TMS LITE SDN BHD Company Registration No: 200401033463 / 671971-V

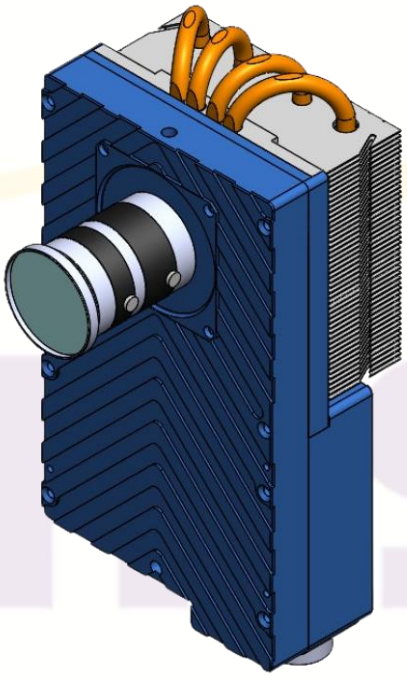
Business Address:
8, Jalan Cassia Selatan 3/1,
Taman Perindustrian Batu Kawan,
14110 Simpang Ampat,
Pulau Pinang, Malaysia.

Tel: +604-588 6393

Correspondence Address:
No. 2A-2, Tingkat Kenari 5,
Desa Ria, Sungai Ara,
11900 Bayan Lepas,
Pulau Pinang, Malaysia.

Tel: +604-646 8428

Email: sales@tms-lite.com | support@tms-lite.com | quotation@tms-lite.com



PTP-40-320X80-4.2V-40A

USER MANUAL

Table of Contents

Table of Contents 2
Chapter 1: Specifications 5
Chapter 2: Connector Description 6
2.1 Power Connector 6
2.2 RS232 Communication Connector 6
2.3 I/O Signal Connector 7
2.4 Liquid Lens Connector 7
Chapter 3: Operation Mode 8
3.1 Power Off Mode 8
3.2 Constant Mode 8
3.3 Test Mode 8
3.4 Strobe Mode 8
3.5 Timing Diagram 9
Chapter 4: Input Signal 10
Chapter 5: Output Signal 10
Chapter 6: Loading an External Image 11
6.1 Prepare and Convert the Image 11
6.1.1 | Design the Image 11
6.1.2 | Convert the Image to Frame BIN 11
6.2 Transfer the Frame BIN File to the SD Card 11
6.2.1 | Power OFF the Controller 11
6.2.2 | Remove the SD Card 11
6.2.3 | Connect the SD Card Reader to the PC 12
6.2.4 | Upload the Frame BIN File to the SD Card 12
6.2.5 | Disconnect the SD Card Reader 12
6.2.6 | Remove the SD Card from the Reader 12
6.3 Install the SD Card into the Eviyos Controller 12
6.3.1 | Insert the SD Card into the Controller 12

6.3.2 | Power ON the Controller 12
6.4 Display the Uploaded Image Using Constant Mode 13
6.4.1 | Select Constant Mode / Strobe Mode 13
6.4.2 | Select the Lighting Pattern 13
6.4.3 | Configure the Output Current 13
6.4.4 | Activate the Lighting Output 13
6.4.5 | Verify the Lighting Pattern 13
6.5 Important Notes 14
Chapter 7: Dimension Drawing 15
Chapter 8: Wiring Diagram 16
Chapter 9: Communication Interface & DLL Integration 17
9.1 Communication Interface 17
9.2 Setup Guideline 17
9.2.1 | Open Visual Studio 17
9.2.2 | Copy the .dll File 17
9.2.3 | Add the .dll File to Your Project 17
9.2.4 | Include the Library in Your Code 18
9.2.5 | Declare Object Type 18
9.2.6 | Verify and Build Your Project 18
9.2.7 | Resolve Any Issues 18
9.3 Board Error Definition 19
9.4 Eviyos SDK Value Conventions 20
9.4.1 | Operation Mode Enumeration 20
9.4.2 | Light-source Current Selector 20
9.5 Standard BoardError Handling Pattern 20
9.6 DLL Library Function 21
9.6.1 | Overview 21
9.6.2 | List of functions 21
9.6.3 | API Function Reference 23

Firmware Changelog.....33

since 2004
www.tms-lite.com

Chapter 1: Specifications

Parameter	Value
ELECTRICAL SPECIFICATIONS	
Supply Voltage	4.2 VDC
Supply Current	40 A
CONTROLLER SPECIFICATIONS	
Pixel Pitch	40 µm
Number of Pixels	320 x 80
Light Emitting Area	12.8 x 3.2 mm^2
Luminous Flux	85 MNits
Type of Lens Mount	C-Mount
System Modes	Test, Constant, Strobe
Communication Interface	RS-232
Digital Input & Digital Output	1 (Input), 1 (Output)
Intensity	25%, 50%, 75%, 100%
Strobe Delay / Strobe Width / Output Width / Output Delay	Adjustable: 0 – 9999 ms (1 ms/step)
Input Trigger Signal	3.3 – 24 V, >5 mA Supports rising/falling-edge trigger
Output Trigger Signal	Supports rising/falling-edge output trigger
Trigger Delay	~1 ms
TEMPERATURE SPECIFICATIONS	
Operating Temperature	Max: 150°C
Storage Temperature	Max: 125°C
Junction Temperature	Max: 150°C

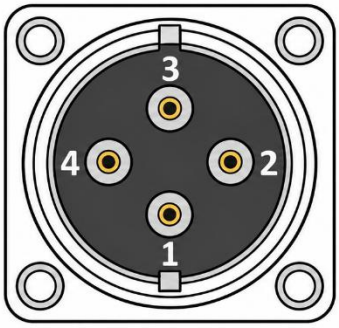
Chapter 2: Connector Description

2.1 Power Connector

WS20 4 PIN FEMALE CONNECTOR PINOUT

(Casing Side View)

(View from the rear side of the panel connector)



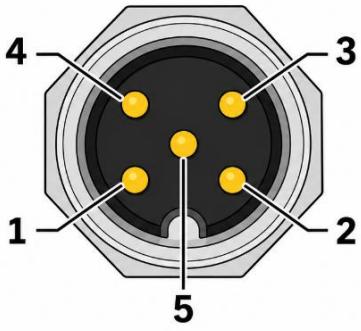
PIN No.	Signal
1	4.2V POWER
2	-
3	GND
4	-

2.2 RS232 Communication Connector

M12 MALE CONNECTOR PINOUT

Front View

(Male Connector – Pin Side)

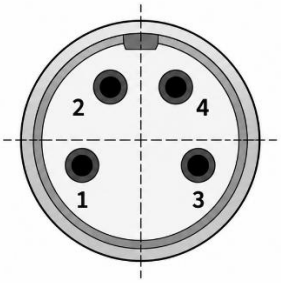


PIN No.	Signal
1	-
2	RX
3	TX
4	-
5	GND

2.3 I/O Signal Connector

M8 FEMALE CONNECTOR PINOUT

Front View
(Female Connector – Pin Side)

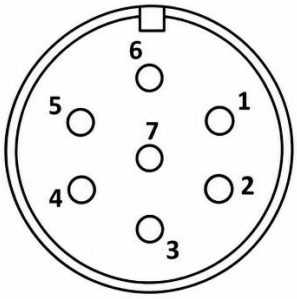


PIN No.	Signal
1	IN +
2	IN -
3	OUT +
4	OUT -

2.4 Liquid Lens Connector

HIROSE 7 PIN FEMALE CONNECTOR PINOUT

(Casing Side View)



PIN No.	Signal
1	5V
2	RX
3	TX
4	GND
5	GND
6	N/A
7	N/A

Chapter 3: Operation Mode

3.1 Power Off Mode

Power Off Mode disables the light output. No illumination is emitted until another operating mode is selected. This mode can be used to stop light output when illumination is not required.

3.2 Constant Mode

Constant Mode provides continuous illumination with adjustable output current.

The output current can be configured to one of four predefined levels: 25%, 50%, 75%, or 100%.

3.3 Test Mode

Test Mode allows the user to select and run a predefined lighting pattern to verify that the lighting operates as expected. The selected pattern provides a practical way to check lighting functionality and identify unexpected output behavior during setup, inspection, or troubleshooting.

- Select a predefined lighting pattern for testing.
- Confirm that the lighting output follows the selected pattern as expected.

3.4 Strobe Mode

Strobe Mode provides triggered, time-controlled illumination for synchronized imaging and inspection applications. Upon detection of a valid **external input trigger signal**, the controller executes the configured lighting sequence **once per trigger event**, independent of the duration (pulse width) of the incoming trigger signal.

The following **four strobe parameters** are configurable:

Parameter	Description
Strobe Delay	Defines the delay between detection of the external input trigger and activation of the lighting output.
Strobe Width	Defines the duration for which the lighting output remains ON during each strobe event.
Output Delay	Defines the delay between the trigger event and activation of the external output signal.
Output Width	Defines the duration of the external output signal.

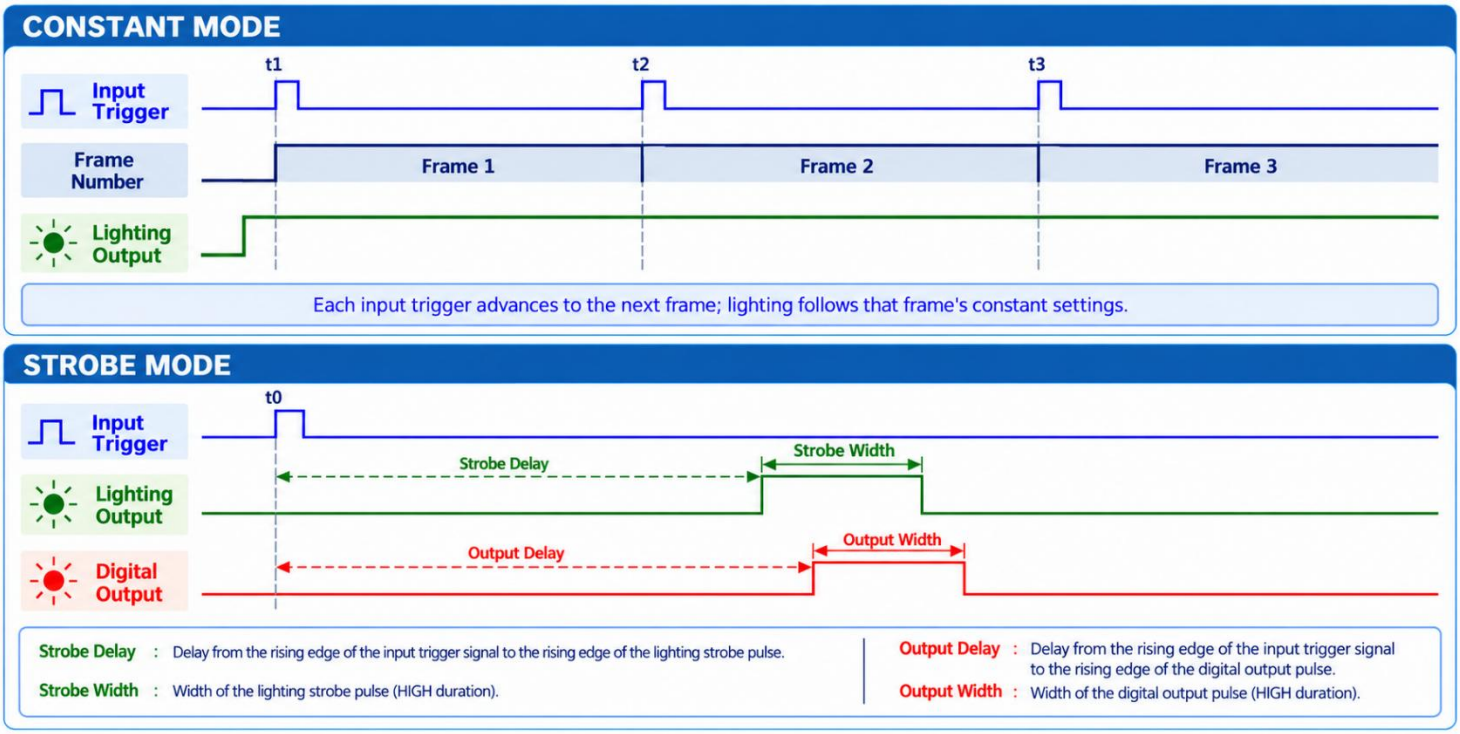
Timing Parameters

All timing-related parameters can be configured within the range of **0 to 9,999 counts**, with a resolution of **1 ms per count**. This corresponds to an adjustable timing range of:

0 to 9999 ms, in 1 ms increments.

The timing parameters include **Strobe Delay, Strobe Width, Output Delay, and Output Width**.

3.5 Timing Diagram



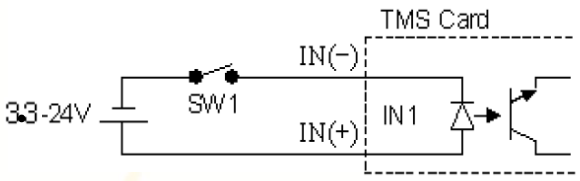
Chapter 4: Input Signal

Input signal is used to turn on the lighting under Strobe/Trigger mode.

The lighting output is turned on after a delay time defined by Strobe Delay, while the turn on duration is defined by Strobe Width.

Acceptable voltage for external input signal is from 3.3-24VDC.

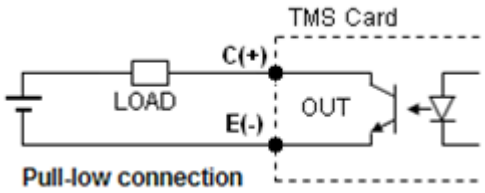
The external trigger signal can be configured as rising edge or falling edge trigger.

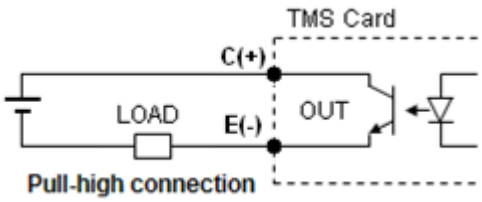


Chapter 5: Output Signal

The output signal is optional and is used to trigger external devices such as smart camera. The output signal is only available in Strobe Mode.

The output pulse is triggered after a delay defined by Output Delay, while the turn on duration is defined by Output Width.





Chapter 6: Loading an External Image

The Eviyos controller supports customized lighting patterns through **Frame BIN files** stored on an SD card. Users can design their own lighting patterns, convert them into the required Frame BIN format using the **PTP Driver Test** software, and transfer the generated files to the SD card.

Once the Frame BIN file has been transferred successfully, the controller can display the selected lighting pattern continuously in **Constant Mode / Strobe Mode**, with adjustable output current levels of 25%, 50%, 75%, or 100%.

Follow the steps below to prepare, transfer, and display a custom lighting pattern.

6.1 Prepare and Convert the Image

6.1.1 | Design the Image

1. Create the desired lighting pattern using an image-editing application
2. Set the image resolution to exactly 320 × 80 pixels.
3. Design the lighting pattern according to the intended application.
4. Save the completed image to the PC.

6.1.2 | Convert the Image to Frame BIN

5. Launch the **PTP Driver Test** software on the PC.
6. Load the prepared 320 × 80 image using the software's external image loading function.
7. Convert the image into the required **Frame BIN (.bin)** format.
8. Save the generated Frame BIN file to the PC.
9. Verify that the file has been generated successfully.

6.2 Transfer the Frame BIN File to the SD Card

6.2.1 | Power OFF the Controller

10. Turn OFF the Eviyos controller completely.
11. Ensure that the controller is no longer powered before removing the SD card.
12. Important: Do not insert or remove the SD card while the controller is powered ON.

6.2.2 | Remove the SD Card

13. Carefully remove the SD card from the Eviyos controller.
14. Insert the SD card into a compatible SD card reader.

6.2.3 | Connect the SD Card Reader to the PC

15. Connect the SD card reader to an available USB port on the PC.
16. Wait until the PC detects the SD card.
17. Open File Explorer and locate the SD card under This PC.
18. Ensure that the SD card can be accessed successfully.

6.2.4 | Upload the Frame BIN File to the SD Card

19. Locate the generated Frame BIN file on the PC.
20. Copy the Frame BIN file.
21. Open the SD card storage location in File Explorer.
22. Paste the Frame BIN file into the required directory on the SD card.
23. Wait until the file transfer is fully completed.
24. Verify that the Frame BIN file is successfully stored on the SD card.

6.2.5 | Disconnect the SD Card Reader

25. Close any files or folders currently accessing the SD card.
26. Safely eject the SD card through the operating system.
27. Disconnect the SD card reader from the PC.

6.2.6 | Remove the SD Card from the Reader

28. Carefully remove the SD card from the SD card reader.
29. Ensure that the SD card is ready to be reinstalled into the controller.

6.3 Install the SD Card into the Eviyos Controller

6.3.1 | Insert the SD Card into the Controller

30. Ensure that the Eviyos controller remains powered OFF.
31. Insert the SD card containing the Frame BIN file into the controller's SD card slot.
32. Ensure that the SD card is properly inserted.

6.3.2 | Power ON the Controller

33. Turn ON the Eviyos controller.
34. Wait for the controller to complete its initialization process.
35. The Frame BIN file stored on the SD card is now available for use, provided it follows the controller's required file format and storage configuration

6.4 Display the Uploaded Image Using Constant Mode

After transferring the Frame BIN file to the SD card and powering ON the controller, users can display and verify the customized lighting pattern using Constant Mode / Strobe Mode.

6.4.1 | Select Constant Mode / Strobe Mode

36. Set the Eviyos controller to Constant Mode / Strobe Mode.
37. Ensure that the controller has successfully entered Constant Mode / Strobe Mode.

6.4.2 | Select the Lighting Pattern

38. Select the desired lighting pattern from the Frame BIN file stored on the SD card.
39. Ensure that the correct lighting pattern has been selected.

6.4.3 | Configure the Output Current

40. Configure the output current according to the lighting requirements.
41. Select one of the four available output current levels:
    25% – Low output current.
    50% – Medium output current.
    75% – High output current.
    100% – Maximum output current.

6.4.4 | Activate the Lighting Output

42. Enable the lighting output in Constant Mode.
43. The controller will continuously display the selected lighting pattern using the configured output current.

6.4.5 | Verify the Lighting Pattern

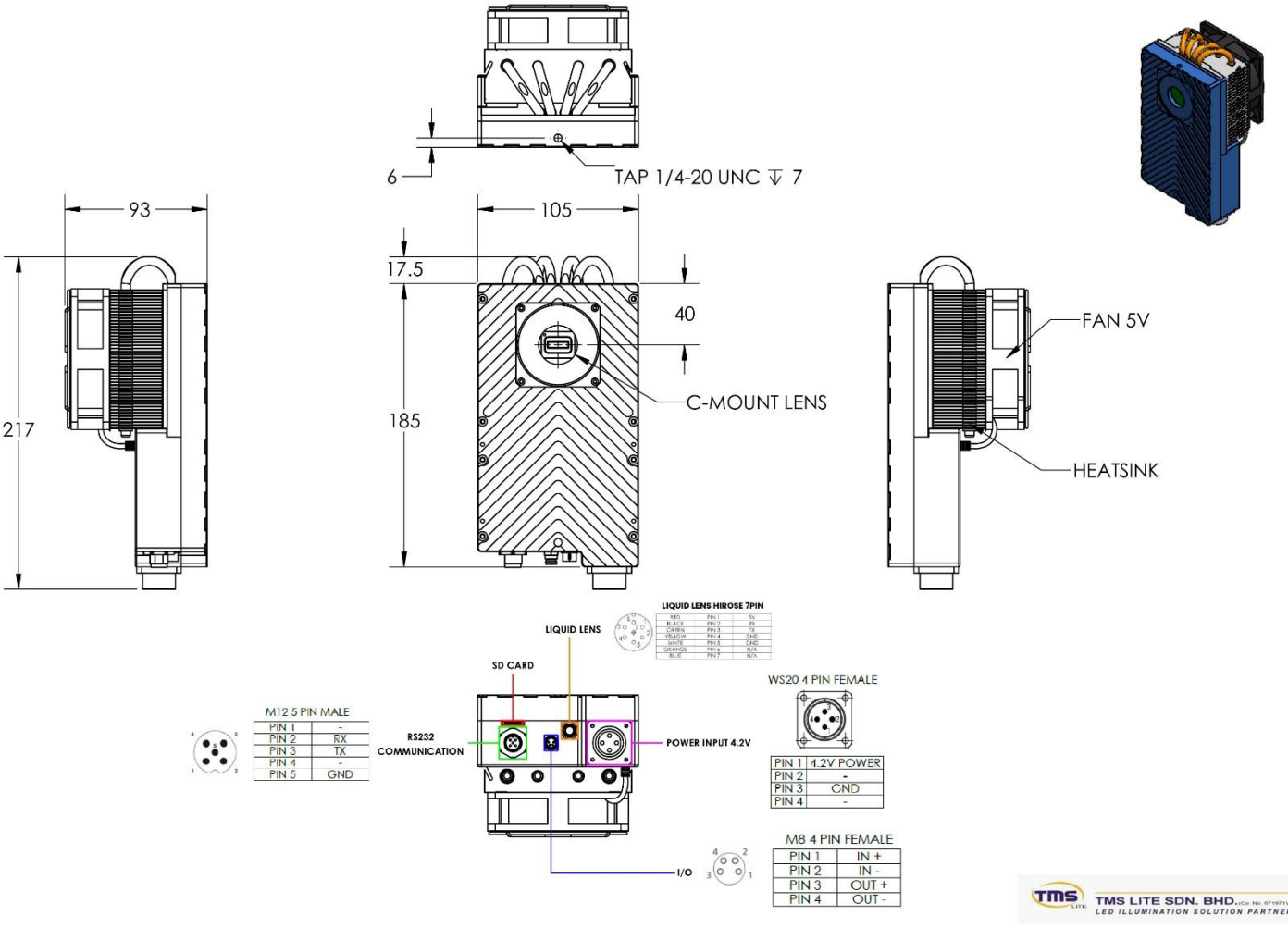
44. Observe the lighting output.
45. Verify that the displayed pattern matches the original designed image.
46. Ensure that the lighting output is stable and operates as expected.
47. If the displayed pattern is incorrect, check the Frame BIN file and repeat the preparation or transfer procedure if necessary.

6.5 Important Notes

- **Always power OFF the Eviyos controller before inserting or removing the SD card.**
- The designed image must have a resolution of **320 × 80 pixels**.
- Use the **PTP Driver Test** software to convert the image into the required Frame BIN format.
- Ensure that the Frame BIN file is successfully transferred to the SD card.
- Do not disconnect the SD card reader during file transfer.
- Safely eject the SD card before removing it from the PC.
- Ensure that the SD card is properly inserted before powering ON the controller.
- Use **Constant Mode / Strobe Mode** to display the uploaded lighting pattern continuously.
- The output current can be configured to **25%, 50%, 75%, or 100%**.

since 2004
www.tms-lite.com

Chapter 7: Dimension Drawing



TMS LITE SDN. BHD. LED ILLUMINATION SOLUTION PARTNER
www.tms-lite.com

TMS LITE SDN BHD Company Registration No: 200401033463 / 671971-V

Business Address:
8, Jalan Cassia Selatan 3/1,
Taman Perindustrian Batu Kawan,
14110 Simpang Ampat,
Pulau Pinang, Malaysia.

Tel: +604-588 6393

Correspondence Address:
No. 2A-2, Tingkat Kenari 5,
Desa Ria, Sungai Ara,
11900 Bayan Lepas,
Pulau Pinang, Malaysia.

Tel: +604-646 8428

Email: sales@tms-lite.com | support@tms-lite.com | quotation@tms-lite.com

Chapter 8: Wiring Diagram

since 2004
www.tms-lite.com

Chapter 9: Communication Interface & DLL Integration

9.1 Communication Interface

Parameter	Protocol Setting
Interface	RS-232 serial
Baud rate	57600 bit/s
Data bits	8
Parity	None
Stop bits	1

9.2 Setup Guideline

9.2.1 | Open Visual Studio

- Launch Visual Studio.
- Open your existing C# project or create a new one targeting .NET Framework 4.7.2 or above.

9.2.2 | Copy the .dll File

- Copy the .dll file to a suitable directory within your project. A common practice is to create a lib folder within your project directory to store external libraries.

9.2.3 | Add the .dll File to Your Project

Right-click on Your Project:

- In the Solution Explorer, right-click on your project name.
- Select Add > Reference.

Browse for the .dll File:

- In the Reference Manager, click Browse at the bottom.
- Navigate to the directory where you placed the .dll file (e.g., lib folder).
- Select the .dll file and click Add.

LCLibrary.dll

Verify the Reference:

- Ensure that the .dll file appears in the list of selected components.
- Click OK to close the Reference Manager.

9.2.4 | Include the Library in Your Code

- Open the source files where you need to use the external library.
- Add the appropriate using statement at the top of these files. For example:
  using LCLibrary;

9.2.5 | Declare Object Type

```
using LCLibrary;

EviyosDLL dll = new EviyosDLL();
```

9.2.6 | Verify and Build Your Project

- Ensure that your code is using the classes and methods provided by the external library correctly.
- Save all changes.
- Build your project by clicking Build > Build Solution (or pressing Ctrl+Shift+B).

9.2.7 | Resolve Any Issues

- If there are any issues during the build, ensure that the library is compatible with your project's .NET version. (Minimum .Net Framework 4.7.2)
- Check for any missing dependencies that the external library might need.

Example Directory Structure

Here is an example of how your project directory might be organized:

```
MyProject/
├── lib/
│   └── LCLibrary.dll
├── src/
│   └── YourSourceFile1.cs
├── MyProject.sln
└── MyProject.csproj
```

9.3 Board Error Definition

EviyosDLL Library use BoardError for operations that report communication, initialization, validation, or controller-read status. A caller should proceed only when the returned value is BoardError.OK.

```
public enum BoardError
{
    OK = 0,
    CONNECTION_FAILED,
    COMPORT_NOT_OPEN,
    FAILED_TO_WRITE,
    FAILED_TO_READ,
    FAILED_TO_INIT,
    LENGTH_OUT_OF_RANGE,
    OUT_OF_RANGE
}
```

Value	BoardError	Definition
0	OK	The operation was completed successfully.
1	CONNECTION_FAILED	The connection could not be established.
2	COMPORT_NOT_OPEN	The configured COM port is not open.
3	FAILED_TO_WRITE	The command could not be transmitted.
4	FAILED_TO_READ	The controller response could not be read.
5	FAILED_TO_INIT	The controller could not be initialized.
6	LENGTH_OUT_OF_RANGE	The supplied length is outside the supported range.
7	OUT_OF_RANGE	The supplied value is outside the supported range.

9.4 Eviyos SDK Value Conventions

9.4.1 | Operation Mode Enumeration

Value	OperationMode	Meaning
0	POWER_OFF	Controller power-off state
1	CONSTANT	Normal constant illumination
2	TESTING	Testing mode
3	STROBE	Normal strobe mode
4	INVALID	Invalid / fallback state; cannot be set

9.4.2 | Light-source Current Selector

Selector	Current
0	25%
1	50%
2	75%
3	100%

9.5 Standard BoardError Handling Pattern

```
BoardError result = controller.ReadParameter();

if (result != BoardError.OK)
{
    // Stop the current operation and handle/report the error.
    return;
}

// Safe to continue with controller operations.
```

9.6 DLL Library Function

9.6.1 | Overview

Connection	ComportConnect(int portnum) / ComportDisconnect()
Serial Settings	57600 baud, 8 data bits, 1 stop bit, parity None
Read status	BoardError return; check BoardError.OK
Current selector	0 = 25%, 1 = 50%, 2 = 75%, 3 = 100%
Frame count	ReadMaxFrame() populates MaxFrame; indices 0 through MaxFrame-1

9.6.2 | List of functions

```
➜ byte ComportConnect(int portnum);
➜ void ComportDisconnect();
➜ BoardError ReadDLLVersion(ref string version);
➜ BoardError ReadFirmwareVersion(ref string version);
➜ BoardError ReadParameter();
➜ BoardError ReadRemarks(ref ushort value);
➜ BoardError SetRemarks(ushort value, bool verify = false);
➜ BoardError ReadTemperature(int index, ref int value);
➜ BoardError ReadVoltage(int index, ref ushort value);
➜ BoardError ReadAutoState(ref int value);
➜ BoardError ReadCurrent(ref int value);
➜ BoardError ReadInput(ref int value);
➜ BoardError ReadOperationMode(ref OperationMode mode);
➜ BoardError SetOperationMode(OperationMode mode, bool verify = false);
➜ BoardError SetCurrent(int value, bool verify = false);
➜ BoardError ReadStartupCurrent(ref int value);
➜ BoardError SetStartupCurrent(int value, bool verify = false);
➜ BoardError ReadNormalDuration(ref int value);
➜ BoardError SetNormalDuration(int value, bool verify = false);
➜ BoardError ClearScreen(bool verify = false);
➜ BoardError NextFrame(bool verify = false);
➜ BoardError PreviousFrame(bool verify = false);
➜ BoardError ReadTestImage(ref int value);
```

```
➜ BoardError SetTestImage(int value, bool verify = false);
➜ BoardError ReadCurrentFrame(ref int value);
➜ BoardError ReadMaxFrame(ref int value);
➜ BoardError SetCurrentFrame(int value, bool verify = false);
➜ BoardError SendBitmap(Bitmap bitmap);
➜ BoardError SendBitmapFast(Bitmap bitmap);
➜ BoardError SendData(List<byte> data);
➜ BoardError SendData(byte[] data);
➜ BoardError SetOutput(bool value, bool verify = false);
➜ BoardError Pause(bool verify = false);
➜ BoardError Resume(bool verify = false);
➜ BoardError StrobeLighting(bool verify = false);
➜ BoardError ReadStrobeWidth(ref int value);
➜ BoardError SetStrobeWidth(int value, bool verify = false);
➜ BoardError ReadStrobeDelay(ref int value);
➜ BoardError SetStrobeDelay(int value, bool verify = false);
➜ BoardError ReadOutputWidth(ref int value);
➜ BoardError SetOutputWidth(int value, bool verify = false);
➜ BoardError ReadOutputDelay(ref int value);
➜ BoardError SetOutputDelay(int value, bool verify = false);
➜ BoardError SaveEEPROM(bool verify = false);
➜ BoardError ReadEEPROM(bool verify = false);
➜ BoardError SetSDEnabled(bool value, bool verify = false);
➜ BoardError ReadSDStatus(ref bool enable);
➜ BoardError SetDrawingEnabled(bool value, bool verify = false);
➜ BoardError SetLensCurrent(int value, bool verify = false);
➜ BoardError ReadLensCurrent(ref int value);
➜ BoardError ConnectLens(ref bool connected);
```

9.6.3 | API Function Reference

Connection

1. byte ComportConnect(int portnum);

Description	Opens the requested COM port using: Baud Rate = 57600 Data Bits = 8 Stop Bits = 1 Parity = None After opening the port, communication is verified by reading the firmware version.
Parameter	int portnum COM port number.
Return	byte 0 = Connected successfully 1 = Connection / communication error

2. void ComportDisconnect();

Description	Closes the currently opened serial COM port.
Parameter	NONE
Return	NONE

Version

3. BoardError ReadDLLVersion(ref string version);

Description	Returns the current Eviyos DLL interface version.
Parameter	1) ref string version Reference string receiving the driver version.
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

4. BoardError ReadFirmwareVersion(ref string version);

Description	Reads the controller firmware version.
Parameter	1) ref string version Reference string receiving the firmware version.
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

Diagnostic / Sensor

5. BoardError ReadRemarks(ref ushort value);

Description	Reads the 16-bit notebook/remarks register from the Eviyos controller.
Parameter	1) ref ushort value Reference value that receives the 16-bit notebook/remarks register.
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

6. BoardError SetRemarks(ushort value, bool verify = false);

Description	Sets the 16-bit notebook/remarks register on the Eviyos controller.
Parameter	1) ushort value 16-bit notebook/remarks register to write. 2) bool verify = false Whether to verify the controller OK response internally.
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

7. BoardError ReadTemperature(int index, ref int value);

Description	Reads the temperature sensor reading from the Eviyos controller. Sensor index: 1-2.
Parameter	1) int index One-based sensor index. 2) ref int value Reference value that receives the temperature sensor reading.
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

8. BoardError ReadAutoState(ref int value);

Description	Reads the automatic state from the Eviyos controller.
Parameter	1) ref int value Reference value that receives the automatic state.
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

9. BoardError ReadCurrent(ref int value);

Description	Reads the current selection (0 = 25%, 1 = 50%, 2 = 75%, 3 = 100%) from the Eviyos controller.
Parameter	1) ref int value Reference value receiving selector 0 (25%), 1 (50%), 2 (75%), or 3 (100%).
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

10. BoardError ReadInput(ref int value);

Description	Reads the digital input state from the Eviyos controller.
Parameter	1) ref int value Reference value that receives the digital input state.
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

Operation Mode

11. BoardError ReadOperationMode(ref OperationMode mode);

Description	Reads the controller operation mode.
Parameter	1) ref OperationMode mode Reference value receiving POWER_OFF, CONSTANT, TESTING, STROBE, or INVALID.
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

12. BoardError SetOperationMode(OperationMode mode, bool verify = false);

Description	Changes the controller operation mode. INVALID is not accepted. With verify=true, mode is read back and compared.
Parameter	1) OperationMode mode Requested mode (excluding INVALID). 2) bool verify = false Read back and compare the controller operation mode when true.
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

Current / Timing

13. BoardError ReadStartupCurrent(ref int value);

Description	Reads the current selector (0 = 25%, 1 = 50%, 2 = 75%, 3 = 100%).
Parameter	1) ref int value Reference value receiving selector 0 (25%), 1 (50%), 2 (75%), or 3 (100%).
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

14. BoardError SetCurrent(int value, bool verify = false);

Description	Sets the current selector on the Eviyos controller: 0 = 25%, 1 = 50%, 2 = 75%, 3 = 100%.
Parameter	1) int value Current selector to write: 0 = 25%, 1 = 50%, 2 = 75%, or 3 = 100%. 2) bool verify = false Whether to verify the controller OK response internally.
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

15. BoardError ReadStartupCurrent(ref int value);

Description	Reads the startup current selector (0 = 25%, 1 = 50%, 2 = 75%, 3 = 100%).
Parameter	1) ref int value Reference value receiving selector 0 (25%), 1 (50%), 2 (75%), or 3 (100%).
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

16. BoardError SetStartupCurrent(int value, bool verify = false);

Description	Sets the startup current selector: 0 = 25%, 1 = 50%, 2 = 75%, 3 = 100%.
Parameter	1) int value Startup current selector to write: 0 = 25%, 1 = 50%, 2 = 75%, or 3 = 100%. 2) bool verify = false Whether to verify the controller OK response internally.
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

17. BoardError ReadNormalDuration(ref int value);

Description	Reads the normal mode auto next duration from the Eviyos controller.
Parameter	1) ref int value Reference value that receives the normal mode auto next duration. (in 10 millisecond/step)
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

18. BoardError SetNormalDuration(int value, bool verify = false);

Description	Sets the normal mode auto next duration on the Eviyos controller. Value: 0-9999; (in 10 millisecond/step)
Parameter	1) int value Normal mode auto next duration to write. 2) bool verify = false Whether to verify the controller OK response internally.
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

Frame / Test Image

19. BoardError ClearScreen(bool verify = false);

Description	Clears the displayed image.
Parameter	1) bool verify = false Whether to verify the controller OK response internally.
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

20. BoardError NextFrame(bool verify = false);

Description	Advances to the next frame.
Parameter	1) bool verify = false Whether to verify the controller OK response internally.
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

21. BoardError PreviousFrame(bool verify = false);

Description	Moves to the previous frame.
Parameter	1) bool verify = false Whether to verify the controller OK response internally.
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

22. BoardError Pause(bool verify = false);

Description	Pauses the Eviyos controller auto next frame.
Parameter	1) bool verify = false Whether to verify the controller OK response internally.
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

23. BoardError Resume(bool verify = false);

Description	Resumes the Eviyos controller auto next frame.
Parameter	1) bool verify = false Whether to verify the controller OK response internally.
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

24. BoardError ReadTestImage(ref int value);

Description	Reads the selected test image from the Eviyos controller.
Parameter	1) ref int value Reference value that receives the selected test image.
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

25. BoardError SetTestImage(int value, bool verify = false);

Description	Sets the selected test image on the Eviyos controller. Value: 0-13.
Parameter	1) int value Selected test image to write. 2) bool verify = false Whether to verify the controller OK response internally.
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

26. BoardError ReadCurrentFrame(ref int value);

Description	Reads the current frame index from the Eviyos controller.
Parameter	1) ref int value Reference value that receives the current frame index.
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

27. BoardError ReadMaxFrame(ref int value);

Description	Reads and caches the frame count used for frame-index validation. Refreshes the MaxFrame property used by SetCurrentFrame.
Parameter	1) ref int value Reference integer receiving the maximum frame count.
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

28. BoardError SetCurrentFrame(int value, bool verify = false);

Description	Selects a frame from the cached frame range [0, MaxFrame - 1]. Value: 0 through MaxFrame-1; call ReadMaxFrame first.
Parameter	1) int value Zero-based frame index. 2) bool verify = false Whether to verify the controller OK response internally.
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

29. BoardError SetDrawingEnabled(bool value, bool verify = false);

Description	Sets the drawing enable state on the Eviyos controller.
Parameter	1) bool value Drawing enable state to write. 2) bool verify = false Whether to verify the controller OK response internally.
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

Bitmap / Raw Data

30. BoardError SendBitmap(Bitmap bitmap);

Description	Transmits the first 320 x 80 pixels as 8-bit grayscale samples. Bitmap must be at least 320 x 80. Sends the first 320 x 80 pixels as one-byte grayscale samples.
Parameter	1) Bitmap bitmap Source bitmap at least 320 x 80 pixels in size.
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

31. BoardError SendBitmapFast(Bitmap bitmap);

Description	Packs each bitmap pixel into a monochrome bit (1 when brightness >= 128), MSB-first. Positive-sized bitmap; converts brightness to a 1-bit stream with threshold 128.
Parameter	1) Bitmap bitmap Bitmap containing one or more pixels.
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

32. BoardError SendData(List<byte> data);

Description	Transmits a byte list raw frame.
Parameter	1) List<byte> data Raw bytes to transmit.
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

33. BoardError SendData(byte[] data);

Description	Transmits a byte array raw frame.
Parameter	1) byte[] data Raw bytes to transmit.
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

Output / Strobe

34. BoardError SetOutput(bool value, bool verify = false);

Description	Sets the output enable state on the Eviyos controller.
Parameter	1) bool value Output enable state to write. 2) bool verify = false Whether to verify the controller OK response internally.
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

35. BoardError StrobeLighting(bool verify = false);

Description	Triggers the controller lighting strobe.
Parameter	1) bool verify = false Whether to verify the controller OK response internally.
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

36. BoardError ReadStrobeWidth(ref int value);

Description	Reads the strobe pulse width from the Eviyos controller.
Parameter	1) ref int value Reference value that receives the strobe pulse width. (in 1 millisecond/step)
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

37. BoardError SetStrobeWidth(int value, bool verify = false);

Description	Sets the strobe pulse width on the Eviyos controller. Value: 0-9999; (in 1 millisecond/step)
Parameter	1) int value Strobe pulse width to write. 2) bool verify = false Whether to verify the controller OK response internally.
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

38. BoardError ReadStrobeDelay(ref int value);

Description	Reads the strobe delay from the Eviyos controller.
Parameter	1) ref int value Reference value that receives the strobe delay. (in 1 millisecond/step)
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

39. BoardError SetStrobeDelay(int value, bool verify = false);

Description	Sets the strobe delay on the Eviyos controller. Value: 0-9999; (in 1 millisecond/step)
Parameter	1) int value Strobe delay to write. 2) bool verify = false Whether to verify the controller OK response internally.
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

40. BoardError ReadOutputWidth(ref int value);

Description	Reads the output pulse width from the Eviyos controller.
Parameter	1) ref int value Reference value that receives the output pulse width. (in 1 millisecond/step)
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

41. BoardError SetOutputWidth(int value, bool verify = false);

Description	Sets the output pulse width on the Eviyos controller. Value: 0-9999; (in 1 millisecond/step)
Parameter	1) int value Output pulse width to write. 2) bool verify = false Whether to verify the controller OK response internally.
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

42. BoardError ReadOutputDelay(ref int value);

Description	Reads the output pulse delay from the Eviyos controller.
Parameter	1) ref int value Reference value that receives the output pulse delay. (in 1 millisecond/step)
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

43. BoardError SetOutputDelay(int value, bool verify = false);

Description	Sets the output pulse delay on the Eviyos controller. Value: 0-9999; (in 1 millisecond/step)
Parameter	1) int value Output pulse delay to write. 2) bool verify = false Whether to verify the controller OK response internally.
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

EEPROM

44. BoardError SaveEEPROM(bool verify = false);

Description	Saves controller configuration into EEPROM.
Parameter	1) bool verify = false Whether to verify the controller OK response internally.
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

45. BoardError ReadEEPROM(bool verify = false);

Description	Loads controller configuration from EEPROM.
Parameter	1) bool verify = false Whether to verify the controller OK response internally.
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

SD Settings

46. BoardError SetSDEnabled(bool value, bool verify = false);

Description	Sets the SD card enable state on the Eviyos controller.
Parameter	1) bool value Sd card enable state to write. 2) bool verify = false Whether to verify the controller OK response internally.
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

47. BoardError ReadSDStatus(ref bool enable);

Description	Reads SD card enable status (1 = enabled). A raw value of 1 means enabled; other values map to false.
Parameter	1) ref bool enable Reference Boolean receiving whether the SD interface is enabled.
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

Liquid Lens

48. BoardError SetLensCurrent(int value, bool verify = false);

Description	Sets the liquid lens drive current on the Eviyos controller. Value: 0-1023
Parameter	1) int value Liquid lens drive current to write. 2) bool verify = false Whether to verify the controller OK response internally.
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

49. BoardError ReadLensCurrent(ref int value);

Description	Reads the liquid lens drive current value from the Eviyos controller.
Parameter	1) ref int value Reference value that receives the liquid lens drive current value.
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

50. BoardError ConnectLens(ref bool connected);

Description	Checks the liquid lens connection state.
Parameter	1) ref bool connected Reference Boolean receiving true when the lens is connected.
Return	BoardError.OK on success; otherwise the returned BoardError failure code.

TMS LITE since 2004 www.tms-lite.com

Firmware Changelog

Version	Release Date	Description
1.0.0.1	2025-07-04	• Initial Release
1.0.0.2	2026-01-22	• Optimize the code • Integrated Liquid Lens
1.0.0.3	2026-10-13	• Command Protocol Standardizing

since 2004
www.tms-lite.com

THANK YOU

Thank you for placing your trusts in us and giving us a chance to provide you with our services. We have always valued our customers and emphasized on providing the best solution to ease their application process. We hope to continue this relationship in the forthcoming year with great understanding and respect.

Thank You!