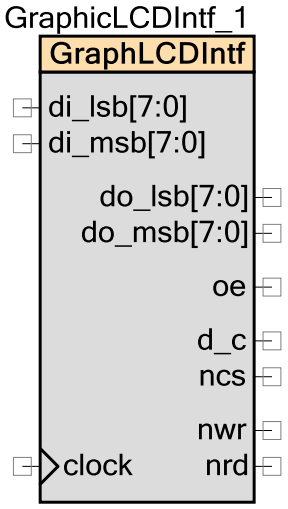
Graphic LCD Interface (GraphicLCDIntf)

1.50

Features

- 8 or 16 bit interface to Graphic LCD Controller
- Compatible with many graphic controller devices
- Read and write transaction
- 2-255 cycles for Read Low Pulse Width
- 1-255 cycles for Read High Pulse Width
- Implements typical i8080 interface



General Description

The Graphic LCD Interface (GraphicLCDIntf) component provides the interface to a graphic LCD controller and driver device. These devices are commonly integrated into an LCD panel. The interface to these devices is commonly referred to as an i8080 interface. This is a reference to the historic parallel bus interface protocol of the Intel 8080 microprocessor.

When to use a GraphicLCDIntf

LCD controllers and driver devices are commonly integrated into an LCD panel. They either include or provide the interface to the frame buffer for the display and they manage that buffer. The GraphicLCDIntf component performs read and write transactions to this controller. These transactions have the following parameters:

- Read or write
- Address. A one bit address driven on the d_c pin
- Data (8 or 16 bits). Sent on “do” for writes and read on “di” for reads.

The GraphicLCDIntf component supports a large number of controllers. There are three parameters to use when you configure this component.

- Clock frequency: The frequency for the clock driving this component is often limited by minimum pulse width low for the write signal. (This value can be found in the respective Graphic LCD Controller data sheet). The write pulse is low for a single clock period, so set the clock frequency to satisfy this requirement.

- Read pulse width high: This setting in the customizer is measured in clock cycles. The clock period times the number of cycles set for the pulse width high must satisfy the requirement for read pulse width high for the controller.
- Read pulse width low: This setting is made in the same way that the read pulse width high setting is made. The timing for the read pulse width low must satisfy the controller's requirement for the read pulse width and the requirement for read access time. The data will be sampled one clock cycle before the end of the active low read pulse, so the pulse width must be long enough that the access time will be satisfied

The following lists the settings for the applicable LCD controller:

Solomon Systech SSD1289

- Clock frequency: 20 MHz (50 ns)
- Read pulse width high: 10 clock cycles (500 ns)
- Read pulse width low: 10 clock cycles (500 ns)

Solomon Systech SSD2119

- Clock frequency: 25 MHz (40ns)
- Read pulse width high: 13 clock cycles (500 ns)
- Read pulse width low: 13 clock cycles (500 ns)

Himax HX8347A

- Clock frequency: 28.5 MHz (35 ns)
- Read pulse width high: 3 clock cycles (105 ns)
- Read pulse width low: 11 clock cycles (385 ns)

ILITEK ILI9325

- Clock frequency: 20 MHz (50 ns)
- Read pulse width high: 3 clock cycles (150 ns)
- Read pulse width low: 3 clock cycles (150 ns)

Epson S1D13743

- Clock frequency: 33 MHz (33.3ns)
- Read pulse width high: 2 clock cycles (67 ns)
- Read pulse width low: 5 clock cycles (167 ns)

Input/Output Connections

This section describes the various input and output connections for the GraphicLCDIntf component. An asterisk (*) in the list of I/Os indicates that the I/O may be hidden on the symbol under the conditions listed in the description of that I/O.

clock

Clock that operates this component. This component operates entirely from a single clock connected to the component.

di_lsb[7:0]

Lower 8 bits of the input data bus. Used for data during a read transaction.

These signals should be connected to an input pin on the device and the "Input Synchronized" selection for these pin should be disabled. The signals themselves are inherently synchronized since they are being driven based on synchronous output signals.

di_msb[7:0] *

Upper 8 bits of the input data bus. Used for data during a read transaction. Only present for 16-bit interface mode.

These signals should be connected to an input pin on the device and the "Input Synchronized" selection for these pin should be disabled. The signals themselves are inherently synchronized since they are being driven based on synchronous output signals.

do_lsb[7:0]

Lower 8 bits of the output data bus. Used for data during a write transaction.

do_msb[7:0] *

Upper 8 bits of the output data bus. Used for data during a write transaction. Only present for 16-bit interface mode.

oe

Output enable for the data bus. Normally connected to the output enable of the Input/Output pin component for the data buses. Refer to the schematic macro to see how this signal is used.

d_c

Data/Command signal. Indicates a data transaction when high and command transaction when low.

ncs

Active low chip select.

nwr

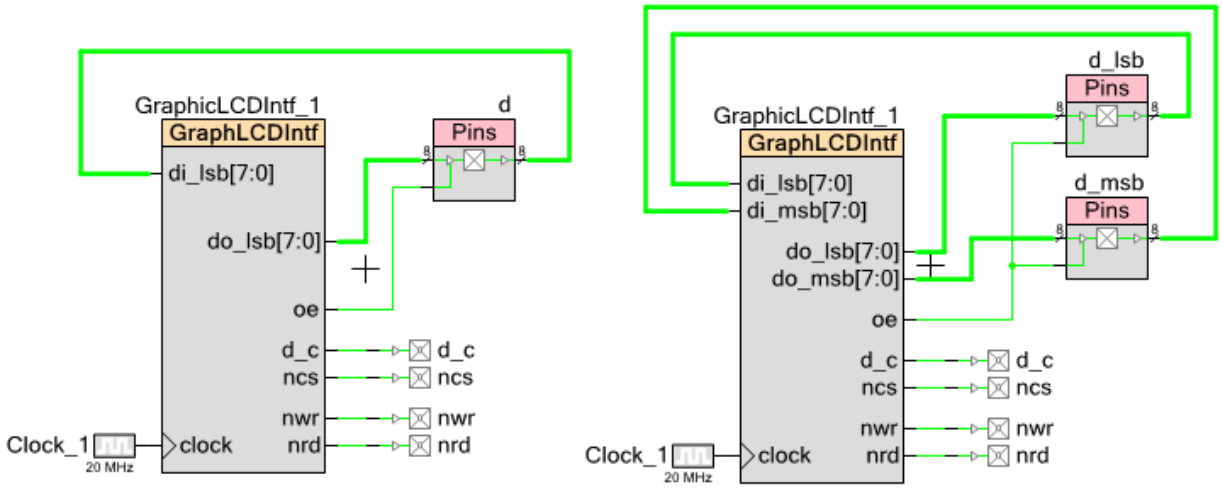
Active low write control signal.

nrd

Active low read control signal.

Schematic Macro Information

Two macros are supplied in addition to the standard symbol entry in the catalog. One macro is for an 8-bit implementation connected to pins and a clock. The other is for a 16-bit implementation connected to pins and a clock.

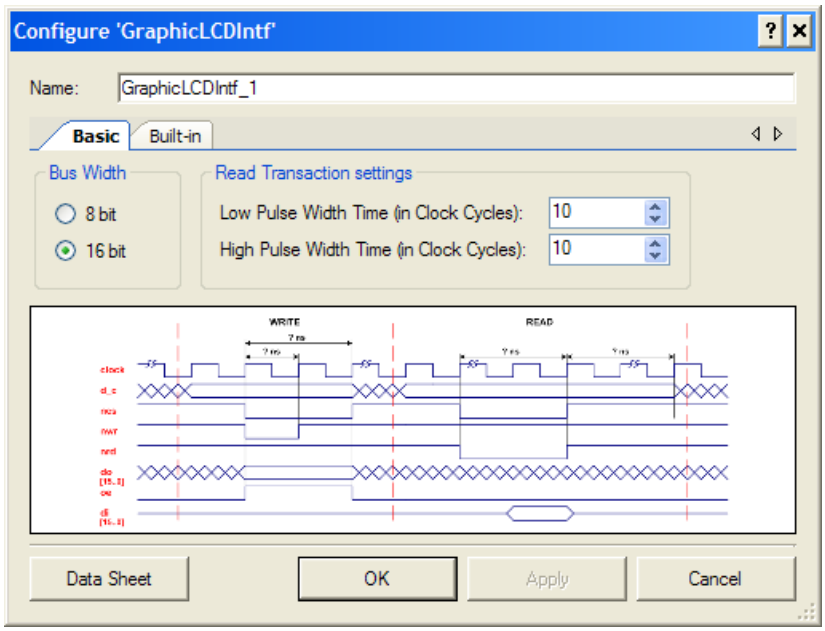


For each of the macros, the clock is set to 20 MHz and the pulse width settings are left at the default. These are the correct settings for the SSD1289 Controller used on CY8CKIT-032 Graphics LCD Interface Kit.

The "Input Synchronized" option is unchecked on all of the data pins and the generation of APIs for all of the pins is turned off.

Parameters and Setup

Drag a GraphicLCDIntf component onto your design and double click it to open the Configure dialog. The default GraphicLCDIntf settings are the proper settings for operation with the Solomon Systech SSD1289 device used on the CY8CKIT-032 Graphics LCD Interface Kit.



Bus Width

Determines whether an 8- or 16-bit parallel interface to a graphic LCD controller is supported. The default setting is 16.

Low Pulse Width Time

Determines the number of clock cycles required for the read pulse width low for the controller. This value can be set between 2 and 255 clock cycles (minimum is 2 because the read value needs to be sampled one clock before the end of the pulse). The default setting is 10.

High Pulse Width Time

Determines the number of clock cycles required for read pulse width high for the controller. This value can be set between 1 and 255 clock cycles. The default setting is 10.

Clock Selection

There is no internal clock in this component. You must attach a clock source. This component operates from a single clock connected to the component.

Placement

The GraphicLCDIntf is placed throughout the UDB array and all placement information is provided to the API through the *cyfitter.h* file.

Resources

Resolution	Digital Blocks					API Memory (Bytes)		Pins (per External I/O)
	Datapaths	Macro cells	Status Registers	Control Registers	Counter7	Flash	RAM	
8 bits	1	11	2	0	0	TBD	TBD	12
16 bits	2	11	3	0	0	TBD	TBD	20

Application Programming Interface

Application Programming Interface (API) routines allow you to configure the component using software. The following table lists and describes the interface to each function. The subsequent sections cover each function in more detail.

By default, PSoC Creator assigns the instance name "GraphicLCDIntf_1" to the first instance of a component in a given design. You can rename the instance to any unique value that follows the syntactic rules for identifiers. The instance name becomes the prefix of every global function name, variable, and constant symbol generated for the component. For readability, the instance name used in the following table is "GraphicLCDIntf".

Function	Description
void GraphicLCDIntf_Init(void)	Initializes or Restores default GraphicLCDIntf configuration
void GraphicLCDIntf_Enable(void)	Enables the GraphicLCDIntf
void GraphicLCDIntf_Start(void)	Starts the GraphicLCDIntf interface.
void GraphicLCDIntf_Stop(void)	Disables the GraphicLCDIntf interface.
void GraphicLCDIntf_Write8(uint8 d_c, uint8 data)	Initiates a write transaction on the 8-bit parallel interface.
void GraphicLCDIntf_Write16(uint8 d_c, uint16 data)	Initiates a write transaction on the 16-bit parallel interface.
uint8 GraphicLCDIntf_Read8(uint8 d_c)	Initiates a read transaction on the 8-bit parallel interface.
uint16 GraphicLCDIntf_Read16(uint8 d_c)	Initiates a read transaction on the 16-bit parallel interface.
void GraphicLCDIntf_Sleep(void)	Saves configuration and disables the GraphicLCDIntf
void GraphicLCDIntf_WakeUp(void)	Restores configuration and enables the GraphicLCDIntf
void GraphicLCDIntf_SaveConfig(void)	Saves configuration of GraphicLCDIntf
void GraphicLCDIntf_RestoreConfig(void)	Restores configuration of GraphicLCDIntf

Global Variables

Variable	Description
GraphicLCDIntf_initVar	Indicates whether the Graphic LCD Interface has been initialized. The variable is initialized to 0 and set to 1 the first time GraphicLCDIntf_Start() is called. This allows the component to restart without reinitialization in after the first call to the GraphicLCDIntf_Start() routine. If reinitialization of the component is required the variable should be set to 0 before the GraphicLCDIntf_Start() routine is called. Alternately, the Graphic LCD Interface can be reinitialized by calling the GraphicLCDIntf_Init() and GraphicLCDIntf_Enable() functions.

void GraphicLCDIntf_Init(void)

Description: Initializes/Restores default GraphicLCDIntf configuration provided with customizer. Only static component configuration that defines Read Low and High Pulse Widths will be restored to their initial values.

Parameters: None

Return Value: None

Side Effects: This will re-initialize the component but it will not clear data from the FIFO's, and it will not reset the component hardware state machine. Current transaction will be performed on the bus.

void GraphicLCDIntf_Enable(void)

Description: Enables the GraphicLCDIntf interface.

Parameters: None

Return Value: None

Side Effects: None

void GraphicLCDIntf_Start(void)

Description: Enables Active mode power template bits or clock gating as appropriate. Configures the component for operation.

Parameters: None

Return Value: None

Side Effects: None

void GraphicLCDIntf_Stop(void)

Description: Disables Active mode power template bits or gates clocks as appropriate.

Parameters: None

Return Value: None

Side Effects: None

void GraphicLCDIntf_Write8(uint8 d_c, uint8 data)

Description: Initiates a write transaction on the 8-bit parallel interface. The write is a posted write, so this function will return before the write has actually completed on the interface. If the command queue is full, this function will not return until space is available to queue this write request.

Parameters: d_c: Data (1) or Command (0) indication. Passed to the d_c pin

data: Data sent on the do_lsb[7:0] pins

Return Value: None

Side Effects: None

void GraphicLCDIntf_Write16(uint8 d_c, uint16 data)

Description: Initiates a write transaction on the 16-bit parallel interface. The write is a posted write, so this function will return before the write has actually completed on the interface. If the command queue is full, this function will not return until space is available to queue this write request.

Parameters: d_c: Data (1) or Command (0) indication. Passed to the d_c pin

data: Data sent on the do_msb[7:0] (most significant byte) and do_lsb[7:0] (least significant byte) pins

Return Value: None

Side Effects: None

uint8 GraphicLCDIntf_Read8(uint8 d_c)

Description: Initiates a read transaction on the 8-bit parallel interface. The read will execute after all currently posted writes have completed. This function will wait until the read completes and then returns the read value.

Parameters: d_c: Data (1) or Command (0) indication. Passed to the d_c pin.

Return Value: 8-bit read value from the di_lsb[7:0] pins

Side Effects: None

uint16 GraphicLCDIntf_Read16(uint8 d_c)

Description: Initiates a read transaction on the 16-bit parallel interface. The read will execute after all currently posted writes have completed. This function will wait until the read completes and then returns the read value.

Parameters: d_c: Data (1) or Command (0) indication. Passed to the d_c pin.

Return Value: 16-bit read value from the di_msb[7:0] (most significant byte) and di_lsb[7:0] (least significant byte) pins

Side Effects: None

void GraphicLCDIntf_Sleep(void)

Description: Saves GraphicLCDIntf configuration and non-retention register values. Disables Active mode power template bits or clock gating as appropriate.

Parameters: None

Return Value: None

Side Effects: None

void GraphicLCDIntf_WakeUp(void)

Description: Restores GraphicLCDIntf configuration and non-retention register values. Enables Active mode power template bits or clock gating as appropriate.

Parameters: None

Return Value: None

Side Effects: None

void GraphicLCDIntf_SaveConfig(void)

Description: Saves the user configuration of GraphicLCDIntf non-retention registers. The compile-time component configuration that defines Read Low and High Pulse Widths will be stored.

Parameters: None

Return Value: None

Side Effects: None

void GraphicLCDIntf_RestoreConfig(void)

Description: Restores the configuration of GraphicLCDIntf non-retention registers. The API is called by GraphicLCDIntf_Wakeup to restore component non-retention registers.

Parameters: None

Return Value: None

Side Effects: If this API will be called before GraphicLCDIntf_SaveConfig the component configuration for Read Low and High Pulse Widths will be restored to their values provided with customizer.

Sample Firmware Source Code

See the example provided with the CY8CKIT-032 Graphics LCD Interface Kit. Besides initialization of the controller, this component is typically used exclusively by the Segger emWin Graphics component.

Functional Description

Bus Transactions

This interface can perform either a read or a write transaction. These transactions have the following parameters:

- Read or write
- Address. In this case it is just a one bit address driven on the d_c pin
- Data (8 or 16 bits). Sent on “do” for writes and read on “di” for reads.

The implementation assumes that the CPU sends a command byte to the component using a FIFO (the same FIFO that is used for data). That command byte indicates read or write and provides the d_c bit.

Idle Condition

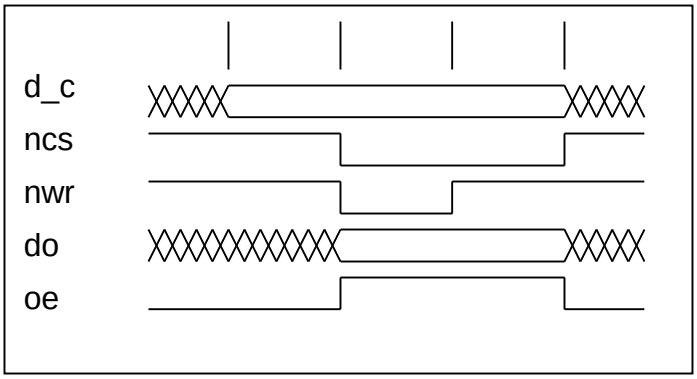
When neither a read nor a write is occurring on the interface the interface is in the idle state. The values for the output pins in that condition are:

- d_c: don’t care (may be left at its last state)
- ncs: 1
- nwr: 1
- nrd: 1
- do: don’t care (may be left at its last state)
- oe: 0

In the description of the read and write transactions, any signal not listed is idle.

Write Transaction

The timing diagram for a write transaction on the parallel interface is shown as follows:

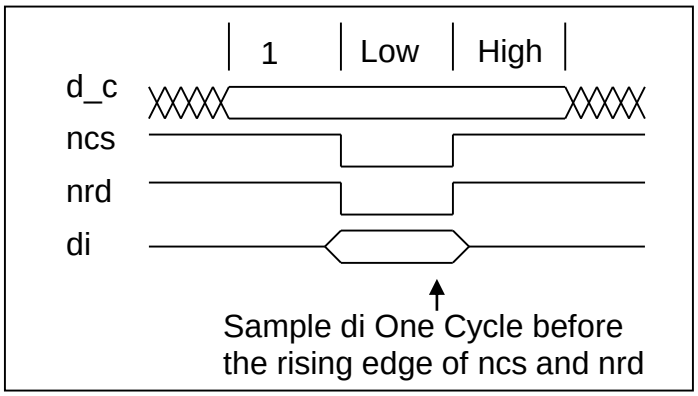


This diagram shows that the write transaction requires three clock cycles. The timing diagram is the same regardless of the bit width. This transaction can be immediately preceded or followed by another read or write transaction or may be in the idle state before or after a write transaction.

The interface to the CPU allows the CPU to make posted write requests (request a write providing the address and data and then proceed before the transaction is actually completed on parallel bus). The implementation allows the CPU to have two write requests outstanding without stalling the CPU.

Read Transaction

The timing diagram for a read transaction on the parallel interface is shown as follows:

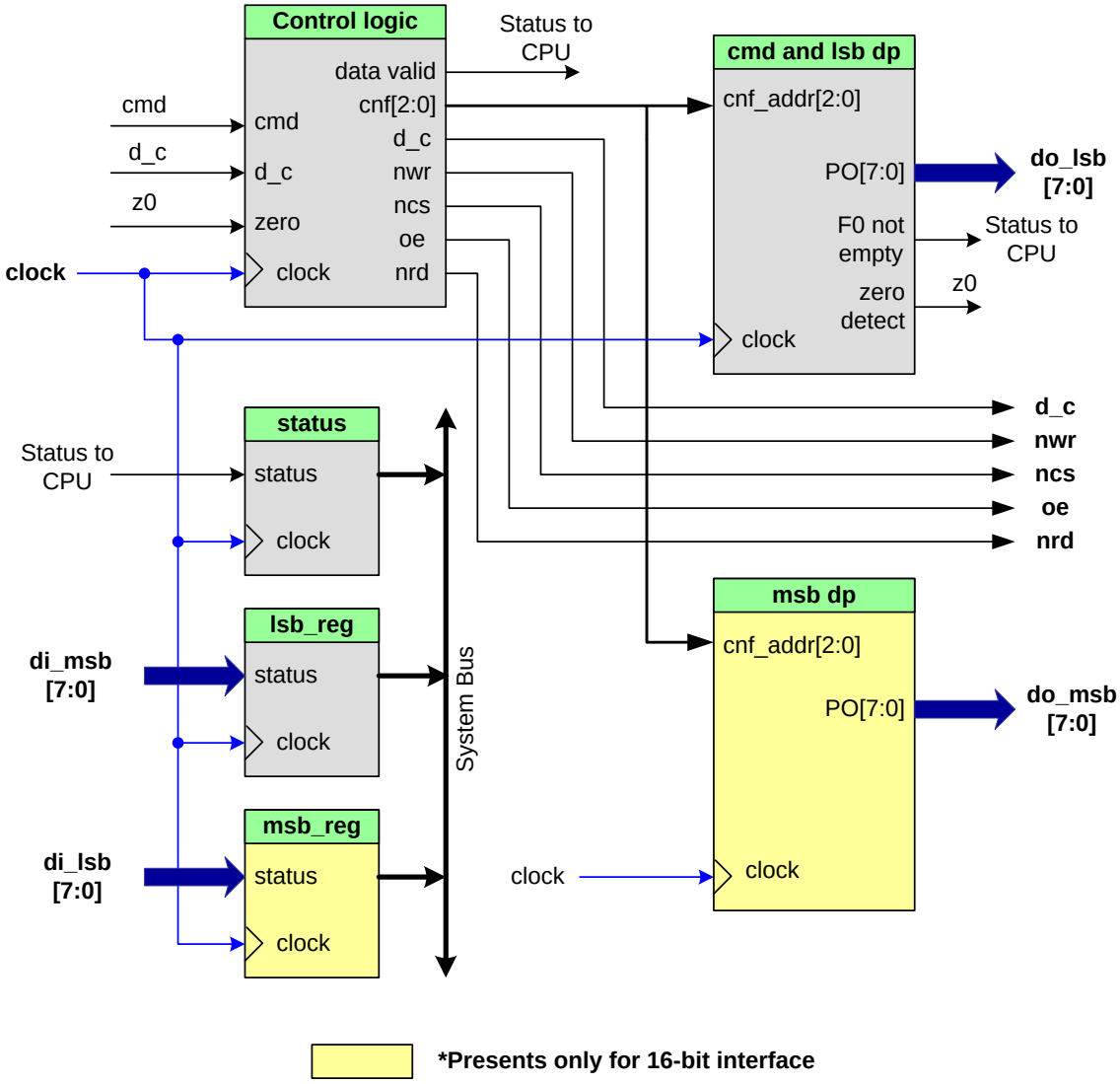


This diagram shows that the read transaction requires a variable number of clock cycles depending on the setting for the high and low read pulse widths. The timing diagram is the same regardless of the bit width. Note that the data input is sampled one clock cycle before the end of the ncs and nrd low pulses. This transaction can be immediately proceeded or followed by another read or write transaction or may be in the idle state before or after a read transaction.

The ordering of reads and writes is maintained (reads occur before posted writes have completed). Reads will require the CPU to wait for the completion of the read transaction before proceeding.

Block Diagram and Configuration

The GraphicLCDIntf component is implemented as a set of configured UDBs. The implementation is shown in the following block diagram.



Registers

GraphicLCDIntf_STATUS_REG

Bits	7	6	5	4	3	2	1	0
Value	reserved						data_valid	F0_half_ empty

- F0_half_empty: if set there is at least two bytes of room in the command/data FIFO

- data_valid: set if read data is valid for the CPU. This bit is cleared when CPU reads the register.

GraphicLCDIntf_DIN_LSB_DATA_REG

Bits	7	6	5	4	3	2	1	0
Value	di_lsb[7:0]							

- Lower 8 bits of the input data bus for read transaction

The register value may be read by the user with the GraphicLCDIntf_Read8() API function for 8-bit interface and is the least significant byte of returned value from GraphicLCDIntf_Read16() API function for 16-bit interface.

GraphicLCDIntf_DIN_MSB_DATA_REG

Bits	7	6	5	4	3	2	1	0
Value	di_msb[7:0]							

- Upper 8 bits of the input data bus for read transaction

The register value is the most significant byte of returned value from GraphicLCDIntf_Read16() API function for 16-bit interface.

Note: DIN_LSB_DATA_REG and DIN_MSB_DATA_REG bits will be cleared when CPU firmware reads these registers.

References

Not applicable

DC and AC Electrical Characteristics

5.0V/3.3V DC and AC Electrical Characteristics

Parameter	Typical	Min	Max	Units	Conditions and Notes
Input					
Input Voltage Range	---		Vss to Vdd	V	
Input Capacitance	---		---	pF	
Input Impedance	---		---	Ω	
Maximum Clock Rate	---		67	MHz	

Component Changes

Version 1.50 is the first release of the GraphicLCDIntf component.

© Cypress Semiconductor Corporation, 2010-2016. This document is the property of Cypress Semiconductor Corporation and its subsidiaries, including Spansion LLC (“Cypress”). This document, including any software or firmware included or referenced in this document (“Software”), is owned by Cypress under the intellectual property laws and treaties of the United States and other countries worldwide. Cypress reserves all rights under such laws and treaties and does not, except as specifically stated in this paragraph, grant any license under its patents, copyrights, trademarks, or other intellectual property rights. If the Software is not accompanied by a license agreement and you do not otherwise have a written agreement with Cypress governing the use of the Software, then Cypress hereby grants you a personal, non-exclusive, nontransferable license (without the right to sublicense) (1) under its copyright rights in the Software (a) for Software provided in source code form, to modify and reproduce the Software solely for use with Cypress hardware products, only internally within your organization, and (b) to distribute the Software in binary code form externally to end users (either directly or indirectly through resellers and distributors), solely for use on Cypress hardware product units, and (2) under those claims of Cypress’s patents that are infringed by the Software (as provided by Cypress, unmodified) to make, use, distribute, and import the Software solely for use with Cypress hardware products. Any other use, reproduction, modification, translation, or compilation of the Software is prohibited.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS DOCUMENT OR ANY SOFTWARE OR ACCOMPANYING HARDWARE, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. To the extent permitted by applicable law, Cypress reserves the right to make changes to this document without further notice. Cypress does not assume any liability arising out of the application or use of any product or circuit described in this document. Any information provided in this document, including any sample design information or programming code, is provided only for reference purposes. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. Cypress products are not designed, intended, or authorized for use as critical components in systems designed or intended for the operation of weapons, weapons systems, nuclear installations, life-support devices or systems, other medical devices or systems (including resuscitation equipment and surgical implants), pollution control or hazardous substances management, or other uses where the failure of the device or system could cause personal injury, death, or property damage (“Unintended Uses”). A critical component is any component of a device or system whose failure to perform can be reasonably expected to cause the failure of the device or system, or to affect its safety or effectiveness. Cypress is not liable, in whole or in part, and you shall and hereby do release Cypress from any claim, damage, or other liability arising from or related to all Unintended Uses of Cypress products. You shall indemnify and hold Cypress harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of Cypress products.

Cypress, the Cypress logo, Spansion, the Spansion logo, and combinations thereof, PSoC, CapSense, EZ-USB, F-RAM, and Traveo are trademarks or registered trademarks of Cypress in the United States and other countries. For a more complete list of Cypress trademarks, visit cypress.com. Other names and brands may be claimed as property of their respective owners.