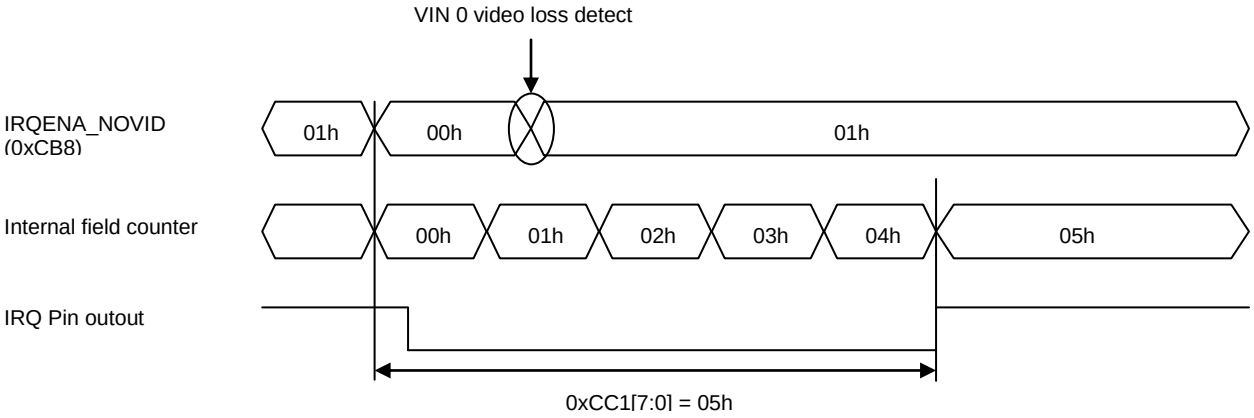
[0xCBA, 0xCBB] : Motion detection IRQ status clear by writing “1” and 0xCC0[7] == “0” for 16 channel.

[0xCBC, 0xCBD] : Blind detection IRQ status clear by writing “1” and 0xCC0[7] == “0” for 16 channel.

[0xCBE, 0xCBF] : Night detection IRQ status clear by writing “1” and 0xCC0[7] == “0” for 16 channel.

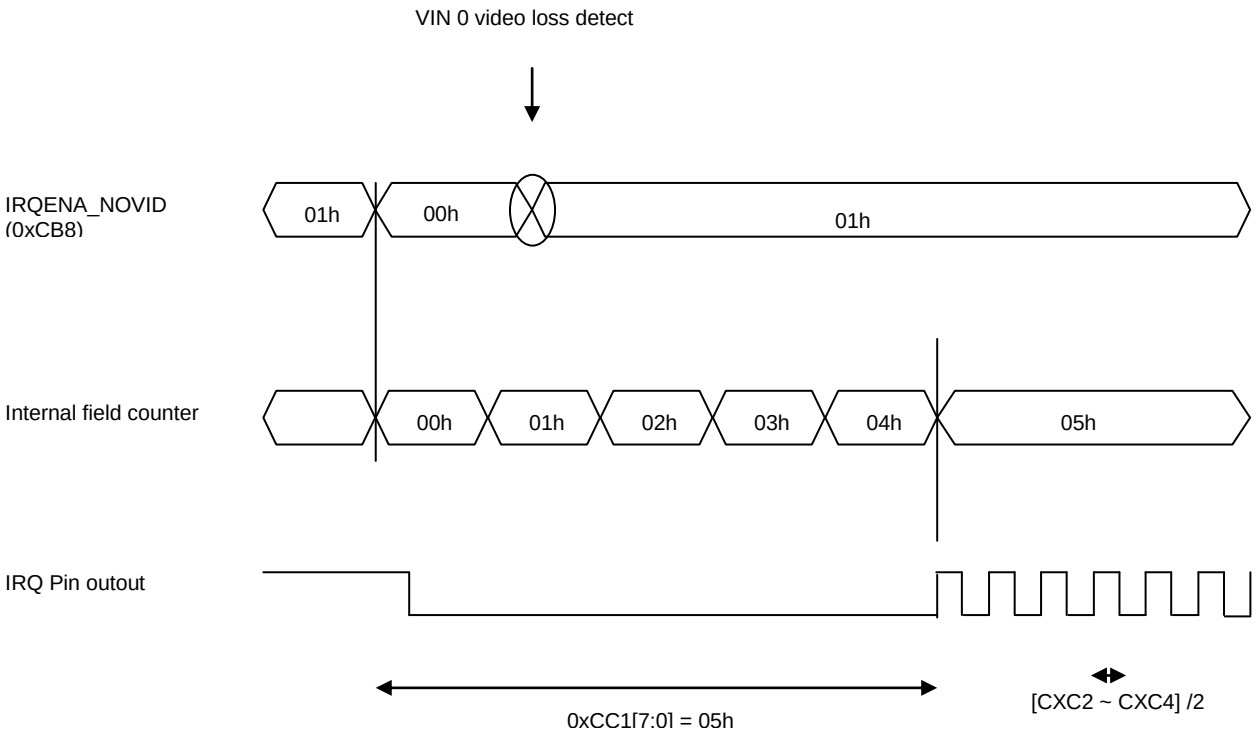
[0xCC0[6]] : interrupt polarity (“0” : positive, “1” : negative)

[0xCC1] : Control the interrupt generation period (The unit is field).
0 : Immediate generation of interrupt when any Interrupt happens
: :
255 : Interrupt generation by the duration of the [0xCC1]



[0xCC0[5]] : irq status repeat enable

[0xCC2 ~ 0xCC4] : irq status repeat period



Motion Box Setting

The TW2880 supports 16 MD arrayed boxes that have programmable cell size up to 16x16. The MD arrayed box can be used to make table menu or display motion detection information via the MDBOX_MODE (0x550[6] ~ 0x55F[6]) register. The MD arrayed box is displayed on each path by the MDBOX_EN (0x550[7] ~ 0x55F[7]) register.

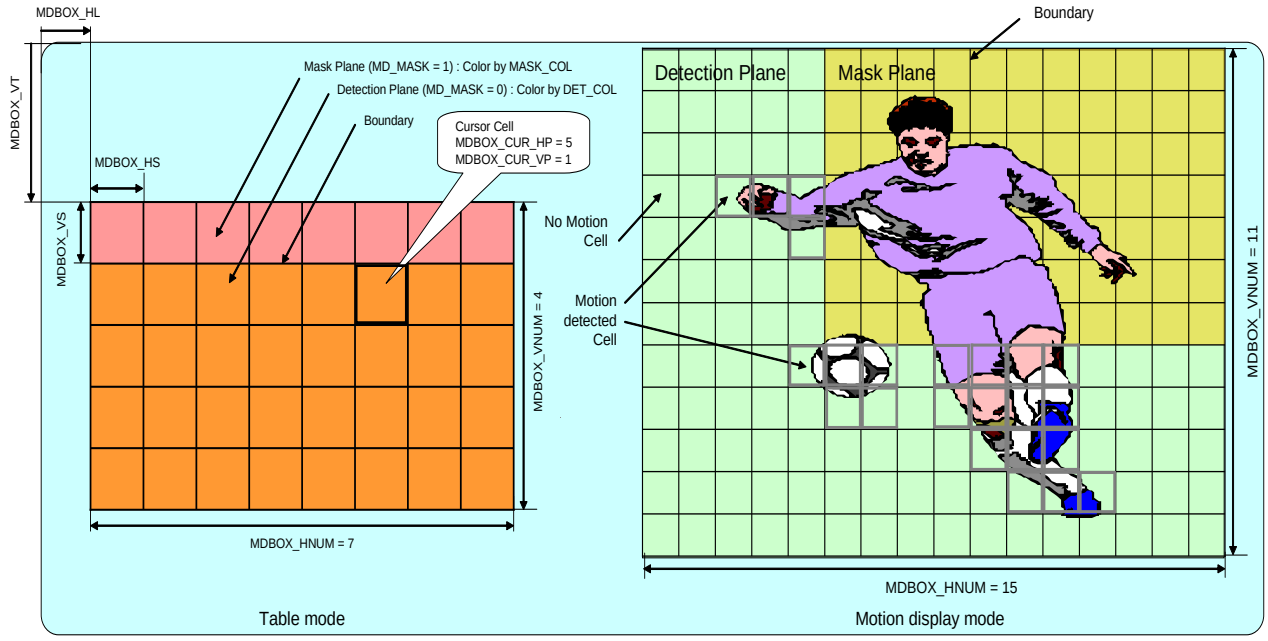
For each MD arrayed box, the number of row and column cells is defined via the MDBOX_HNUM(0x5E8[3:0] ~ 0x5EF[7:4]) and MDBOX_VNUM (0x5F0[3:0] ~ 0x5F7[7:4]) registers. The horizontal and vertical location of left top is controlled by the MDBOX_HL (0x568 ~ 0x587) register and the MDBOX_VT (0x558 ~ 0x5A7) registers. The horizontal and vertical size of each cell is defined by the MDBOX_VS (0x5C8 ~ 0x5E7) registers and the MDBOX_HS (0x5A8 ~ 0x5C7) registers. Therefore, the whole size of the MD arrayed box is the same as the sum of cells in row and column.

The boundary of MD arrayed box is enabled by the MDBOX_BNDEN (0x550[4] ~ 0x55F[4]) register and its color is controlled via the MDBOX_BNDCOL register.

Especially the TW2880 provides the function to indicate cursor cell inside MD arrayed box. The cursor cell is enabled by the MDBOX_CUREN (0x550[5] ~ 0x55F[5]) register and the displayed location is defined by the MDBOX_CURHP (0x5F8 ~ 0x5FF) and MDBOX_CURVP (0x48B ~ 0x492) registers. Its color is a reverse color of cell boundary. It is useful function to control motion mask region.

The plane of MD arrayed box is separated into mask plane and detection plane. The mask plane represents the cell defined by MD_MASK (0x800 ~ 0xBD7) register. The detection plane represents the motion-detected cell excluding the mask cells among whole cells. The mask plane of MD arrayed box is enabled by the MDBOX_MSKEN (0x550[3] ~ 0x55F[3]) register and the detection plane is enabled by the MDBOX_DETEN (0x550[2] ~ 0x55F[2]) register. The color of mask plane is controlled by the MASK_COL register and the color of detection plane is defined by the DET_COL register. The mask plane of MD arrayed box shows the mask information according to the MD_MASK registers automatically and the additional narrow boundary of each cell is provided to display motion detection via the MDBOX_DETEN register and its color is a reverse cell boundary color. The plane can be mixed with video data by the MDBOX_MIX (0x550[1:0] ~ 0x55F[1:0]) register. Even in the horizontal / vertical mirroring mode, the video data and motion detection result can be matched via the MDBOX_HINV and MDBOX_VINV registers.

The TW2880 has 16 MD arrayed boxes so that 16 video channels can have its own MD arrayed box for motion display mode. To overlay mask information and motion result on video data properly, the scaling ratio of video should be matched with MD arrayed box size.



Register Settings

0x493[7:0] : Red color for motion box out boundary

0x494[7:0] : Green color for motion box out boundary

0x495[7:0] : Blue color for motion box out boundary

0x496[7:0] : Red color for motion box inner boundary

0x497[7:0] : Green color for motion box inner boundary

0x498[7:0] : Blue color for motion box inner boundary

0x499[7:0] : Red color for motion box mask area

0x49A[7:0] : Green color for motion box mask area

0x49B[7:0] : Blue color for motion box mask area

0x49C[7:0] : Red color for motion box plane area

0x49D[7:0] : Green color for motion box plane area

0x49E[7:0] : Blue color for motion box plane area

0x550 : MD Box control register

[7] : MD Box enable

[6] : operation mode select, “0” : table mode, “1” : motion display mode

[5] : cursor cell enable, “0” : display, “1” : enable

[4] : out boundary cell enable, “0” : display, “1” : enable

[3] : masking plane enable, “0” : display, “1” : enable

[2] : detection plane enable, “0” : display, “1” : enable

[1:0] : mixing control

“00” = 75% original pixel value / 25% plane (boundary) color mix
“01” = 50% original pixel value / 50% plane (boundary) color mix
“10” = 25% original pixel value / 75% plane (boundary) color mix

“11” = plane (boundary) color

0x550 ~ 0x55F : channel 1 ~ channel 16

0x560 : MD Box line width control register

[3:2] : vertical line width control

“00” = 1 line, “01” = 2 line, “10” = 3 line, “11” = 4 line

[1:0] : horizontal pixel width control

“00” = 1 pixel, “01” = 2 pixel, “10” = 3 pixel, “11” = 4 pixel

0x560 ~ 0x567 : channel 1 ~ channel 16

0x568[7:0] : MD Box horizontal left position LSB

0x569[2:0] : MD Box horizontal left position MSB

0x568 ~ 0x587 : channel 1 ~ channel 16

0x588[7:0] : MD Box vertical top position LSB

0x589[2:0] : MD Box vertical top position MSB

0x588 ~ 0x5A7 : channel 1 ~ channel 16

0x5A8[7:0] : horizontal cell size LSB

0x5A9[2:0] : horizontal cell size MSB

0x5A8 ~ 0x5C7 : channel 1 ~ channel 16

0x5C8[7:0] : vertical cell size LSB

0x5C9[2:0] : vertical cell size MSB

0x5C8 ~ 0x5E7 : channel 1 ~ channel 16

0x5E8[3:0] : horizontal motion cell number

0x5E8 ~ 0x5EF : channel 1 ~ channel 16

0x5F0[3:0] : vertical motion cell number

0x5F0 ~ 0x5F7 : channel 1 ~ channel 16

0x5F8[3:0] : horizontal cursor position

0x5F8 ~ 0x5FF : channel 1 ~ channel 16

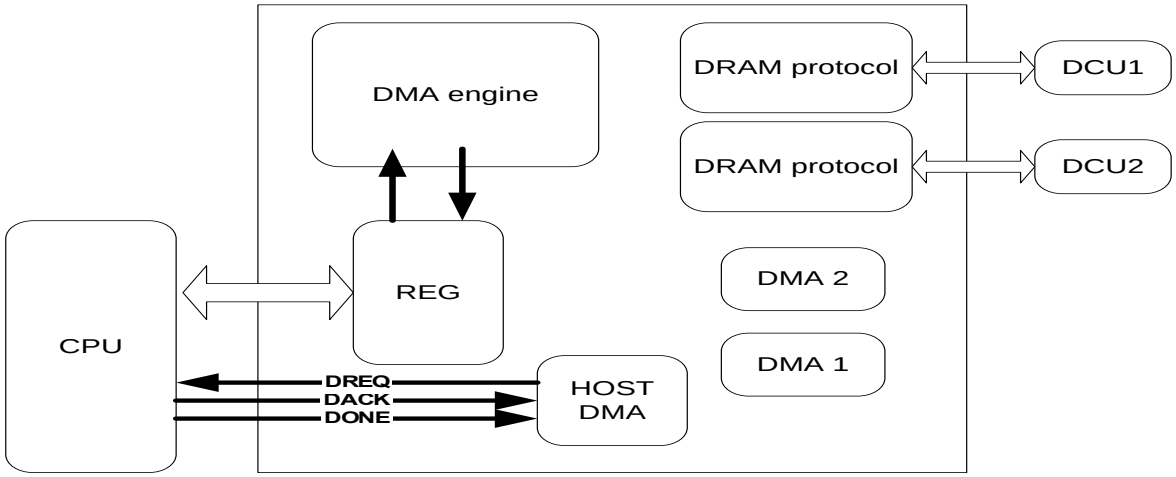
0x48B[3:0] : vertical cursor position

0x48B ~ 0x492 : channel 1 ~ channel 16

Section 8: DMA Function

Introduction

The TW2880's direct memory access controller (DMAC) is a second-generation platform block capable of performing complex data movements through 2 programmable channels, with minimal intervention from the host processor. The hardware micro-architecture includes a DMA engine that performs source and destination address calculations, and the actual data movement operations, along with a DRAM-based memory containing the transfer control descriptors (TCD) for the channels. DMAC can be programmed to move data between the host processor and TW2880's off-chip memory or between two locations in the off-chip DRAM independently with the minimum help from CPU. Two independent DMA channels are supported.



TW2880 DMAC

Features

DMAC subsystem features are summarized as follows:

- DMA Engine
- 2 independent channels that can move data between
  - Host processor and TW2880's LCD or recording memory
  - Two memory areas in TW2880's LCD or recording memory
- Programmable source / destination starting address

DMA Engine

TW2880 DMA operations include:

- SDRAM to SDRAM block moves (DRAM data copy)
- Host to SDRAM moves (OSG data transfer)

A DMA operation begins when software enables a DMA channel, after setting the source and destination starting addresses, transfer count, bus transaction size, and lock feature. The DMA Engine moves the data block, and the DMA operation ends when the number of bytes specified by the transfer count has been reached. A DMA operation may also end early by programmer. When a DMA operation ends, an interrupt is sent to the host processor. Status register in each channel can be used to identify the event that caused the interrupt: a normal operation end or any one of several types of error ends.

The exact transfer count of each DMA operation is controlled by Transfer Size Count Registers (0x242 ~0x243, 0x246 ~ 0x247). The data moving sequence is performed as a series of DRAM transactions. The burst memory transaction size is controlled by DCU and is not changeable from DMAC.

The source and destination starting addresses are set by firmware programmers. These addresses are incremented by the DMAC. All addresses are physical addresses. The DMA control information can be set by direct CPU writes to DMAC registers or alternatively, this DMA control information can be read from DMA descriptors stored in the off-screen memory.

DRAM INTERFACE

DMAC is a low priority DCU client. When transferring data to/from DRAM, the normal handshake is followed. If both channels are activated, an arbiter inside the DMAC will do the arbitration to share the DRAM bandwidth. In addition, the programmer needs to make sure that the addresses are correct in order to prevent overwriting. During the transactions, if time out situation occurs, both DMA channel will report the status and let host CPU decide if a software reset is needed.

EXTERNAL DMA DREQ/DACK PROTOCOL

There are two types of external DMA request/acknowledge protocols (Single service Demand, Single service Handshake). Each type defines how the signals like DMA request and acknowledge are related to these protocols.

Basic DMA Timing

The DMA service means performing paired Reads and Writes cycles during DMA operation, which can make one DMA operation. Figure 55 shows the basic Timing in the DMA operation.

- The setup time and the delay time of DREQ and DACK are the same in all the modes.
- If the completion of DREQ meets its setup time, it is synchronized twice and then DACK is asserted.
- After assertion of DACK, DMA requests the bus and if it gets the bus, it performs its operations.

DACK will be deasserted when DMA operation is completed.

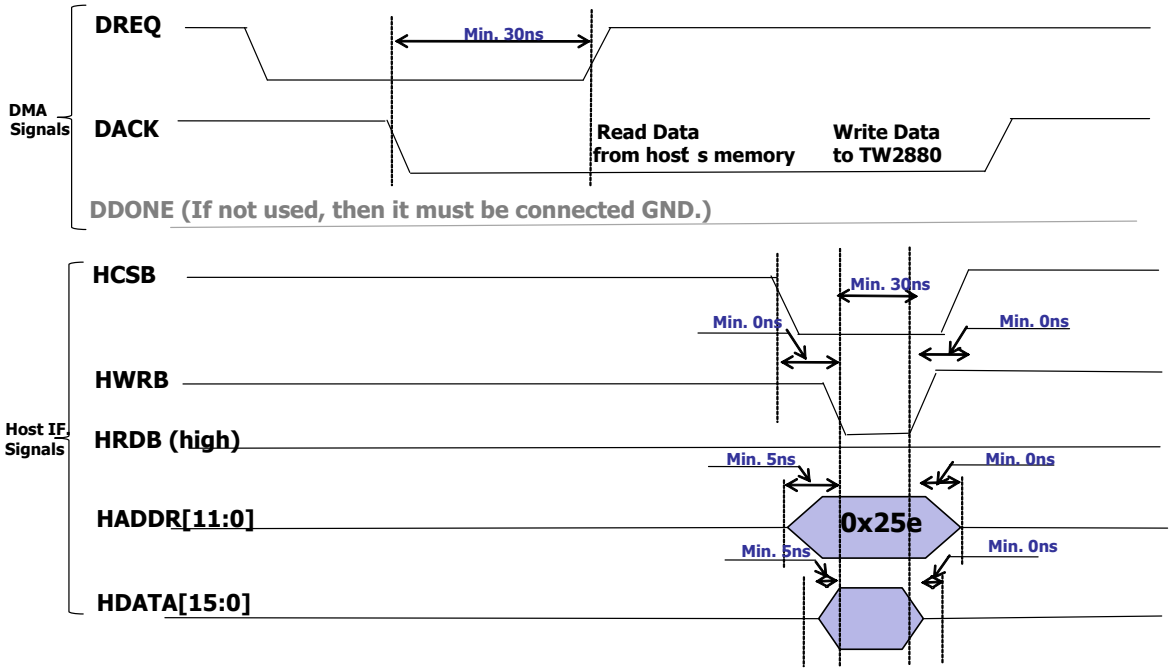


FIGURE 55. BASIC DMA TIMING DIAGRAM

DEMAND / HANDSHAKE MODE COMPARISON

Demand and Handshake modes are related to the protocol between DREQ and DACK. Figure 56 shows the differences between the two modes.

At the end of one transfer (Single/Burst 4 transfer), DMA checks the state of double- synched DREQ.

Demand Mode

If DREQ remains asserted, the next transfer starts immediately. Otherwise, it waits for DREQ to be asserted (described in a processor point of view).

Handshake Mode

If DREQ is negated, DMA negates DACK. Otherwise, it waits until DREQ is negated.

Caution: DREQ has to be asserted (low) only after the negation (high) of DACK (described in a processor point of view).

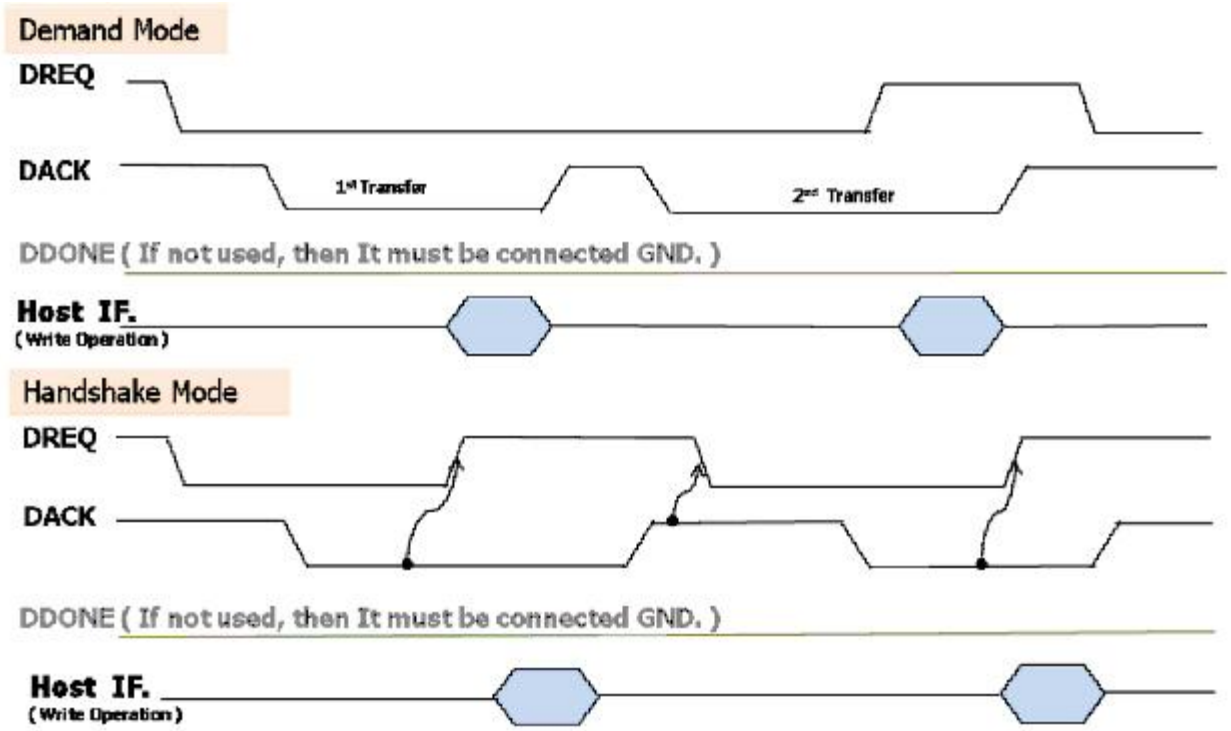


FIGURE 56. DEMAND/HANDSHAKE MODE COMPARISON (EXAMPLE: 2 TIMES TRANSACTION)

Transfer Size

There are two different transfer sizes: Single and Burst 4.

DMA holds the bus firmly during the transfer of the chunk of data. Thus, other bus masters cannot get the bus.

Burst 4 Transfer Size

Four sequential reads and Writes respectively are performed in the Burst 4 Transfer.

* Note: Single Transfer size: One read and one write are performed.

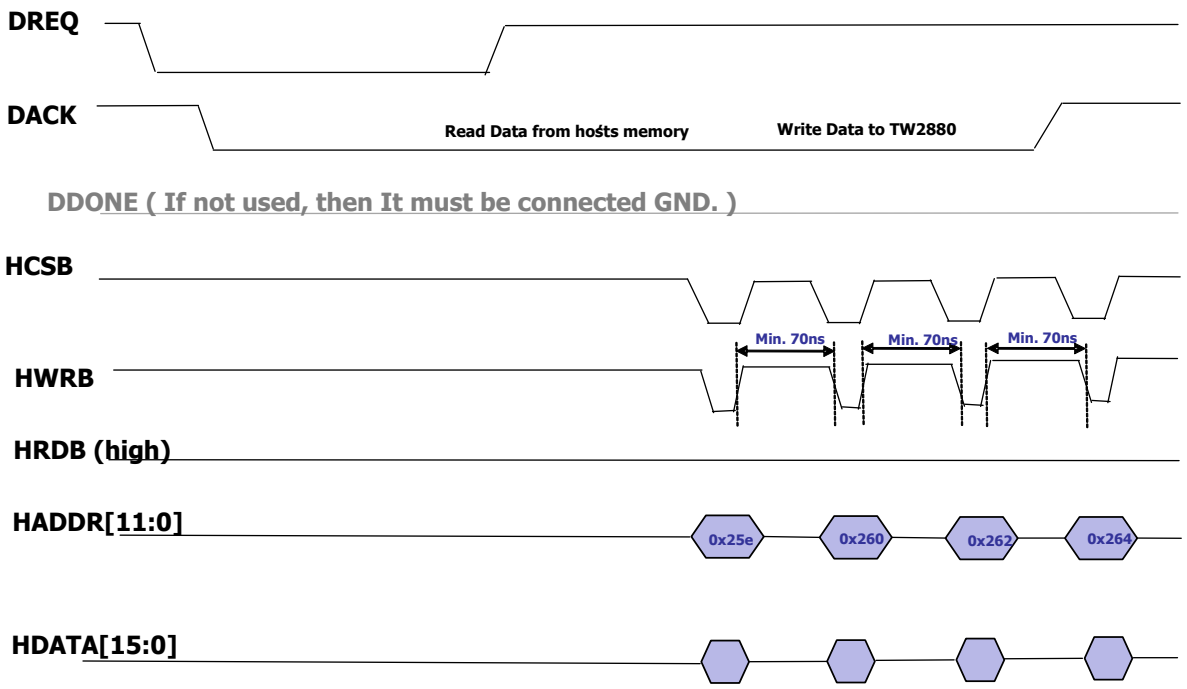


FIGURE 57. BURST 4 TRANSFER SIZE

EXAMPLES

Single Service in Demand Mode with Single Transfer Size

The assertion of DREQ is need for every single transfer (Single service mode). The operation continues while the DREQ is asserted (Demand mode), and one pair of Read and Write (Single transfer size) is performed.

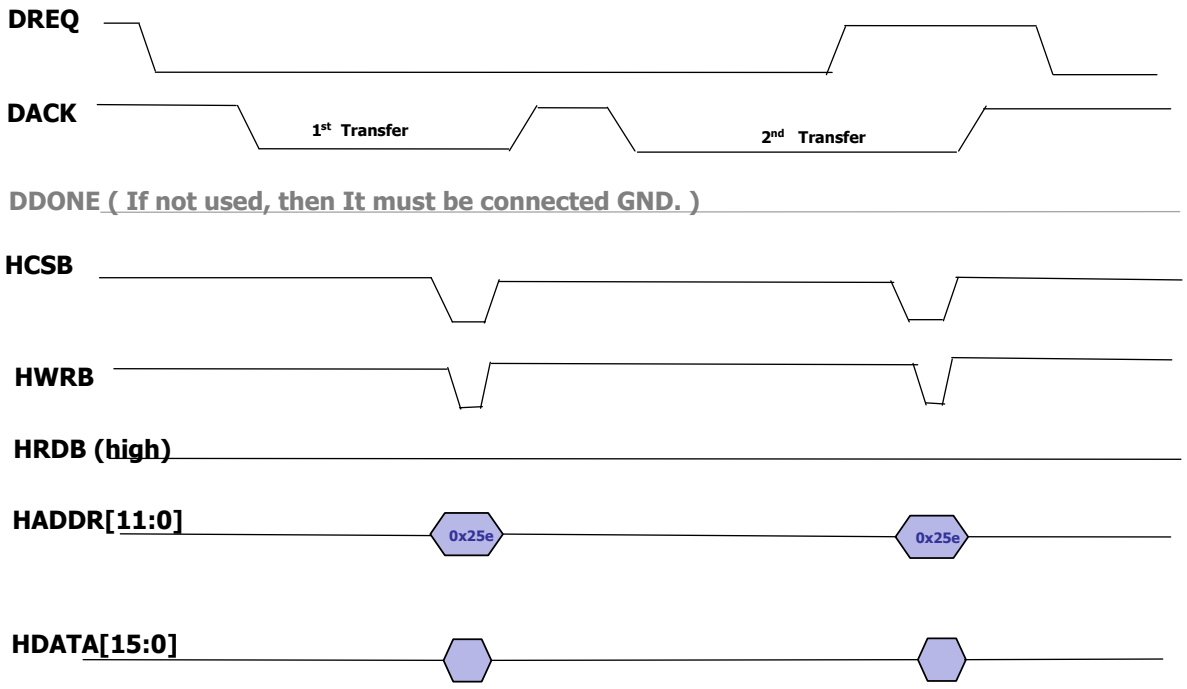


FIGURE 58. SINGLE SERVICE IN DEMAND MODE WITH SINGLE TRANSFER SIZE

Single Service in Handshake Mode with Single Transfer Size

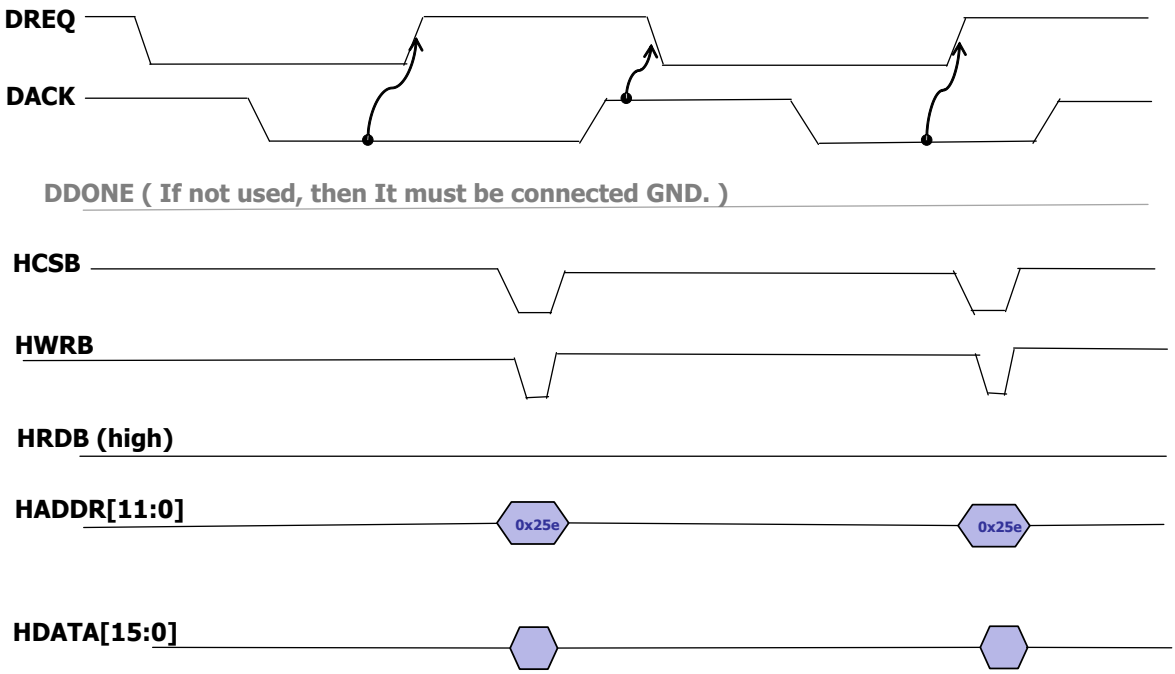


FIGURE 59. SINGLE SERVICE IN HANDSHAKE MODE WITH SINGLE TRANSFER SIZE

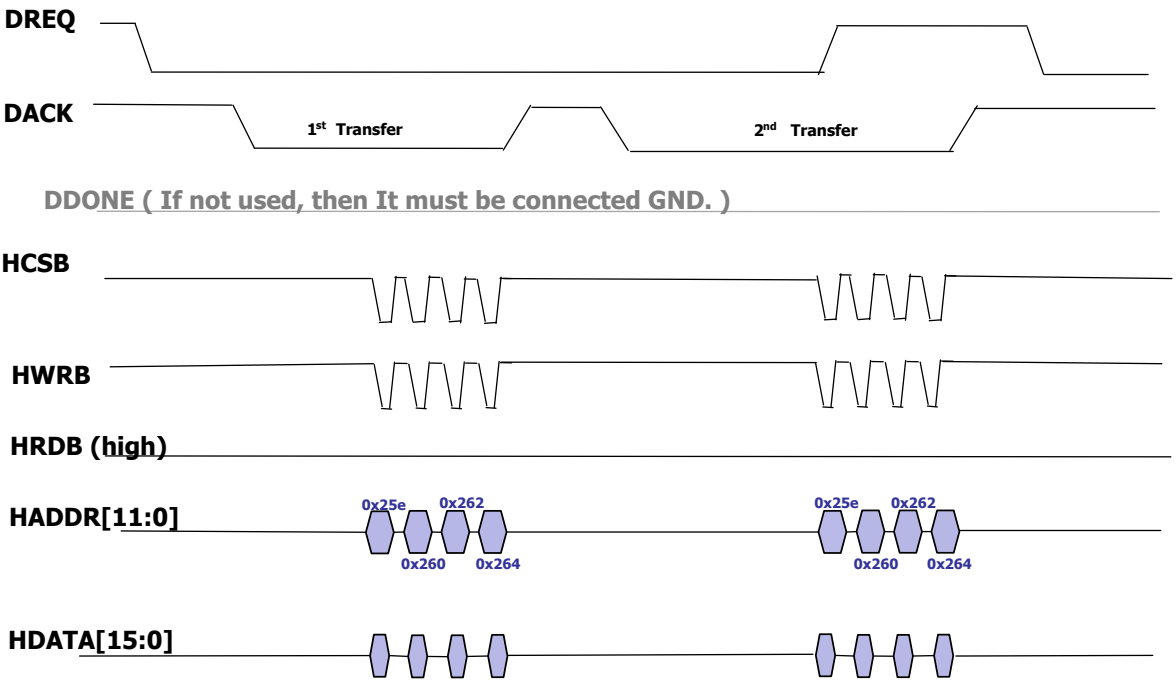


FIGURE 60. BURST 4 SERVICE IN DEMAND MODE WITH SINGLE TRANSFER SIZE

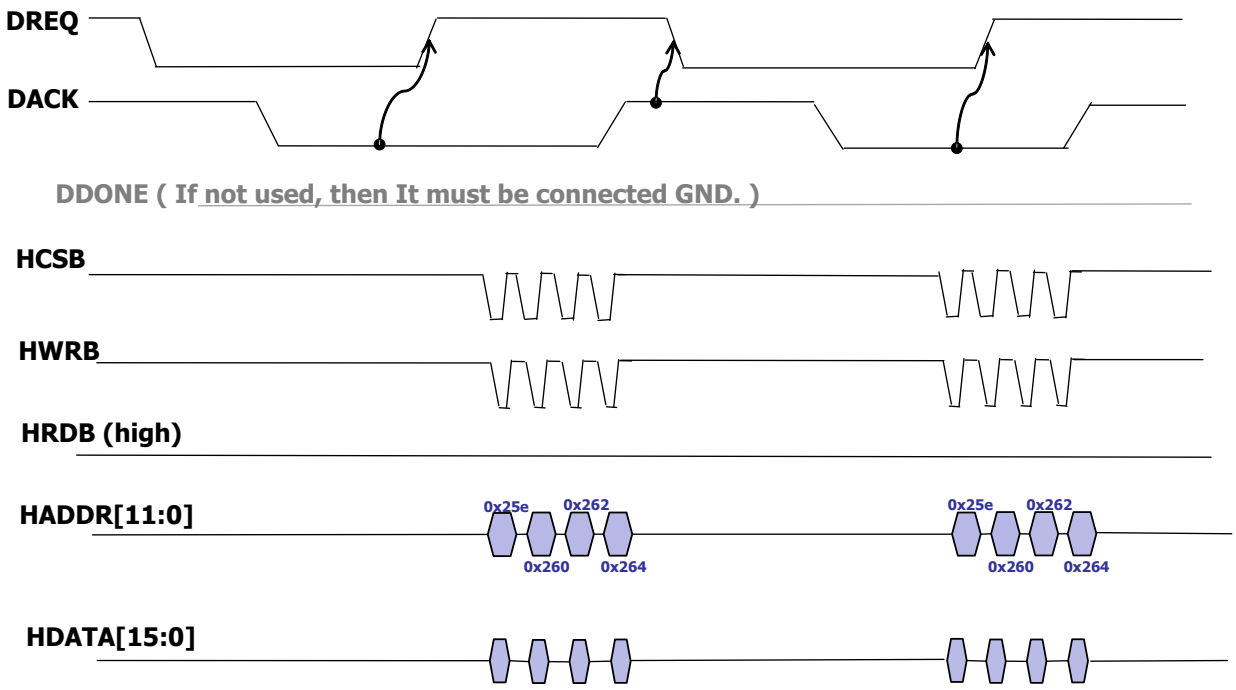
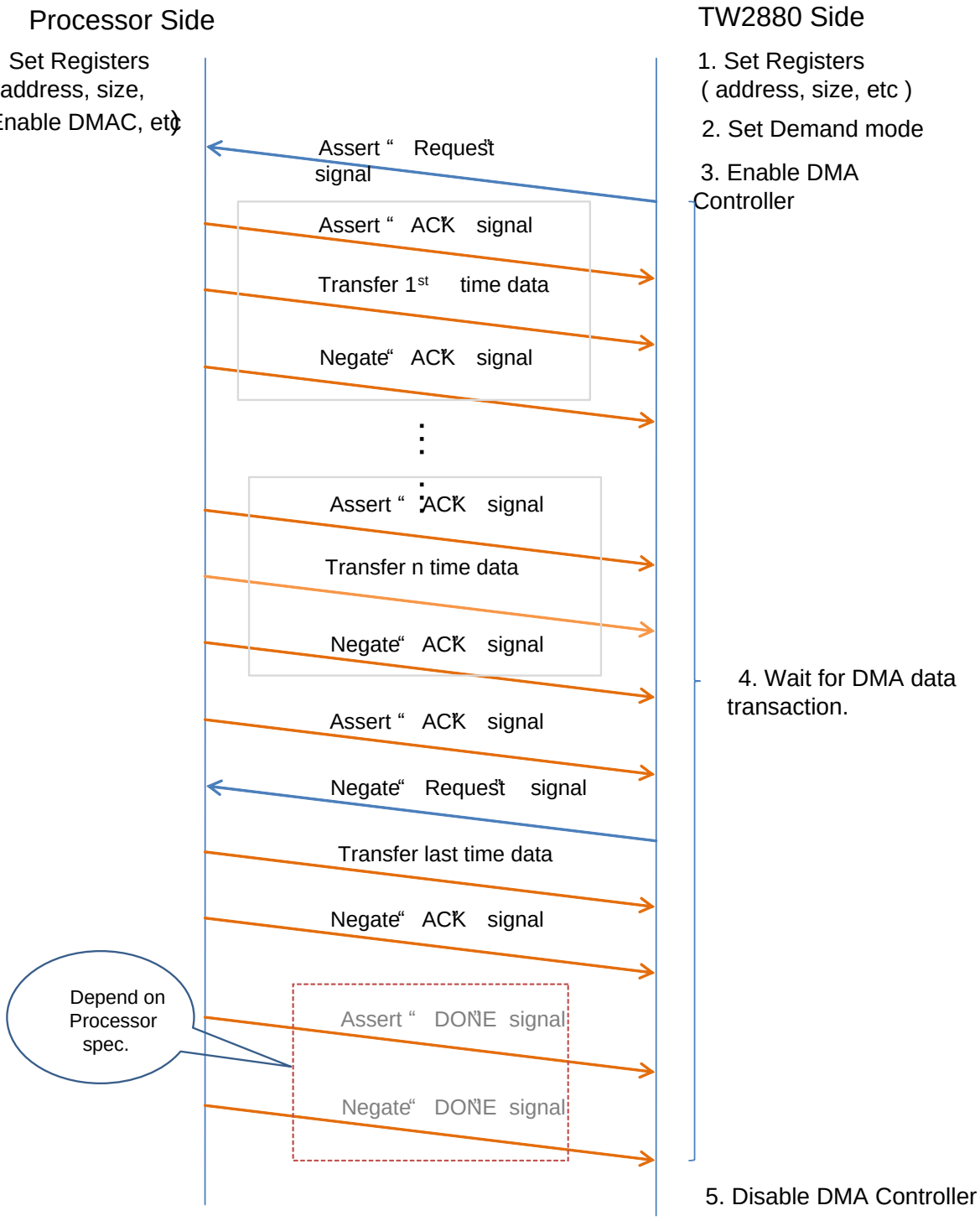


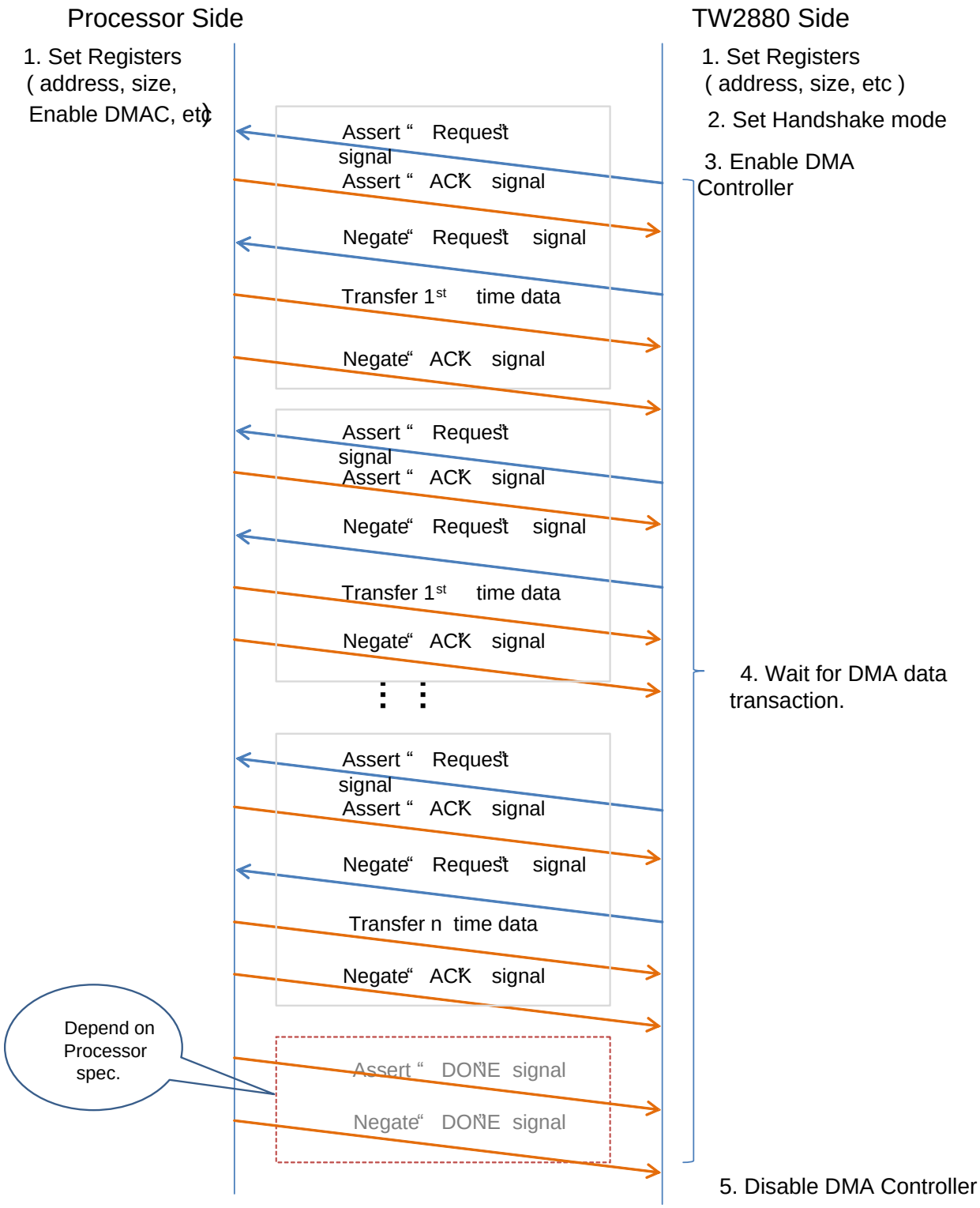
FIGURE 61. BURST 4 SERVICE IN HANDSHAKE MODE WITH SINGLE TRANSFER SIZE

DMA Function Software Example

Data Flow for DMA demand mode

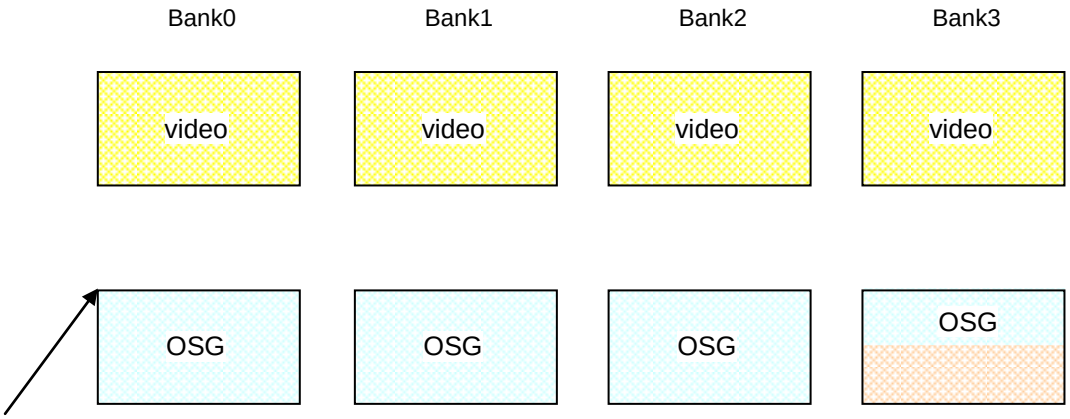


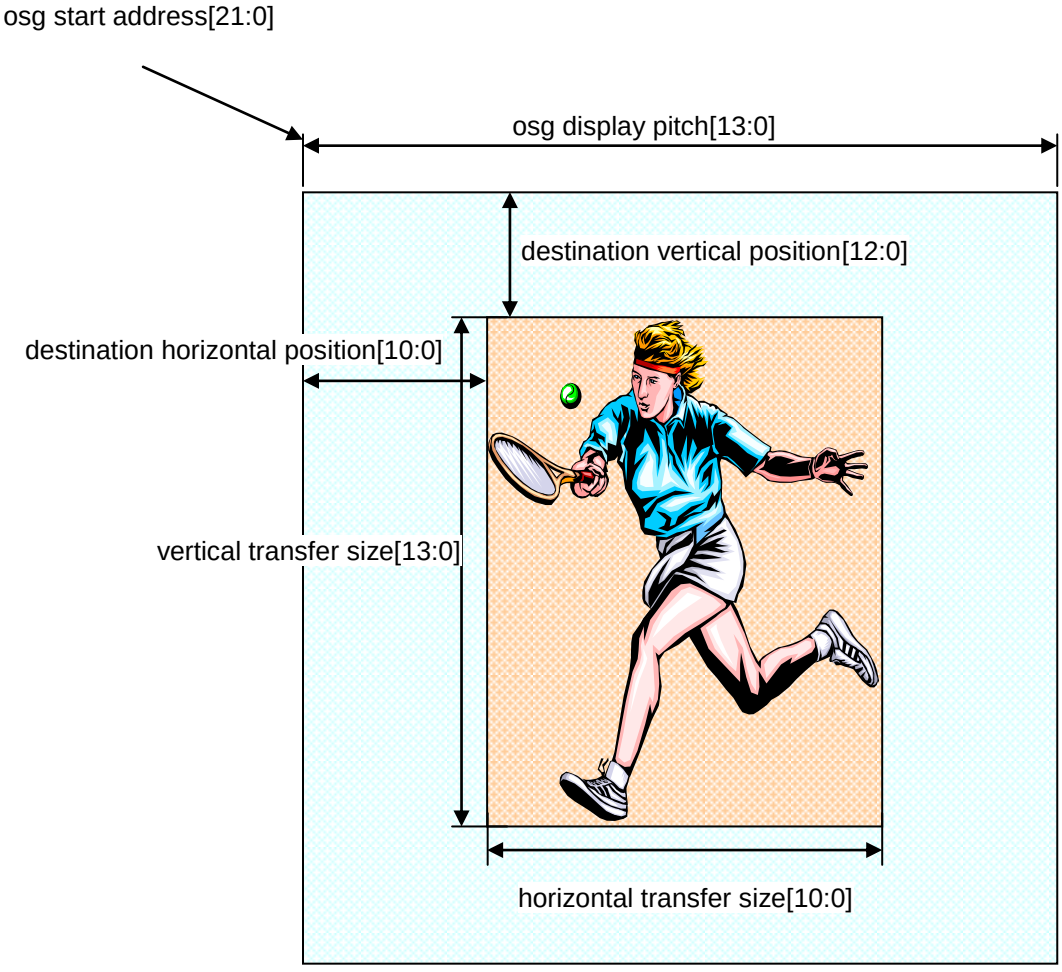
Data Flow for DMA handshake mode



Register Setting Example

Host to SDRAM Moves (OSG Data Transfer)





0x26A[7:0]: Host DMA start address LSB 1

0x26B[7:0]: Host DMA start address LSB 2

0x26C[5:0]: Host DMA start address MSB

0x268[7:0]: Host DMA display pitch LSB (1 pixel unit)

0x269[5:0]: Host DMA display pitch MSB

0x240[7:0]: Host DMA Destination vertical position LSB (line unit)

0x241[4:0]: Host DMA Destination vertical position MSB

0x242[7:0]: Host DMA vertical size LSB

0x243[5:0]: Host DMA vertical size MSB

0x244[7:0]: Host DMA destination horizontal position LSB (4 pixel unit)

0x245[2:0]: Host DMA destination horizontal position MSB

0x246[7:0]: Host DMA horizontal transfer size LSB

0x247[3:0]: Host DMA horizontal transfer size MSB

0x247[4:4]: Host DMA control (“0” : single service, “1” : not used)

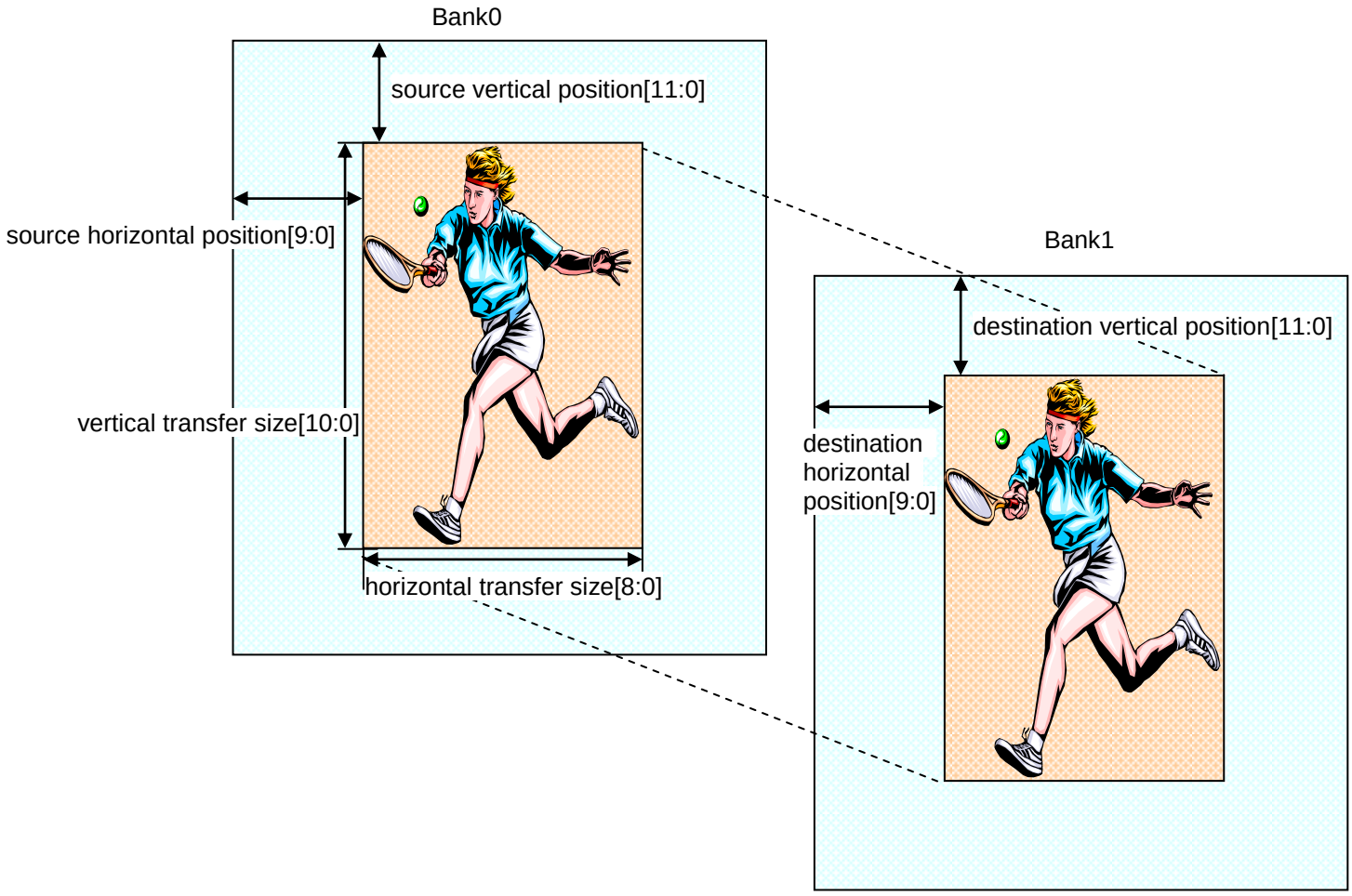
0x247[5:5] (“0” : demand mode, “1” : handshake mode)

0x249[4:4] (“0” : little endian, “1” : big endian)

0x249[5:5] (“0” : unit mode, “1” : 4 burst mode)

0x248[1:0]: Host & OSG DMA enable

DRAM Data Copy (Display DRAM)



0x230[7:0]: Display DRAM source vertical position LSB

0x231[3:0]: Display DRAM source vertical position MSB

0x231[5:4]: Display DRAM source bank

0x232[7:0]: Display DRAM destination vertical position LSB

0x233[3:0]: Display DRAM destination vertical position MSB

0x233[5:4]: Display DRAM destination bank

0x234[7:0]: Display DRAM vertical transfer size LSB

0x235[2:0]: Display DRAM vertical transfer size MSB

0x236[7:0] : Display DRAM source horizontal position LSB

0x237[1:0] : Display DRAM source horizontal position MSB

0x238[7:0] : Display DRAM destination horizontal position LSB

0x239[1:0] : Display DRAM destination horizontal position MSB

0x23A[7:0] : Display DRAM horizontal transfer size LSB

0x23B[0:0] : Display DRAM horizontal transfer size MSB

0x23C[0] : copy start

DRAM Data Copy (Record DRAM)

0x250[7:0] : Record DRAM source vertical position LSB

0x251[3:0] : Record DRAM source vertical position MSB

0x251[5:4] : Record DRAM source bank

0x252[7:0] : Record DRAM destination vertical position LSB

0x253[3:0] : Record DRAM destination vertical position MSB

0x253[5:4] : Record DRAM destination bank

0x254[7:0] : Record DRAM vertical transfer size LSB

0x255[2:0] : Record DRAM vertical transfer size MSB

0x256[7:0] : Record DRAM source horizontal position LSB

0x257[1:0] : Record DRAM source horizontal position MSB

0x258[7:0] : Record DRAM destination horizontal position LSB

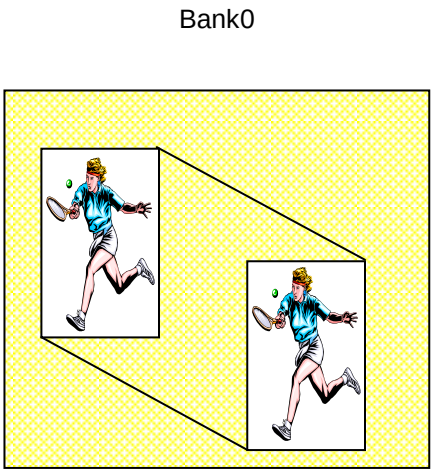
0x259[1:0] : Record DRAM destination horizontal position MSB

0x25A[7:0] : Record DRAM horizontal transfer size LSB

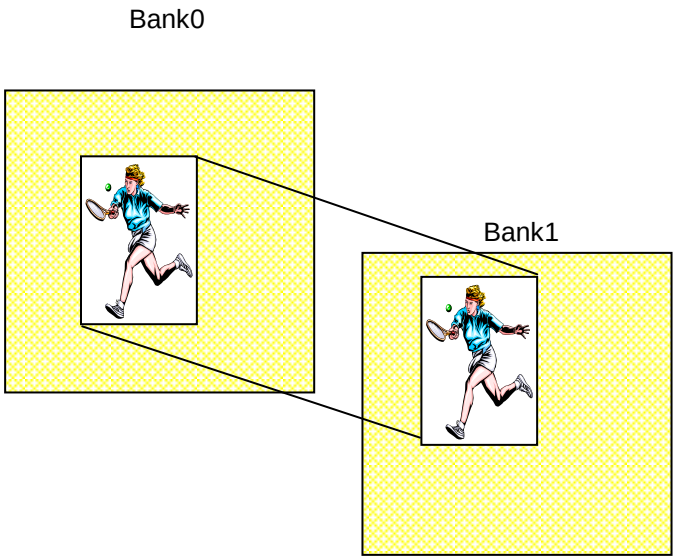
0x25B[0:0] : Record DRAM horizontal transfer size MSB

0x25C[0:0] : Record DRAM DMA enable

Ex) DRAM Copy to same bank



Ex) DRAM Copy to another bank



DMA Function Firmware Example

Introduction

TW2880 DMA function can be useful for writing OSG data or Graphic image such as Company logo at the WIN32 (channel window 32) on the main display or dual display area. DMA unit write a rectangular image to SDRAM at the position (x, y) from DMA memory start address. When host write Graphic image, the data format should be YUV 4:2:2.

There are two ways Host can write OSG data to SDRAM using DMA unit. One is DMA unit directly write the data to SDRAM. The other way is DMA unit pass the data to OSG unit, and OSG unit write the data to SDRAM. It operates depend on Reg 0x17E[1:0] setting. If OSG data is compressed, then CPU should use DMA through OSG write mode, and in this case, CPU should set OSG related registers with correct values as well as DMA registers.

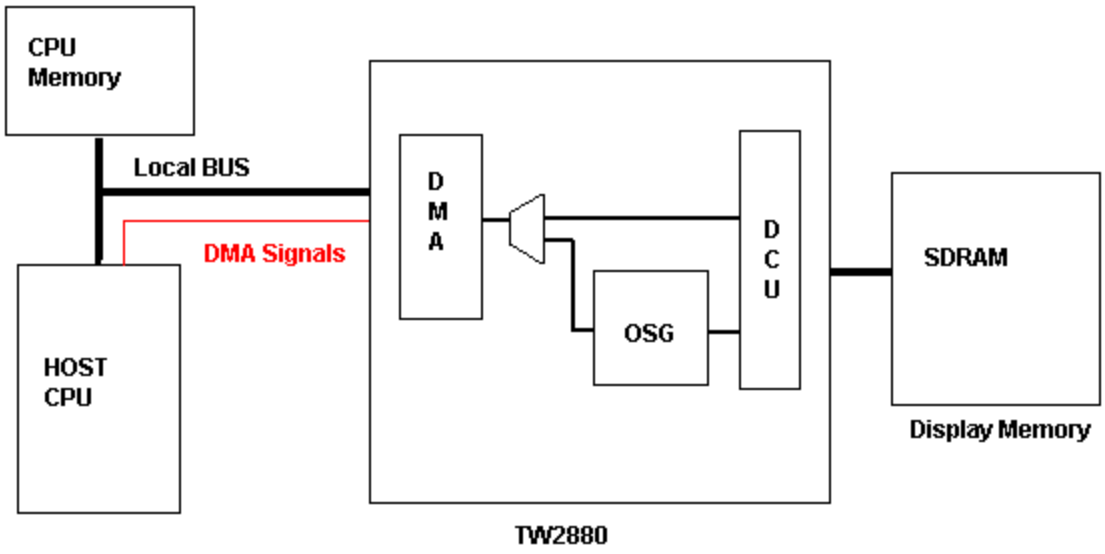


FIGURE 62. DMA OPERATION DIAGRAM

* DCU: RGB DRAM Control Unit

Local Bus: Host Interface using HCS (Hot Chip Select Signal) etc.

DMA Write Mode Sequence

Step 1 : Disable OSG Pass : Reg 0x17E[1:0] = 00

Step 2 : Set DMA Write Memory Start Address and DMA Pitch.

- If you write OSG data to OSG display area, DMA pitch and Memory start address should be same as OSG's.
- if you write Graphic image to main or dual display area:
  - DMA pitch = main display Pitch Reg value(Reg 0x210) * 16
  - DMA memory write start address should be 0x00.

Step 3 : Reset DMA unit: Reg 0x20d[6]: 1 -> 0

Step 4 : Set DMA Destination position (x, y), DMA Destination size(w, h),

- Position x, Width w: 4 pixel unit.

Setp 5 : Set DMA control mode : Single/Burst 4 mode, Hand/Demand mode.

- TW2880 cannot support Whole mode.

Step 6 : Set CPU DMA unit properly

Step 7 : Assign DMA Data Port: Reg 0x25E

Step 8 : Set DMA Write Enable: Reg 0x248[1:0]=11

Step 9 : Wait until DMA processing is done.

Step 10 : Clear DMA write Enable

- If data is for WIN32(channel window 32), CPU can use DMA COPY function to write same Graphic image to different bank of display area or if necessary, CPU can write different graphic image to each memory bank of the display memory for little bit animation effect.
    - OSG Data write firmware example is at “/tw2880/osg.c” in our reference source code.

    void OsgLoadBmpByDMA(U32 saddr, U16 dx, U16 dy, U32 hand, U32 burst)

    saddr : Source data start address in CPU side memory

    dx, dy : Destination position (x,y) in TW2880 side memory

    hand : hand mode enable

    burst : burst mode enable

- Graphic Image Data write firmware example is at “/tw2880/hostif.c” in our reference source code.

    int WriteDisplayToHostDMA(U32 addr, U16 dx, U16 dy)

    addr : Source data start address in CPU side memory

    dx, dy : Destination position (x,y) in TW2880 side memory

DMA Through OSG Write Mode Sequence

Step 1 : Disable OSG Wait Enable: Reg 0x162[4]=0

Step 2 : Set OSG write position(x,y), width and height(w,h)

Step 3 : Set OSG Mode (2bit Expension/Compression/RGB Format)

Step 4 : Set DMA Write Memory Start Address and DMA Pitch.

Step 5 : DMA pitch and Memory start address should be same as OSG's.

Step 6 : Reset DMA unit: Reg 0x20d[6]: 1 -> 0

Step 7 : Set DMA Count value(Reg 24e:24d:24c:24b)

Count Value is 2 byte unit. It should be multiplied by 4 in burst mode.

Step 8 : Set DMA control mode : Single/Burst 4 mode, Hand/Demand mode.

Step 9 : Enable OSG Pass : Reg 0x17E[1:0] = 11

Step 10 : Enable OSG Write Start Enable

Step 11 : Set CPU DMA unit properly

Step 12 : Assign DMA Data Port: Reg 0x25E

Step 13 : Set DMA Write Start Enable: : Reg 0x248[1:0]=11

Step 14 : Wait until DMA processing is done.

Step 15 : Clear DMA write Enable

Step 16 : Confirm OSG Busy signal (Reg 0x101[0]) = 0

- Compressed OSG Data write firmware example is at "/tw2880/osg.c" in our reference source code.

void OsgLoadBmpByDMA2OSG(U32 addr, U16 dx, U16 dy, U32 hand, U32 burst)

addr : Source data start address in CPU side memory

dx, dy : Destination position (x,y) in TW2880 side memory

hand : hand mode enable

burst : burst mode enable

Section 9: Audio Interface

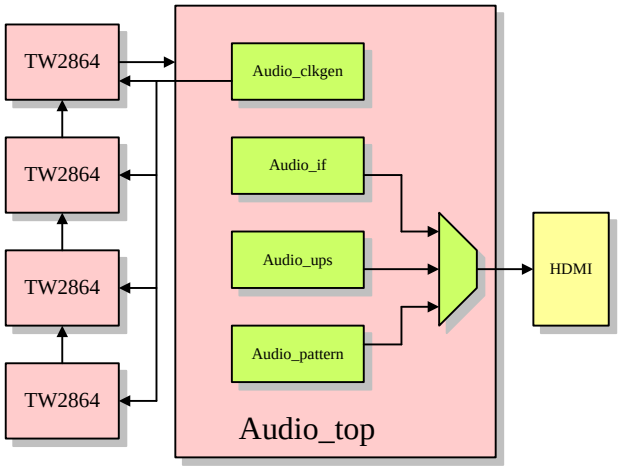
Introduction

Sitting between the TW2864 and TW2880's HDMI transmitter audio interface is a bridging block, which provides two important functions in TW2880's audio system: (1) converts the I²S data stream from TW2864 and (2) provides up sample option. The audio input format can be either I²S or SPDIF. However, HDMI IP core cannot support I²S format output directly from TW2864 because the data rate is too high for HDMI core. Audio interface will convert high data rate to low data rate and select required channel from I²S data source. Another function for audio interface is convert 8k or 16k sample rate to 32k sample rate since HDMI can only support audio sampling rate with 32k and up.

Features

- Audio clock generation (8k/16k/32k) for clock master mode
- Select one channel from 16 channels in I²S data stream
- Sample rate converter from 8k or 16k to 32k
- Audio pattern generation (1k/2k/4k/8k sin wave)
- Master clock is 256*fs
- Data rate from TW2864 is 256*fs and has 16 channel
- Data rate to HDMI is 32*fs and only has 1 channel, left and right are same
- 16-bit data

Block Diagram



Audio_clkgen is used for clock master mode. When sample rate is 8k or 16k, TW2880 provide audio master clock for TW2864. TW2864 generate sync and data. This module also generate 32k clock for up sample.

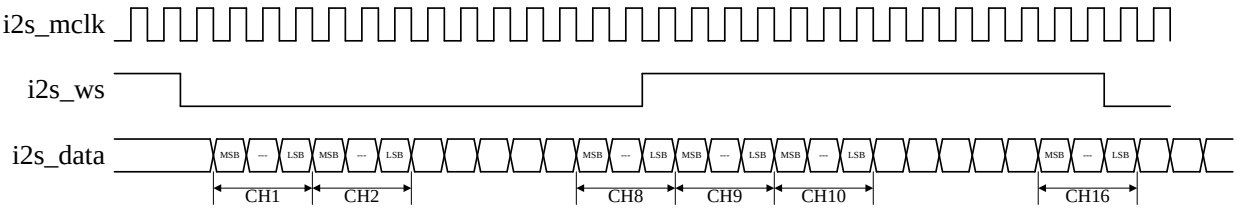
Audio_ups is used for up sample audio data from 8k or 16k to 32k. This module also selects one of the 16 channels.

Audio_if is used for select one channel from 16 channels without up sample

Audio_pattern generates sine wave pattern. There are four patterns: 1k, 2k, 4k or 8k.

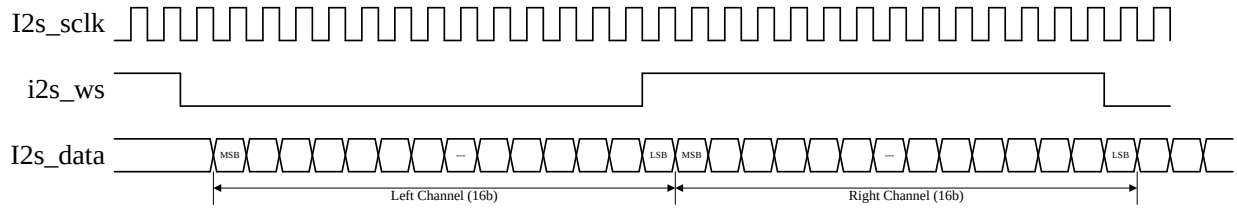
Timing Diagram

INPUT TIMING



i2s_mclk is from TW2864 (slave mode) or from TW2880 (master mode). It is 256 times of sample rate fs. For example, if sample rate fs is 32kHz, i2s_mclk is 256*32k=8.192MHz. I2s_ws is word select signal from TW2864. The frequency of i2s_ws is sample rate fs. I2s_data is data from TW2864. The data rate is 256fs. There are totally 16 channel data in one cycle of i2s_ws. Each data has 16bit. The sequence of channel can be programmed in TW2864.

OUTPUT TIMING



Output signals will go to HDMI link. The frequency of i2s_sclk is 1/8 of i2s_mclk frequency. It is 32*fs. Left channel and right channel have same data. Each data has 16bit. I2s_mclk will keep same with input mclk.

Working Mode

CLOCK SLAVE MODE

When TW2864 provides clock, TW2880 is in slave mode. TW2864 must run in 32k, 44.1k or 48k sample rate. Master clock is 256*fs. Data is 16bit.

CLOCK MASTER MODE

When TW2880 provides clock, TW2880 is in master mode. TW2864 runs in clock slave mode and sync master mode. In this mode, TW2880 can support 8k, 16k or 32k sample rate. In 8k, 16k sample rate, up sample is needed.

Register Setting Guide

HDMI AUDIO REGISTERS

1. Register [0x7A: 0x03] to [0x7A: 0x05]: ACR N value. This N value is calculated by this equation:128*fs = f(vclk) * N / CTS. In addition, N must meet: 128*fs / 1500Hz<= N <= 128*fs / 300 Hz. Here N and CTS are integer. N must be set by firmware, and CTS is calculated by hardware and can be read. The following table shows the recommended N value in different case.

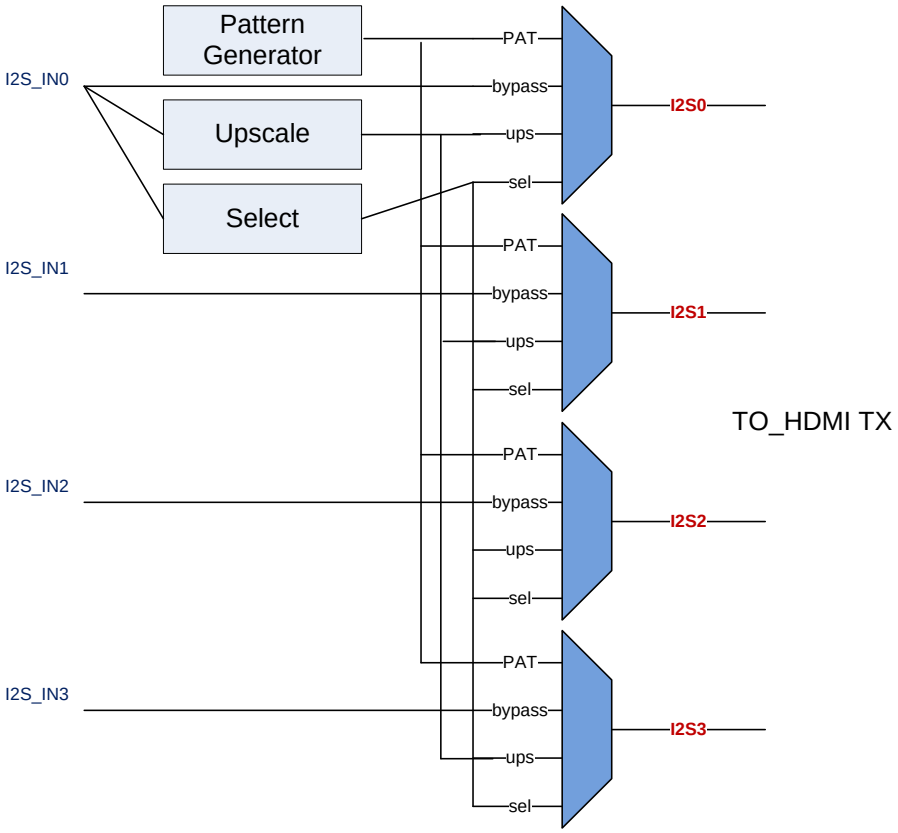
FS	VCLK	N	CTS
32kHz	148.5MHz	4096	148500
32kHz	74.25MHz	4096	74250
32kHz	108MHz	4096	108000
44.1kHz	148.5MHz	6272	165000
44.1kHz	74.25MHz	6272	82500
44.1kHz	108MHz	6272	12000
48kHz	148.5MHz	6144	148500
48kHz	74.25MHz	6144	74250
48kHz	108MHz	6144	108000

2. Sample Rate registers: [0x7A: 0x21] bit[3:0], [0x7A: 0x22] bit[7:4]. User must set correct sample rate using following table:

3	2	1	0	FS
0	0	1	1	32k
0	0	0	0	44.1k
0	0	1	0	48k

3. Register: [0x7A: 0x14] = 0x11, audio enable
4. Register: [0x7A: 0x2F] = 0x21, HDMI control
5. Register: [0x7A: 0x1D] = 0x40, I^2S control

TW2880 HDMI Audio Datapath



AUDIO INTERFACE REGISTERS

MODE	0X228	0X229
S	0x00	0x00
M 8K	0x00	0x60
M 16K	0x00	0x64
M 32K	0x00	0x48

1. TW2880 slave mode setting: [0x228] = 0x00, [0x229] = 0x00
2. TW2880 master mode 8k setting: [0x228] = 0x00, [0x229] = 0x60
3. TW2880 master mode 16k setting: [0x228] = 0x00, [0x229] = 0x64
4. TW2880 master mode 32k setting: [0x228] = 0x00, [0x229] = 0x48

TW2864 REGISTERS

MODE	0XCF	0XD2	0XDB	0XF0	0XF1	0XF2	0XF3	0XF4	0XF5	0XF8
S 32k	0x83	0x03	0xC1	0x0E	0xD6	0x26	0xDE	0x15	0x02	0xC4
S 44.1k	0x83	0x03	0xC1	0x65	0x85	0x35	0xBC	0xDF	0x02	0xC4
S 48k	0x83	0x03	0xC1	0x15	0x41	0x3A	0xCD	0x20	0x03	0xC4
M8K/16K/32K	0x83	0x03	0xC0							0xC4

1. TW2880 slave mode 32k setting: [0xCF] = 0x83, [0xD2] = 0x03, [0xDB] = 0xC1, [0xF0] = 0x0E, [0xF1] = 0xD6, [0xF2] = 0x26, [0xF3] = 0xDE, [0xF4] = 0x15, [0xF5] = 0x02, [0xF8] = 0xC4
2. TW2880 slave mode 44.1k setting: [0xCF] = 0x83, [0xD2] = 0x03, [0xDB] = 0xC1, [0xF0] = 0x65, [0xF1] = 0x85, [0xF2] = 0x35, [0xF3] = 0xBC, [0xF4] = 0xDF, [0xF5] = 0x02, [0xF8] = 0xC4
3. TW2880 slave mode 48k setting: [0xCF] = 0x83, [0xD2] = 0x03, [0xDB] = 0xC1, [0xF0] = 0x15, [0xF1] = 0x41, [0xF2] = 0x3A, [0xF3] = 0xCD, [0xF4] = 0x20, [0xF5] = 0x03, [0xF8] = 0xC4
4. TW2880 master mode 8k/16k/32k setting: [0xCF] = 0x83, [0xD2] = 0x03, [0xDB] = 0xC0, [0xF8] = 0xC4

Register Table

ADDRESS	R/W	DEFAULT	DESCRIPTION
0x228	R/W	0	AUDIO_CTRL[7:0] [7:6]: Port_sel [5]: Bypass_i2s [4]: Mute [3:0]: Ch_sel
0x229	R/W	0	AUDIO_CTRL[15:8] [7]: Asclk_sel [6]: Audio_master [5]: Ups_en [4]: Pat_en [3:2]: Rate_sel [1:0]: Pat_freq_sel

Register Description

AUDIO CONTROL 1 REGISTER – 0X228

BIT	R/W	DEFAULT	DESCRIPTION
7:6	R/W	0	**Port_sel[1:0]** There are four i²s data input pin. This register selects which pin will be sent to HDMI. 00: port 1 01: port 2 10: port 3 11: port 4
5	R/W	0	**Bypass_i2s** When enable this bit, i²s data will be directly sent to HDMI. For TW2864 I²S data, this bit must be set to 0
4	R/W	0	**Mute** 0: normal sound 1: mute
3:0	R/W	0	**Ch_sel** Select one of the 16 channels from TW2864

AUDIO CONTROL 2 REGISTER – 0X229

BIT	R/W	DEFAULT	DESCRIPTION
7	R/W	0	**Asclk_sel** ASCLK is for SPDIF format. This clock can be selected from external clock or from ck108 on chip. 0: external pin 1: internal 108MHz
6	R/W	0	**Audio_master** 0: slave mode, TW2864 provide clock 1: master mode, TW2880 provide clock
5	R/W	0	**Ups_en** 0: no up sample 1: up sample from 6k or 8k to 32k, only useful in master mode
4	R/W	0	**Pat_en** This bit enable internal pattern
3:2	R/W	0	**Rate_sel** Select sample rate, only useful in master mode 00: 8k 01: 16k 10, 11: 32k
1:0	R/W	0	**Pat_freq_sel** Pattern selection 00: 1k 01: 2k 10: 4k 11: 8k

Section 10: Differences Between C2 and B1

The Register Revision List for Recording Unit

Separated 'wr_page' Reference

In the TW2880B, there is only one 'wr_page' reference for 16 record write buffers and 16 SPOT write buffers. This reference is selected by the the register 0xC56[2:0]. Only one reference is not enough because we supports up to 8 read ports, 1 network port and 4 SPOTs. When read port is off and on, there is tearing.

In the TW2880C, each write buffer can select its own 'wr_page' reference by setting the the register 0xC6A[4:0] and 0xC6B[3:0]. This new scheme can turn on by setting the the register 0xC6B[4]

TABLE 26 THE REGISTER FOR SEPARATED 'WR_PAGE' REFERENCE

ADDR	TW2880B1	TW2880C2	DESCRIPTION
0xC6A[4:0]		New	**WR_BUF_ADDR** wr_buffer index for selecting 'wr_page' reference from read port **0 ~ 15 : recording write buffer 0 ~ 15** **16 ~ 31 : SPOT write buffer 0 ~ 15**
0xC6B[3:0]		New	**wr_page_sel** wr_page reference selection value 0 = port1 wr_page reference 1 = port2 wr_page reference 2 = port3 wr_page reference 3 = port4 wr_page reference 4 = port5 wr_page reference 5 = port6 wr_page reference 6 = port7 wr_page reference 7 = port8 wr_page reference 8 = network port wr_page reference 9 = SPOT1 wr_page reference 10 = SPOT2 wr_page reference 11 = SPOT3 wr_page reference 12 = SPOT4 wr_page reference **the others = port1 wr_page reference**
0xC6B[4]		New	**WR_PAGE_SEP** 0 = Use one wr_page reference **1 = Use separated wr_page reference according to the write buffer**

New Write Buffer Mapping for Read Port

In the TW2880B, there is bug for mapping write buffers to read ports. In the TW2880C, we fix this problem so that you can assign write buffers to each read port by setting the the register 0xCFD[4:0] and 0xCFE[7:0]. This new mapping can be turned on by setting the the register 0xCFD[5].

TABLE 27. THE REGISTER FOR NEW WRITE BUFFER MAPPING OF READ PORT

ADDR	TW2880B1	TW2880C2	DESCRIPTION
0xCFD[4:0]		New	**EN_ADDRI** Read port index 0 = port5, recording write buffer 0 ~ 7 selection 1 = port5, recording write buffer 8 ~ 15 selection 2 = port5, SPOT write buffer 0 ~ 7 selection 3 = port5, SPOT write buffer 8 ~ 15 selection 4 = port6, recording write buffer 0 ~ 7 selection 5 = port6, recording write buffer 8 ~ 15 selection 6 = port6, SPOT write buffer 0 ~ 7 selection 7 = port6, SPOT write buffer 8 ~ 15 selection 8 = port7, recording write buffer 0 ~ 7 selection 9 = port7, recording write buffer 8 ~ 15 selection 10 = port7, SPOT write buffer 0 ~ 7 selection 11 = port7, SPOT write buffer 8 ~ 15 selection 12 = port8, recording write buffer 0 ~ 7 selection 13 = port8, recording write buffer 8 ~ 15 selection 14 = port8, SPOT write buffer 0 ~ 7 selection 15 = port8, SPOT write buffer 8 ~ 15 selection 16 = network, recording write buffer 0 ~ 7 selection 17 = network, recording write buffer 8 ~ 15 selection 18 = network, SPOT write buffer 0 ~ 7 selection 19 = network, SPOT write buffer 8 ~ 15 selection
0xCFE[7:0]		New	**EN_DATA** New write buffer mapping control data According to the value of EN_ADDRI(0xCFD[4:0]), this value select write buffer for each read port 0 = disable, 1= enable
0xCFD[5]		New	**NEW_EN** New method for mapping wr_buffers to read ports 00 = original mapping 01 = new mapping, each read port selects write buffers by setting the the register 0xCFD[4:0] and 0xCFE[7:0]

New Field Signal Generation Scheme in the Field Interleaved Mode

In the TW2880B, there is bug for generating field signal when field of original source is not correct.

In the TW2880C, we fixed that problem. If you want to use new field signal generation scheme, you should turn on new schem by setting the the register 0xC6C[6].

TABLE 28. THE REGISTER FOR NEW FIELD SIGNAL GENERATION SCHEME IN THE FIELD INTERLEAVED MODE

ADDR	TW2880B1	TW2880C2	DESCRIPTION
0xC6C[6]		New	**FLD_HVCNT_SEL** Field generation option in the hvcnt_rout 0 = Old one 1 = New one

New Non-Real Time Field Interleaved Mode

In the TW2880B, there is bug for non-real time field interleaved mode. Frame buffer is not increased until the number of table index is done and we cannot control frame rate by using the channel table setting.

In the TW2880C, we fixed that problem. If you want to use new non-real time field interleaved mode, you just set the register 0xC7A[4:0].

TABLE 29. THE REGISTER FOR NEW NON-REAL TIME FIELD INTERLEAVED MODE

ADDR	TW2880B1	TW2880C2	DESCRIPTION
0xC7A[4:0]		New	**FLD_BANK_INC for Port 5 ~ Port 9:** Read bank number control in the field mode 0 = Original 1 = Bank is increased with real time frame rate

Bitmapped OSD

This function remains the same as B1 version.

Audio Interface Block

This function remains the same as B1 version.

Play Back Unit

PB unit has received a major change in TW2880C. First, the frame control method has updated to reflect the PAL mode stop and go issue. The input pin set to the PB unit has increased to 6 sets. A new field only saving mode is included.

TABLE 30. THE REGISTER REVISION LIST OF PLAY BACK UNIT

ADDR	TW2880B1	TW2880C2	DESCRIPTION
0x6F0, 0x6F1	N/A	PB display method	Control FRSC and field saving mode
0x6FC	same	same	Many new options, please read data sheet
0x6F9	N/A	New	External correction period
0x6FF	N/A	New	Internal correction period
0x6BA, 0x6B5	N/A	New	PAL tester timing replace
0x6B9	same	same	Add non standard ignore and IRG mask
0x3FF	same	same	Add PBX2_SEL, PBX4_SEL
0x3C6	same	same	Add RECX2_SEL, RECX4_SEL

Live Unit

Live unit has one a major change in TW2880C. The frame control method has updated to reflect the PAL mode stop and go issue.

TABLE 31. THE REGISTER REVISION LIST OF LIVE UNIT

ADDR	TW2880B1	TW2880C2	DESCRIPTION
0x6E8 - 0x6EF	N/A	Live display method	Control FRSC mode in different situations
0x3F8 – 0x3FB	N/A	Nonstd def	Non-standard definition control the registers
0x6F9	N/A	New	External correction period
0x6FF	N/A	New	Internal correction period
0x6BA, 0x6B5	N/A	New	PAL tester timing replace
0x6B9	same	same	Add non standard ignore and IRG mask
0x400		Modified	Weave mode up-scale enable => If [3] is '1', weave mode up-scale turn on

OSG

Several bugs in TW2880-B1 version are fixed now. This includes the FIFO overrun in low-resolution mode. OSG starting position limits and width limit. We also include a new feature: external OSG in slave mode.

TABLE 32. OSG BUG CORRECTION LIST

ADDR	TW2880B1	TW2880C2	DESCRIPTION
0x13D	same	same	Increase bitmap buffer start address to 24-bit for external OSG.
0x140, 0x141	N/A	New	External OSG Transparent color
0x17E[4]	N/A	New	Convert YCrCb data to RGB data

DMON Unit

ADDR	TW2880B1	TW2880C2	DESCRIPTION
LCD MODULE			
0x400		Modified	Weave mode up-scale enable => If [3] is '1', weave mode up-scale turn on
RGB_INTERFACE MODULE			
0x6B5 0x6BA		New	Sync time use test pattern sync enable (Write Mode) (0x6B5 : Ch1~Ch8, 0x6BA : Ch9~Ch16)
0x6B5 0x6BA		New	Channel non-standard status (Read Mode) (0x6B5 : Ch1~Ch8, 0x6BA : Ch9~Ch16)
0x6B9		Modified	non-standard control => If [7:6] is '2'b11', enable control bits Non-standard interrupt mask on/off => If [5] is '1', interrupt mask off (interrupt on)
DOWN_SCALER MODULE			
0x3F8		New	Non-standard upper line limit divide 2 for NTSC mode
0x3F9		New	Non-standard lower line limit divide 2 for NTSC mode
0x3FA		New	Non-standard upper line limit divide 2 for PAL mode
0x3FB		New	Non-standard lower line limit divide 2 for PAL mode

Host DMA

This function remains the same as B1 version.

OSD

This function remains the same as B1 version.

LCD Display Unit

This unit has several new functions and modifications. SBOX window has increased from four to eight.

ADDR	TW2880B1	TW2880C2	DESCRIPTION
0x505	same	same	Bit 7 is PHSYNC[9] bit

Simple OSD Unit

This function remains the same as B1 version.

DRAM Arbitration Control Unit

In TW2880C, we have implemented LCD Priority Arbitration and REC Priority Arbitration registers (register 0x280h ~ 0x285h) to control the priority of the clients. TW2880C's DRAM arbitration is implemented with round robin scheme, that means each client get the same priority treatment and will take turn to access DRAM with equal opportunity. However, in certain configurations, some clients may need DRAM service more often than other units. To accommodate this situation, the user can set the corresponding bit for that client, then this client will skip the round robin arbitration loop and get higher bandwidth service.

For example, LCD and RGBW requests are more critical than the other clients in a 3D 1080p native mode situation are. To support this situation, the user can set register 0x280h[0] and 0x280h[3] to one to let LCD and RGBW have higher priority.

LCD PRIORITY ARBITRATION 1 — 0X280 (NEW)

BIT	R/W	DEFAULT	DESCRIPTION
7:0	R/W	0	**Priority Arbitration[7:0]** Enable Clients arbitrate with high priority. The following table shows client enable bit of Priority Arbitration register for each module 0 : lcd 1 : dmon 2 : di_wr 3 : rgbw 4 : di_rd

BIT	R/W	DEFAULT	DESCRIPTION
			5 : spot_osd 6 : osgrd1 7 : osgrd2 8 : osgrd3 9 : dm_osgrd1 10 : dm_osgrd2 11 : host_dma 12 : osgw 13 : freeze 14 : cpu 15 : lcd_mouse 16 : dm_mouse

LCD PRIORITY ARBITRATION 2 — 0X281 (NEW)

BIT	R/W	DEFAULT	DESCRIPTION
7:0	R/W	0	Priority Arbitration[15:8] Enable Clients arbitrate with high priority

LCD PRIORITY ARBITRATION 3 — 0X282 (NEW)

BIT	R/W	DEFAULT	DESCRIPTION
7:0	R/W	0	Priority Arbitration[23:16] Enable Clients arbitrate with high priority

REC PRIORITY ARBITRATION 1 — 0X284 (NEW)

BIT	R/W	DEFAULT	DESCRIPTION
7:0	R/W	3	Priority Arbitration[7:0] Enable Clients arbitrate with high priority Following table show client enable bit of Priority Arbitration register for each module 0 : recw 1 : spw 2 : qcif

BIT	R/W	DEFAULT	DESCRIPTION
			3 : rout1 4 : rout2 5 : rout3 6 : rout4 7 : rout5 8 : rout6 9 : rout7 10 : rout8 11 : net 12 : spot1 13 : spot2 14 : spot3 15 : spot4 16 : mdw 17 : mdr 18 : dma_cpu2 19 : reserved

REC PRIORITY ARBITRATION 2 — 0X285 (NEW)

BIT	R/W	DEFAULT	DESCRIPTION
7:0	R/W	0	Priority Arbitration[15:8] Enable Clients arbitrate with high priority

Privacy Windows Unit

ADDR	TW2880B1	TW2880C2	DESCRIPTION
0xE4F-0xEDF	N/A	New	Privacy windows control
0xDD0-0xDFF	N/A	New	Privacy windows control

SPOT

SPOT unit has some minor updates.

ADDR	TW2880B1	TW2880C2	DESCRIPTION
0xF12[7]	N/A	New	SPOT1 Enable independent address read in quad mode
0xF72[7]	N/A	New	SPOT2 Enable independent address read in quad mode

ADDR	TW2880B1	TW2880C2	DESCRIPTION
0xFB1[7]	N/A	New	SPOT3 Enable independent address read in quad mode
0xFD2[7]	N/A	New	SPOT4 Enable independent address read in quad mode
0xF36[7:3] 0xF38[7:3] 0xF3A[7:3] 0xF3C[7:3]	N/A	New	SP1_FRC_CTL[4:0]- SP4_FRC_CTL[4:0] [4] Auto Correction Jumps two field when set to 1. One field if this bit set to 0. [3] Enable Correction [2:0] Select one of quad spot wr pages as reference 0: Select spot wr page 0,1,2,3 1: Select spot wr page 4,5,6,7 2: Select spot wr page 7,8,9,10 3: Select spot wr page 11,12,13,14 4-7: use quad position to select as reference
0xF36[2:0] 0xF38[2:0] 0xF3A[2:0] 0xF3C[2:0]	RW	0	SP1_OSD_CHNUM_VPOS[10:8] – SP4_OSD_CHNUM_VPOS[10:8] **Channel number information vertical position offset to each channel vertical start position. It is one pixel unit.**
0xF21[7:0]	N/A	New	SPOT memory address offset [7:0]
0xF89[7:4]	N/A	New	SPOT memory address offset [12:8]
0xFCB[1:0]	N/A	New	[1] spot4 power down control [0] spot3 power down control

CLKGEN

Some PB input clock control.

ADDR	TW2880B1	TW2880C2	DESCRIPTION
0x23F	N/A	New	PB Clock delay control [1:0] PB1 clk select 0, 1, 2, 3 ns delay [3:2] PB2 clk select 0, 1, 2, 3 ns delay [5:4] PB3 clk select 0, 1, 2, 3 ns delay [7:6] PB4 clk select 0, 1, 2, 3 ns delay
0x21C	N/A	New	PB Clock Invert control [4] PB1 clk invert [5] PB2 clk invert [6] PB3 clk invert [7] PB4 clk invert

Section 11: Firmware Change Summary

Rev.1.57

Release Date: 12/19/2010

ISSUE	TYPE	DESCRIPTION
None	None	First Relese for TW2880C
New DRAM Mapping to save memory bandwidth	Fixed Chip Bug	Add 0x26d, 0x05 in tbl_tw2880.txt
Support HDMI Receiver SIL9135	New	Add to test 4 HD support.
		SIL9135 has only one I2C address selection pin, so that we need to add I2C bus switch, which can allow to use 4 SIL9135 HDMI receivers. I2C BUS switch – Access command “i 71 1” or “I 71 2”. /tw2880/hostif.c WriteI2CCmd() /main/monitor.c MonWriteI2CCmd()
		Initialize SIL9135 HDMI receiver to get BT1120 16bit 4:2:2 YC format data stream. * Why HDMI Input color is wrong? HDMI Receiver SIL9135 : Reg0x4a = 0x9a or 0x92. Comming HD input is YVU or RGB format. If YUV is comming, then 4:2:2 bypass. Otherwise turn RGB2YUV convertor then 4:2:2. /device/sil9135.c
Dual Monitor OSG at low resolution display	Fixed Chip Bug	Don’t need to cut every 124 pixels in horizontal for dual monitor OSG. Just use DmOsgWindowInit().
OSG Writing/Block Fill	Fixed Chip Bug	Use original OsgBlockFill() function instead of OsgBlockFill2().

Rev.1.58

Release Date: 12/21/2010

ISSUE	TYPE	DESCRIPTION
Auto Mode 16D1 108MHz noise	Workaround	(1) Set rec port clk to 54MHz first then change 108MHz (??) (2) WQL(RegC58[5:4]) = 1 /tw2880/rec.c InitRecforMdsp()
STOP & GO	Fixed Chip Bug	Need to control Live Correction Reg0x6fc[5]. (1) PB Auto mode + Live : 0x6fc[5]=1 (Turn OFF correction) (2) PB normal mode + Live : 0x6fc[5]=0 (Turn ON correction) * PB correction bit is always disabled. /tw2880/pb.c SetRGBMode()

Rev.1.59

Release Date: 12/22/2010

ISSUE	TYPE	DESCRIPTION
OSG Protection Address	Workaround/ (New Chip bug)	(1) Based on mixium display memory area, calculate OSG_BASE_ADDRESS. /tw2880/osg.c InitOsg() max_osg_base = (PHR+dmPHR * 16)*(PVR+100)/4; *TW2880 maximum display resolution is 1080p (PHR=1920, PVR=1080), Dual monitor resolution is XGA(dmPHR=1024). (2) To set OSG Protection address only once in System_Init(), delete InitOsg() in other place. Otherwise, PROT_EN hold display when display resolution changing.

ISSUE	TYPE	DESCRIPTION
		(3) Add SetOsgProtectionMemoryAddrBase(). /tw2880/osg.c * OSG Protection address Reg writing sequence. - 6fc[7:6]=0, 6fc[4]=1 - set OSG_PROTECT_BASE reg. - 6fc[7:6]=1, 6fc[4]=1
Support 1080i@50Hz	New	Add in tbl_tw2880.txt
HFPORCH[8:0]	New	Change 9bit HFPORCH to support 1080p50Hz
Support 4HD mode using REC as PB input	New	(1) When REC port 1~8 is disable, REC pin become input for HD. (Default is all OFF) (2) Be careful about data input/output direction. It may kill chips. Add “mdsp mode 14” command /tw2880/disp InitMdsp_4HD_PB() /tw2880/rec.c SetRecPort() InitRecforMdsp() /tw2880/system.c InitTw2880() SoftwareReset() * Leave softreset REC_enc, REC_dec. This mode need OPTION SW #2=LOW before POWER ON. This mode uses SIL9135 HD daughter board.
SPOT3 shaking	Workaround	Add sequence for SPOT register setting: (1) Set Reg 0xfa2[7:6] = 10 first. (2) Set other spot registers - spot buffer, scalers, etc (3) set new added spot registers WriteP(0xf9b, 0x0b); WriteP(0xfcb, 0x0b); WriteP(0xf07, 0x00); //WriteP(0xf12, 0x84);

ISSUE	TYPE	DESCRIPTION

		WriteP(0xf36, 0x60);
		
		WriteP(0xf38, 0x60);
		WriteP(0xf67, 0x00);
		//WriteP(0xf72, 0x84);
		
		WriteP(0xf3a, 0x60);
		WriteP(0xf97, 0x00);
		//WriteP(0xfa2, 0x84);
		
		WriteP(0xf3c, 0x60);
		WriteP(0xfc7, 0x00);
		//WriteP(0xfd2, 0x84);
		
		/tw2880/spot.c
		SpotMemoryStart()
		SpotInit()

Rev.1.60

Release Date: 12/28/2010

ISSUE	TYPE	DESCRIPTION
Support SPOT Rotate frequency commnd	New	Add command “spot rot time n”, n=1~10. /tw2880/spot.c sRotateTime SpotTest()

Rev.1.66

Release Date: 5/27/2011

ISSUE	TYPE	DESCRIPTION
OSG Protection function recover	Chip Bug Fix	Removed firmware workaround solution of TW2880C1 Reg 0x6FC[7:6]=0
Display Memory Bandwidth	New	In case of 1080p HD display, TW2880 can have better performance in memory bandwidth when memory pitch is multiple of 2048 Byte. Both Display_Pitch(Reg0x210)*16 and OSG_Pitch(Reg0x219:0x218, Reg0x1CD:0x1CC) changed to 2048. /include/config.h #define OSGPITCH 2048 /tw2880/system.c ChangeMainDisplayTiming() /tw2880/disp.c InitDualMonitor() /tw2880/osg.c InitOsg() For 6CH mode, I changed Dualmonitor Memory writing position under the main display. Osg memory position is below after DualMonitor.
Fixed 32 Window Noise issue	Chip Bug Fix	Diskplay Memory controller arbitration priority: Reg0x280 = 0x0b Record port clock sequential control: 54MHz -> 108MHz
REC New Field Switch mode	New	Added REC new field switch reference program: 1. Reg0xC77[5:0]= [rport9 - rport5], Reg0xC7A[5:0]= [rport9 - rport5]

		2. Recbuf : Set frame mode 3. Recport: Set field mode (rport5~8 only support) Set Interlaced Frame Interleave
ISSUE	**TYPE**	**DESCRIPTION**
		4. Recpin : 27MHz (HalfD1 filed SW case) 5. Reg0xE46[7:0]= [rport9 - rport0] : Formatter 6. PB DNS : Vertical Target size will be double because of only one field. ex) V.Source size=240, V.Target size=240 (1:1 ratio, HalfD1 case) (7.old field sw mode disable: RegCF2:CF1=0) - Created Reference : D1 one field switch mode ("mdsp mode 1c", "mdsp mode 1d") - Why jumping? if you turn on 2D, you could see jumping because of only one field. Better use Weave mode. /tw2880/disp.c /tw2880/rec.c