PmodCLP Library Reference Manual

Revision: February 14, 2012

1300 NE Henley Court, Suite 3
Pullman, WA 99163
(509) 334 6306 Voice | (509) 334 6300 Fax

Introduction

The PmodCLP is a 16x2 character LCD module that uses two Pmod connectors to present a 3.3V, 8-bit parallel data interface to system boards. It is based on a Sunlike LCD panel that uses a Samsung KS0066 (or equivalent) LCD controller. The module can be attached to any number of Digilent system boards to create a character LCD subsystem. This document describes the structure and the usage of the library functions that manage the PmodCLP. The purpose of this ClpLib library is to provide user access to the PmodCLP functionality.

Overview

The module is capable of displaying any one of more than 200 predefined characters in each of the 32 display locations (organized as 16 characters on two rows). Most characters use ASCII codes (see the Samsung KS0066 data sheet for a complete list of character codes). The module can also execute a variety of instructions, such as erasing specific characters, setting different display modes, scrolling, and displaying user-defined characters.

Library Operation

Library Interface

The header file ClpLib_config.h is a configuration file to setup the target platform for the specific PMOD connections being used (default: JA and JB).

The header file ClpLib.h contains the declarations for the actual functions that are available to the user for this module. Actual definitions for these functions are found within ClpLib.c.

Defining PMOD Connectors Used

The Cerebot 32Mx4 Pmod connectors configuration offers a position for PmodCLP where data pins correspond to ordered bits in a port: JA-JB connectors. Using this configuration ensures better performance for read/write operations and therefore is recommended. This option is specified by _CONTIGUOUS_DATA being defined in ClpLib_Config.h. This is the default option, no need to edit ClpLib_Config.h.

The user may decide not to use JA-JB Pmod connectors for CLP (known as Scatterd data pins). This is also supported for ClpLib, with worse performance because data pins are in arbitrary order.
In order to use this option, edit ClpLib_Config.h:

- Comment out the definition of _CONTIGUOUS_DATA
- By default, for non-ordered data pins, definitions are made for JC-JD position.
- If the using other positions than default JC-JD position, then the user should edit the bit definitions for RS, RW, E, BL, D0 – D7.

CLP Library Functions

Functions are implemented at multiple levels. The lowest level implements the hardware required cycles, while the higher level functions are basically calling other functions in order to implement their functionality.
The implementation of the lowest level functions is done to support two approaches of connecting the PMODs: with ordered data pins (JA-JB) and with scattered (arbitrary order) data pins (default JC-JD, any other combination is supported). Read more in Define which PMOD connectors are used for CLP.

Lowest Level Functions:

These functions implement the read or write cycles required in CLP datasheets. It is important that they comply with required timing from the datasheet. On the Cerebot32Mx4 platform in the specified configuration, no additional wait periods for read and write cycles needed to be implemented.

BYTE ClpReadByte(void)

Parameters:
none

Implements a CLP read sequence and returns the read byte. The function is used to read data (RS pin of PmodCLP set before calling this function) or to read status (RS cleared before calling this function). The function implements two approaches of handling data pins, according to _CONTIGUOUS_DATA defined or not.

void ClpWriteByte(BYTE bData)

Parameters:
none

Implements a CLP write sequence for the specified byte. The function is used to write data (RS set before calling this function) or to write commands (RS cleared before calling this function). The function implements two approaches of handling data pins, according to _CONTIGUOUS_DATA defined or not.

Medium Level Functions:

BYTE ClpReadStatus(void)

Parameters:
none
Reads the status of the CLP. It clears the RS and calls the ClpReadByte function.

void ClpWaitUntilNotBusy(void)

Parameters:

none

Waits until the status of the CLP is not busy. This function relies on ClpReadStatus and verifies specific busy status bits.

void ClpWriteCommand(BYTE bCmd)

Parameters:
bCmd the command code byte

Writes the specified byte as command. When the device is ready (using ClpWaitUntilNotBusy) it clears the RS and writes a byte using ClpWriteByte.

void ClpWriteDataByte(BYTE bData)

Parameters:
bData the data byte

Writes the specified byte as data. When the device is ready (using ClpWaitUntilNotBusy) it sets the RS and writes a byte using ClpWriteByte.

Upper Level Functions:

void ClpDisplaySet(BYTE bDisplaySetOptions)

Parameters:
bDisplaySetOptions – display options, possible options (to be OR-ed):
- displaySetOptionDisplaOn – display ON
- displaySetOptionCursorOn – cursor ON
- displaySetBlinkOn – cursor blink ON

Sets the display options. It builds the command code and writes it using ClpWriteCommand.

void ClpDisplayClear(void)

Parameters:
none

Clears the display and returns the cursor home (upper left corner). It writes the specific command using ClpWriteCommand.

void ClpReturnHome(void)

Parameters:
none

Returns the cursor home (upper left corner). It writes the “Return Home” command using ClpWriteCommand.

void ClpInit(BYTE bDisplaySetOptions)

Parameters:

bDisplaySetOptions – display options, possible options (to be OR-ed)

- displaySetOptionDisplayOn – display ON
- displaySetOptionCursorOn – cursor ON
- displaySetBlinkOn – cursor blink ON

Performs the initializing sequence. As specified in the datasheet, it calls the following functions in the specified time sequence:

- ClpWriteCommand for function init command
- ClpDisplaySet providing bDisplaySetOptions argument
- ClpDisplayClear

void ClpSetWriteDdramPosition(BYTE bAdr)

Parameters:

bAdr – the write location. The position in DDRAM where the next data writes will put bytes.

Sets the DDRAM write Position. It builds the command code and writes it using ClpWriteCommand.

void ClpWriteStringAtPos(char *szLn, int idxLine, BYTE idxPos)

Parameters:

szLn – string to be written
idxLine – line where the string will be written
idxPos – the starting position of the string within the line

Writes the specified string at the specified position on the specified line. It sets the corresponding write position (using ClpSetWriteDdramPosition) and then writes data bytes (using ClpWriterDataByte). Strings longer than 0x27 are trimmed.

void ClpSetWriteCgramPosition(BYTE bAdr)

Parameters:

bAdr – the write location. The position in CGRAM where the next data writes will put bytes.

Sets the DDRAM write position. It builds the command code and writes it using ClpWriteCommand.

void ClpWriteBytesAtPosCgram(BYTE *pBytes, BYTE len, BYTE bAdr)

Parameters:

pBytes – pointer to the string of bytes

len - the number of bytes to be written
bAdr – the position in CGRAM where bytes will be written

Writes the specified number of bytes to CGRAM starting at the specified position. It sets the corresponding write position (using ClpSetWriteCgramPosition) and then writes data bytes (using ClpWriteDataByte).

void ClpDisplayShift(BOOL fRight)

Parameters:
fRight – fTrue in order to shift right, else fFalse to shift left

Shifts the entire displayed text one position right or left, depending on the fRight parameter.

void ClpCursorShift(BOOL fRight)

Parameters:
fRight – fTrue in order to shift right, else fFalse to shift left

Shifts the cursor one position right or left, depending on the fRight parameter

Configuration Functions:

void ClpPinsConfigure(void)

Parameters:
none

Configures pins involved in CLP management as outputs. Data pins are not configured here. They will be configured as output / input in the corresponding write / read functions.

Other Functions:

void ClpSetBackLight(BOOL fBl)

Parameters:
fBl – fTrue in order to set the backlight ON, else fFalse to turn it OFF

Sets the backlight.