SITARA™ ARM® PROCESSORS

BOOt camp

LCD Controller

LAB: None

Author: Texas Instruments®, Sitara™ ARM® Processors

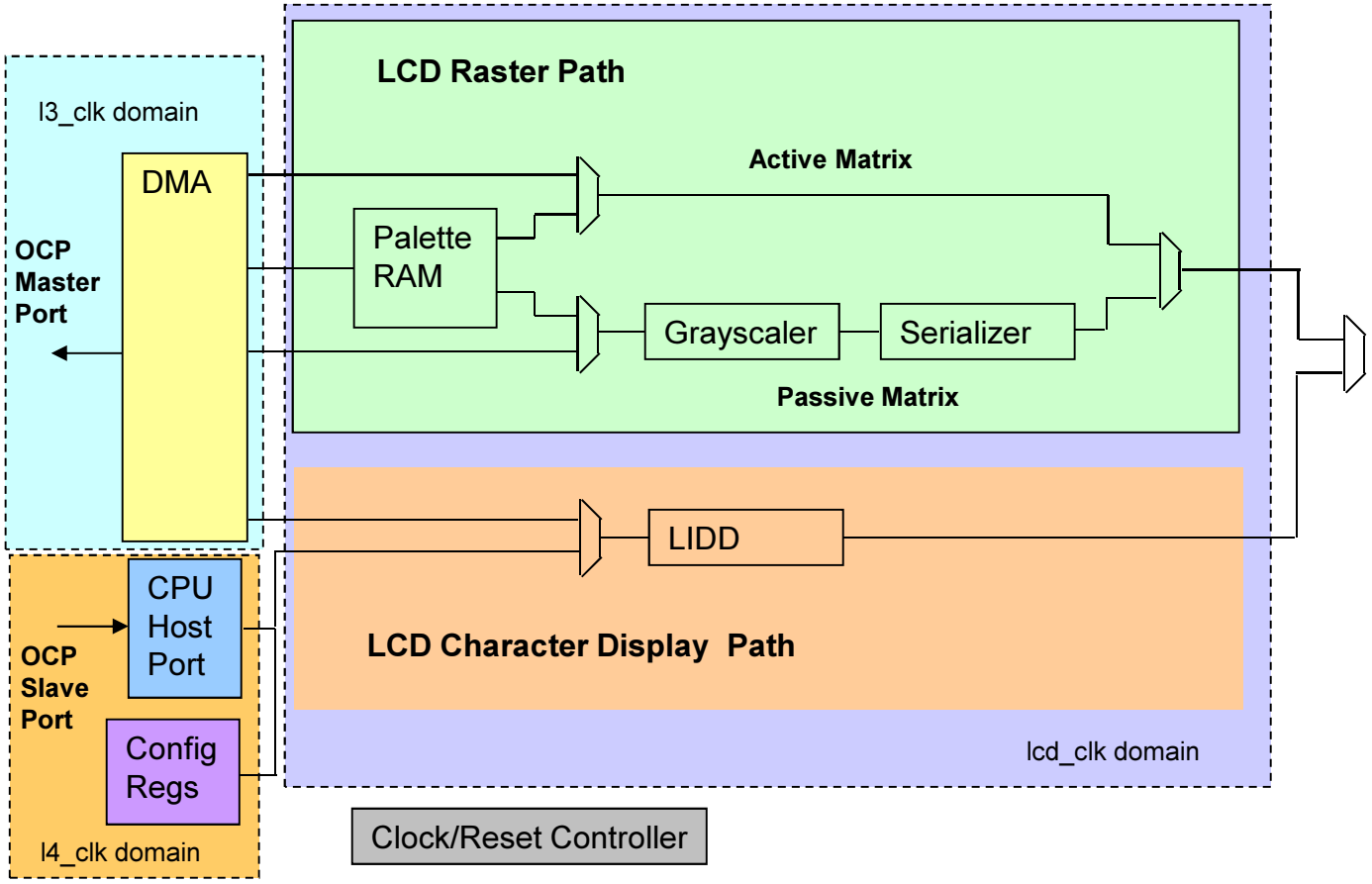
Sept 2012

LCDC Features

- LIDD Character/Graphic Based Panels with internal frame buffer
  - Supports 2 Panels (CS0 and CS1) with independent and programmable bus timing parameters when in asynchronous Hitachi, Motorola and Intel modes
  - Supports 1 Panel (CS0) with programmable bus timing parameters when in synchronous Motorola and Intel modes
  - Can be used as a generic 16 bit address/data interleaved MPU bus master with no external stall
- Passive Matrix LCD Panels
  - Panel types including STN, DSTN, and C-DSTN
  - AC Bias Control
- Active Matrix LCD Panels
  - Panel types including TN TFT
- OLED Panels
  - Passive Matrix (PM OLED) with frame buffer and controller IC inside the Panel
  - Active Matrix (AM OLED)

LIDD = LCD Interface display Driver

High Level Block Diagram



RASTER MODE

Raster - Features

- Single panel support
- Programmable pixel display modes (1, 2, 4, 8, 12, 16, and 24 bit)
- Pallet RAM support for 1, 2, 4, and 8-bit pixel modes
- Passive Matrix Panel Support
- Active Matrix Panel Support
- 24-bit Packed data mode
- 126MHz maximum pixel clock

Raster – Pixel CLK, Frame Rate, & Resolution

Number of PCLKs Per Line = PPL + HSW + HFP + HBP

Number of PCLKs Per Frame = (LPP + VSW + VFP + VBP) * PCLKs Per Line

Refresh/Frame Rate = PCLK Freq / PCLKs Per Frame

```
PCLK = Pixel Clock
PPL  = Pixels Per Line (X resolution of LCD)
LPP  = Lines Per Panel ( Y resolution of LCD)
HSW  = Horizontal Sync Width
HFP  = Horizontal Front Porch
HBP  = Horizontal Back Porch
VSW  = Vertical Sync Width
VFP  = Vertical Front Porch
VBP  = Vertical Back Porch
```

Raster – Memory Bandwidth

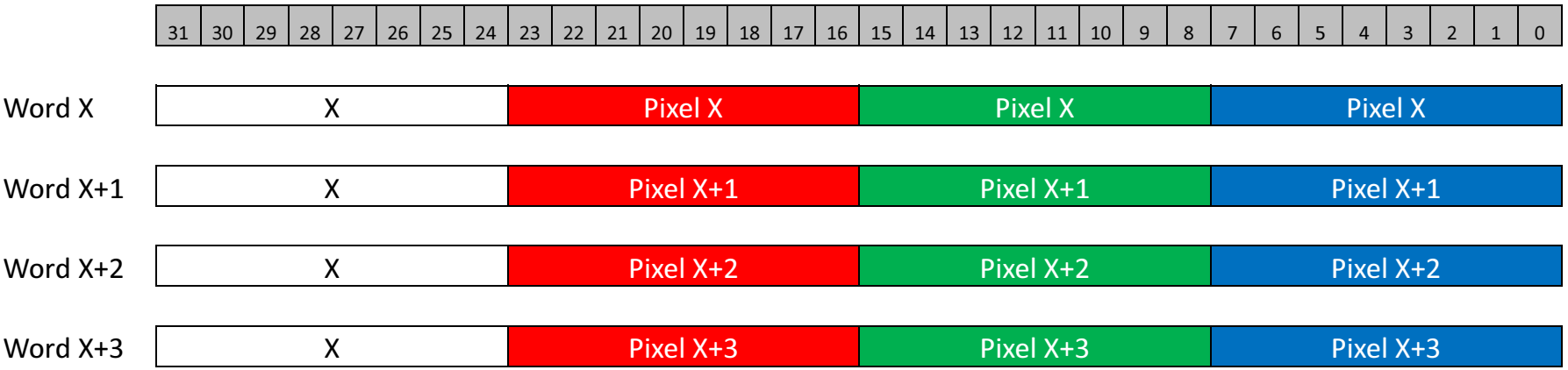
To calculate the memory bandwidth requirement for a 16-bit or 24-bit display:

MB/s = PPL x LPP x Bytes Per Pixel x Frame Rate

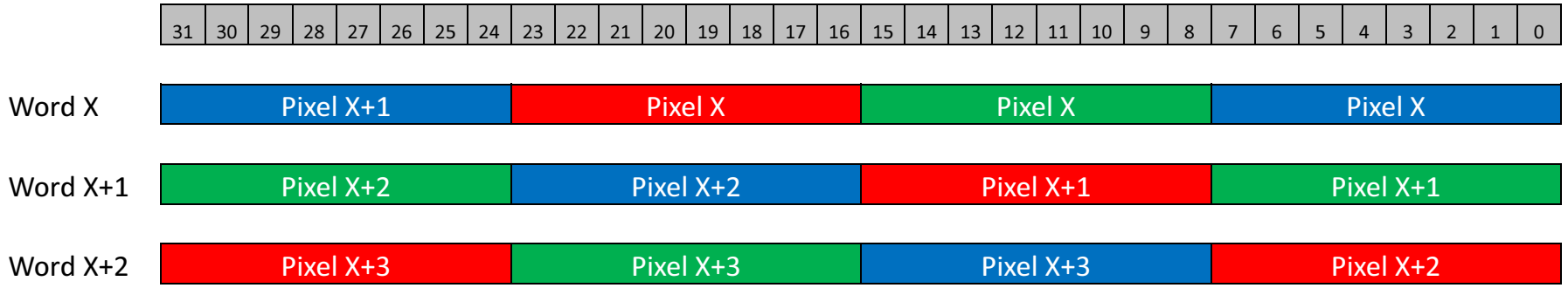
When LCDC is 16-bit mode the Bytes Per Pixel = 2
When LCDC is set to 24-bit mode the Bytes Per Pixel = 4

The higher the refresh rate and LCD resolution, the high the bandwidth requirement!

Raster – 24bit Data Format



24-bit Unpacked Format

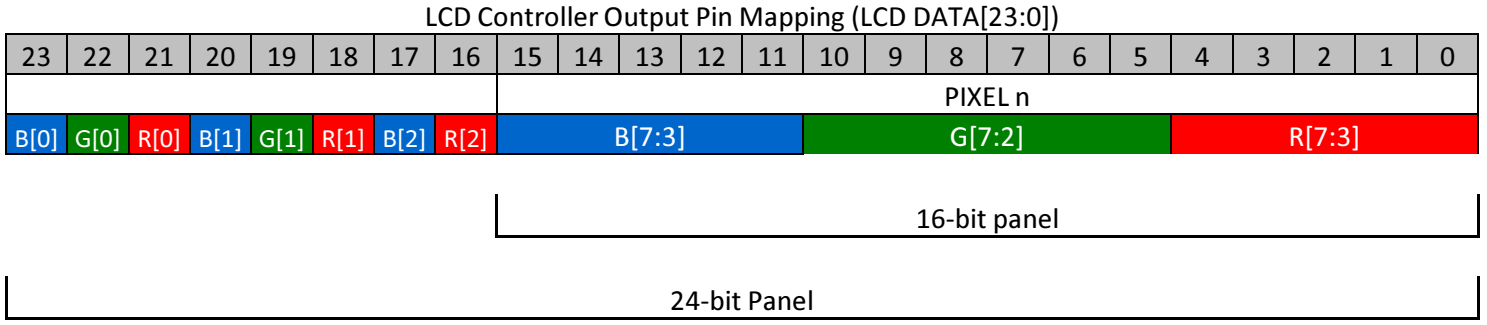


24-bit Packed Format

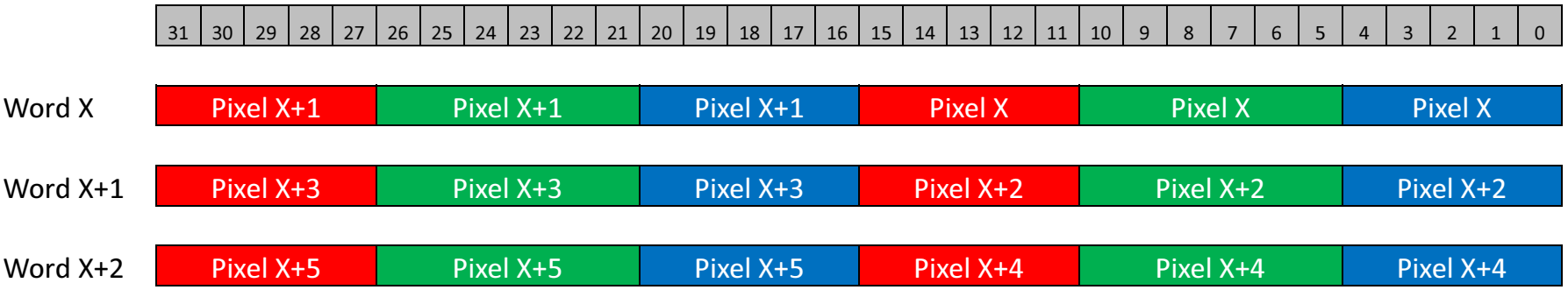
Using packed format reduces frame buffer size by 25%

LCDC Connections – 24-bit RGB Mode

The blue and red color assignments to the LCD data pins are reversed when operating in RGB888 (24bpp) mode compared to RGB565 (16bpp) mode. In order to correctly display RGB888 data from the SGX, or any source formatted as RGB in memory, it is necessary to connect the LCD panel as shown in. Using the LCD Controller with this connection scheme limits the use of RGB565 mode. Any data generated for the RGB565 mode requires the red and blue color data values be swapped in order to display the correct color.



Raster – 16-bit Data Format

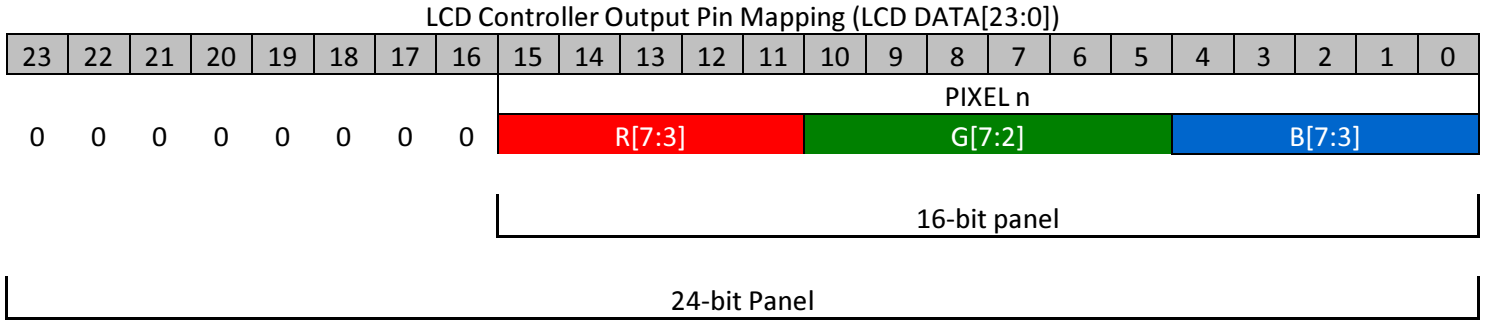


The frame buffer size for 16-bit is 50% less than the unpacked 24-bit.

However the is a noticeable reduction in color depth.

LCDC Connections – 16-bit RGB Mode

When operating the LCD Controller in RGB565 mode the LCD panel should be connected as shown in Figure 3. Using the LCD Controller with this connection scheme limits the use of RGB888 mode. Any data generated for the RGB888 mode requires the red and blue color data values be swapped in order to display the correct color.



LIDD MODES

LIDD - Features

- Intel 8080, Motorola 68K, and Hitachi Modes
- 2 Asynchronous Panels (same mode) or 1 Synchronous Panel
- Up to 42MHz clock rate using OPP100
- Can be used as a generic 16 bit address/data interleaved MPU bus master (external stall not supported).
- DMA support for DATA (Always uses fb0)
- CPU support for DATA and COMMAND/ADDRESS

LIDD - Signal Mapping

Ball Name	Raster	Motorola SYNC	Motorola ASYNC	INTEL SYNC	INTEL ASYNC	Hitachi
lcd_pclk	PCLK	EN	EN	Rs	Rs	N/A
lcd_hsync	HSYNC	Dir	Dir	Ws	Ws	Dir
lcd_vsync	VSYNC	Ale	Ale	Ale	Ale	Ale
lcd_ac_bias_en	DATA EN	Cs0	Cs0	Cs0	Cs0	E0
lcd_memory_clk	N/A	Mclk	Cs1	Mclk	Cs1	E1

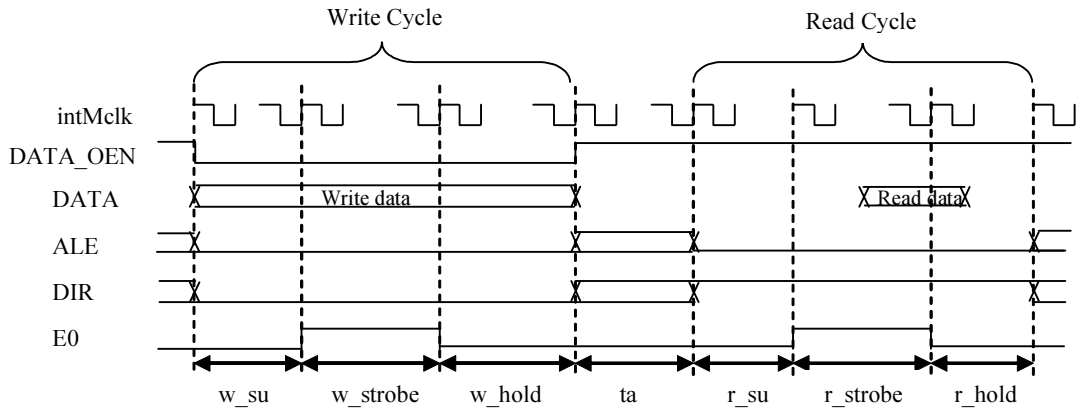
PCLK = Pixel Clock
HSYNC = Horizontal Sync
VSYNC = Vertical Sync
DATA EN = Data Enabled/Valid Data
EN = Enable
Rs = Read Strobe

Dir = Direction
Ws = Write Strobe
Ale = Address Latch enable
Mclk = Memory clock
Ex = Enable x
Csx = Chip Select x

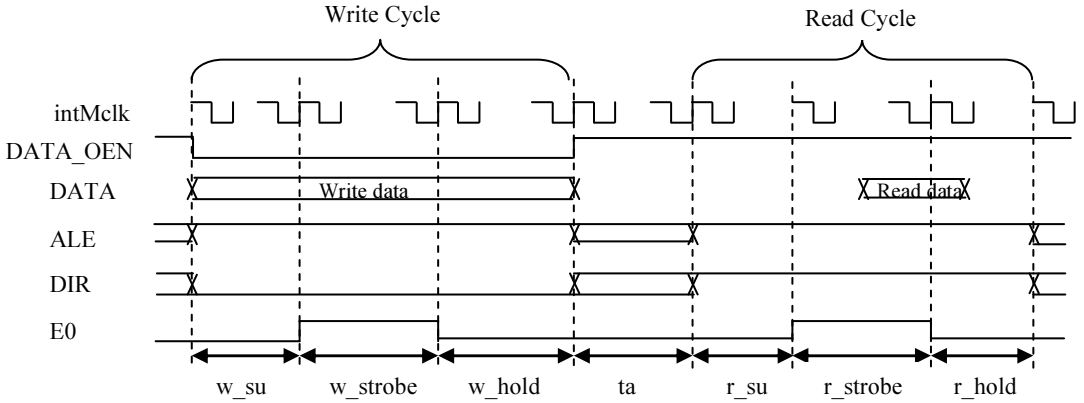
This table shows the mapping of AM335x ball names to the mode functional names

LIDD – Hitachi Asynchronous Interface

Command Mode

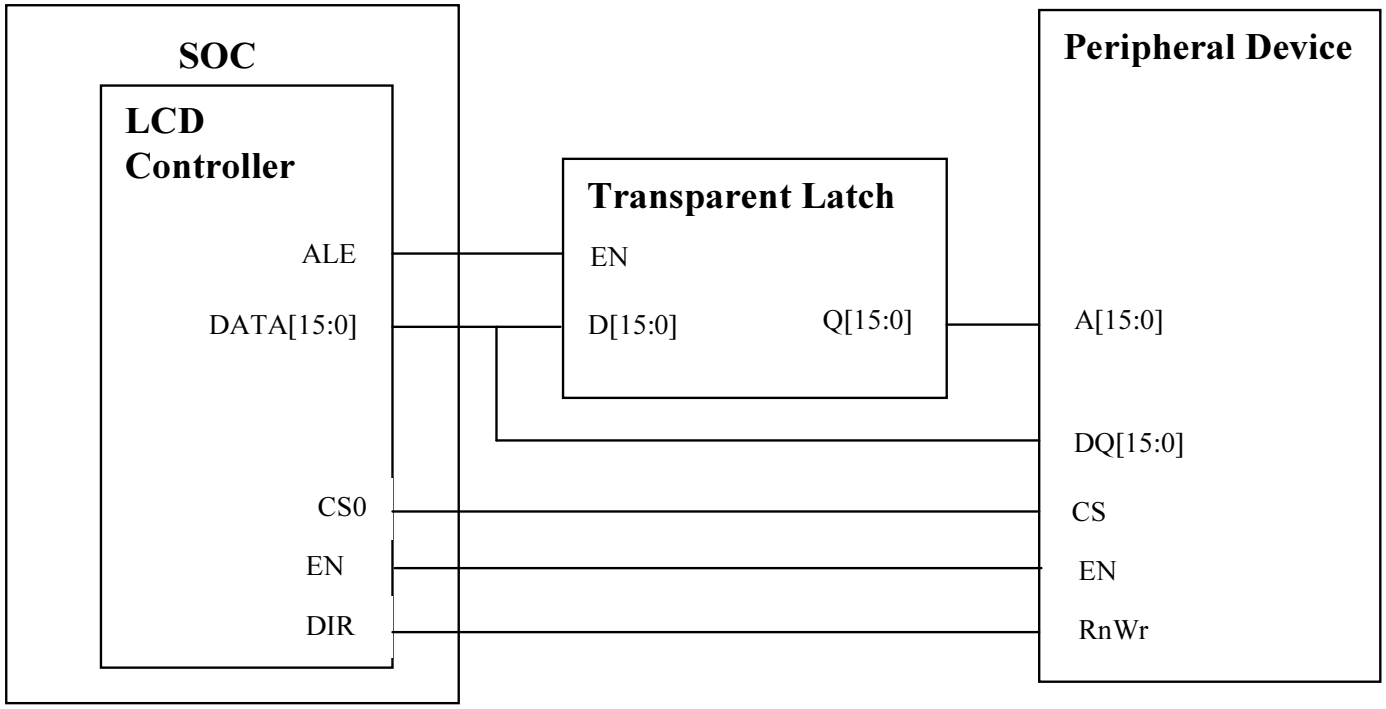


Data Mode



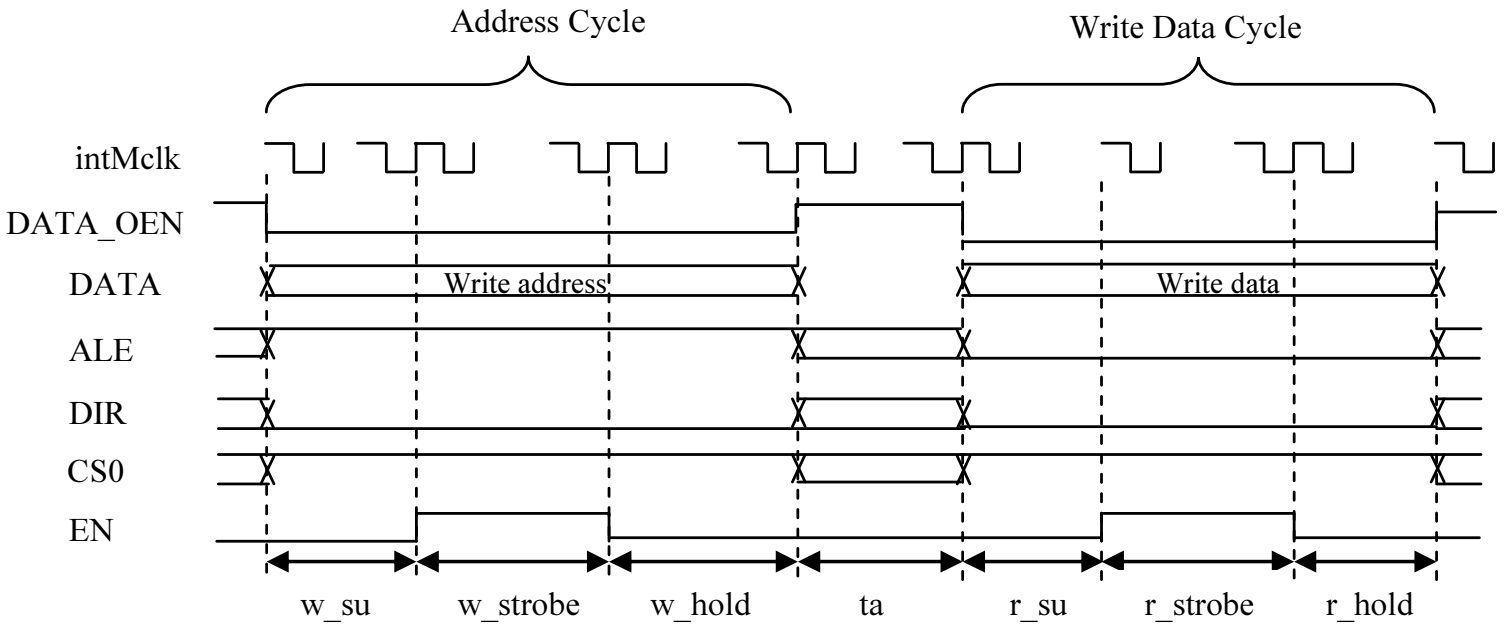
The Hitachi style bus model is shown here. intMclk is the internal bus clock and is shown for reference only. ALE is low during the transaction if the Host is reading or writing a command. A command is initiated by writing to the LIDD_CSx_ADDR register. DIR determines whether the bus transaction is a read or a write. E0, shown to be active high, is the data strobe for device 0. For a read, external read data is expected to be ready the intMCLK cycle just before E0 goes inactive

LIDD – Motorola 68K Interface



The Motorola 68k mode uses a shared 16-bit address/data bus. The address is sent first, latched externally, and then data is sent to complete the bus transaction. The external logic required for using the shared address/data bus is shown here.

LIDD – Motorola 68K Write Transaction



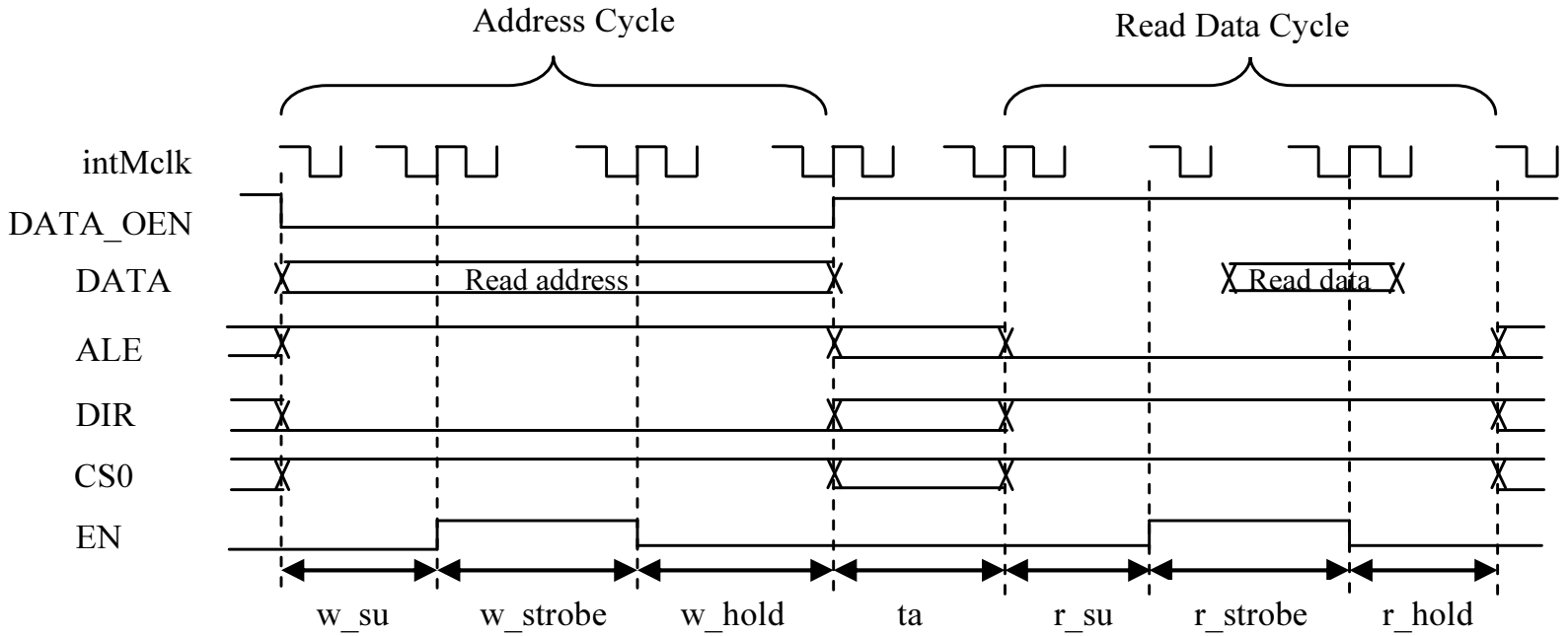
Signals are shown active high. DIR is shown to be high for a Read and low for a Write.

The address is written to the external latch when the CPU writes the target location to LIDD_CSx_ADDR. This CPU write activates ALE for the transaction.

The CPU next writes the data to LIDD_CSx_DATA. ALE is inactive this transaction cycle. CS0 is active for both the address and data transactions. EN is used as a strobe signal to the external device.

DATA_OEN is used to tri-state an output buffer. It is active low when valid write data is available.

LIDD – Motorola 68K Read Transaction

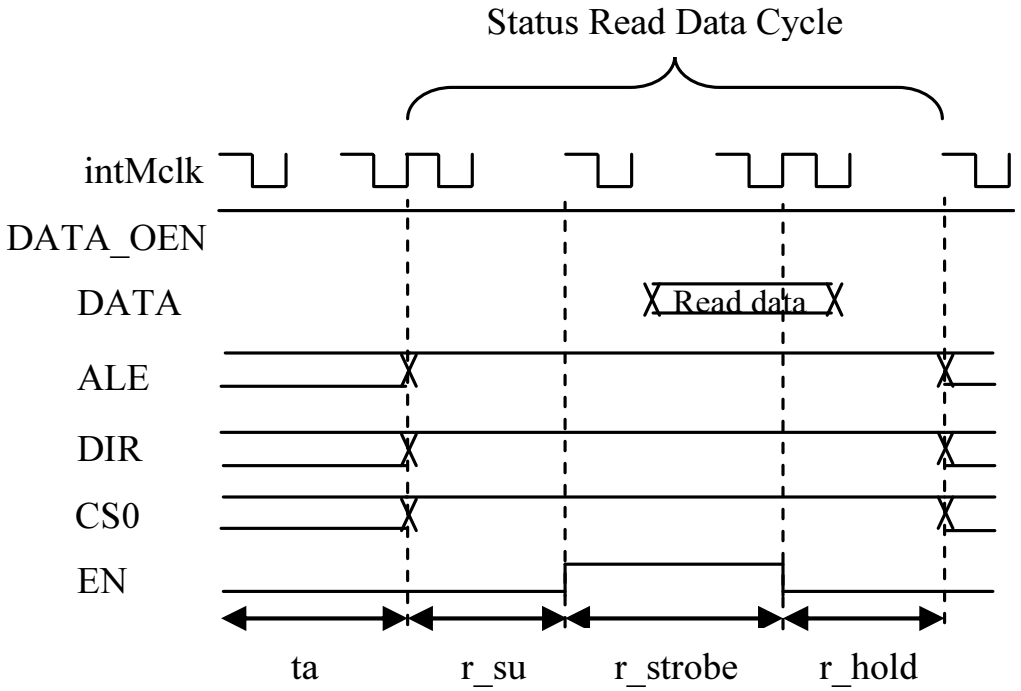


The address is loaded into the external latch when the CPU writes a value to LIDD_CSx_ADDR. ALE is active and DIR is low (write).

The read transaction is initiated when the CPU reads from the LIDD_CSx_DATA register. ALE is inactive for a data read transaction. DIR is high (read). External read data is expected to be ready the intMCLK cycle just before EN goes inactive. CS0 is active for both the address and data transactions. EN is used as a strobe signal to the external device. External read data is expected to be ready the intMCLK cycle just before EN goes inactive.

DATA_OEN is used to tristate an output buffer. It is active low when valid write data is available

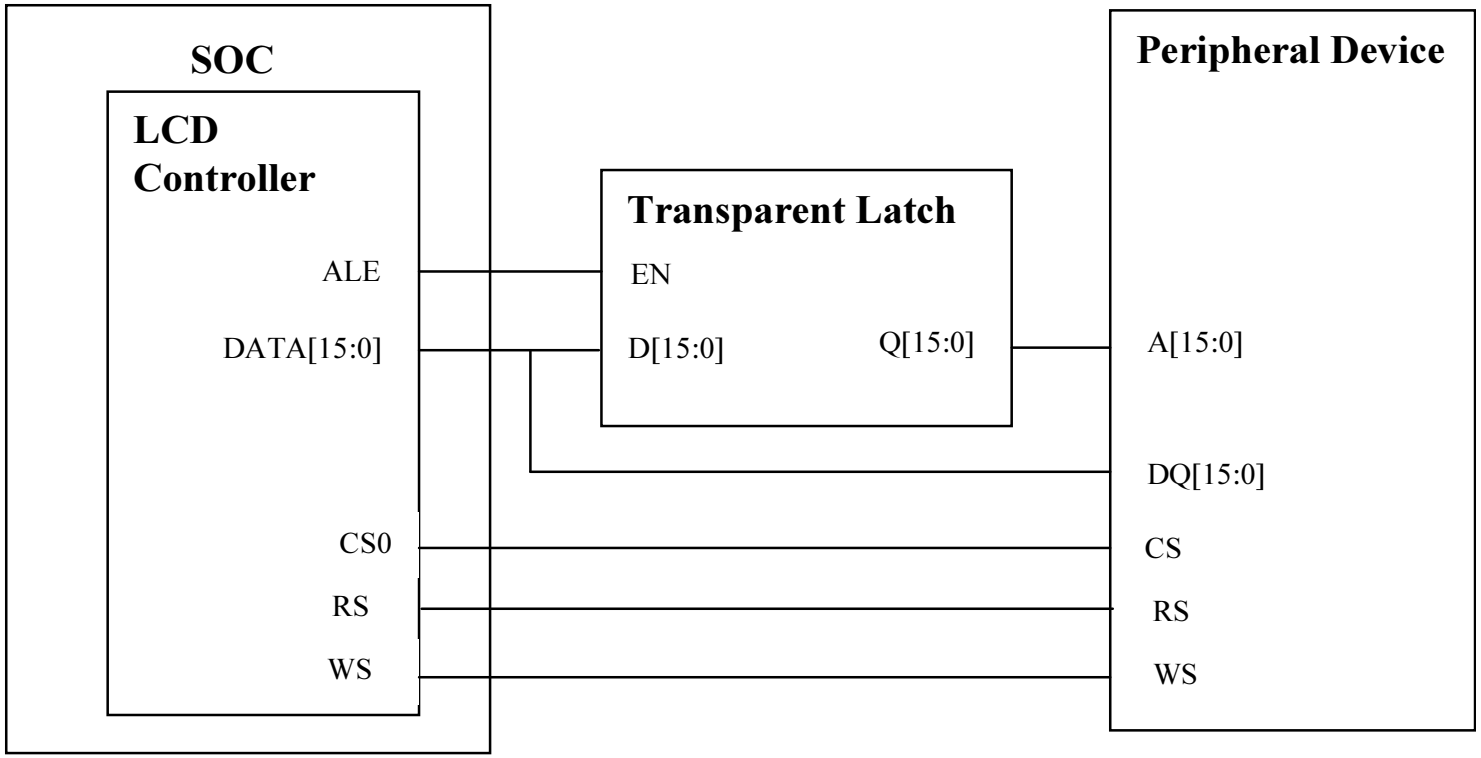
LIDD – Motorola 68K Read Status Transaction



The Motorola bus allows a transaction to read a single status vector from the target peripheral. This status read transaction does not require an address, since it returns only a single vector. The CPU initiates a status read transaction when it reads from the LIDD_CSx_ADDR register.
ALE is active, signaling a read-status transaction. CS0 is active. EN is used as a strobe signal to the external device. External read data is expected to be ready the intMCLK cycle just before EN goes inactive.

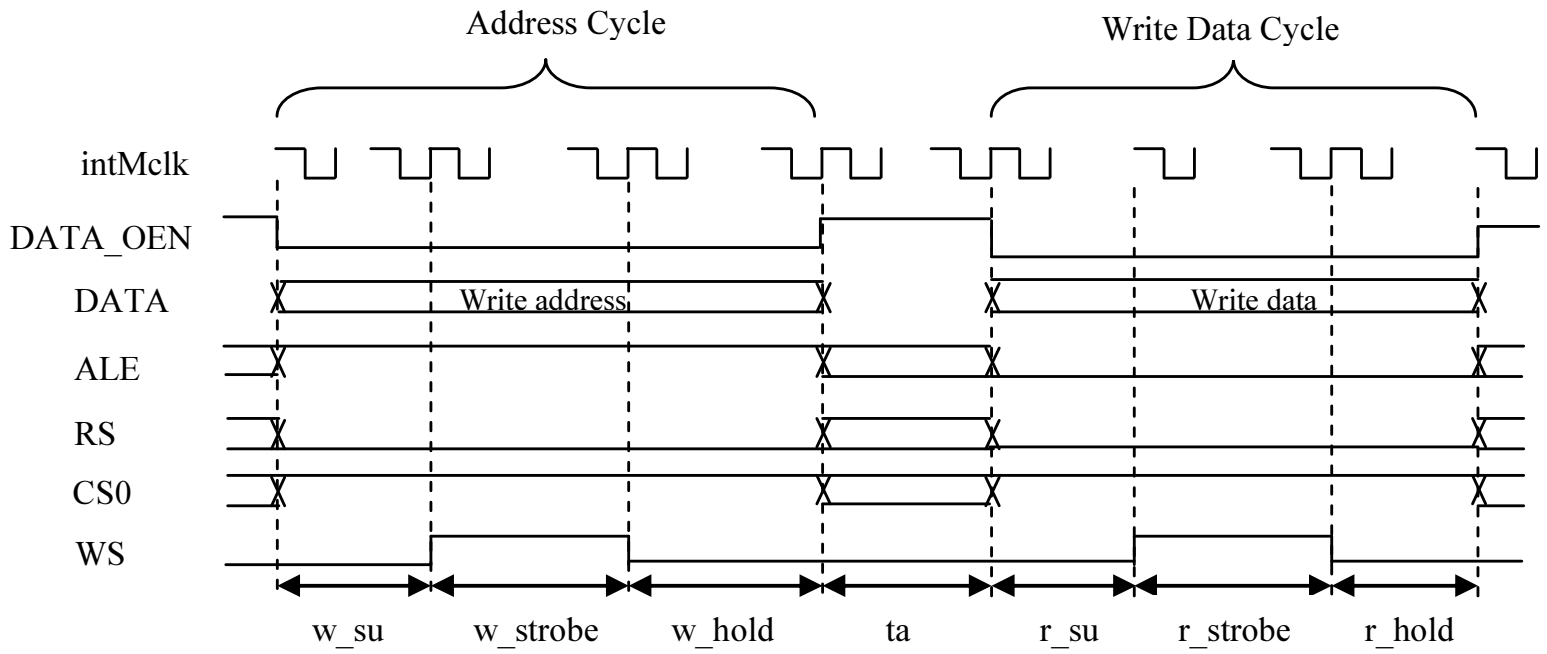
DATA_OEN is used to tristate an output buffer. It is active low when valid write data is available

LIDD – Intel Interface



The Intel interface mode uses a shared 16-bit address/data bus. The address is sent first, latched externally, and then data is sent to complete the bus transaction.

LIDD – Intel Write Transaction

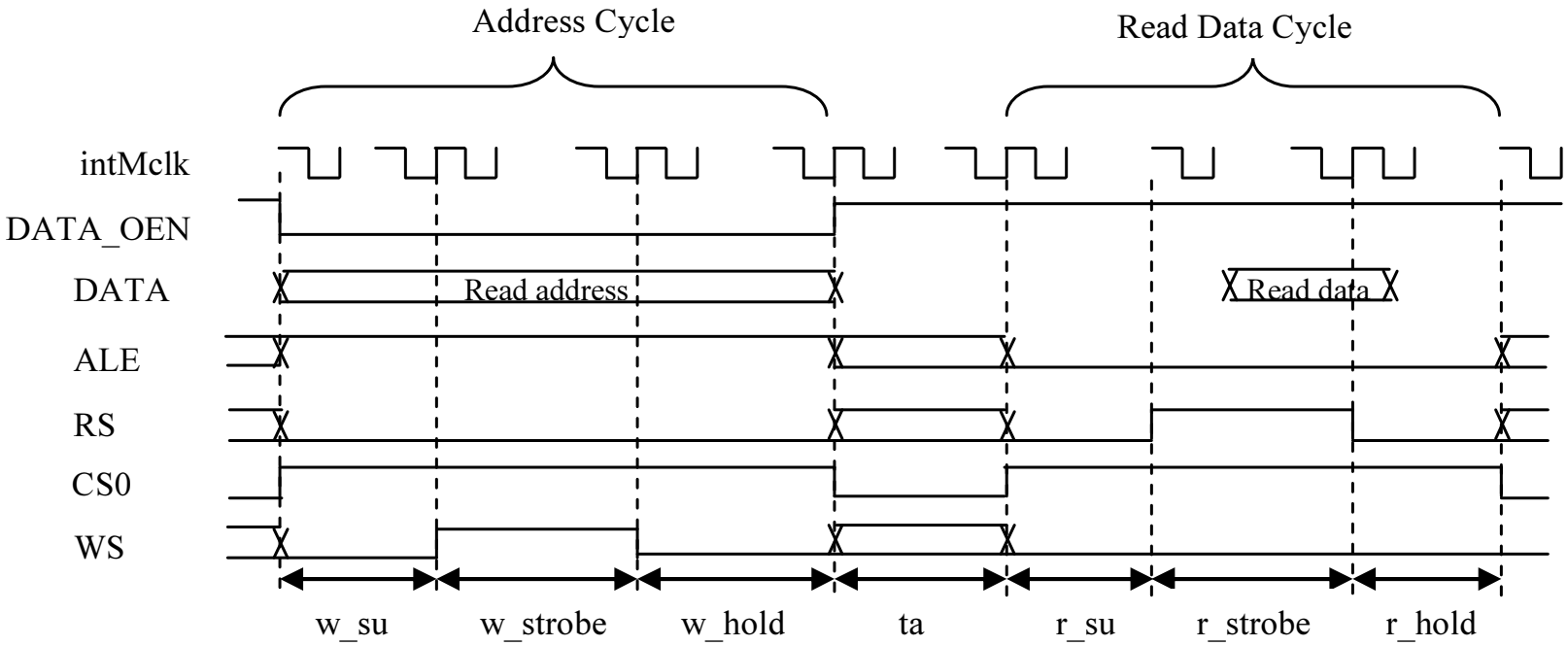


Signals are shown active high.

The address is written to the external latch when the CPU writes the target location to LIDD_CSx_ADDR. This CPU write activates ALE for the transaction.

The CPU next writes the data to LIDD_CSx_DATA. ALE is inactive this transaction cycle. CS0 is active for both the address and data transactions. WS is used as a write strobe signal to the external device. DATA_OEN is used to tri-state an output buffer. It is active low when valid write data is available.

LIDD – Intel Read Transaction

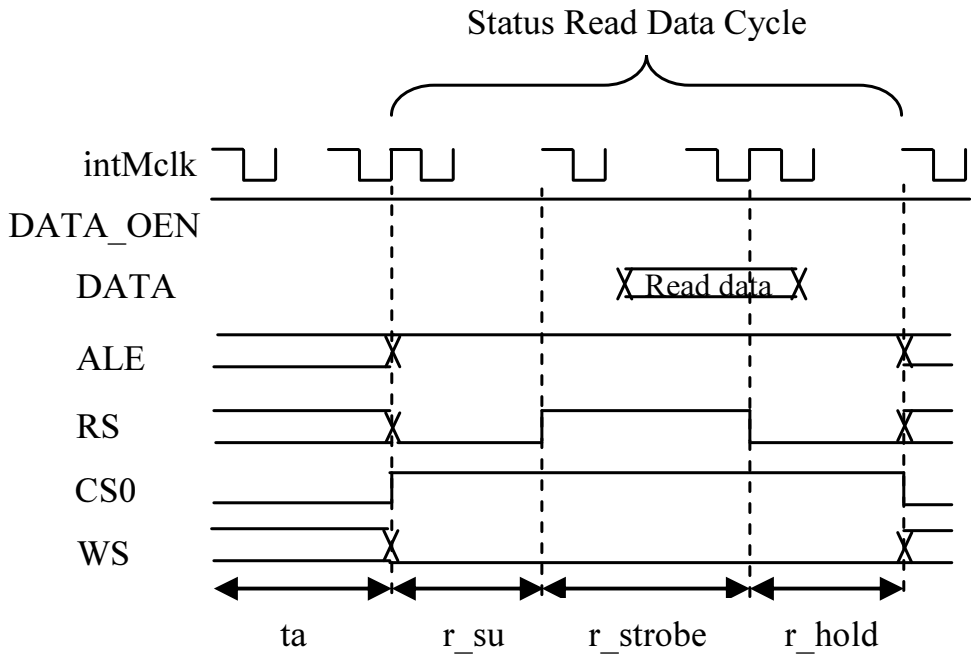


The address is loaded into the external latch when the CPU writes a value to LIDD_CSx_ADDR. This CPU write activates ALE for the transaction.

The read transaction is initiated when the CPU reads from the LIDD_CSx_DATA register. ALE is inactive for a data read. CS0 is active for both the address and data transactions. External read data is expected to be ready the intMCLK cycle just before the RS read strobe goes inactive.

DATA_OEN is used to tri-state an output buffer. It is active low when valid write data is available

LIDD – Intel Read Status Transaction



The Intel bus allows a transaction to read a single status vector from the target peripheral. This status read transaction does not require an address, since it returns only a single vector. The CPU initiates a status read transaction when it reads from the LIDD_CSx_ADDR register. ALE is active, signaling a read-status transaction. CS0 is active. RS is used as a strobe signal to the external device. External read data is expected to be ready the intMCLK cycle just before RS goes inactive.

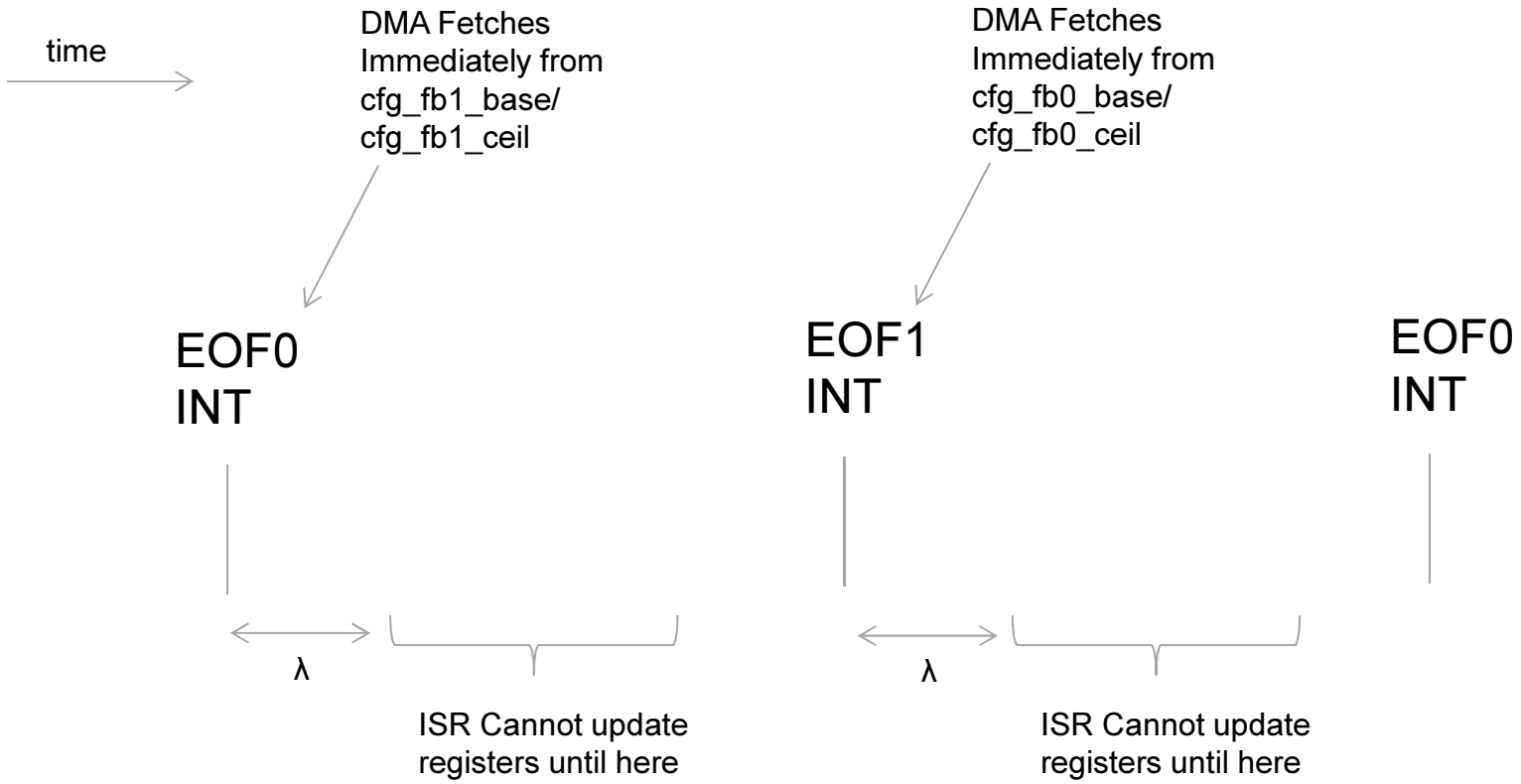
DATA_OEN is used to tri-state an output buffer. It is active low when valid write data is available.

LIDD - DMA Mode Issue

DETAILS	After a DMA transfer is complete, the CFG_LIDD_DMA_EN bit is driven low and the read/write pointers of the Asynchronous FIFO are designed to reset. However, only the write pointer gets reset and therefore the FIFO sequence is corrupt.
WORKAROUND	After receiving LIDD Frame Done interrupt, modify the CLKC_ENABLE register to perform software reset in the L3 and LIDD clock domains after each DMA transfer. This will return the write and read FIFO pointers to their default values and allow for proper FIFO operation

BUFFER MANAGEMENT

Timing Relationships

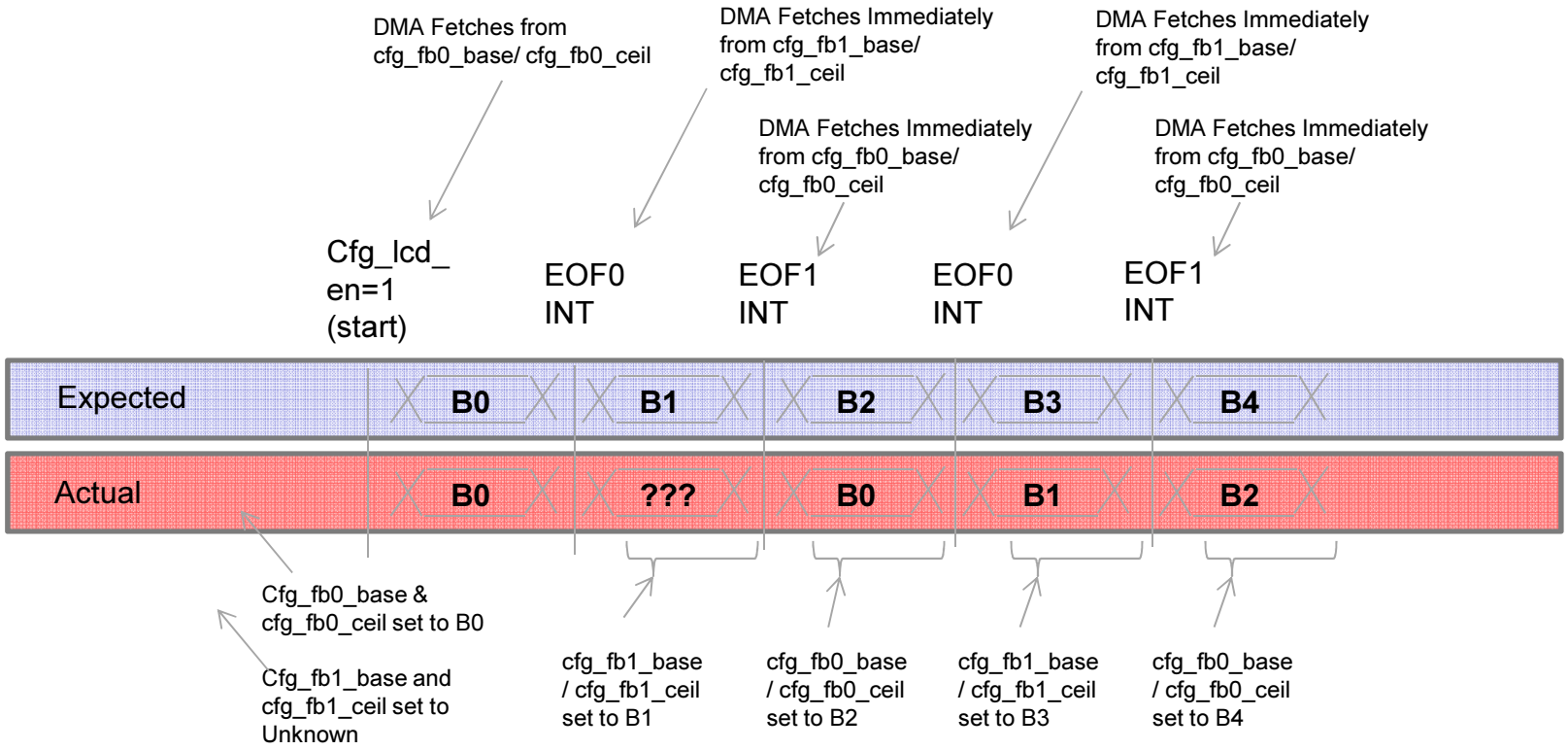


After an EOF0 or EOF1 interrupt, it takes a certain latency λ before the ISR can set cfg_fb0_base, cfg_fb0_ceil, cfg_fb1_base, and cfg_fb1_ceil.

Meanwhile, the DMA uses the current values of those registers for the next frame immediately after firing the interrupt.

λ = 10 cycles of the L3 or L4 clocks (whichever is slower)

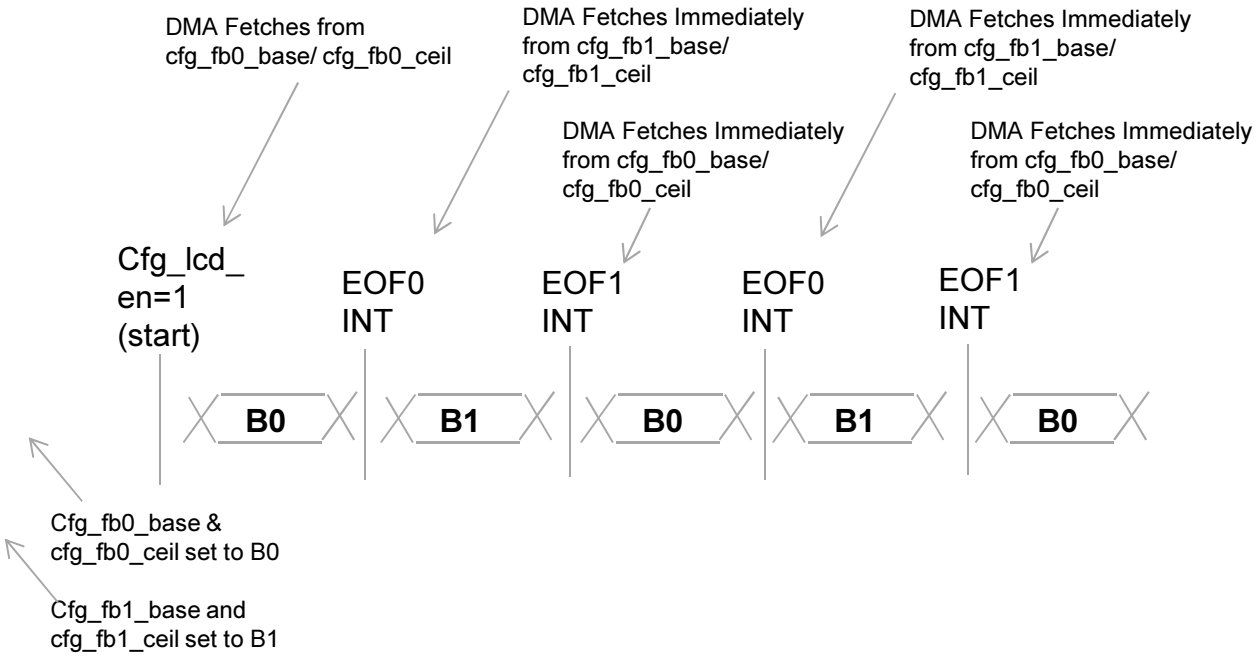
Incorrect Ping Pong Buffer Usage



The intent is to try and set the upcoming framebuffer location at each EOF0/EOF1 ISR with the expected result shown in Blue.

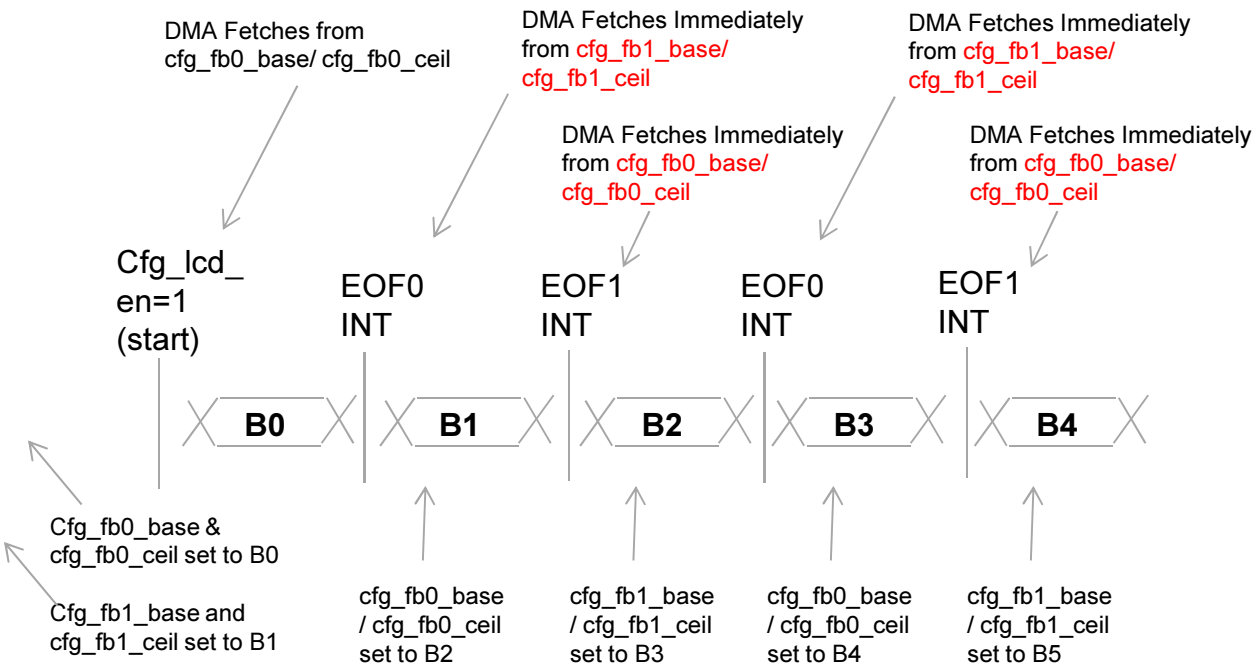
However, the hardware displays the contents of frames two back in time. The Actual results are in Red.

Use Case 1: Recommended Ping Pong Buffer Usage



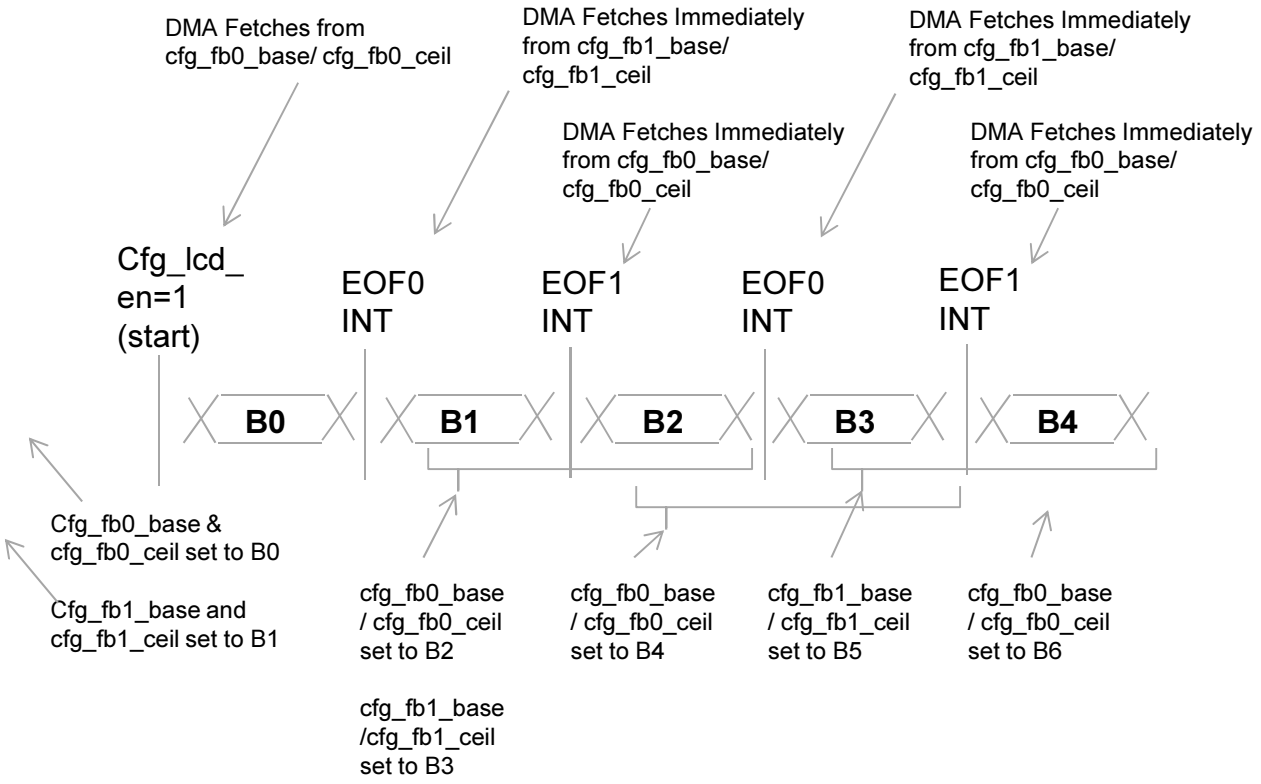
Two frames, B0 and B1, are needed at the start. However, B0 can be a black frame (the Host creates a black frame in DDR). In this case, a video display framebuffer must be available at the start and for every ISR.

Host CPU Keep Out Regions



The DMA engine uses a base address and a ceiling address to define a framebuffer. It is imperative that these base-ceiling pairs match up when the hardware reads them. Say the Host CPU updates only the base register and has not had a chance to update the ceiling register when the DMA engine does the read. This framebuffer, as understood by the DMA engine, will be way off. It will lead to sync error interrupts and possibly affect system performance. The Red bars denote keep-out regions where the Host CPU cannot update the same config registers that are being read by the DMA engine.

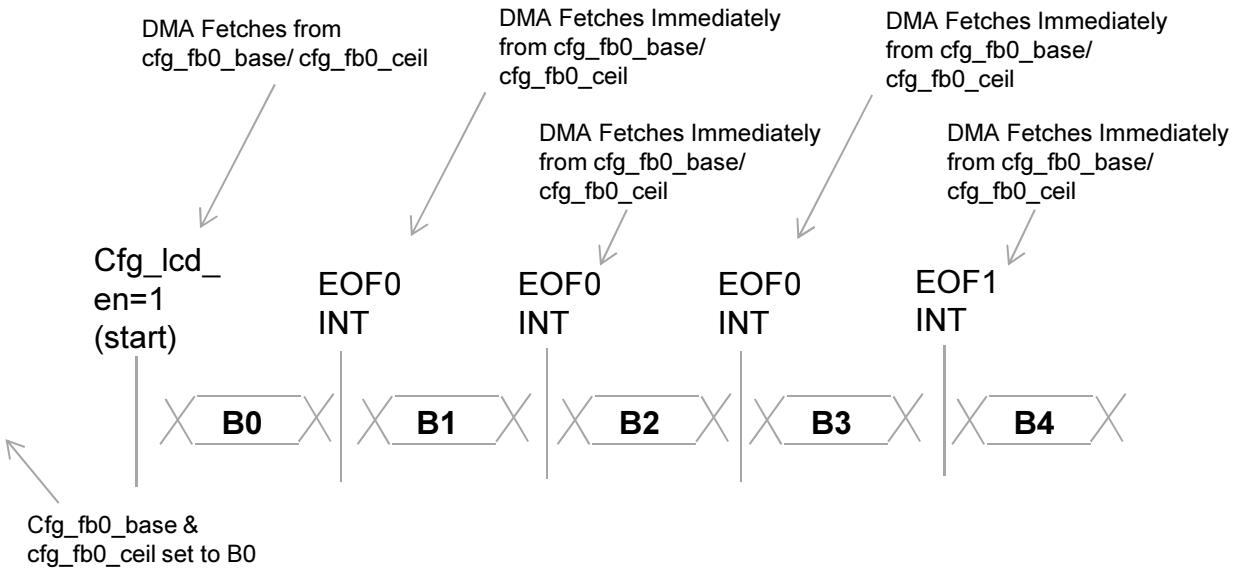
Use Case 2: Using Ping Pong Buffers Where Host CPU Has Large ISR Latency



B0, B1, and B2 can be black frames. A video framebuffer must be available every EOF interrupt. On the first EOF0 ISR, configure fb0 (only if B2 is not a Black frame) and fb1. Then, subsequent EOF1 ISRs update fb0 and EOF0 ISRs update fb1.

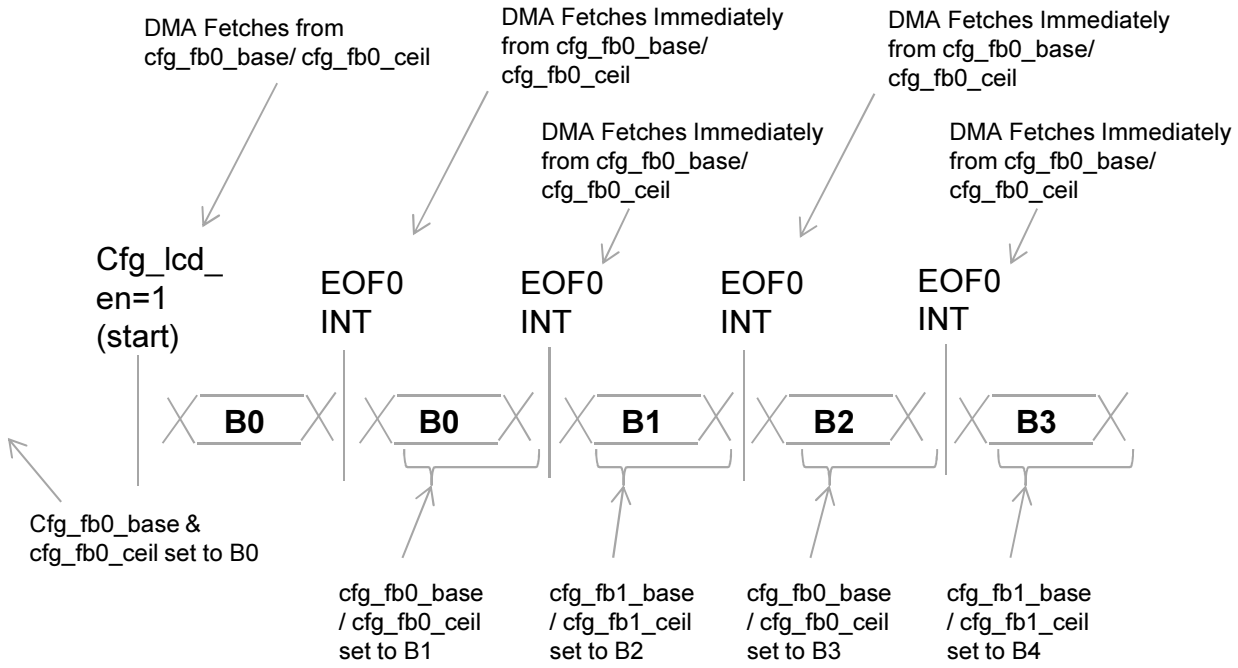
The ISRs after EOF1 now have up to two frame times to perform config register updates before the register values are used. However, there is one additional frame time of output latency.

Use Case 3: Single Buffer Usage Model



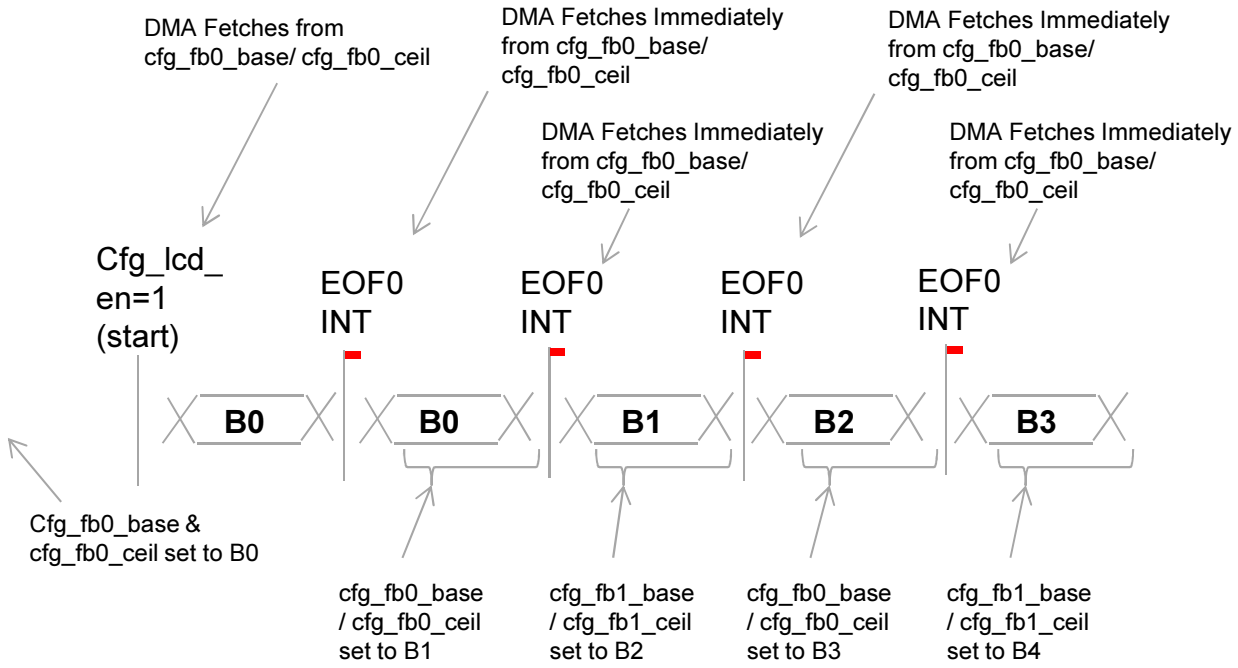
The Single Buffer Usage Model was designed for the case where the framebuffer pointers are static. A fixed display buffer in DDR has new frames copied into it by the CPU or DMA.

Use Case 4: Updating Single Buffer Pointers in Runtime



B0 can be a black frame. A video framebuffer must be available every EOF0 interrupt, which is used to trigger the ISR that updates cfg_fb0_base and cfg_fb0_ceil. This scenario is very similar to Use Case 1, which uses Ping Pong Buffers.

Race Condition: Changing Single Buffer Pointers in Use Case 4



For Use Case 4, there is a Race Condition in which the Host CPU must not update cfg_fb0_base and cfg_fb0_ceil until after the DMA engine has read the values in the CPU Keep-out regions shown in Red. Practically, though, the interrupt latency for the CPU is expected to be long enough to ensure that the configuration registers cannot change in the Keep-out region. The ISR must update the pointers before the next interrupt, though.

LCDC COMAPARED TO DSS

Feature Comparison

Display modes :	**DSS**	**LCDC**
Programmable pixel display modes (1, 2, 4, 8, 12, 16, and 24 bit-per-pixel modes)	YES[1]	YES
Programmable display size	YES	YES[2]
Pallet Size	24-bit	12-bit
Maximum Programmable pixel rate	75MHz	126MHz
Display support :	**DSS**	**LCDC**
Passive & Active Matrix panel.	YES	YES
Remote Frame Buffer support through the RFBI module.	2 Panels	2 Panels
Signal processing :	**DSS**	**LCDC**
Overlay support	YES	NO[3]
Video resizer : upsampling (up to x8) downsampling (down to 1/4)	YES	NO
Rotation 90°, 180°, and 270°	YES[4]	NO
Transparency color key (source and destination)	YES	NO
Programmable video color space conversion YcbCr 4:2:2 into RGB	YES	NO
Gamma curve support	YES	NO
Mirroring support	YES	NO
Programmable Color Phase Rotation (CPR)	YES	NO

Notes
1. DSS Also supports 18 bit modes
2. Pixels Per Line must be multiple of 16
3. Overlay support using software and SGX Under development.
4. Using EDMA