AN11494

Cascading NXP LCD segment drivers

Rev. 1 — 12 February 2014 Application note

Document information

Info	Content
Keywords	PCF8576C, PCA8576C, PCF8576D, PCA8576D, PCA8576F, PCF8532, PCF8533, PCA8533, PCF8534, PCA8534, PCF8562, PCF85132, PCA85132, PCF85133, PCA85133, PCF85134, PCA85134, PCF85162, PCA85162, PCF85176, PCA85176, PCA85232, PCA85233, PCA85262, PCA85276, LCD driver, segment driver, cascading, COG, Chip-On-Glass, Display
Abstract	This application note aims to assist designers using NXP LCD segment drivers in making a cascaded design. Cascading means combining more than one LCD driver in a design in such a way that makes them appear to the rest of the application – especially to the microcontroller and the display - as one larger LCD driver capable of driving a larger display than the individual LCD drivers are able to. Depending on which devices are cascaded, cascading can be easy and straight forward on the one hand, or need some attention to details on the other hand in order to achieve a properly working set-up.

Revision history

Rev	Date	Description
1	20140212	Initial release

Contact information

For additional information, please visit: http://www.nxp.com

For sales office addresses, please send an email to: salesaddresses@nxp.com

1. Introduction

The NXP LCD driver portfolio consists of three families: segment drivers, character drivers and graphic drivers (dot matrix). This application note deals with segment drivers only, because cascading as meant in this application note is only possible with the segment drivers. However, not all segment drivers offer this feature. There is no principal reason why cascading could not be implemented in character drivers or dot matrix drivers. They are all based on the same principle. A liquid crystal cell is driven by two voltages, a backplane and a segment voltage (for segment drivers) respectively a row and column voltage (for character and dot matrix drivers). The voltage experienced by the liquid crystal cell is the difference between the two voltages. In order to avoid a DC component across the cell and in order to use multiplexed addressing, the voltages supplied to the liquid crystal cell vary over time, looking like staggered waveforms.

In the context used here, cascading means combining more than one LCD driver in a design in such a way that makes all of them appear to the rest of the application as one larger LCD driver, capable of driving a larger display than the individual LCD drivers are able to. Cascading is normally fairly easy and straight forward, but some combinations of LCD drivers need a more sophisticated approach in order to achieve a properly working set-up. This application note deals with these subjects.

The most simple cascade consists of one master and one or several slaves. They share the same I^2C-address and therefore appear as one LCD driver to the microcontroller. Inside the cascade the individual drivers are differentiated by using a 2-bit or 3-bit hardware subaddress (A0, A1 and A2). These hardware subaddresses can be set by connecting the respective pins (A0, A1 and A2) to either V_{SS} or V_{DD} to make them either LOW (logic 0) or HIGH (logic 1). Thus up to eight segment drivers can be used and differentiated using the same I^2C-address. For very large displays it may be useful to cluster two of these simple cascades to a joint cascade with two masters having one or several slaves each. This requires two I^2C-addresses which can be configured by using the SA0 pin. The SA0 pin allows to set the LSB of the I^2C-address to logic 0 or logic 1. In combination with the hardware subaddressing this means that – depending on the type – up to sixteen segment drivers can be differentiated on the same I^2C-bus.

Some segment drivers have less than three pins for the hardware subaddressing. The number of possible drivers on one bus is accordingly reduced.

Segment drivers connected in a cascade are synchronized by the $\overline{\text{SYNC}}$ signal to allow the backplane signals from only one device in the cascade to be shared. This arrangement is cost-effective in large LCD applications since the backplane outputs of only one device need to be through-plated to the backplane electrodes of the display. The other cascaded segment drivers contribute additional segment outputs. Their common outputs (backplane outputs) can either be connected together to enhance the drive capability or they can be left open-circuit.

2. Overview of NXP segment drivers

This section gives an overview of a part of the NXP segment driver portfolio for which this application note is valid, along with their I^2C subaddress and size of the integrated Display RAM (DRAM). Each bit in the DRAM corresponds to one segment to be displayed. In some cases the available DRAM is larger than the number of segments that

can be driven. In such cases this superfluous part is unused but it must be taken into account when writing the software for the application. This will be described later.

Table 1 shows an overview of these segment drivers with the relevant data for cascading. For the sake of completeness, some other smaller drivers have been included in the table. It shows that not all segment drivers can be cascaded, most however can. Also not all cascadable segment drivers can be combined with each other. In order to make a simple and straight forward cascade, it is necessary to know the determining parameters.

Table 1. Overview of segment drivers

Overview of segment drivers with relevant data for cascading. Where PCx is written, both PCF and PCA are meant.

Type number	Multiplex rate versus number of segments				I^2C slave address(es)	Cascad-able	Compa-tible with others	Internal display RAM size	Command length	Sub address pins
	1:1	1:2	1:3	1:4						
PCF2111C	-	64	-	-	N.A., CBUS	no	N.A.	64 bit	N.A.	-
PCF2112C	32	-	-	-	N.A., CBUS	no	N.A.	32 bit	N.A.	-
OM4068	32	64	96	-	N.A., SPI	yes	no	96 bit	N.A.	-
PCF8577C	32	64	-	-	74h	yes	no	64 bit	1 byte	A0, A1, A2; pins shared
PCF8566	24	48	72	96	7Ch and 7Eh	yes	yes	24 x 4 = 96 bit	1 byte	A0, A1, A2
PCF8562 PCx85162 PCA85262	32	64	96	128	70h and 72h	yes	yes	40 x 4 = 160 bit	1 byte	A0, A1, A2
PCx8576C	40	80	120	160	70h and 72h	yes	yes	40 x 4 = 160 bit	1 byte	A0, A1, A2
PCx8576D PCA8576F PCx85176 PCA85276	40	80	120	160	70h and 72h	yes	yes	40 x 4 = 160 bit	1 byte	A0, A1, A2
PCx8534A PCx85134	60	120	180	240	70h and 72h	yes	yes	80 x 4 = 320 bit	2 bytes	A0, A1, A2
PCx8533 PCx85133 PCA85233	80	160	240	320	70h and 72h	yes	yes	80 x 4 = 320 bit	2 bytes	A0, A1, A2
PCF8532 PCx85132 PCA85232	160	320	480	640	70h and 72h	yes	yes	160 x 4 = 640 bit	2 bytes	A0, A1

The command structure (1 byte or 2 bytes) is not the same for all drivers in this table. Combining drivers with different command structure in one cascade is possible, but has software implications. This is described in section 6 ”Combining different product types in one cascade”.

As the portfolio of NXP segment drivers gets continuously expanded, this table may not include all segment drivers that can be cascaded with other devices mentioned here. In addition, NXP carries several segment drivers which can’t be cascaded with the devices in this AN.

3. What is cascading

In electronics, a general description of cascading could be a series of components or networks of which the output of each serves as the input for the next. In the context used here, cascading means combining more than one LCD driver in a design in such a way that makes all of them appear to the rest of the application – especially to the microcontroller and the display – as one large LCD driver, capable of driving a larger display than the individual LCD drivers would be able to. Here it is the clock signal that is output from device N in the cascade to device N+1, whilst backplane signals are shared. Large display configurations of up to sixteen drivers can be recognized on the same I^2C-bus by using the 3-bit hardware subaddress (A0, A1 and A2, in some cases only A0 and A1 are present) and the programmable I^2C-bus slave address (SA0).

For most of the NXP segment drivers two I^2C slave addresses (0111 000 and 0111 001) are reserved. This is illustrated in Table 2.

Table 2. I^2C slave address byte

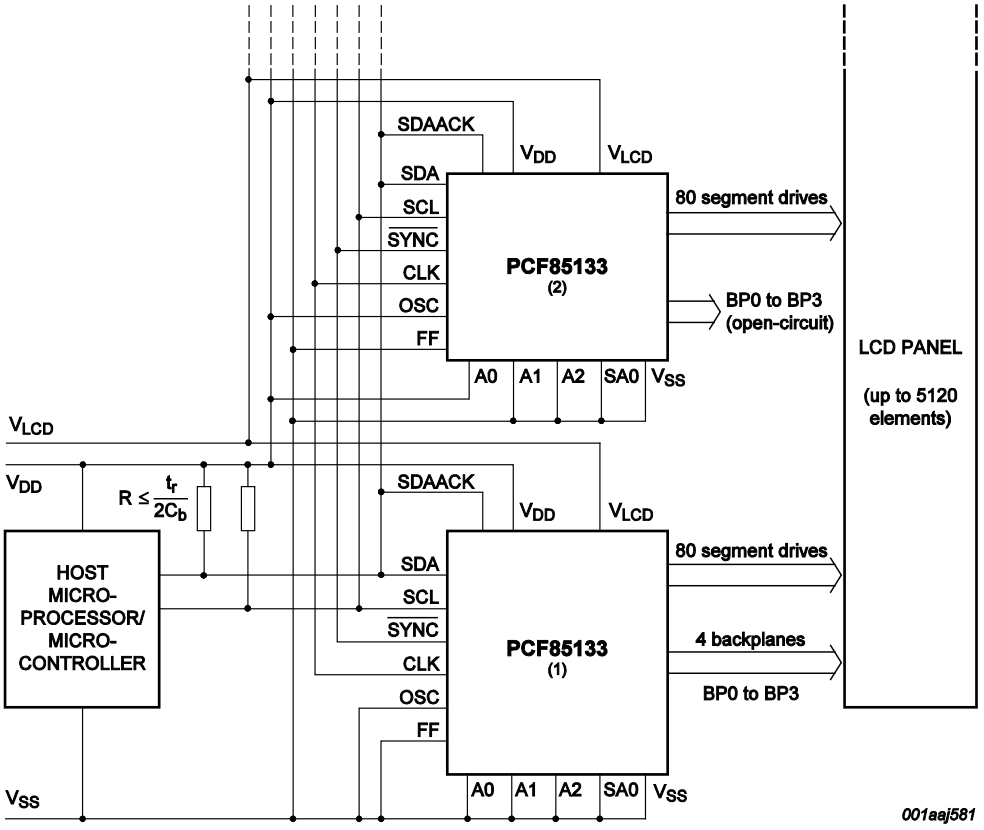
	Slave address							
Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	MSB							**LSB**
	0	1	1	1	0	0	SA0	R/W

The least significant bit of the slave address is bit R / $\overline{W}$. The drivers covered by this application note are all write-only devices and thus they will not respond to a read access. Therefore the LSB should always be logic 0. The second bit of the slave address byte is defined by the level tied at input SA0. Having two reserved slave addresses allows the following on the same I^2C-bus:

- Up to 16 (8 hardware subaddresses) or up to 8 (4 hardware subaddresses) drivers for large LCD applications, using two clusters in a joint cascade
- The use of two types of LCD multiplex drive modes, even on the same display. For example, one driver could drive small segments in mux 1:4, while another driver drives large segments in static mode. Note that this is not cascading.

Cascaded drivers must be synchronized using the $\overline{SYNC}$ pins. They can share the backplane signals from one of the devices in the cascade. Such an arrangement is cost-effective in large LCD chip-on-glass applications since the backplane outputs of only one device need to be through-plated to the backplane electrodes of the display. The other cascaded segment drivers contribute additional segment outputs. Their common outputs

(backplane outputs) can either be connected together with those of other drivers to enhance the drive capability or they can be left open-circuit. Since the signals are synchronized, it is also possible to use some common outputs from the master and some from a slave in order to come to a more optimized or easier layout. An example is shown in Fig 1.



(1) Is master (OSC connected to V_{SS}).

(2) Is slave (OSC connected to V_{DD}).

Fig 1. Example of cascaded configuration with two LCD drivers

The lower device in Fig 1 is the master. By connecting pin OSC to V_{SS}, the internal oscillator is enabled. On pin CLK the clock signal (from the internal oscillator) is available to be used as input for slave devices in the cascade. Of course it is also possible to use an external signal, in that case pin OSC of the master must be connected to V_{DD}.

The upper device is the slave. Pin OSC is connected to V_{DD}, thus disabling the internal oscillator and enabling pin CLK as an external clock input. The clock signal is provided by the master device.

In non-cascaded applications, pins $\overline{\text{SYNC}}$ and CLK must be left open circuit.

3.1 The SYNC pin

In order to achieve synchronous operation between cascaded chips, a $\overline{\text{SYNC}}$ pin has been provided. The only time that $\overline{\text{SYNC}}$ is likely to be needed is if synchronization is accidentally lost (e.g. by noise in adverse electrical environments, or by the definition of a multiplex mode when segment drivers with different I^2C addresses are cascaded).

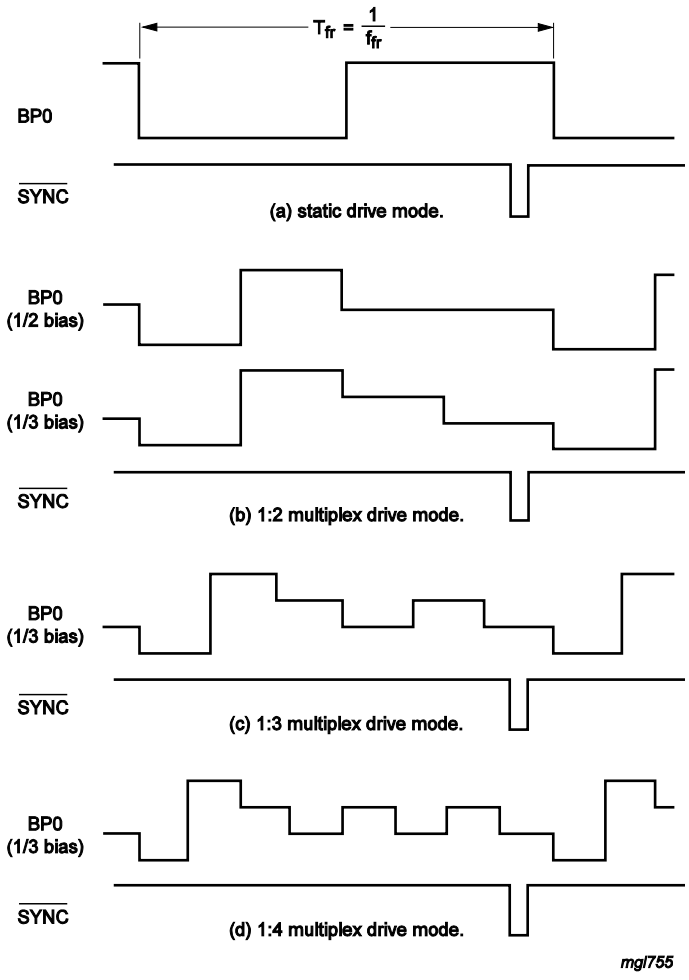


Fig 2. Synchronization and backplane signals for the various drive modes

The $\overline{\text{SYNC}}$ pin is organized as an input/output, the output selection being realized as an open-drain driver. It is not a 3-state pin. It is either output or input. That is, a sense circuit (input) is connected to this pin as well as an open drain output circuit. If the open drain output transistor is conducting, the $\overline{\text{SYNC}}$ pin will be low. If the open drain output transistor is not conducting the $\overline{\text{SYNC}}$ pin acts as an input and the segment driver is monitoring this pin. The $\overline{\text{SYNC}}$ pin is only actively driven LOW but never driven actively HIGH. Its HIGH level is achieved via an internal pull-up. This internal pull-up is weak and therefore the $\overline{\text{SYNC}}$ pin as output is not able to drive a significant load. Under the

influence of a strong external noise signal, in very rare cases the $\overline{SYNC}$ line may be affected. In this case an external pull-up resistor in the range from 100 kΩ to 10 kΩ can be connected. (This requires the $\overline{SYNC}$ signal of the cascade to be available on the LCD module connector if the external pull-up resistor is mounted on the PCB.)

Upon power-up, all cascaded segment drivers drive the $\overline{SYNC}$ pin actively LOW. Once a driver has finished its initialization and is ready for operation it releases the $\overline{SYNC}$ pin (does not drive it low anymore) and starts now to monitor the $\overline{SYNC}$ pin.

If all cascaded drivers have released their $\overline{SYNC}$ pin (none is driving it actively LOW anymore) then the $\overline{SYNC}$ line goes HIGH due to the internal pull ups.

In normal operation, the $\overline{SYNC}$ line is then actively driven LOW by each segment driver after the onset of its last active backplane signal in order to give a synchronization pulse, and released all other times. Therewith it can be guaranteed that once synchronization between several cascaded drivers is lost, resynchronization is possible. The timing relationships between the backplane waveforms and the $\overline{SYNC}$ signal for the various multiplex modes are shown in Fig 2.

If only the backplanes are used from one driver, the $\overline{SYNC}$ signal is necessary to align the output signals. Also when the backplanes from both (or more) drivers are used and drive a separate portion of the display, it is recommended to connect all $\overline{SYNC}$ pins together in order to avoid artifacts.

The contact resistance between the $\overline{SYNC}$ pins of cascaded devices must be controlled. If the resistance is too high, then the device may not be able to synchronize properly. This is particularly applicable to COG applications. Table 3 shows the limiting values for contact resistance.

Table 3. SYNC contact resistance

Number of devices	Maximum contact resistance
2	6000 Ω
3 to 5	2200 Ω
6 to 10	1200 Ω
11 to 16	700 Ω

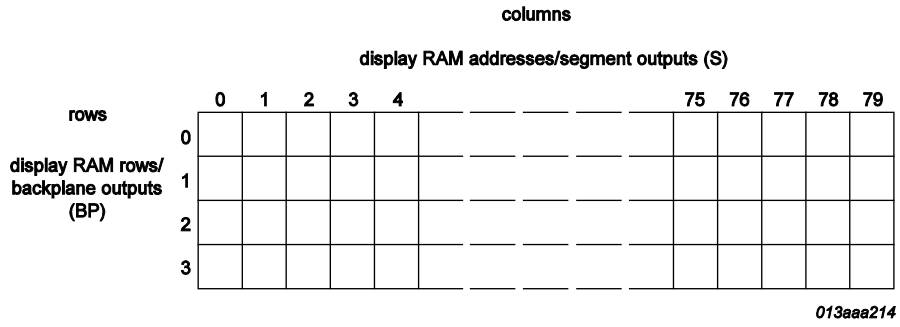
4. Display RAM

The display RAM (display latch) is a static S x 4 bit RAM which stores the LCD data. The number of segments, S, depends on the driver used. In general, the more segment outputs are present, the larger the internal display RAM. Refer to Table 1 for an overview. Sometimes the display RAM is larger than strictly required for a given number of segment outputs. This will affect the software used in a cascading application.

There is a one-to-one correspondence between

- the bits in the RAM bitmap and the LCD elements
- the RAM columns and the segment outputs
- the RAM rows and the backplane outputs

A logic 1 in the RAM bitmap indicates the on-state of the corresponding LCD element; similarly, a logic 0 indicates the off-state.



The display RAM bitmap shows the direct relationship between the display RAM addresses and the segment outputs and between the bits in a RAM word and the backplane outputs.

Fig 3. Display RAM bitmap

The display RAM bit map in Fig 3, is an example for a device with 80 segment outputs. Rows 0 to 3 correspond with the backplane outputs BP0 to BP3, and columns 0 to 79 correspond with the segment outputs S0 to S79. In multiplexed LCD applications the segment data of the first, second, third and fourth row of the display RAM are time-multiplexed with BP0, BP1, BP2 and BP3 respectively.

When display data is transmitted to the LCD driver, the received display bytes are stored in the display RAM in accordance with the selected LCD drive mode. The data is stored as it arrives and depending on the current multiplex drive mode the bits are stored singularly, in pairs, triples or quadruples. Therefore:

- In static mode, data is stored one bit at a time after which the data pointer increases. The eight transmitted data bits are all placed into row 0 as one byte.
- In 1:2 multiplex mode, two bits are stored and then the data pointer increases. The eight transmitted data bits are placed in pairs into row 0 and 1 as four successive 2-bit RAM words
- In 1:3 multiplex mode, three or two bits (3 + 3 + 2) are stored and then the data pointer increases. The eight transmitted data bits are placed in triples into row 0, 1 and 2 as three successive 3-bit RAM words, with bit 3 of the third address left unchanged.
- In 1:4 multiplex mode, four bits are stored and then the data pointer increases. The eight transmitted data bits are placed in quadruples into row 0, 1, 2 and 3 as two successive 4-bit RAM words.

This is shown in Fig 4.

4.1 Data pointer

The addressing mechanism for the display RAM is realized using a data pointer. This allows the loading of an individual display data byte, or a series of display data bytes, into

any location of the display RAM. The sequence commences with the initialization of the data pointer by the load-data-pointer command. Following this command, an arriving data byte is stored at the display RAM address indicated by the data pointer. The filling order is shown in Fig 4. After each byte is stored, the content of the data pointer is automatically incremented by a value dependent on the selected LCD drive mode, as described before.

4.2 Subadress counter

The storage of display data is determined by the content of the subaddress counter. Storage is allowed only when the content of the subaddress counter matches with the hardware subaddress applied to A0, A1 and if present, A2. The subaddress counter value is defined by the device-select command. If the content of the subaddress counter and the hardware subaddress do not match, then data storage is inhibited but the data pointer is incremented as if data storage had taken place. The subaddress counter is also incremented when the data pointer overflows. When the data pointer overflows it actually wraps around to 0. This happens to all drivers in the cascade (assuming now that they are all of the same type). Because the subaddress counter was incremented by one, the storing of data continuous in the next device in the cascade.

The storage arrangements described lead to extremely efficient data loading in cascaded applications. When a series of display bytes are sent to the display RAM, automatic wrap-over to the next LCD driver in the cascade occurs when the last RAM address is exceeded. Like this, the complete arrangement of cascaded devices appears to the microcontroller as one larger LCD driver.

However, this is only true if the command structure of all devices in the cascade is identical, and if the display RAM ends when the last segment is written, i.e. if the display RAM has exactly the size as required in order to address all segments. In this case wrapping over to the next device works seamlessly. Table 1 shows that this is not always the case. In such a case, the application software has to modify the data pointer (set it to 0) and subaddress counter when the next device in the cascade must be addressed.

4.3 Writing over the RAM address boundary

In all multiplex modes, depending on the setting of the data pointer, it is possible to exceed the RAM address boundary while filling the RAM. If the LCD driver is part of a cascade, the additional bits will fall into the next device that then also generates the I^2C acknowledge signal. If the LCD driver is a single device or the last device in a cascade, the additional bits will be discarded and no acknowledge signal will be generated.

Also when the display RAM is filled up to and including exactly the last address, no acknowledge will be generated for the last data byte being sent, if the LCD driver is the last driver in a cascade or a single device. Once the data bits of this last address have been written, the data pointer wraps over to 0 and the subaddress counter increments. The next driver in the cascade will then acknowledge, but of course only if it is present.

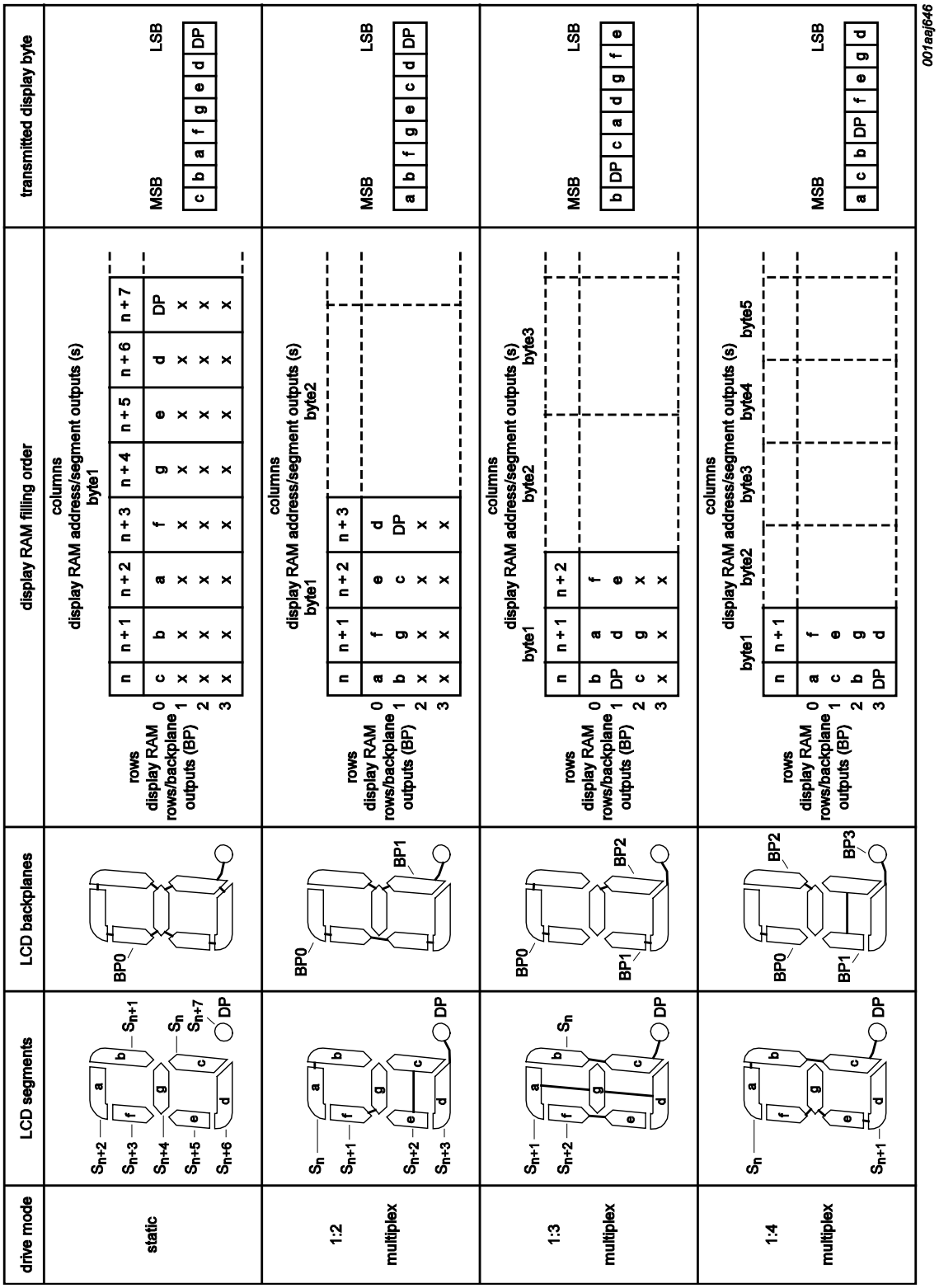


Fig 4. Relationships between LCD layout, drive mode, display RAM filling order, and display data transmitted over the I²C-bus

5. Simple and straight forward cascading

Cascading NXP LCD drivers is easy and straight forward. Fig 1 shows an example.

Only one master but multiple slaves are allowed in a cascade. All devices in the cascade have to use the same clock, whether it is supplied externally or provided by the master. If an external clock is used, all drivers in the cascade must be configured such as to receive the clock from that external source (pin OSC connected to V_{DD}). Thereby it must be ensured that the clock tree is designed such that the clock propagation delay from the clock source to all LCD drivers is as equal as possible, since otherwise synchronization artifacts may occur.

In the standard cascading configuration, all LCD drivers in the cascade share the same I^2C address. The individual drivers in the cascade can be differentiated by using a 2-bit or 3-bit hardware subaddress (A0, A1 and A2). These hardware subaddresses are set by connecting the respective pins to either V_{SS} or V_{DD} to make them logic 0 or logic 1.

The hardware subadress is combined with a subaddress counter. Storage is allowed only when the content of the subaddress counter match with the hardware subaddress applied to the pins A0, A1 and if present A2. This has been described in detail in section 4. If also the size of the display RAM matches exactly the number of segments that can be driven, cascading is very easy. No special precautions are required and to the microcontroller, the cascade looks like a larger LCD driver than the individual drivers in the cascade.

Table 4 gives an overview of the driver combinations that result in very simple cascades, and where no further software precautions are necessary.

Table 4. Recommended and easy combinations to build a cascade

Master	Slave	Last slave in cascade
	Any subaddress	Highest subaddress used
PCF8532	PCF8532	
PCA85132	PCA85132	
PCF85132	PCF85132	
PCA85232	PCA85232	
PCA8533	PCA8533	
PCF8533	PCF8533	
PCA85133	PCA85133	
PCF85133	PCF85133	
PCA85233	PCA85233	
PCA8576C	PCA8576C	
PCF8576C	PCF8576C	
PCA8576D	PCA8576D	
PCF8576D	PCF8576D	PCF8562
PCA8576F	PCA8576F	
PCA85176	PCA85176	PCA85162
PCF85176	PCF85176	PCF85162
PCA85276	PCA85276	PCA85262
PCF8577C	PCF8577C	

6. Combining different product types in one cascade

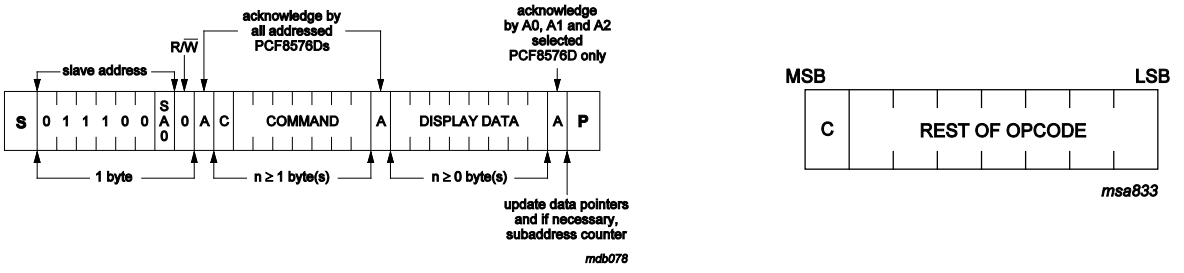
In most cases a cascade will be built using identical drivers. However, sometimes different combinations may be required. In some cases this is also straight forward, in other cases more difficult, and some combinations are not possible or recommended.

6.1 Command structures

The command structure or command length that is being used for a certain LCD driver depends on the number of segments that can be driven, because a larger number of segments results in a larger display RAM. A larger display RAM requires more address bits. The command load-data-pointer is used to set the DRAM address. The continuation bit C is used to indicate whether the next byte contains a command, or whether the next byte contains display data. Depending on whether 8 bits (one byte) are enough to include the address information and the continuation bit, the continuation bit can be included in the command byte, or sent in a separate byte. As a consequence, not all devices use the same command structure. There are three different versions.

6.1.1 Version 1

All commands include a continuation bit C. This bit is also included in the byte used to set the DRAM address. If the continuation bit C = 1, the control bytes continue and the next byte will be a command too. If the continuation bit C = 0, the byte is the last command byte in the transfer and the next byte will be regarded as display data. In addition, at least one bit is required to indicate which command is being sent. In this case this is the command to set the address, which is the command load-data-pointer. This leaves 6 bits for the actual address. See Fig 5.



a. I2C-bus protocol

b. Format command byte. Bit 7 = C, Bit 6 = 0

Fig 5. Command structure with only one command byte

With 6 bits up to 2^6 = 64 addresses can be used. In 1:4 multiplex mode, this allows for 256 segments. This control structure is used for the xxx62 and xxx76 types, where the addresses range from 0 to 39.

As a reminder, note that when two or more of these drivers are cascaded, 64 addresses are not enough to cover the total address space. In order to be able to address any location in the total DRAM, the command device-select must be used. The device-select command allows defining the subaddress counter value. The subadress counter is also incremented when the data pointer overflows, whereby the data pointer wraps around to 0.

6.1.2 Version 2

If larger segment drivers are used with a larger display RAM, more than 6 bits are required for the address pointer and this has consequences for the command structure. The bit C is no longer included in the load-data-pointer command, but sent in a separate control byte. The load-data-pointer command now contains always a 0 in bit 7. Bits 0 to 6 represent the address. The I^2C-bus protocol now changes and is shown in Fig 6. The sequence is initiated with a START condition (S) from the I^2C-bus master which is followed by the slave address. All drivers whose SA0 inputs correspond to bit 0 of the slave address respond by asserting an acknowledge in parallel.

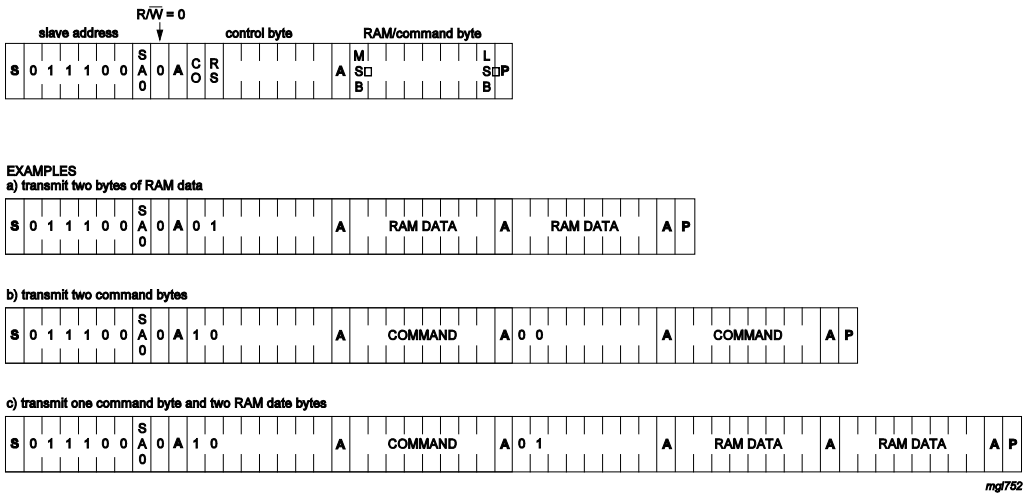
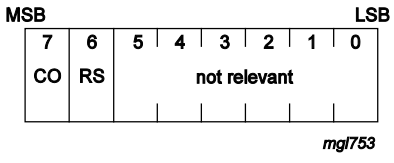


Fig 6. I^2C-bus protocol when two command bytes are required

After acknowledgement, the control byte is sent defining whether the next byte is a command or contains RAM information. The control byte contains also an RS bit (register select) at position b6. The other bits are don't care. This is shown in Fig 7. In between two command bytes always this control byte is included. Once RAM data is started to be sent, no further control bytes are required.



(1) C0 = 0; last control byte.

(2) C0 = 1: control bytes continue.

(3) RS = 0; data is a command byte.

(4) RS = 1; data is a display byte.

Fig 7. Control byte format when two command bytes are required

This command structure is used for the xxx34 and xxx33 types. The possible addresses here range from 0 to 79.

6.1.3 Version 3

The xxx32 types need 8 bits to address the full display RAM. Therefore the load-data-pointer command has been changed to two commands for the four most significant bits (load-data-pointer-MSB) and the four least significant bits (load-data-pointer-LSB). Fig 6 and Fig 7 are equally valid for these devices, but here two address command bytes must be sent in order to transfer the full address. The address range is from 0 to 159.

6.2 Combining devices with the same command structure

Cascading is easiest when all LCD drivers used in the cascade are of the same type and with a DRAM exactly fitting the number of segments that can be driven. In that case cascading is straight forward and the total cascade will really look to the rest of the application as one larger LCD driver, which is according to the definition as given before. No further software precautions are necessary. An example is shown in Fig 1 and recommended combinations in Table 4. The cascade does not have to be limited to two devices and can be extended.

When all devices are of the same type, referring to the device datasheet, or sections 3 and 4 of this application note, will provide all relevant information.

It is also possible to combine different type numbers. As long as the command structure is the same, cascading is still easy. If a device is included with a larger DRAM than the number of segments that can be driven (for example xxx62 variants or xxx34 variants), that LCD driver should be the last driver in the cascade. That is, this LCD driver should have the highest subaddress. In this case, no special software precautions (changing the data pointer and subaddress counter) must be taken.

6.2.1 RAM addressing when display RAM size does not match

In cascaded applications where the DRAM size does not match the number of segments that can be driven, each LCD driver which is not the last in the cascade must be addressed separately. This situation would occur for example when a cascade is built using the combinations given in Table 5.

Table 5. Easy combinations to build a cascade

The combinations below are easy as well but require some software intervention

Master	Slave
	Any subaddress
PCA8534A	PCA8534A
PCF8534A	PCF8534A
PCA85134	PCA85134
PCF85134	PCF85134
PCF8562	PCF8562
PCA85162	PCA85162
PCF85162	PCF85162
PCA85262	PCA85262

The way of working is as follows. Initially, the first driver is selected by sending the device-select command matching the first device's hardware subaddress. Then the data pointer is set to the preferred display RAM address by sending the load-data-pointer command.

Once the display RAM of the first LCD driver has been written, the second driver is selected by sending the device-select command again. This time however the command matches the second device's hardware subaddress. Next the load-data-pointer command is sent to select the preferred display RAM address of the second driver.

This last step is very important because during writing data to the first driver, the data pointer of the second driver is incremented. In addition, the hardware subaddress should not be changed whilst the device is being accessed on the I^2C-bus interface.

If more than two LCD drivers are combined, the same method of writing to the DRAM is applicable for those as well. However, a better option would then be to use other LCD drivers which can drive a higher number of segments.

6.3 Combining devices with different command structure

It is also possible to make other combinations, but this is not cascading as was meant in the earlier given definition.

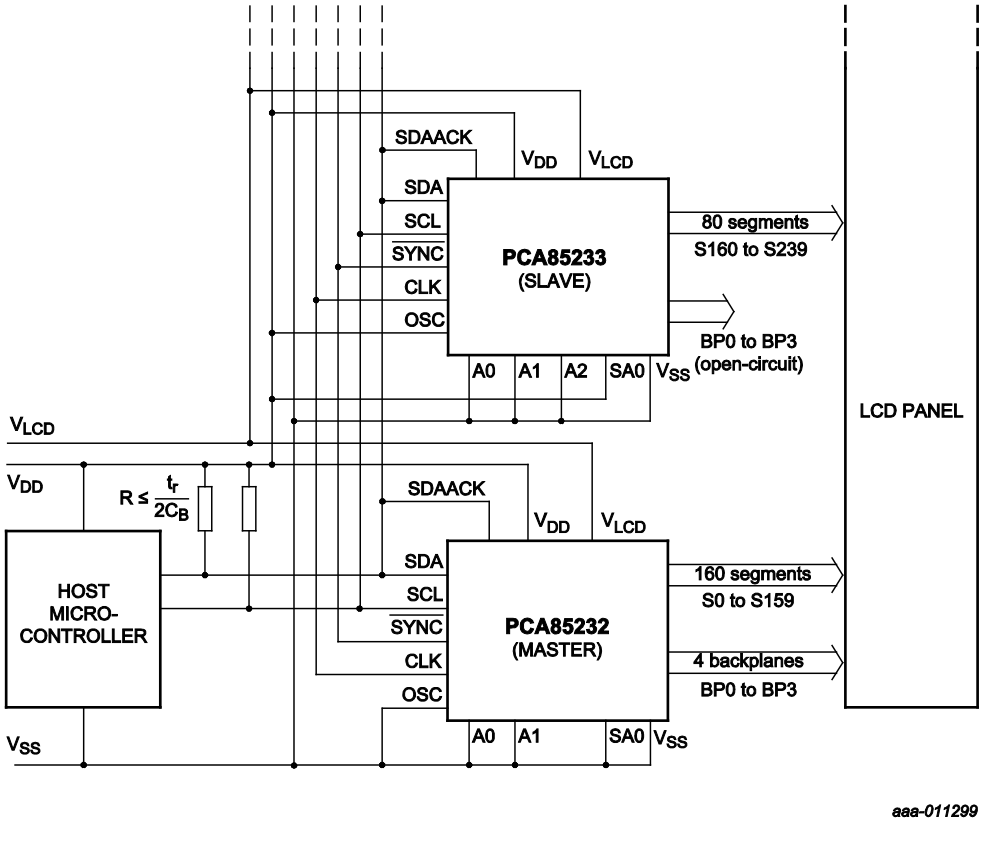


Fig 8. Combining PCA85232 and PCA85233 using two I^2C-bus addresses

Due to the differences in command structure, it is not possible to combine LCD drivers with command structure version 1, with drivers using version 2 or 3.

It is possible to combine drivers using command structure versions 2 and 3. Two examples are given, both using PCA85232 and PCA85233.

In the first example in Fig 8, PCA85232 uses I²C-bus address 70h while PCA85233 uses slave address 72h. Therefore they can both be independently addressed on the same I²C-bus. Master device PCA85232 drives the display with its segment and backplane outputs, while slave device PCA85233 contributes additional segments. Both devices can use any subaddress and the most logical is to use subadress 0 for both, realized by connecting all subaddress pins (A0, A1, A2) to V_{SS}.

In the second example in Fig 9, both master and slave device use the same I²C-bus address. PCA85232 (master) has hardware subaddress 0 while the slave device PCA85233 uses subaddress 2.

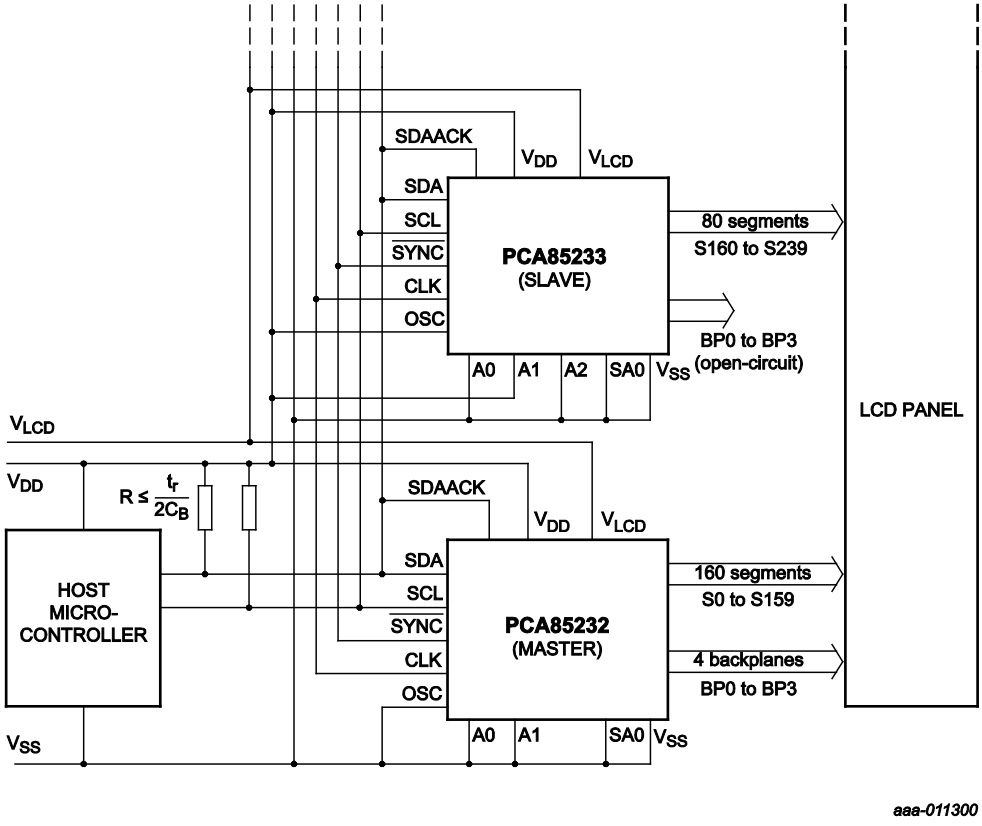


Fig 9. Combining PCA85232 and PCA85233 using only one I²C-bus address

Note that the subaddress now is not set 1 higher, but 2 higher than the master, in order to account for the larger display RAM size of PCA85232 which is twice the size of that of PCA85233. Care must be taken during the development of the software that the load-data-pointer command is sent in the correct sequence. The load-data-pointer-LSB command of PCA85232 should be sent first (0100 0000), followed by the load-data-pointer-MSB command (0000 0000), thus setting also the data pointer of the slave device PCA85233 to 0h.

If the display RAM of the master - PCA85232 - is written to from the start address 0, both data pointers of PCA85232 and PCA85233 increment while writing data. After writing to address 79, the data pointer of PCA85233 wraps around to 0 and the subaddress counter of PCA85233 increments by one, while the data pointer of PCA85232 continuous and increments to 80. Since the display RAM of PCA85232 is exactly twice the size of the display RAM of PCA85233, at the end of the RAM of PCA85232, both data pointers wrap around to 0. Now the subaddress of PCA85232 is 1 and the subaddress of PCA85233 equals 2. This is why the hardware subaddress of PCA85233 must be set to 2.

6.4 Different electrical properties

The supply voltage ranges, both the logic operating voltage V_{DD} and the LCD operating voltage V_{LCD} are not the same for all NXP LCD drivers. Combining LCD drivers with different operating voltage ranges results in the limits for the cascade being determined by the driver with the highest minimum supply voltage and the device with the lowest maximum supply voltage.

Similarly, also the possible frame frequencies are not the same for all LCD drivers. The limits of the cascade will be set by the device with the highest minimum value and the device with the lowest maximum value.

In mixed cascading configurations, care has to be taken that the specifications of the individual cascaded devices are met at all times.

6.5 Difference PCF8566, PCA8576C and PCF8576C compared to other drivers

The diffusion process used for PCF8566, PCA8576C and PCF8576C requires V_{DD} to be at substrate potential. As a consequence, V_{LCD} must be negative with respect to V_{DD}. If the LCD driving voltage is more than ($V_{DD} – V_{SS}$), then V_{LCD} is also negative with respect to V_{SS}. In the special case that the LCD driving voltage is the same as ($V_{DD} – V_{SS}$), V_{LCD} can simply be connected to V_{SS}. This is indicated in the block diagram in Fig 10. This block diagram is representative for these three drivers.

For comparison Fig 11 shows the block diagram representative for the other LCD drivers discussed in this document, including PCF8576D. These use a different diffusion process in which V_{SS} is at substrate level. This means that V_{LCD} must be positive with respect to V_{SS}. If the LCD driving voltage is more than ($V_{DD} – V_{SS}$), then V_{LCD} is also positive with respect to V_{DD}. In the special case when the LCD driving voltage is the same as ($V_{DD} – V_{SS}$), V_{LCD} can simply be connected to V_{DD}.

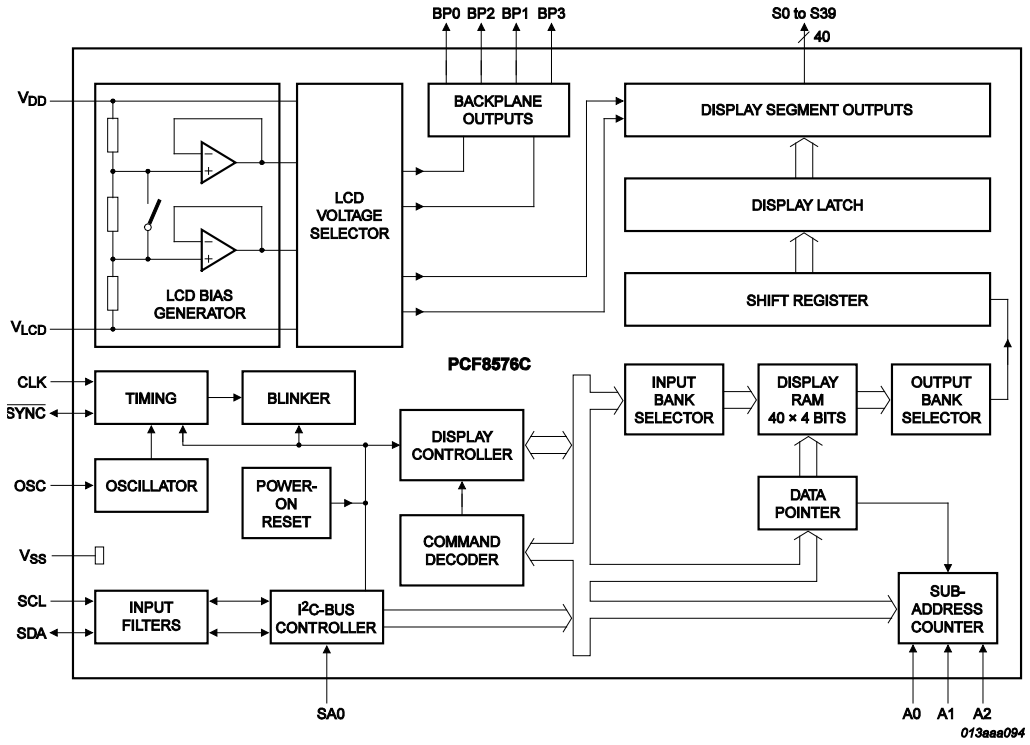


Fig 10. Block diagram of PCF8576C

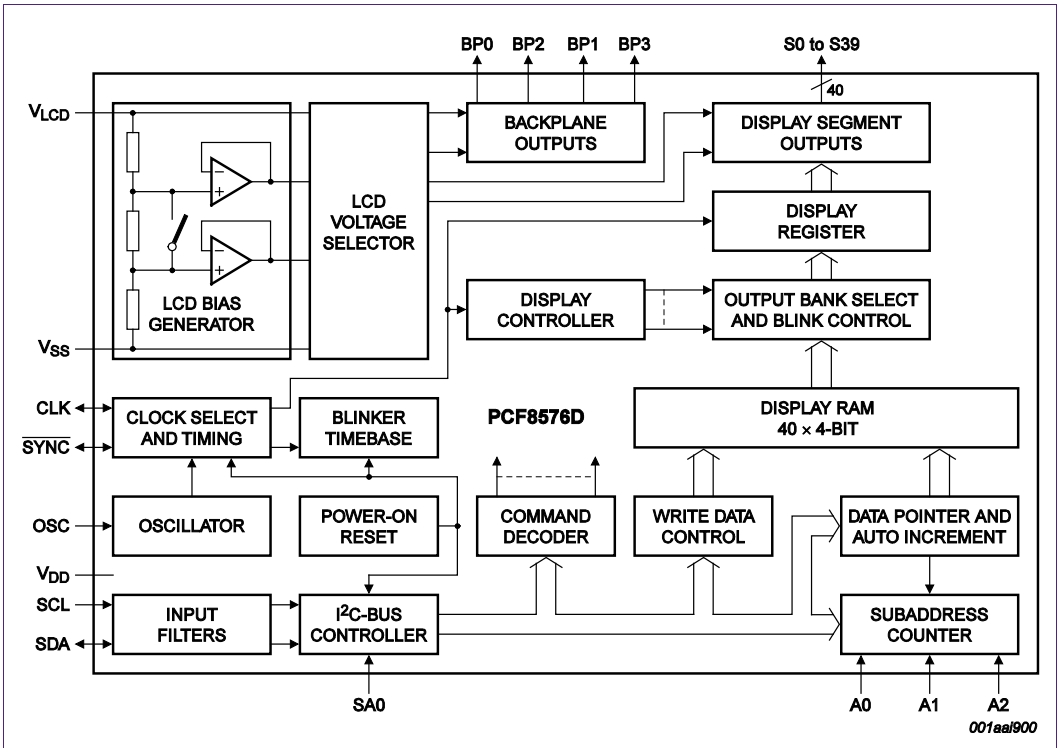


Fig 11. Block diagram of PCF8576D

Conclusion:

1. In the special case of the LCD driving voltage being the same as (V_{DD} – V_{SS}) across the whole operating temperature range, the only required change in system hardware when changing from PCx8576C to drivers like PCF8576D or PCA85176 is connecting V_{LCD} to V_{DD} instead of connecting it to V_{SS}. Note however that in this case V_{LCD} is limited to the maximum operating value of V_{DD}.
2. PCx8576C and PCF8576D et al are software compatible. However, the low power command of PCx8576C will be ignored by the other LCD drivers. Furthermore the I^2C-bus frequency of PCx8576C is limited to 100 kHz whereas the other drivers are Fast mode I^2C devices with a clock frequency up to 400 kHz.

PCx8576C should only be cascaded with other PCx8576C or with PCF8566.

6.5.1 PCF8566

Two I^2C-bus slave addresses (0111 110 and 0111 111) are reserved for PCF8566. The PCF8566 can only be cascaded with other PCF8566 (up to 16 devices). Straight forward cascading with other devices as meant in this application note will not work, because all other devices have a different I^2C-bus slave address. PCF8566 can be combined in one cascade with PCF8576C when the software is written such that it takes care of the different I^2C-bus addresses. The supply voltage architecture is identical to that of PCA8576C and PCF8576C.

6.6 PCF8577C

PCF8577C can be cascaded, but it is not compatible with other drivers in the portfolio. It can only be cascaded with other PCF8577C.

7. References

The documents listed below provide further useful information. They are available at NXP's website.

[1] The datasheets of the segment drivers used.

[2] **AN10170** – Design guidelines for COG modules with NXP monochrome LCD drivers

[3] **AN11267** – EMC and system level ESD design guidelines for LCD drivers

[4] **AN11491** – Design and application guidelines for the COG LCD drivers PCF8538 and PCA8538

[5] **R_10015** – Chip-On-Glass (COG) – a cost-effective and reliable technology for LCD displays

8. Legal information

8.1 Definitions

Draft — The document is a draft version only. The content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included herein and shall have no liability for the consequences of use of such information.

8.2 Disclaimers

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information.

In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory.

Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of NXP Semiconductors.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors accepts no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

Evaluation products — This product is provided on an "as is" and "with all faults" basis for evaluation purposes only. NXP Semiconductors, its affiliates and their suppliers expressly disclaim all warranties, whether express, implied or statutory, including but not limited to the implied warranties of non-infringement, merchantability and fitness for a particular purpose. The entire risk as to the quality, or arising out of the use or performance, of this product remains with customer.

In no event shall NXP Semiconductors, its affiliates or their suppliers be liable to customer for any special, indirect, consequential, punitive or incidental damages (including without limitation damages for loss of business, business interruption, loss of use, loss of data or information, and the like) arising out the use of or inability to use the product, whether or not based on tort (including negligence), strict liability, breach of contract, breach of warranty or any other theory, even if advised of the possibility of such damages.

Notwithstanding any damages that customer might incur for any reason whatsoever (including without limitation, all damages referenced above and all direct or general damages), the entire liability of NXP Semiconductors, its affiliates and their suppliers and customer's exclusive remedy for all of the foregoing shall be limited to actual damages incurred by customer based on reasonable reliance up to the greater of the amount actually paid by customer for the product or five dollars (US$5.00). The foregoing limitations, exclusions and disclaimers shall apply to the maximum extent permitted by applicable law, even if any remedy fails of its essential purpose.

Translations – A non-English (translated) version of a document is for reference only. The English version shall prevail in case of any discrepancy between the translated and English versions.

8.3 Trademarks

Notice: All referenced brands, product names, service names and trademarks are property of their respective owners.

9. List of figures

10. List of tables

11. Contents

For more information, please visit: http://www.nxp.com
For sales office addresses, please send an email to: salesaddresses@nxp.com

Date of release: 12 February 2014
Document identifier: AN11494