HDJD-J822

Programming Example

Application Note 5071

Introduction

This application note describes the programming example of a HDJD-J822 system based on the following options:

a) Internal reference voltage is used.
b) GAINX, GAINY and GAINZ are set to 1X.
c) Sensor used is positive slope sensor.
d) LED driver is active high enabled.
e) Pins A1 and A0 are tied to DVSS.

In normal mode, the programming example is done with the following choices.

a) DUTY_LIMIT (DUTY_LIMIT_HI : DUTY_LIMIT_LO) value is at default value 0x0F32.
b) BETA, Brightness scalar value is 0xFF at the default brightness value.
c) Trim offset is not enabled.

Functionality / Mode of Operation

There are two operating modes. They are as follows:

a) Optical Feedback (normal mode)
b) Open Loop (calibration)

Open Loop mode is done only once when performing calibration to obtain thirty-one calibration data registers.

Optical Feedback or normal mode is always used when the thirty-one calibration data registers are available.

Writing a Data Byte to a Register

Figure 1 shows the protocol to write a byte to a register.

The master first generates a START condition. Then it sends the HDJD-J822 device write address to indicate that the master wants to write to the device. The HDJD-J822 device will then acknowledge the master.

The master then writes the register address it wants to access and waits for the HDJD-J822 to acknowledge. After acknowledgement is received from HDJD-J822, the master then writes the new register value. The HDJD-J822 will acknowledge the new value, and the master will generate a STOP condition to end the data transfer.

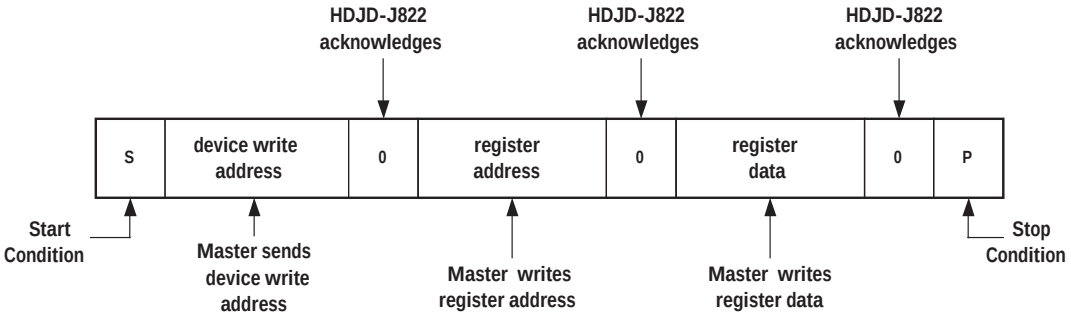

Figure 1. Register byte write protocol.

Reading a Data Byte from a Register

Figure 2 shows the protocol to read a byte from a register.

To read from a register, the master first generates a START condition. Then it sends the HDJD-J822 device write address to indicate the master wants to write to the device. The HDJD-J822 will then acknowledge the master.

The master writes the register address it wants to access and waits for the HDJD-J822 to acknowledge. The master then generates a repeated START condition and sends the HDJD-J822 device read address to indicate that the master wants to read from the device. The HDJD-J822 will then acknowledge the master.

The master reads the register data sent by the HDJD-J822. The master then sends a 'no acknowledge signal' followed by a STOP condition to end the data transfer.

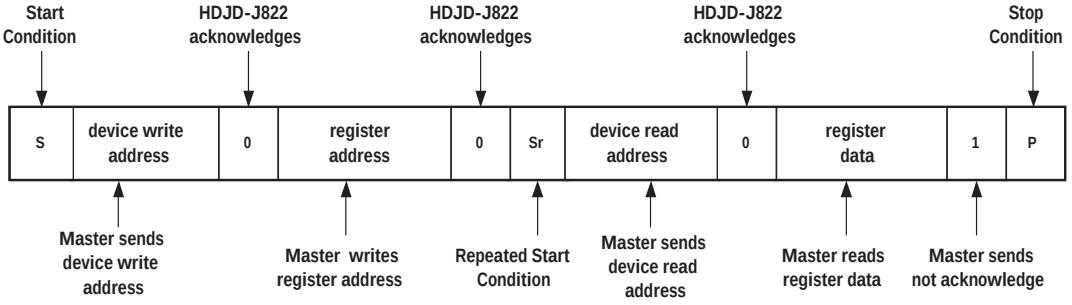

Figure 2. Register byte read protocol.

Device Write and Device Read Address.

The addresses depend on the external pins of the HDJD-J822 as follows.

A1	A0	Device write address	Device read address
0	0	0xA8	0xA9
0	1	0xAA	0xAB
1	0	0xAC	0xAD
1	1	0xAE	0xAF

Figure 3. External A1, A0 pins and device addresses truth table.

In this programming example, the HDJD-J822 pins A1 and A0 are tied to DVSS. The device write address is 0xA8 and the device read address is 0xA9, i.e., the format becomes as follows:

FOR WRITE:
<START><0xA8><Register address> <Register data><STOP>

e.g., To write to the CTRL1 register (address 0x01), with a value 0x00, the I2C data sequence is S[0xA8][0x01][0x00]P

FOR READ:
<START><0xA8><Register address><START><0xA9><Register data><STOP>

e.g., To read the status register (address 0x09), the I2C data sequence is S[0xA8][0x09]S[0xA9][*ST*]P

Notes:
Blue italic variable '*ST*' represents a memory location to store the value read from the HDJD-J822.

The hexadecimal values in square brackets do not include the acknowledgement bit. The acknowledgement bit is checked for a '0' (acknowledged) during writing process. During the read process, the master or controller must send a '1' (not acknowledged) to the HDJD-J822.

Calibration Mode

Figures 4 shows the flow chart for the calibration procedure.

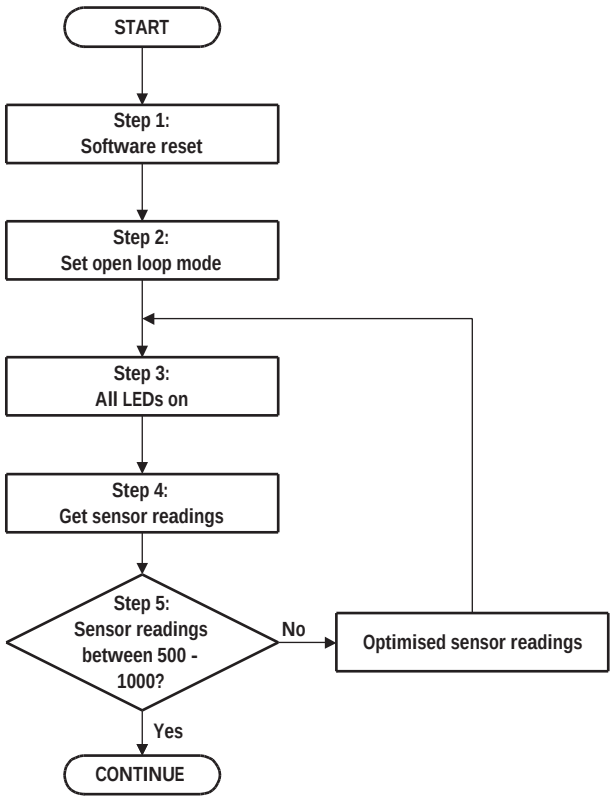

Figure 4. Calibration flow chart.

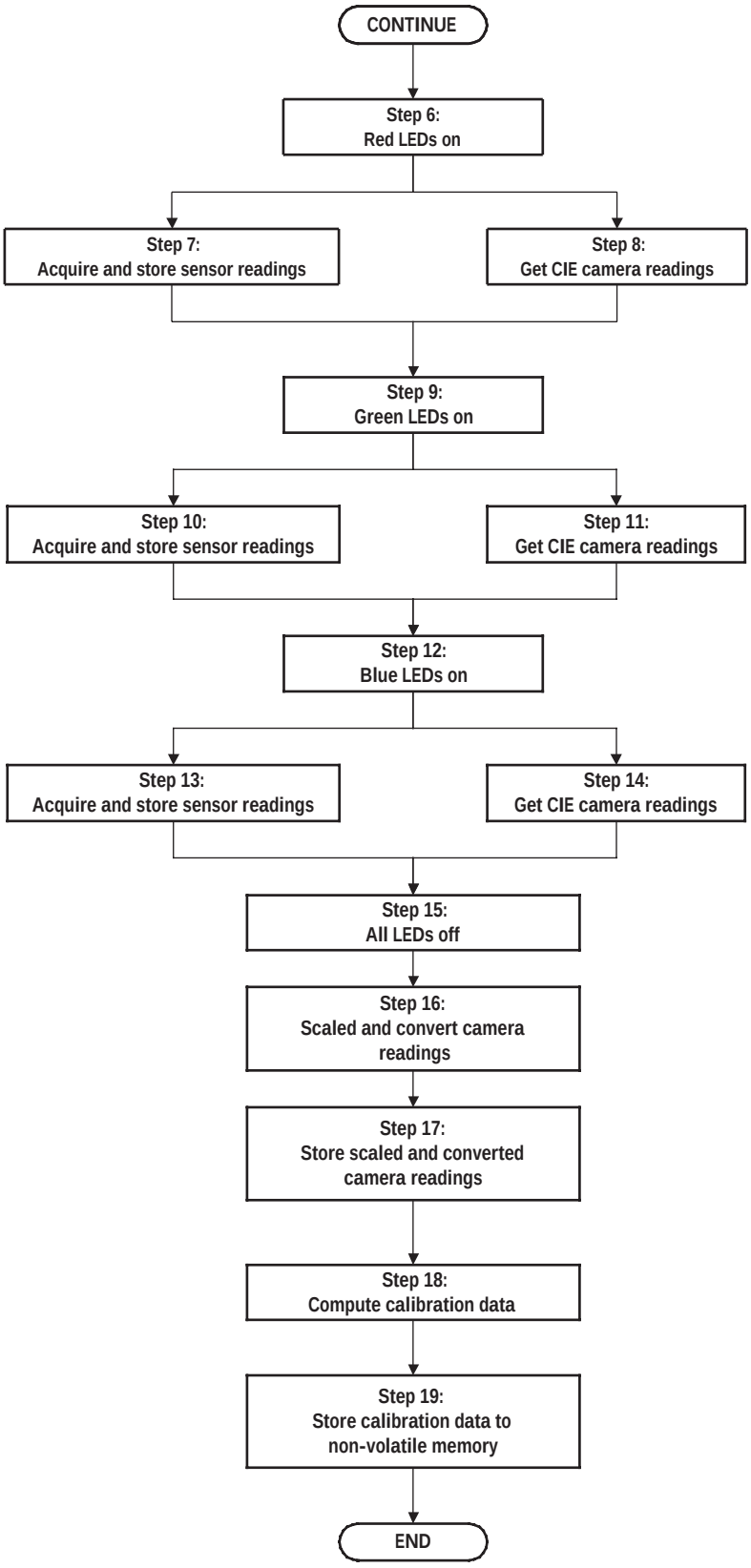

Figure 4. Calibration flow chart (continued).

CALIBRATION: STEP 1 – 5

Step	I2C Data Sequence	Comments
1.1	S[0xA8][0x01][0x01]P	Software reset.
1.2	S[0xA8][0x09]S[0xA9][*ST*]P	Check *ST* = 0x02 to ensure reset is completed.
2.1	S[0xA8][0x03][0x20]P	Set to open loop mode.
3.1	S[0xA8][0x0B][0xFF]P, S[0xA8][0x0C][0x0F]P	Set blue, green and red LED duty to 0x0FFF. (100% duty cycle)
3.2	S[0xA8][0x0D][0xFF]P, S[0xA8][0x0E][0x0F]P	
3.3	S[0xA8][0x0F][0xFF]P, S[0xA8][0x10][0x0F]P	
3.4	S[0xA8][0x01][0x02]P	Turn on all LEDs.
4.1	S[0xA8][0x02][0x02]P	Acquire sensor values.
4.2	S[0xA8][0x02]S[0xA9][*R2*]P	If *R2* = 0x00, then sensor acquisition is completed.
5.1	S[0xA8][0x84]S[0xA9][*ZL*]P, S[0xA8][0x85]S[0xA9][*ZH*]P	Check sensor values Z, Y and X i.e. (*ZH:ZL*), (*YH:YL*) and (*XH:XL*) are between 500 to 1000 or 0x01F4 to 0x03E8.
5.2	S[0xA8][0x86]S[0xA9][*YL*]P, S[0xA8][0x87]S[0xA9][*YH*]P	
5.3	S[0xA8][0x88]S[0xA9][*XL*]P, S[0xA8][0x89]S[0xA9][*XH*]P	

Note: Blue color alphabet indicates that data is read from HDJD-J822 and stored to memory location.

If the sensor values obtained are greater than 1000 or 0x03E8, the following actions can be taken:

a) Decrease the peak LED current.
b) Increase the distance between the sensing area and the sensor. Ensure no external ambient light interference if this is done.
c) Insert neutral density grey filter.
d) Reduce the gain of the sensor channel voltage.
e) Use an external operational amplifier to get the sensor voltages below the ADC voltage reference.
f) Use a higher external voltage reference. If an external voltage reference is used, the I2C data sequences described in this application note need to be modified. Make the changes and refer to the AN5070 application note, *Using the HDJD-J822 Color Management System Feedback Controller ASIC*.

CALIBRATION: STEP 6 – 8

Step	I2C Data Sequence	Comments
6.1	S[0xA8][0x01][0x00]P	Turn on only red LEDs at 100% duty cycle.
6.3	S[0xA8][0x0B][0x00]P, S[0xA8][0x0C][0x00]P	
6.4	S[0xA8][0x0D][0x00]P, S[0xA8][0x0E][0x00]P	
6.5	S[0xA8][0x0F][0xFF]P, S[0xA8][0x10][0x0F]P	
6.6	S[0xA8][0x01][0x02]P	
7.1	S[0xA8][0x02][0x02]P	Acquire sensor values.
7.2	S[0xA8][0x02]S[0xA9][*R2*]P	If *R2* = 0x00, then sensor acquisition is completed.
7.3	S[0xA8][0x84]S[0xA9][*ZL*]P, S[0xA8][0xFA][*ZL*]P	Read sensor values and store to input registers for calibration data processing.
7.4	S[0xA8][0x85]S[0xA9][*ZH*]P, S[0xA8][0xFB][*ZH*]P	
7.5	S[0xA8][0x86]S[0xA9][*YL*]P, S[0xA8][0xFC][*YL*]P	
7.6	S[0xA8][0x87]S[0xA9][*YH*]P, S[0xA8][0xFD][*YH*]P	
7.7	S[0xA8][0x88]S[0xA9][*XL*]P, S[0xA8][0xFE][*XL*]P	
7.8	S[0xA8][0x89]S[0xA9][*XH*]P, S[0xA8][0xFF][*XH*]P	
8.0	Get CIE camera readings	Assume readings are X_R, Y_R, Z_R.

Note: Red color alphabet indicates that data from memory location is written to HDJD-J822.

CALIBRATION: STEP 9 – 11

Step	I2C Data Sequence	Comments
9.1	S[0xA8][0x01][0x00]P	Turn on only green LEDs at 100% duty cycle.
9.2	S[0xA8][0x0B][0x00]P, S[0xA8][0x0C][0x00]P	
9.3	S[0xA8][0x0D][0xFF]P, S[0xA8][0x0E][0x0F]P	
9.4	S[0xA8][0x0F][0x00]P, S[0xA8][0x10][0x00]P	
9.5	S[0xA8][0x01][0x02]P	
10.1	S[0xA8][0x02][0x02]P	Acquire sensor values.
10.2	S[0xA8][0x02]S[0xA9][*R2*]P	If *R2* = 0x00, then sensor acquisition is completed.
10.3	S[0xA8][0x84]S[0xA9][*ZL*]P, S[0xA8][0xF4][*ZL*]P	Read sensor values and store to input registers for calibration data processing.
10.4	S[0xA8][0x85]S[0xA9][*ZH*]P, S[0xA8][0xF5][*ZH*]P	
10.5	S[0xA8][0x86]S[0xA9][*YL*]P, S[0xA8][0xF6][*YL*]P	
10.6	S[0xA8][0x87]S[0xA9][*YH*]P, S[0xA8][0xF7][*YH*]P	
10.7	S[0xA8][0x88]S[0xA9][*XL*]P, S[0xA8][0xF8][*XL*]P	
10.8	S[0xA8][0x89]S[0xA9][*XH*]P, S[0xA8][0xF9][*XH*]P	
11.0	Get CIE camera readings	Assume readings are X_G, Y_G, Z_G.

CALIBRATION: STEP 12 – 16

Step	I2C Data Sequence	Comments
12.1	S[0xA8][0x01][0x00]P	Turn on only blue LEDs at 100% duty cycle.
12.2	S[0xA8][0x0B][0xFF]P, S[0xA8][0x0C][0x0F]P	
12.3	S[0xA8][0x0D][0x00]P, S[0xA8][0x0E][0x00]P	
12.4	S[0xA8][0x0F][0x00]P, S[0xA8][0x10][0x00]P	
12.5	S[0xA8][0x01][0x02]P	
13.1	S[0xA8][0x02][0x02]P	Acquire sensor values.
13.2	S[0xA8][0x02]S[0xA9][*R2*]P	If *R2* = 0x00, then sensor acquisition is completed.
13.3	S[0xA8][0x84]S[0xA9][*ZL*]P, S[0xA8][0xEE][*ZL*]P	Read sensor values and store to input registers for calibration data processing.
13.4	S[0xA8][0x85]S[0xA9][*ZH*]P, S[0xA8][0xEF][*ZH*]P	
13.5	S[0xA8][0x86]S[0xA9][*YL*]P, S[0xA8][0xF0][*YL*]P	
13.6	S[0xA8][0x87]S[0xA9][*YH*]P, S[0xA8][0xF1][*YH*]P	
13.7	S[0xA8][0x88]S[0xA9][*XL*]P, S[0xA8][0xF2][*XL*]P	
13.8	S[0xA8][0x89]S[0xA9][*XH*]P, S[0xA8][0xF3][*XH*]P	
14.0	Get CIE camera readings	Assume readings are X_B, Y_B, Z_B.
15.0	S[0xA8][0x01][0x00]P	Turn off all LEDs.
16.0	Scaled and convert all values to hexadecimal.	

Gather all the camera measurements and tabulate in Table 1.

Red LED on	Green LED on	Blue LED on
X_R	X_G	X_B
Y_R	Y_G	Y_B
Z_R	Z_G	Z_B

Table 1. Tabulated camera measurements.

Find the maximum value in the table. Let's say, the maximum value is V_m

Determine the camera scale value. The camera scale value depends on the application. For multicolor application, set camera scale to 700. For white points only application, set camera scale to 1000.

The scale ratio is calculated using the equation below.

$$\text{Scale ratio} = \frac{\text{camera scale}}{\text{Vm}}$$

Multiply all the elements in the table by the scale ratio and convert all the values to hexadecimal values. You should get the camera scale value as the maximum value.
Table 2 shows the scaled and converted values as follows:

Red LED ON	Green LED ON	Blue LED ON
XR	*XG*	*XB*
YR	*YG*	*YB*
ZR	*ZG*	*ZB*

Table 2. Scaled and converted hexadecimal values.

CALIBRATION: STEP 17

Store the scaled and converted values in Table 2 to input registers at address 0xDC to 0xED for calibration data processing.

Step	I2C Data Sequence	Comments
17.1	S[0xA8][0xDC][*ZBL*]P, S[0xA8][0xDD][*ZBH*]P	Write *ZB: ZBL* (low byte) and *ZBH* (high byte).
17.2	S[0xA8][0xDE][*YBL*]P, S[0xA8][0xDF][*YBH*]P	Write *YB: YBL* (low byte) and *YBH* (high byte).
17.3	S[0xA8][0xE0][*XBL*]P, S[0xA8][0xE1][*XBH*]P	Write *XB: XBL* (low byte) and *XBH* (high byte).
17.4	S[0xA8][0xE2][*ZGL*]P, S[0xA8][0xE3][*ZGH*]P	Write *ZG: ZGL* (low byte) and *ZGH* (high byte).
17.5	S[0xA8][0xE4][*YGL*]P, S[0xA8][0xE5][*YGH*]P	Write *YG: YGL* (low byte) and *YGH* (high byte).
17.6	S[0xA8][0xE6][*XGL*]P, S[0xA8][0xE7][*XGH*]P	Write *XG: XGL* (low byte) and *XGH* (high byte).
17.7	S[0xA8][0xE8][*ZRL*]P, S[0xA8][0xE9][*ZRH*]P	Write *ZR: ZRL* (low byte) and *ZRH* (high byte).
17.8	S[0xA8][0xEA][*YRL*]P, S[0xA8][0xEB][*YRH*]P	Write *YR: YRL* (low byte) and *YRH* (high byte).
17.9	S[0xA8][0xEC][*XRL*]P, S[0xA8][0xED][*XRH*]P	Write *XR: XRL* (low byte) and *XRH* (high byte).

CALIBRATION: STEP 18 – 19

Step	I2C Data Sequence	Comments
18.1	S[0xA8][0x02][0x01]P	Start calibration data processing.
18.2	S[0xA8][0x02]S[0xA9][*R2*]P	Check *R2* value. If 0x00, calibration data processing is completed.
19.1	S[0xA8][0x8A]S[0xA9][*C0*]P	
19.2	S[0xA8][0x8B]S[0xA9][*C1*]P	
19.3	S[0xA8][0x8C]S[0xA9][*C2*]P	
19.4	S[0xA8][0x8D]S[0xA9][*C3*]P	
19.5	S[0xA8][0x8E]S[0xA9][*C4*]P	
19.6	S[0xA8][0x8F]S[0xA9][*C5*]P	
19.7	S[0xA8][0x90]S[0xA9][*C6*]P	
19.8	S[0xA8][0x91]S[0xA9][*C7*]P	
19.9	S[0xA8][0x92]S[0xA9][*C8*]P	
19.10	S[0xA8][0x93]S[0xA9][*C9*]P	
19.11	S[0xA8][0x94]S[0xA9][*C10*]P	
19.12	S[0xA8][0x95]S[0xA9][*C11*]P	
19.13	S[0xA8][0x96]S[0xA9][*C12*]P	
19.14	S[0xA8][0x97]S[0xA9][*C13*]P	
19.15	S[0xA8][0x98]S[0xA9][*C14*]P	
19.16	S[0xA8][0x99]S[0xA9][*C15*]P	Read calibration data and store to non-volatile memory.
19.17	S[0xA8][0x9A]S[0xA9][*C16*]P	
19.18	S[0xA8][0x9B]S[0xA9][*C17*]P	
19.19	S[0xA8][0x9C]S[0xA9][*C18*]P	
19.20	S[0xA8][0x9D]S[0xA9][*C19*]P	
19.21	S[0xA8][0x9E]S[0xA9][*C20*]P	
19.22	S[0xA8][0x9F]S[0xA9][*C21*]P	
19.23	S[0xA8][0xA0]S[0xA9][*C22*]P	
19.24	S[0xA8][0xA1]S[0xA9][*C23*]P	
19.25	S[0xA8][0xA2]S[0xA9][*C24*]P	
19.26	S[0xA8][0xA3]S[0xA9][*C25*]P	
19.27	S[0xA8][0xA4]S[0xA9][*C26*]P	
19.28	S[0xA8][0xA5]S[0xA9][*C27*]P	
19.29	S[0xA8][0xA6]S[0xA9][*C28*]P	
19.30	S[0xA8][0xA7]S[0xA9][*C29*]P	
19.31	S[0xA8][0xA8]S[0xA9][*C30*]P	

All calibration data have been read and stored to non-volatile memory. The calibration procedure is completed.

Normal Operating Mode

Figure 5 shows the normal operating procedure flow chart.

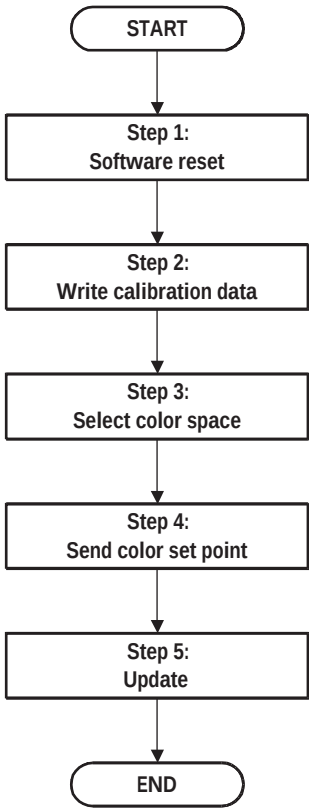

Figure 5. Normal operating flow chart.

NORMAL OPERATION: STEP 1 – 2

Step	I2C Data Sequence	Comments
1.1	S[0xA8][0x01][0x01]P	Software reset.
1.2	S[0xA8][0x09]S[0xA9][*ST*]P	Check *ST* = 0x02 to ensure reset is completed.
2.1	S[0xA8][0x8A][*C0*]P	
2.2	S[0xA8][0x8B][*C1*]P	
2.3	S[0xA8][0x8C][*C2*]P	
2.4	S[0xA8][0x8D][*C3*]P	
2.5	S[0xA8][0x8E][*C4*]P	
2.6	S[0xA8][0x8F][*C5*]P	
2.7	S[0xA8][0x90][*C6*]P	
2.8	S[0xA8][0x91][*C7*]P	
2.9	S[0xA8][0x92][*C8*]P	
2.10	S[0xA8][0x93][*C9*]P	
2.11	S[0xA8][0x94][*C10*]P	
2.12	S[0xA8][0x95][*C11*]P	
2.13	S[0xA8][0x96][*C12*]P	
2.14	S[0xA8][0x97][*C13*]P	
2.15	S[0xA8][0x98][*C14*]P	
2.16	S[0xA8][0x99][*C15*]P	Write calibration data to HDJD-J822.
2.17	S[0xA8][0x9A][*C16*]P	
2.18	S[0xA8][0x9B][*C17*]P	
2.19	S[0xA8][0x9C][*C18*]P	
2.20	S[0xA8][0x9D][*C19*]P	
2.21	S[0xA8][0x9E][*C20*]P	
2.22	S[0xA8][0x9F][*C21*]P	
2.23	S[0xA8][0xA0][*C22*]P	
2.24	S[0xA8][0xA1][*C23*]P	
2.25	S[0xA8][0xA2][*C24*]P	
2.26	S[0xA8][0xA3][*C25*]P	
2.27	S[0xA8][0xA4][*C26*]P	
2.28	S[0xA8][0xA5][*C27*]P	
2.29	S[0xA8][0xA6][*C28*]P	
2.30	S[0xA8][0xA7][*C29*]P	
2.31	S[0xA8][0xA8][*C30*]P	

Next I2C data sequence will be used to specify set point required.

NORMAL OPERATION: STEP 3 – 5

Step	I2C Data Sequence	Comments
3.	S[0xA8][0x04][*CS*]P	Select color space. Write *CS* value. Refer to Table 3.
4.1	S[0xA8][0xEC][*ALO*]P, S[0xA8][0xED][*AHI*]P	Write set point A with values. *AHI: ALO* Refer to Table 4.
4.2	S[0xA8][0xEA][*BLO*]P, S[0xA8][0xEB][*BHI*]P	Write set point B with values. *BHI: BLO* Refer to Table 4.
4.3	S[0xA8][0xE8][*CLO*]P, S[0xA8][0xE9][*CHI*]P	Write set point C with values. *CHI: CLO* Refer to Table 4.
5.	S[0xA8][0x01][0x12]P	Update set point.

The '*CS*' value depends on the color space used. Table 3 shows the possible selections.

'CS' value	Color Space selected
0x08	RGB Color space
0x04	Yu'v' color space
0x02	Yxy color space
0x01	XYZ color space

Table 3. Color space selection values.

The values *AHI*, *ALO*, *BHI*, BLO, *CHI*, *CLO* are written as per format in Table 4. These numbers in Table 4 are decimal numbers.

RGB Color Space	
Variables	**Number format**
AHI	0
ALO	R value from 0 to 255
BHI	0
BLO	G value from 0 to 255
CHI	0
CLO	B value from 0 to 255

Yu'v' Color Space		
Variables	**Number format**	**Comments**
AHI	Y value. Upper 2 bits	Y value range from 0 to 1000
ALO	Y value. Lower 8 bits	
BHI	Integer value (u' * 1000). Upper 2 bits	
BLO	Integer value (u' * 1000). Lower 8 bits	
CHI	Integer value (v' * 1000). Upper 2 bits	
CLO	Integer value (v' * 1000). Lower 8 bits	

Yxy Color Space		
Variables	**Number format**	**Comments**
AHI	Y value. Upper 2 bits	Y value range from 0 to 1000
ALO	Y value. Lower 8 bits	
BHI	Integer value (x * 1000). Upper 2 bits	
BLO	Integer value (x * 1000). Lower 8 bits	
CHI	Integer value (y * 1000). Upper 2 bits	
CLO	Integer value (y * 1000). Lower 8 bits	

XYZ Color Space		
Variables	**Number format**	**Comments**
AHI	X value. Upper 2 bits	X value range from 0 to 1000
ALO	X value. Lower 8 bits	
BHI	Y value. Upper 2 bits	Y value range from 0 to 1000
BLO	Y value. Lower 8 bits	
CHI	Z value. Upper 2 bits	Z value range from 0 to 1000
CLO	Z value. Lower 8 bits	

Table 4. Number format for RGB, Yu'v', Yxy and XYZ color space.

Examples:
1. To use RGB color space with set point R=0xFF, G=0x7F, B= 0x4F, the I2C data sequences are

Step	I2C Data Sequence	Comments
3.	S[0xA8][0x04][0x08]P	Select RGB color space
4.1	S[0xA8][0xEC][0xFF]P, S[0xA8][0xED][0x00]P	Send R value
4.2	S[0xA8][0xEA][0x7F]P, S[0xA8][0xEB][0x00]P	Send G value
4.3	S[0xA8][0xE8][0x4F]P, S[0xA8][0xE9][0x00]P	Send B value
5.	S[0xA8][0x01][0x12]P	Update new color set point

2. To specify D65 white point (x = 0.313, y = 0.329) with Y = 900,
   x value = 0.313 * 1000 = 313 (0x0139)

   y value = 0.329 * 1000 = 329 (0x0149)
   Y = 900 (0x0384)

Step	I2C Data Sequence	Comments
3.	S[0xA8][0x04][0x02]P	Select Yxy color space
4.1	S[0xA8][0xEC][0x84]P, S[0xA8][0xED][0x03]P	Send Y value
4.2	S[0xA8][0xEA][0x39]P, S[0xA8][0xEB][0x01]P	Send x value
4.3	S[0xA8][0xE8][0x49]P, S[0xA8][0xE9][0x01]P	Send y value
5.	S[0xA8][0x01][0x12]P	Update new color set point

Brightness Scalar

The color set point in steps 3 and 4 will select the color and the brightness. The brightness level chosen should be the level that is achievable for the system operating lifetime.

If the brightness scalar of the color is to be changed, the I2C data sequence is:

S[0xA8][0xC4][*BR*]P

BR represents the brightness scalar value. The default brightness scalar value is 255 or 0xFF at 100% brightness.

To set at 75% brightness, the brightness scalar value is 0.75 * 255 = 191 (0xBF). The I2C data sequence is:

S[0xA8][0xC4][0xBF]P

5989-1169EN July 5, 2006