E-paper Display COG Driver Interface Timing

Description	**Detailed information to design a timing controller for 1.44”, 2”, and 2.7” E-paper panels**
Date	**2012/07/27**
Doc. No.	**4P008-00**
Revision	**02**

	Design Engineering		
	Approval	**Check**	**Design**
	李 2012.07.27 欣達	李 2012.07.27 欣達	丁 2012.07.27 昭文

No.18, Shengli 1st St., Rende Dist., Tainan City 71758, Taiwan (R.O.C.)
Tel: +886-6-279-5399 Fax: +886-6-270-5857

Copyright

龍亭新技股份有限公司　Pervasive Displays Inc.

No.18, Shengli 1st St., Rende Dist., Tainan City 71758, Taiwan (R.O.C.)

Tel: +886-6-279-5399

http://www.pervasivedisplays.com

Table of Contents

Revision History

Version	Date	Page (New)	Section	Description
Ver. 01	2012/05/08	All	All	Approval specification first issued
Ver. 02	2012/07/27	6	1.1	Modify "Overview" description
		9	1.2	Add "Input Terminal Pin Assignment" section and description
		11	1.3	Add "Reference Circuit" section
		12	1.4	Modify Flash to memory in the flow chart
		13	1.5	Modify Controller and description
		16	1.6	Modify SCL to SCLK, SDI to SI in the sheet
		17	2	Modify the section name "Write to the Flash" to "Write to the Memory"
		17	2	Modify the description of section 2
		18	3	Modify "Border control" to "BORDER" Add PWM toggle before V_{CC}/V_{DD} turn on
		19	4	Modify the flow chart and description Modify the setting of register 0x06 from 0x1F to 0xFF
		21	5	Modify the section name "Write data from the flash to the EPD" to "Write data from the memory to the EPD"
		21	5.1	Modify the description of section 5
		22	5.1	Add 1.44 frame time for V110 FPL
		23	5.2	Add 1.44" and 2.7" flow chart
		27	5.3	Modify the flow chart and description

Glossary of Acronyms

EPD	Electrophoretic Display (e-Paper Display)
EPD Panel	EPD
TCon	Timing Controller
FPL	Front Plane Laminate (e-Paper Film)
SPI	Serial Peripheral Interface
COG	Chip on Glass
PDI, PDi	Pervasive Displays Incorporated

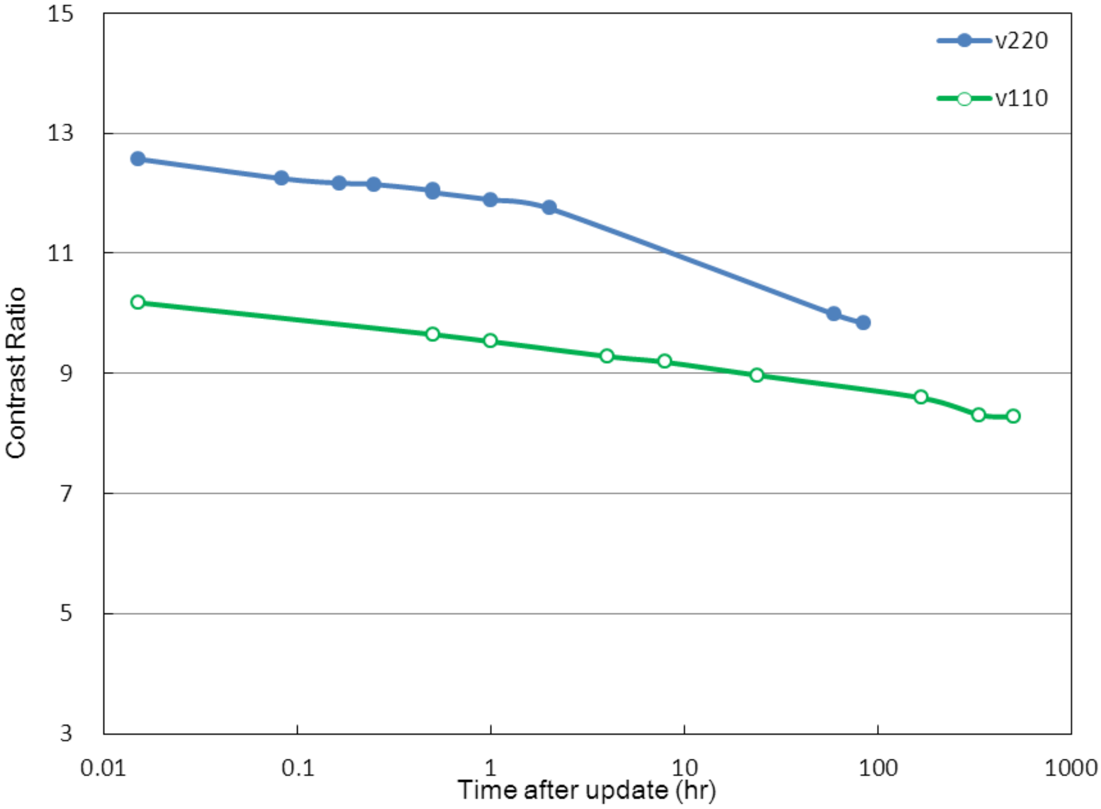
1 General Description

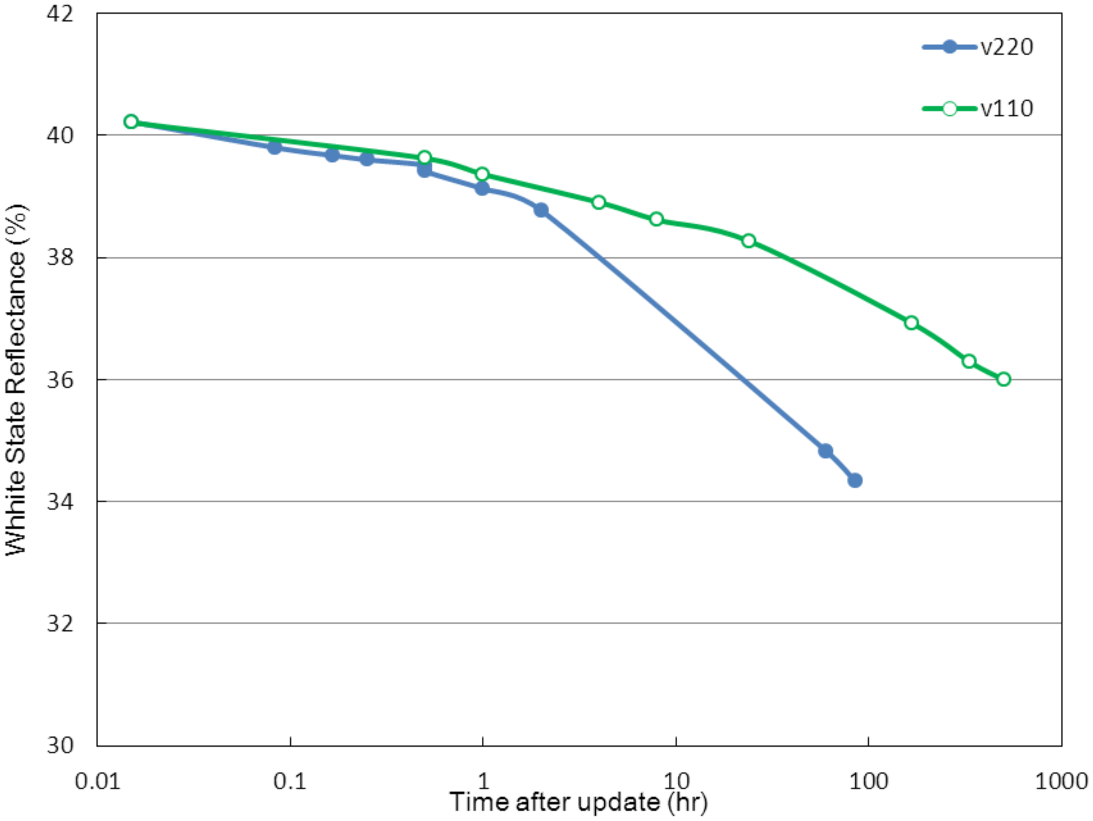
1.1 Overview

This document explains the interface to the COG Driver to operate the EPD for a MCU based solution using two pages of memory buffer. This document applies to 1.44”, 2.0”, and 2.7” EPDs.

Both new and previous display images are stored in memory buffer, then the COG Driver is powered on, initialized, panel updated in stages and then the COG Driver is powered off. Refer to the EPD controller in section 1.5 to see the complete update cycle from Power On, Initialize, Update and Power off. To operate the EPDs for the best sharpness and performance, each update of the panel is divided into a series of stages before the display of the new image pattern is completed. During each stage, frame updates with intermediate image patterns are repeated for a specified period of time. The number of repeated frame updates during each stage is dependent on the MCU speed. After the final stage, the new pattern is displayed.

V110 and V220 are names for two types of Front Plane Laminate which is the PET material that contains the microcapsules that is used to manufacture EPDs. V110 is the current generation of material, and V220 is the next generation of material. The materials are similar but have a few differences. V220 was designed to have a higher contrast ratio and is excellent for e-readers which are frequently updated. For V110 and V220, contrast ratio and white reflectance are compared in the charts below for reference.


15
13
11
9
7
5
3
Contrast Ratio
0.01
0.1
1
10
100
1000
Time after update (hr)
v220
v110


42
40
38
36
34
32
30
Whhite State Reflectance (%)
0.01
0.1
1
10
100
1000
Time after update (hr)
v220
v110

Around the active area of the EPD is a 0.5mm width blank area called the border. It should be connected to V_{DL} (-15V) to keep the border white. After approximately 10,000 updates with the constant voltage, the border color may degrade to a gray level that is not as white as the active area. To prevent this phenomenon, PDI recommends connecting BORDER as described in our documentation so that it can receive a control signal to turn on and off to avoid the degradation.

Section 1 is an overview and contains supporting information such as the overall theory for updating an EPD, SPI timing for PDI's EPDs, as well as current profiles.

Section 2 describes a method to write to memory buffer. Previously updated and new patterns are stored in the memory buffer to compare the old and new image patterns during the update.

Section 3 describes how to power on the COG Driver which consists of applying a voltage and generating the required signals for /CS and /RESET.

Section 4 describes the steps to initialize the COG Driver.

Section 5 describes the details on how to update the EPD from the memory buffer, create a line of data, update in stages, and also power down housekeeping steps.

1.2 Input Terminal Pin Assignment

No	Signal	I/O	Connected to	Function
1	/CS	I	MCU	Chip Select. Low enable
2	BUSY	O	MCU	When BUSY = HIGH, EPD stays in busy state that EPD ignores any input data from SPI.
3	ID	I	Ground	Set SPI interface
4	SCLK	I	MCU	Clock for SPI
5	SI	I	MCU	Serial input from host MCU to EPD
6	SO	O	MCU	Serial output from EPD to host MCU
7	/RESET	I	MCU	Reset signal. Low enable
8	ADC_IN	-	-	Not connected
9	V_{CL}	C	Capacitor	-
10	C42P	C	Charge-Pump Capacitor	-
11	C42M	C		-
12	C41P	C	Charge-Pump Capacitor	-
13	C41M	C		-
14	C31M	C	Charge-Pump Capacitor	-
15	C31P	C		-
16	C21M	C	Charge-Pump Capacitor	-
17	C21P	C		-
18	C16M	C	Charge-Pump Capacitor	-
19	C16P	C		-
20	C15M	C	Charge-Pump Capacitor	-
21	C15P	C		-
22	C14M	C	Charge-Pump Capacitor	-
23	C14P	C		-

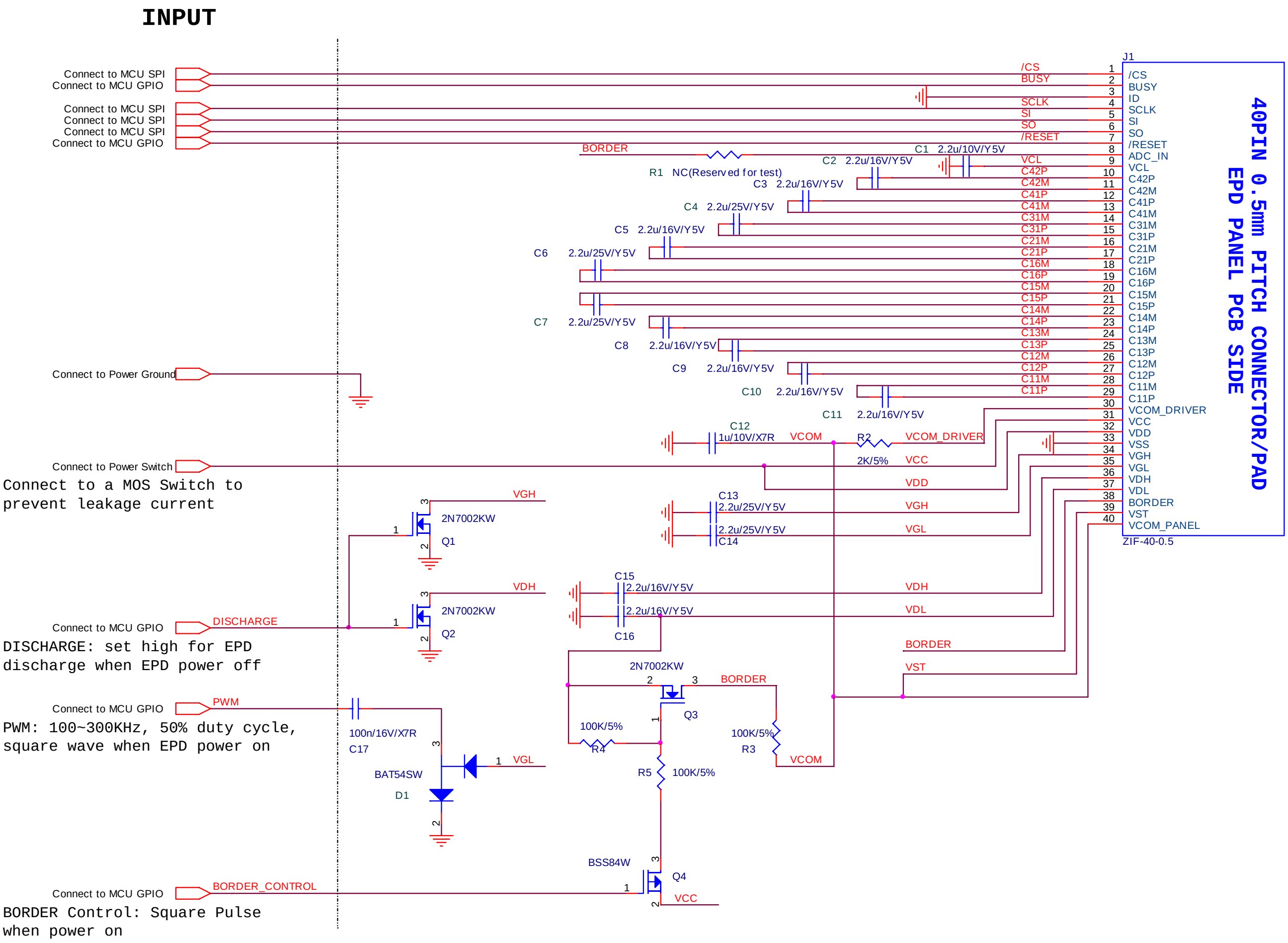
No	Signal	I/O	Connected to	Function
24	C13M	C	Charge-Pump Capacitor	-
25	C13P	C		-
26	C12M	C	Charge-Pump Capacitor	-
27	C12P	C		-
28	C11M	C	Charge-Pump Capacitor	-
29	C11P	C		-
30	V_{COM_DRIVER}	RC	Resistor & Capacitor	The signal duty cycle can drive V_{COM} voltage from source driver IC
31	V_{CC}	P	V_{CC}	Power supply for analog part of source driver
32	V_{DD}	P	V_{DD}	Power supply for digital part of source driver
33	V_{SS}	P	Ground	-
34	V_{GH}	C	Capacitor	-
35	V_{GL}	C	Capacitor	-
36	V_{DH}	C	Capacitor	-
37	V_{DL}	C	Capacitor	-
38	BORDER	I	-	Connect to V_{DL} via control circuit for white frame border
39	V_{ST}	P	V_{COM_PANEL}	-
40	V_{COM_PANEL}	C	Capacitor	V_{COM} to panel

Note:

I:	Input
O:	Output
C:	Capacitor
RC:	Resistor and Capacitor
P:	Power

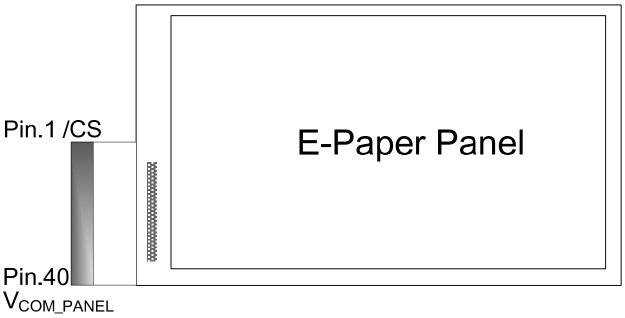
1.3

1.3 Reference Circuit



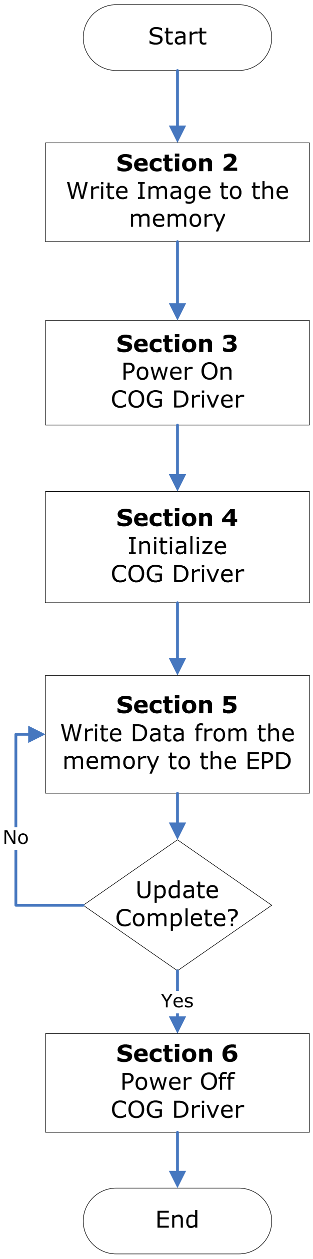
Note:

1. V_{DD} and V_{CC} must be discharged promptly after power off.
2. Pin.1 location



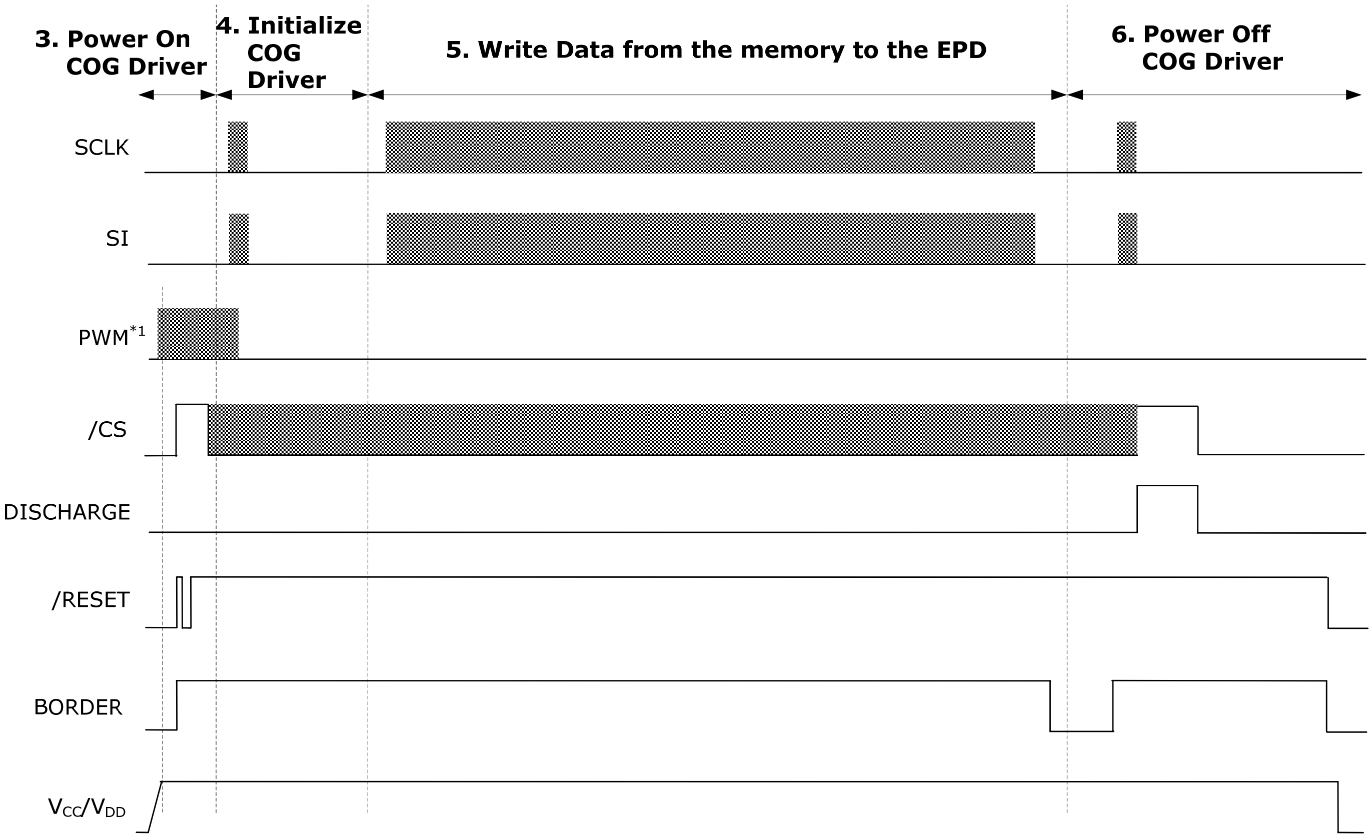
1.4 EPD Driving Flow Chart

The flowchart below provides an overview of the actions necessary to update the EPD. The steps below refer to the detailed descriptions in the respective sections.



1.5 Controller

The diagram below provides a signal control overview during an EPD update cycle. The diagram is segmented into “3. Power On COG Driver”, “4. Initialize COG Driver”, “5. Write data from the memory to the EPD”, and “6. Power Off COG Driver”. The segment number and title matches a section title in this document which contain the details for each segment.



Note:

1. PWM: 100~300 KHz Duty= 50% Square wave.

   The PWM signal starts before V_{CC}/V_{DD} input and stops during the initialization of the COG Driver to ensure there is a negative VGL on the COG Driver. Our reliability testing shows that with low temperature that the COG Driver has the possibility of V_{CC} generating a slightly positive voltage, and the PWM is an effective solution for this condition. Refer to the section 4 of this spec.

1.6 SPI Timing Format

SPI commands are used to communicate between the MCU and the COG Driver. The SPI format used differs from the standard in that two way communications are not used, and CS is pulled high then low between clocks. When setting up the SPI timing, PDI recommends verifying the control signals for the overall waveform in Section 1.5, next verify the SPI command format and SPI command timing both in this section.

The maximum clock speed that the display can accept is 12MHz. The minimum is 4MHz. The SPI mode is 0.

- Below is a description of the SPI Format:

SPI(0xI_1I_2, 0x$D_1D_2D_3D_4$, $D_5D_6D_7D_8$…)

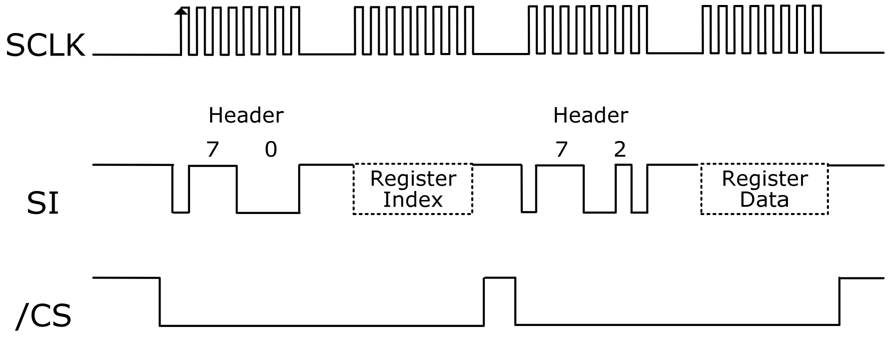
Where:

I_mI_n is the Register Index and the length is 1 byte
$D_{m\sim n}$ is the Register Data. The Register Data length varies from 1, 2, to 8 bytes depending on which Register Index is selected.

Register Index	Number Bytes of Register Data
0x01	8
0x02	1
0x03	1
0x04	1
0x05	1
0x06	1
0x07	1
0x08	1
0x09	2

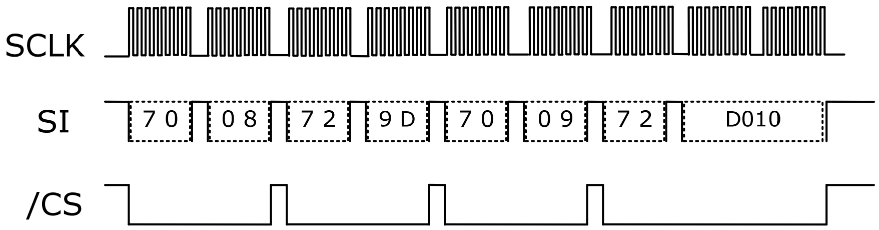
- Before sending the Register Index, the SPI (SI) must send a 0x**70** header command.
- Likewise, the SPI (SI) must send a 0x**72** is the header command prior to the Register Data. The flow chart and detailed description can be found on the next page.

- SPI command signals and flowchart:

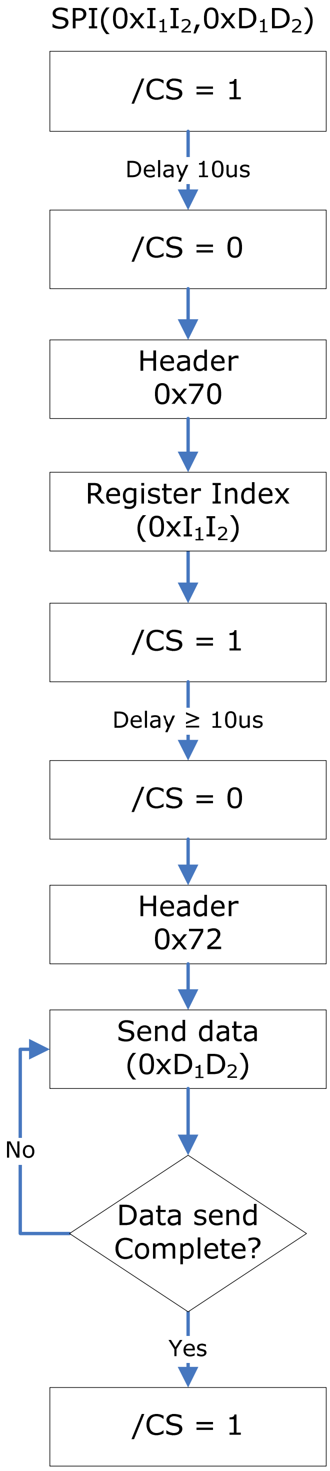


/CS must be set High then Low between Register Index and Register Data

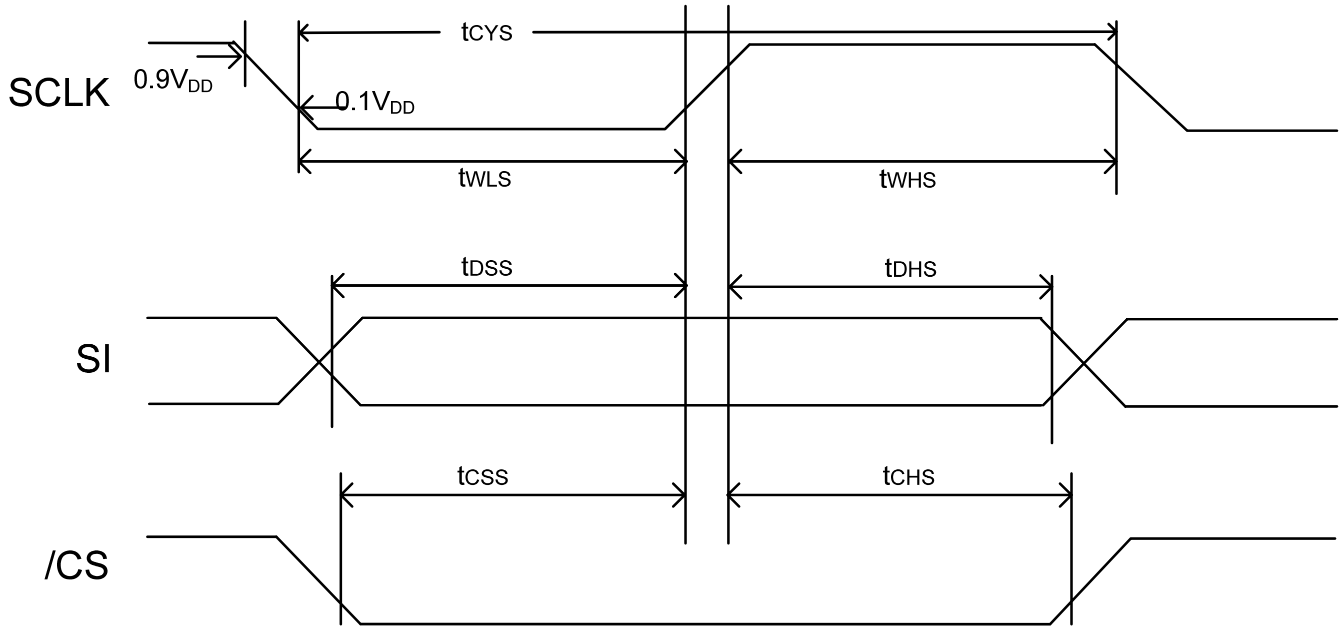
For example:
To send two SPI commands:
SPI((0x08,0x9D) and SPI(0x09, 0xD010)



If register data is larger than two bytes, you must input data continuously without setting Register Index again.



- SPI command timing



VCC = 2.7 to 3.3V **Temp = 0 to +50**℃

Item	Signal	Symbol	Min.	Typ.	Max.	Unit	Remark
Serial clock cycle	SCLK	tCYS	80	-	-	ns	
SCLK high pulse width	SCLK	tWHS	40	-	-	ns	
SCLK low pulse width	SCLK	tWLS	40	-	-	ns	
Data setup time	SI	tDSS	20	-	-	ns	
Data hold time	SI	tDHS	20	-	-	ns	
CSB setup time	/CS	tCSS	16	-	-	ns	
CSB hold time	/CS	tCHS	24	-	-	ns	

2 Write to the Memory

Before powering on COG Driver, the developer should write the new pattern to image buffer, either SRAM or flash memory. The image pattern must be converted to a 1 bit bitmap format (Black/White) in prior to writing.

Two buffer spaces should be allocated to store both previous and new patterns. The previous pattern is the currently displayed pattern. The new pattern will be written to the EPD. The COG Driver will compare both patterns before updating the EPD. The table below lists the buffer space size required for each EPD size.

EPD size	Image resolution(pixels)	Previous + new image Buffer (bytes)
1.44″	128 x 96	3,072
2"	200 x 96	4,800
2.7″	264 x 176	11,616

3 Power On COG Driver

This flowchart describes power sequence for the COG Driver.

1. Start :

   Initial State:

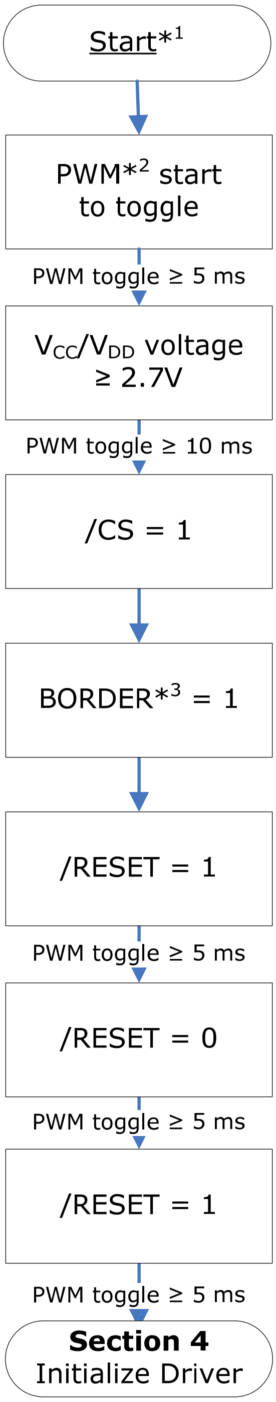
   V_{CC}/V_{DD}, /RESET, /CS, BORDER, SI, SCLK = 0

2. PWM:

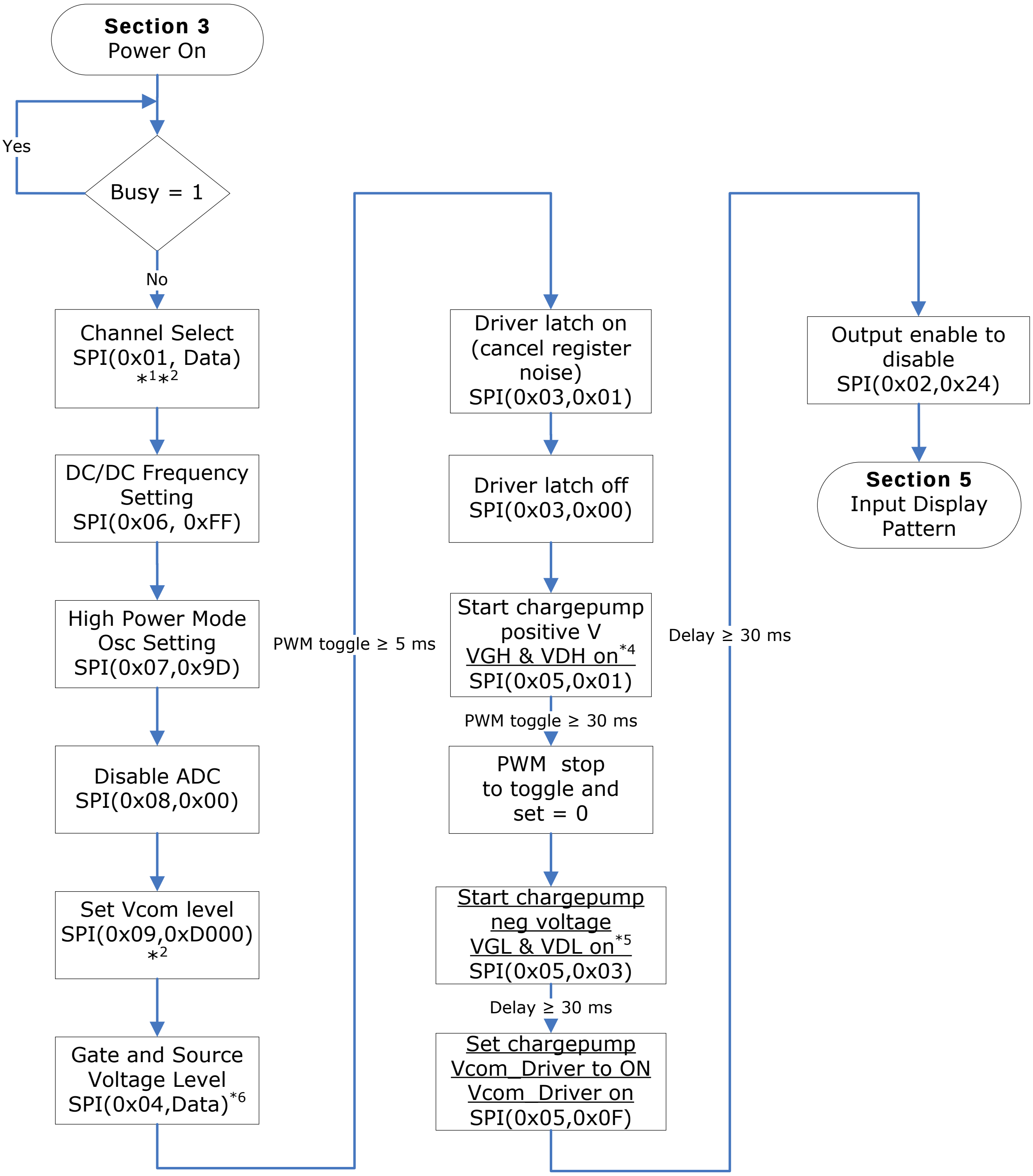
   100~300KHz, 50% duty cycle, square wave to eliminate the potential negative voltages that could occur at low temperature. Keeping PWM toggling until VGL & VDL is on (SPI(0x05,0x03)).

3. BORDER:

   For implement this function, Developer needs to use a pin from Microcontroller to control. BORDER is used to keep a sharp border while taking care of the electronic ink particles.



4 Initialize COG Driver



Note:

1. SPI(0x01, Data):
   - Different by each size
     - 1.44”: SPI(0x01, (0x0000,0000,000F,FF00))
     - 2”: SPI(0x01, (0x0000,0000,01FF,E000))
     - 2.7”: SPI(0x01, (0x0000,007F,FFFE,0000))
   - To send first byte protocol (0x70) before Register Index (0x01), and then send second byte protocol (0x72) before Register Data (0x0000,0000,01FF,E000).

2. If register data is larger than two bytes, the developer must finish sending the data prior to sending another Register Index command.
3. PWM: 100~300KHz, 50% duty cycle, square wave to eliminate the potential negative voltages that could occur at low temperature.
4. Should measure VGH >12V and VDH >8V
5. Should measure VGL <-12V and VDL <-8V
6. Gate and Source Voltage Level is different by each size:
   - Different by each size
     - 1.44”: SPI(0x04,0x03)
     - 2”: SPI(0x04,0x03)
     - 2.7”: SPI(0x04,0x00)

5 Write data from the memory to the EPD

This section describes how data should be sent to the COG Driver which will update the display. The COG Driver uses a buffer to store a line of data and then writes to the display.

5.1 Data Structure

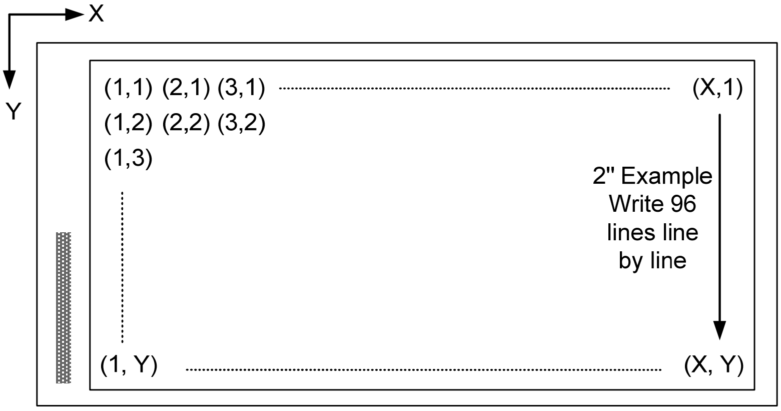
- EPD Resolutions

EPD size	Image resolution(pixels)	X	Y
1.44”	128 x 96	128	96
2"	200 x 96	200	96
2.7”	264 x 176	264	176

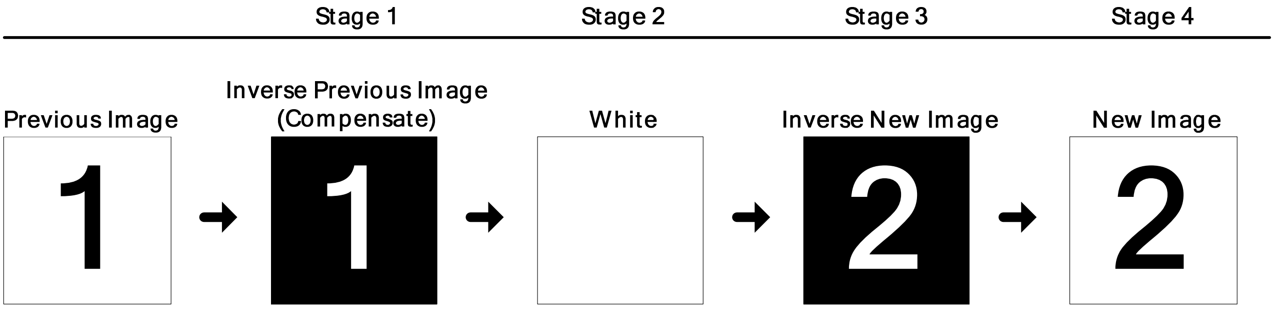
- Data components
  - One Bit – A bit can be W (White), B (Black) or N (Nothing) bits. Using the N bit mitigates ghosting.
  - One Dot/pixel is comprised of 2 bits.
  - One line is the number of dots in a line. For example:
    - The 1.44” uses 128 Dots to represent 1 Line.
    - The 2” uses 200 Dots to represent 1 Line.
    - The 2.7” uses 264 Dots to represent 1 Line.
    - The COG Driver uses a buffer to write one line of data (FIFO) - interlaced

Data Bytes	Scan bytes	Data Bytes
1st – 25th (Even)	1st - 24th	26th – 50th (Odd)
2” Example: Because method to write is interlaced, write the even data bytes for a line {D(200,y),D(198,y), D(196,y), D(194,y)} …. {D(8,y),D(6,y), D(4,y),	2” Example: Write bytes for every scan line {S(1),S(2), S(3), S(4)}…. {S(93),S(94), S(95), S(96)}	2” Example: Write the odd data bytes for a line D(4,y), D(2,y)}{D(1,y),D(3,y), D(5,y), D(7,y)}……… {D(193,y),D(195,y), D(197,y), D(199,y)}

  - One frame of data is the number of lines * rows. For example:
    - The 1.44” frame of data is 96 lines * 128 dots.
    - The 2” frame of data is 96 lines * 200 dots.
    - The 2.7” frame of data is 176 lines * 264 dots.



- One stage is the number of frames used to write an intermediate pattern. This can vary based on the MCU choice. PDI's design writes 16 frames of data per stage, and then 4 stages for 2″ and 1.44″ to update the display from the previous to the new pattern. 2.7″ need 21 frames of data per stage.



Panel Size	FPL	Stage Time (ms)	MCU Frame Time (ms)(Recommend)
1.44″	V110	480	< 50ms
1.44″	V220	480	
2″	V110	480	
2″	V220	480	
2.7″	V110	630	< 70ms
2.7″	V220	630	

5.2 Store a line of data in the buffer

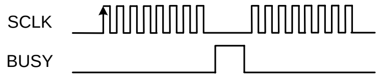
This section describes the details of how to send data to the COG Driver. The COG Driver uses a buffer to update the display line by line.

• 1.44” Input Data Order

Note :

1. When start transfer each Data Byte, users need to check BUSY pin.

Example :



2. If users cannot check BUSY pin, use delay at least 1 usec (10^{-6} second) Between byte-byte data for transfer image data.

Data	bit1	bit0	Input

D(x,y)
x = 1~128
y = 1~96
	1	1	Black (B)

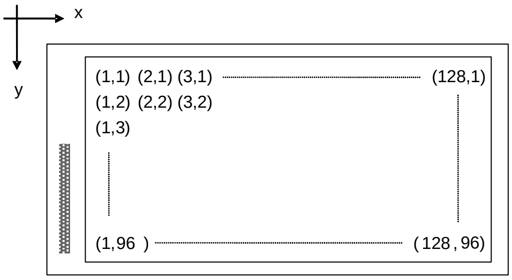
	1	0	White (W)
	0	1	Nothing (N)

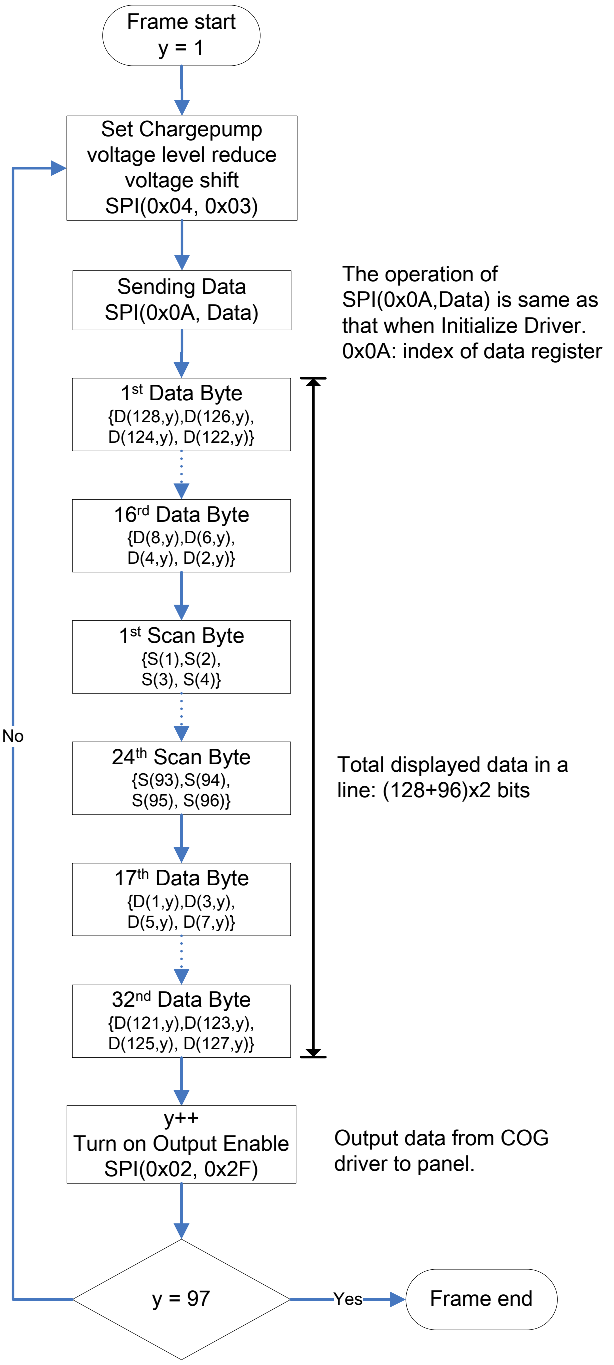
Example:
D(128,y) = Black (B) = 11
D(126,y) = White (W)= 10
D(124,y) = Nothing (N) = 01
D(122,y) = Black (B) = 11
→ 1st Data Byte= 11,10,01,11

Scan	bit1	bit0	Input
S(1) ~S(96)	1	1	Scan on
	0	0	Scan off

Example:
When y = 2,
→ Only S(2) is Scan on (11) while others are Scan off (00). The image represented by Data Bytes will be displayed on 2nd horizontal line (i.e. Dot(1,2) ~ Dot(128,2)).

S(1) = Scan off = 00
S(2) = Scan on = 11
S(3) = Scan off = 00
S(4) = Scan off = 00
:
S(96) = Scan off = 00
→ 1st Scan Byte = 00,11,00,00
→ 2nd ~ 24th Scan Byte = 00,00,00,00




Frame start
y = 1
Set Chargepump voltage level reduce voltage shift SPI(0x04, 0x03)
Sending Data SPI(0x0A, Data)
1st Data Byte {D(128,y),D(126,y), D(124,y), D(122,y)}
16rd Data Byte {D(8,y),D(6,y), D(4,y), D(2,y)}
1st Scan Byte {S(1),S(2), S(3), S(4)}
24th Scan Byte {S(93),S(94), S(95), S(96)}
17th Data Byte {D(1,y),D(3,y), D(5,y), D(7,y)}
32nd Data Byte {D(121,y),D(123,y), D(125,y), D(127,y)}
y++
Turn on Output Enable SPI(0x02, 0x2F)
y = 97
No
Yes
Frame end

The operation of SPI(0x0A,Data) is same as that when Initialize Driver. 0x0A: index of data register

Total displayed data in a line: (128+96)x2 bits

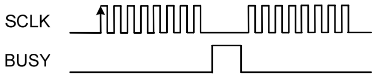
Output data from COG driver to panel.

• 2" Input Data Order

Note :

1. When start transfer each Data Byte, users need to check BUSY pin.

Example :



2. If users cannot check BUSY pin, use delay at least 1 usec (10^{-6} second) Between byte-byte data for transfer image data.

Data	bit1	bit0	Input

D(x,y)
x = 1~200
y = 1~96
	1	1	Black (B)

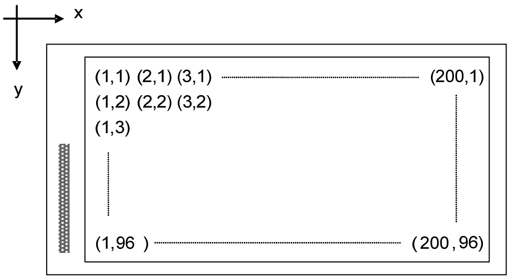
	1	0	White (W)
	0	1	Nothing (N)

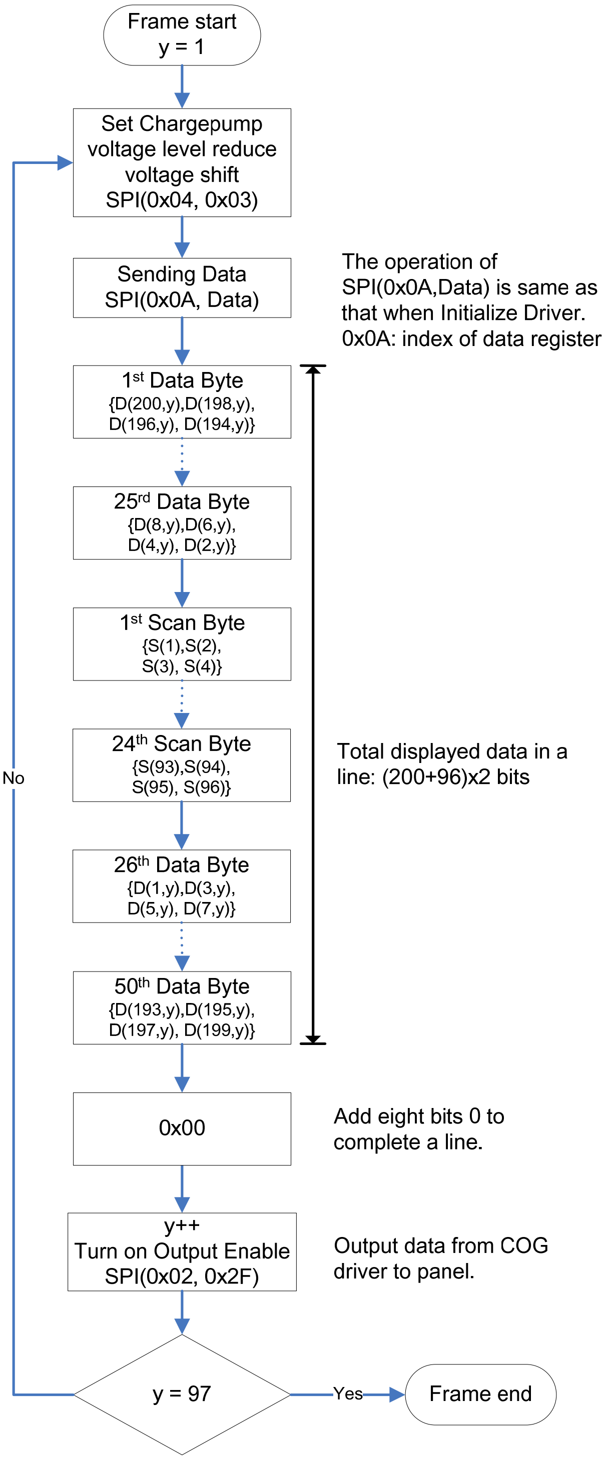
Example:
D(200,y) = Black (B) = 11
D(198,y) = White (W)= 10
D(196,y) = Nothing(N) = 01
D(194,y) = Black (B) = 11
→ 1st Data Byte= 11,10,01,11

Scan	bit1	bit0	Input
S(1) ~S(96)	1	1	Scan on
	0	0	Scan off

Example:
When y = 2,
→ Only S(2) is Scan on (11) while others are Scan off (00). The image represented by Data Bytes will be displayed on 2nd horizontal line (i.e. Dot(1,2) ~ Dot(200,2)).

S(1) = Scan off = 00
S(2) = Scan on = 11
S(3) = Scan off = 00
S(4) = Scan off = 00
:
S(96) = Scan off = 00
→ 1st Scan Byte = 00,11,00,00
→ 2nd ~ 24th Scan Byte = 00,00,00,00





The operation of SPI(0x0A,Data) is same as that when Initialize Driver. 0x0A: index of data register

Total displayed data in a line: (200+96)x2 bits

Add eight bits 0 to complete a line.

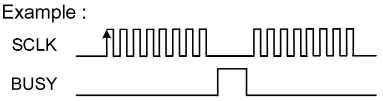
Output data from COG driver to panel.

• 2.7” Input Data Order

Note :

1. When start transfer each Data Byte, users need to check BUSY pin.

Example :



2. If users cannot check BUSY pin, use delay at least 1 usec (10^{-6} second) Between byte-byte data for transfer image data.

Data	bit1	bit0	Input

D(x,y)
x = 1~264
y = 1~176
	1	1	Black (B)

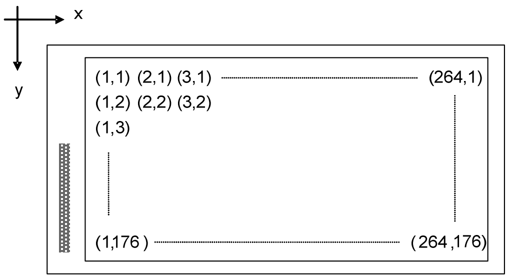
	1	0	White (W)
	0	1	Nothing (N)

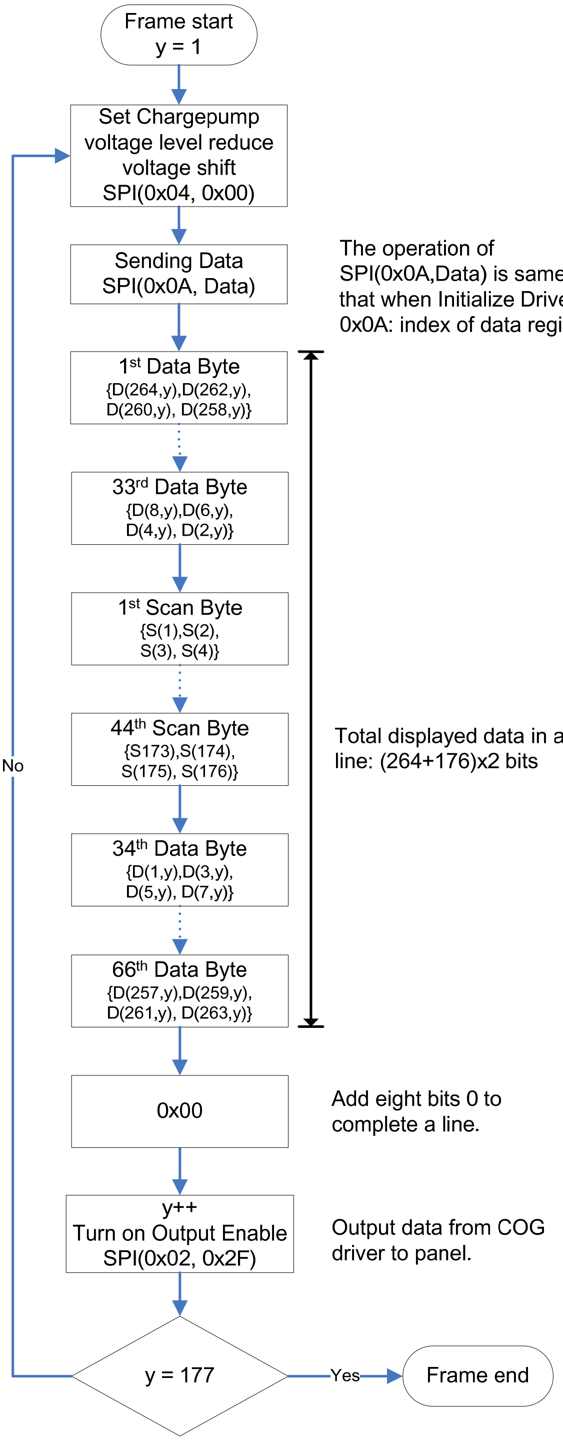
Example:
D(264,y) = Black (B) = 11
D(262,y) = White (W)= 10
D(260,y) = Nothing(N) = 01
D(258,y) = Black (B) = 11
→ 1st Data Byte= 11,10,01,11

Scan	bit1	bit0	Input
S(1) ~S(176)	1	1	Scan on
	0	0	Scan off

Example:
When y = 2,
→ Only S(2) is Scan on (11) while others are Scan off (00). The image represented by Data Bytes will be displayed on 2nd horizontal line (i.e. Dot(1,2) ~ Dot(264,2)).

S(1) = Scan off = 00
S(2) = Scan on = 11
S(3) = Scan off = 00
S(4) = Scan off = 00
:
S(176) = Scan off = 00
→ 1st Scan Byte = 00,11,00,00
→ 2nd ~ 44th Scan Byte = 00,00,00,00





The operation of SPI(0x0A,Data) is same as that when Initialize Driver.
0x0A: index of data register

Total displayed data in a line: (264+176)x2 bits

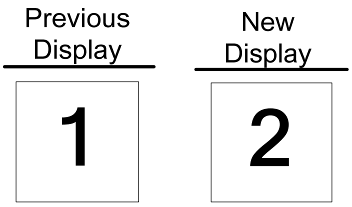
Add eight bits 0 to complete a line.

Output data from COG driver to panel.

5.3 Writing to the display in stages

This section contains the method to write to the display in stages. Each of the 4 stages should be the same use the same number of frames. Rewrite the frame during each stage.

The flow chart that follows describes how to update an image from a previous displayed image stored in memory buffer to a new image also stored in memory buffer. See the sample previous and new images below.



Temperature (℃)	TF[*7]
	V110/V220
≤-10	17
-5 ≥ T > -10	12
5 ≥ T > -5	8
10 ≥ T > 5	4
15 ≥ T > 10	3
20 ≥ T > 15	2
40 ≥ T > 20	1
> 40	0.7

1. The **previous** image stored in memory is used to determine how to write the data for both Stage 1 and Stage 2.

2. The **new** image stored in memory is used to determine how to write the data for both Stage 3 and Stage 4.

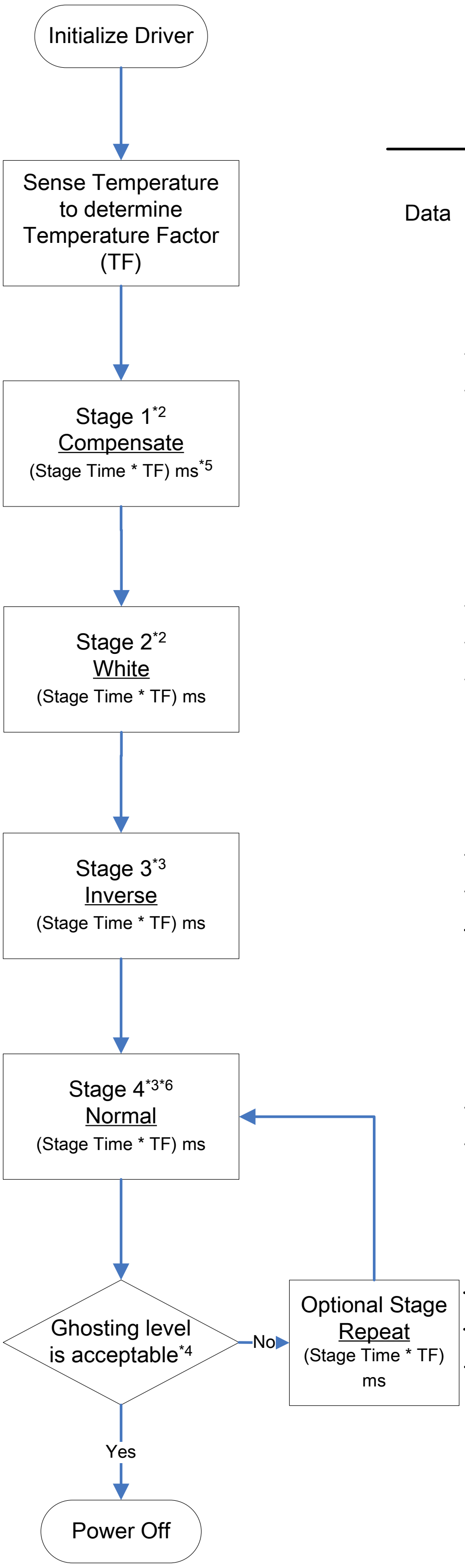
3. Optional: The optical performance is dependent on Stage Time. If the ghosting is at unacceptable level, the EPD can be rewritten and then Stage 4 repeated to write the New image.

4. Panel Size

Panel Size	(Stage Time * TF) ms
1.44"(V110)	480
1.44"(V220)	480
2"(V110)	480
2"(V220)	480
2.7"(V110)	630
2.7"(V220)	630

5. If you use Flash memory for the Section 2, please erase the buffer When Stage 4 is completed.

6. The TF below 0℃ is for reference only. PDI does not guarantee the performance and functionality below 0℃.


Initialize Driver
Sense Temperature to determine Temperature Factor (TF)
Stage 1*2 Compensate (Stage Time * TF) ms*5
Stage 2*2 White (Stage Time * TF) ms
Stage 3*3 Inverse (Stage Time * TF) ms
Stage 4*3*6 Normal (Stage Time * TF) ms
Ghosting level is acceptable*4
No
Optional Stage Repeat (Stage Time * TF) ms
Yes
Power Off

	bit1	bit0	Input	Previous Display
Data	1	1	Black (B)	1
	1	0	White (W)	
	0	0	Nothing (N)	

Stage 1	Data		Display
Previous[*2]	B	W	1
Input	W	B	
Display	W	B	

Stage 2	Data		Display
Previous[*2]	B	W	White
Input	N	W	
Display	W	W	

Stage 3	Data		Display
New[*3]	B	W	2
Input	N	B	
Display	W	B	

Stage 4	Data		New Display
New[*3]	B	W	2
Input	B	W	
Display	B	W	

Stage R	Data		Display
New[*3]	B	W	2
Input	W	B	
Display	W	B	

6 Power off COG Driver

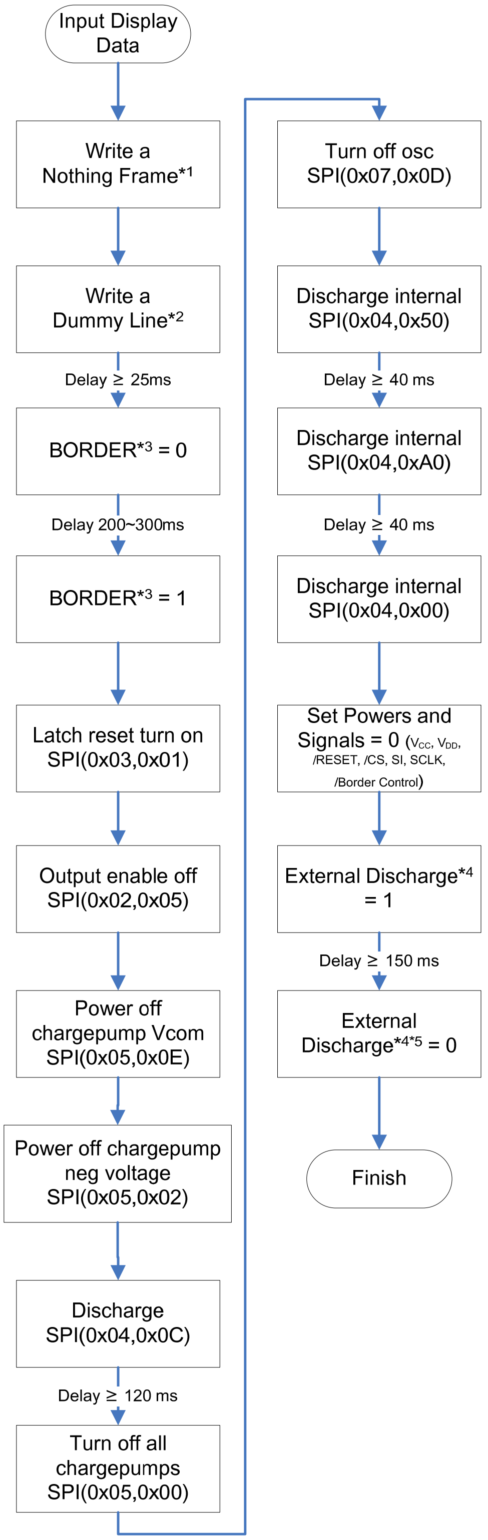
1. Nothing Frame :
   A frame, 96 lines/1.44"&2", 176 lines/2.7", whose all D(x,y) are N(01). Scan Bytes operate normally. Scan lines are still turned on sequentially. This frame will make the image more uniform.

2. Dummy Line :
   A line whose all Data Bytes are 0x55 and Scan Bytes are 0x00. Clear the register data before power off.

3. BORDER :
   For implement this function, users need to use a pin to control from Microcontroller. When = 0, the BORDER is ON and write to white. When = 1,the BORDER is OFF. The reason for using BORDER is to keep a sharp border and not have a charge on the Eink particles . Voltage too long on these will produce a gray effect which is the optimal for long term operation. BORDER is needed in V220 FPL and V110 FPL.

4. External Discharge :
   For implement this function, users need to use a pin from Microcontroller to control. This is important to avoid vertical lines.

5.If you use the Flash memory for pattern store, please recheck flash in this phase and verify the old image flash is erased.


Input Display Data
Write a Nothing Frame*1
Write a Dummy Line*2
Delay ≥ 25ms
BORDER*3 = 0
Delay 200~300ms
BORDER*3 = 1
Latch reset turn on SPI(0x03,0x01)
Output enable off SPI(0x02,0x05)
Power off chargepump Vcom SPI(0x05,0x0E)
Power off chargepump neg voltage SPI(0x05,0x02)
Discharge SPI(0x04,0x0C)
Delay ≥ 120 ms
Turn off all chargepumps SPI(0x05,0x00)
Turn off osc SPI(0x07,0x0D)
Discharge internal SPI(0x04,0x50)
Delay ≥ 40 ms
Discharge internal SPI(0x04,0xA0)
Delay ≥ 40 ms
Discharge internal SPI(0x04,0x00)
Set Powers and Signals = 0 (VCC, VDD, /RESET, /CS, SI, SCLK, /Border Control)
External Discharge*4 = 1
Delay ≥ 150 ms
External Discharge*4*5 = 0
Finish