PCA9661

Parallel bus to 1 channel Fm+ I^{2}C-bus controller

Rev. 1 — 4 August 2011 **Product data sheet**

1. General description

The PCA9661 is an advanced single master mode I^{2}C-bus controller. It is a fourth generation bus controller designed for data intensive I^{2}C-bus data transfers. It has one I^{2}C-bus channel with data rates up to 1 Mbits/s using the Fast-mode Plus (Fm+) open-drain topology. The serial channel has a generous 4352 byte data buffer which makes the PCA9661 the ideal companion to any CPU that needs to transmit and receive large amounts of serial data with minimal interruptions.

The PCA9661 is a 8-bit parallel-bus to I^{2}C-bus protocol converter. It can be configured to communicate with up to 64 slaves in one serial sequence with no intervention from the CPU. The controller also has a sequence loop control feature that allows it to automatically retransmit a stored sequence.

Its onboard oscillator and PLL allow the controller to generate the clocks for the I^{2}C-bus and for the interval timer used in sequence looping. This feature greatly reduces CPU overhead when data refresh is required in fault tolerant applications.

An external trigger input allows data synchronization with external events. The trigger signal controls the rate at which a stored sequence is re-transmitted over the I^{2}C-bus.

Error reporting is handled at the transaction level, channel level, and controller level. A simple interrupt tree and interrupt masks allow further customization of interrupt management.

The controller parallel bus interface runs at 3.3 V and the I^{2}C-bus I/Os logic levels are referenced to a dedicated $V_{DD(IO)}$ input pin with a range of 3.0 V to 5.5 V.

2. Features and benefits

- Parallel-bus to I^{2}C-bus protocol converter and interface
- 1 Mbit/s and up to 30 mA SCL/SDA I_{OL} Fast-mode Plus (Fm+) capability
- Internal oscillator trimmed to 1 % accuracy reduces external components
- 4352-byte buffer for the Fm+ channel
- Three levels of reset: individual software channel reset, global software reset, global hardware $\overline{\text{RESET}}$ pin
- Communicates with up to 64 slaves in one serial sequence
- Sequence looping with interval timer
- Supports SCL clock stretching
- JTAG port available for boundary scan testing during board manufacturing process
- Trigger input synchronizes serial communication exactly with external events
- Maskable interrupts

- Fast-mode Plus I^{2}C-bus capable and compatible with SMBus
- Operating supply voltage: 3.0 V to 3.6 V (device and host interface)
- I^{2}C-bus I/O supply voltage: 3.0 V to 5.5 V
- Latch-up testing is done to JEDEC Standard JESD78 which exceeds 100 mA
- ESD protection exceeds 8000 V HBM per JESD22-A114 and 1000 V CDM per JESD22-C101
- Packages offered: LQFP48

3. Applications

- Add I^{2}C-bus port to controllers/processors that do not have one
- Add additional I^{2}C-bus ports to controllers/processors that need multiple I^{2}C-bus ports
- Converts 8 bits of parallel data to serial data stream to prevent having to run a large number of traces across the entire printed-circuit board
- Entertainment systems
- LED matrix control
- Data intensive I^{2}C-bus transfers

4. Ordering information

Table 1. Ordering information

Type number	Topside mark	Package		
		Name	Description	Version
PCA9661B	PCA9661	LQFP48	plastic low profile quad flat package; 48 leads; body 7 × 7 × 1.4 mm	SOT313-2

5. Block diagram

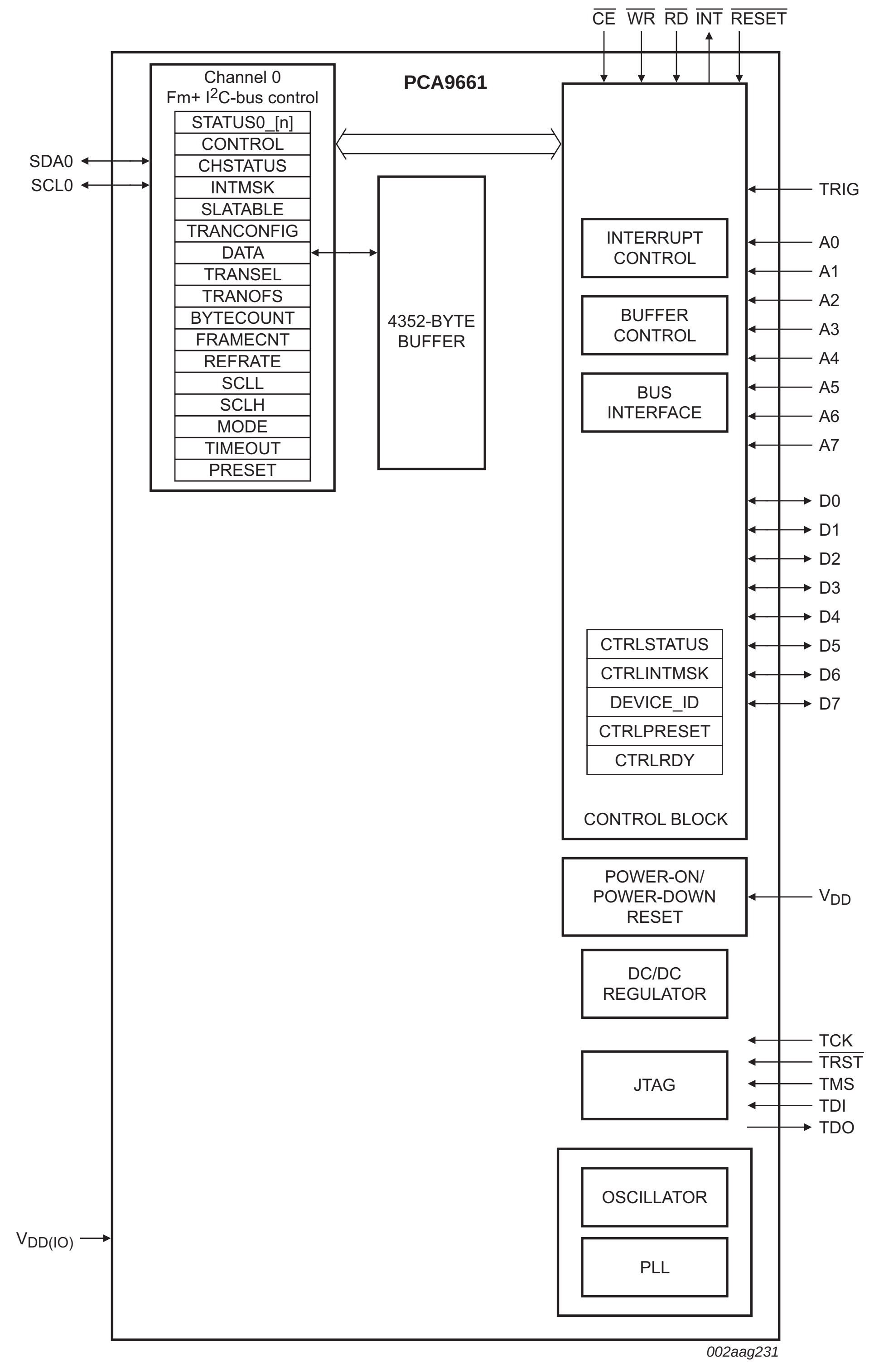


Fig 1. Block diagram

6. Pinning information

6.1 Pinning

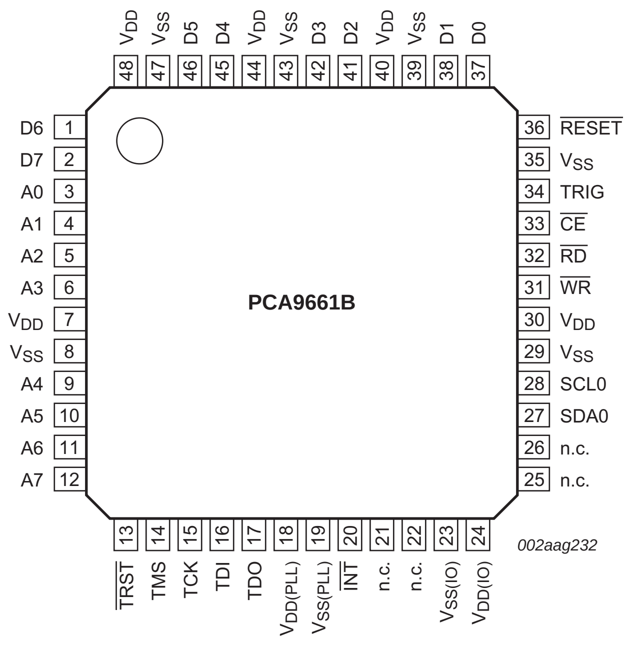


Fig 2. Pin configuration for LQFP48

6.2 Pin description

Table 2. Pin description

Symbol	Pin	Type	Description
A0	3	I	**Address inputs:** selects the bus controller's internal registers and ports for read/write operations. Address is registered when $\overline{CE}$ is LOW and whether $\overline{WR}$ or $\overline{RD}$ transitions LOW. A0 is the least significant bit.
A1	4	I	
A2	5	I	
A3	6	I	
A4	9	I	
A5	10	I	
A6	11	I	
A7	12	I	
D0	37	I/O	**Data bus:** bidirectional 3-state data bus used to transfer commands, data and status between the bus controller and the host. D0 is the least significant bit. Data is registered on the rising edge of $\overline{WR}$ when $\overline{CE}$ is LOW.
D1	38	I/O	
D2	41	I/O	
D3	42	I/O	
D4	45	I/O	
D5	46	I/O	
D6	1	I/O	
D7	2	I/O	

Table 2. Pin description *...continued*

Symbol	Pin	Type	Description
$\overline{TRST}$	13	I	**JTAG test reset input.** For normal operation, hold LOW (V_{SS}).
TMS	14	I	**JTAG test mode select input.** For normal operation, hold HIGH (V_{DD}).
TCK	15	I	**JTAG test clock input.** For normal operation, hold HIGH (V_{DD}).
TDI	16	I	**JTAG test data in input.** For normal operation, hold HIGH (V_{DD}).
TDO	17	O	**JTAG test data out output.** For normal operation, do not connect (n.c.).
$\overline{INT}$	20	O	**Interrupt request:** Active LOW, open-drain, output. This pin requires a pull-up device.
SDA0	27	I/O	**Channel 0 I²C-bus serial data input/output** (open-drain). This pin requires a pull-up device.
SCL0	28	I/O	**Channel 0 I²C-bus serial clock input/output** (open-drain). This pin requires a pull-up device.
$\overline{WR}$	31	I	**Write strobe:** When LOW and $\overline{CE}$ is also LOW, the content of the data bus is loaded into the addressed register. Data are latched on the rising edge of $\overline{WR}$. $\overline{CE}$ may remain LOW or transition with $\overline{WR}$.
$\overline{RD}$	32	I	**Read strobe:** When LOW and $\overline{CE}$ is also LOW, causes the contents of the addressed register to be presented on the data bus. The read cycle begins on the falling edge of $\overline{RD}$. Data lines are driven when $\overline{RD}$ and $\overline{CE}$ are LOW. $\overline{CE}$ may transition with $\overline{RD}$.
$\overline{CE}$	33	I	**Chip Enable:** Active LOW input signal. When LOW, data transfers between the host and the bus controller are enabled on D0 to D7 as controlled by the $\overline{WR}$, $\overline{RD}$ and A0 to A7 inputs. When HIGH, places the D0 to D7 lines in the 3-state condition. During the initialization period, $\overline{CE}$ must transition with $\overline{RD}$ until controller is ready.
TRIG	34	I	**Trigger input:** provides the trigger to start a new frame.
$\overline{RESET}$	36	I	**Reset:** Active LOW input. A LOW level resets the device to the power-on state. Internally pulled HIGH through weak pull-up current.
$V_{DD(IO)}$	24	power	**I/O power supply:** 3.0 V to 5.5 V. Power supply reference for I²C-bus pins.
$V_{SS(IO)}$	23	power	**I/O supply ground.** Can be tied to V_{SS}.
$V_{DD(PLL)}$	18	power	**PLL power supply.**
$V_{SS(PLL)}$	19	power	**PLL supply ground.**
V_{DD}	7, 30, 40, 44, 48	power	**Power supply:** 3.0 V to 3.6 V. All V_{DD} pins should be connected together externally.
V_{SS}	8, 29, 35, 39, 43, 47	power	**Supply ground.** All V_{SS} pins must be tied together externally.
n.c.	21, 22, 25, 26	-	not connected

7. Functional description

7.1 General

The PCA9661 acts as an interface device between standard high-speed parallel buses and the serial I^{2}C-bus. On the I^{2}C-bus, it acts as a master. Data transfer between the I^{2}C-bus and the parallel-bus host is carried out on a buffered basis, using either an interrupt or polled handshake.

7.2 Internal oscillator and PLL

The PCA9661 contains an internal 12.0 MHz oscillator and 156 MHz PLL which are used for all internal and I^{2}C-bus timing. The oscillator and PLL require up to $t_{init(po)}$ to start up and lock after power-up. The oscillator is not shut down if the serial bus is disabled.

7.3 Buffer description

Remark: In the following section a ‘transaction’ is defined as a contiguous set of commands and/or data sent/received to/from a single slave. A ‘sequence’ is a set of transactions stored in the buffer.

The PCA9661 serial channel has a 4352-byte data buffer (see Section 7.3.2 “Buffer size”) that allows several transactions to be executed before an interrupt is generated. This allows the host to request several transactions (up to maximum buffer size on each channel) in a single sequence and lets the PCA9661 perform it without the intervention of the host each time a requested transaction is performed. The host can then perform other tasks while the PCA9661 executes the requested sequences.

By following a simple procedure, the I^{2}C-bus controller can store several I^{2}C-bus transactions directed to different slaves addresses on any of the channels. The transaction stored in the buffer can be of any type, thus reads and writes can be interlaced in a sequence. When multiple slave reads are requested in a sequence, the read data is stored in-line in the sequence and the buffer number must be specified in the TRANSEL to provide the read location and the TRANOFS byte offset value. By default, the TRANOFS is set to 00h. So let us consider the scenario where the host has done the initialization (mode, masks, and other configuration) and writes data into the buffer of one of the three channels.

The host starts by programming the buffer configuration registers TRANCONFIG (number of slaves and bytes per slave) and then the SLATABLE (slave addresses). Then the host programs the TRANSEL (Transaction Data Buffer Selection) and the TRANOFS (byte offset selection) to 00h to set the memory pointers to the beginning of the buffer (the default value is 00h after a power-on or $\overline{\text{RESET}}$). Next, the host transfers the data into DATA until the entire sequence is loaded. If the transaction is a read transaction, the host must write a dummy byte (i.e., FFh) for each expected serial read byte to reserve the memory space in the buffer for the transaction.

Care should be taken so as to not overflow the buffer with excessive read/write commands. In the event of an overflow, represented by the BE bit in the CTRLSTATUS register, will be set to logic 1. The $\overline{\text{INT}}$ pin will be set LOW if the BEMSK bit in the CTRLINTMSK register is logic 0. To recover the channel, a channel reset is required. All configuration and data needs to be checked by the host and resent to the I^{2}C-bus controller. (See Section 7.3.2 “Buffer size”.)

After sending all the commands and data it wanted to the I^{2}C-bus controller, the host writes to the CONTROL register to begin data transmission on the serial channel. The transactions will be sent on the I^{2}C-bus in the order in which the slave addresses are listed in the SLATABLE, separated by a RESTART condition. The last transaction in the sequence will end with a STOP condition.

If during a READ command a NACK on the slave address is received, the buffer space allocated for the read will remain untouched and will contain the last information written in that location. A buffer read on the parallel bus should only be done after a valid buffer state is reached to guarantee data valid (see Section 7.5.1.1 “STATUS0_[n] — Transaction status registers”).

7.3.1 Buffer management assumptions

- Repeated STARTs will be sent between two consecutive transactions.
- After the last operation on a channel is completed, a STOP will be sent.
- In a READ transaction, after the last data byte has been received from a particular slave, a NACK is sent to the slave.

7.3.2 Buffer size

The PCA9661 serial channel has a 4352-byte buffer assigned to it. The contents of the buffers should only be modified during channel idle states.

The buffer size represents the memory allocated for the data block only. The slave address table and configuration bytes are contained in other locations and do not need to be included in the required buffer size calculation.

For example, to calculate the size of the memory needed to write 26 bytes to 10 slaves and to read 2 bytes from 4 slaves (no command bytes required for the read):

10 slaves $\times$ 26 bytes/slave = 260 bytes for the write transactions

4 slaves $\times$ 2 bytes/slave = 8 bytes for the read transactions

A total of 268 bytes of buffer space is required to complete the sequence.

Remark: Note that the bytes required to store the 30 slave addresses are not included in the calculation since they are stored in the SLATABLE register.

7.4 Error reporting and handling

In case of any transaction error conditions, the device will load the transaction error status in the STATUS0_[n], generate an interrupt, if unmasked, by pulling down the $\overline{\text{INT}}$ pin and update the CHSTATUS and CTRLSTATUS registers. The status for the individual SLA addresses will be stored in the STATUS0_[n] registers.

In the event of a NACK from a slave, there are two possible courses of action. The first is that an interrupt will be generated and the current transaction and sequence terminated. The second is that while the WEMSK and/or REMSK is a logic 1, a NACKed byte will be ignored, and the transmission will continue with the next transaction in the sequence until the end of the sequence. The controller will skip the slave address and/or data where the NACK occurred and move on to the next transaction in the sequence. Any error will be reported in the corresponding STATUS0_[n] register (where 'n' is the buffer number of the slave) or the CHSTATUS or CTRLSTATUS registers.

7.5 Registers

The PCA9661 contains several registers that are used to configure the operation of the device, status reporting, and to send and receive data. The device also contains global registers for chip level control and status reporting.

The STATUS0_[n] registers are channel-level direct access registers. The DATA, SLATABLE, TRANCONFIG, and BYTECOUNT registers are auto-increment registers.

The memory access pointer to the DATA registers can be programmed using the TRANSEL and TRANOFS registers. See Section 7.5.1.2 "CONTROL — Control register", for information on the pointer reset bits BPTRRST and AIPTRRST.

Table 3. PCA9661 register address map - direct register access

7	6	5	4	3	2	1	0	Register name	Access	Write access while CH active	Description	Default	Size (bytes)
Channel status register													
0	0	channel 0 transaction number (hex)						STATUS0_[n]	R	no	individual transaction status (direct address)	00h	64
Channel 0 (Fm+) registers													
1	1	0	0	0	0	0	0	CONTROL	R/W	yes[1]	channel 0 control	00h	1
				0	0	0	1	CHSTATUS	R	no	channel 0 status	00h	1
				0	0	1	0	INTMSK	R/W	yes	channel 0 interrupt mask	00h	1
				0	0	1	1	SLATABLE	R/W	no	channel 0 slave address table (auto-increment)	00h	64
				0	1	0	0	TRANCONFIG	R/W	yes, for TRANCOUNT[2]	channel 0 transaction configuration (auto-increment)	00h	65
				0	1	0	1	DATA	R/W	yes	channel 0 data (auto-increment)	00h	bufsize[3]
				0	1	1	0	TRANSEL	R/W	yes	channel 0 transaction data buffer select	00h	1
				0	1	1	1	TRANOFS	R/W	yes	channel 0 transaction data buffer byte offset	00h	1
				1	0	0	0	BYTECOUNT	R	no	channel 0 transmitted byte count (auto-increment)	00h	64
				1	0	0	1	FRAMECNT	R/W	no	channel 0 frame count	01h	1
				1	0	1	0	REFRATE	R/W	no	channel 0 frame refresh rate	00h	1
				1	0	1	1	SCLL	R/W	no	channel 0 clock LOW state	5Eh	1
				1	1	0	0	SCLH	R/W	no	channel 0 clock HIGH state	3Fh	1
				1	1	0	1	MODE	R/W	no	channel 0 mode	92h	1
				1	1	1	0	TIMEOUT	R/W	no	channel 0 time-out	00h	1
				1	1	1	1	PRESET	R/W	yes	channel 0 parallel reset	00h	1

Table 3. PCA9661 register address map - direct register access *…continued*

7	6	5	4	3	2	1	0	Register name	Access	Write access while CH active	Description	Default	Size (bytes)
Global registers													
1	1	1	1	0	0	0	0	CTRLSTATUS	R	yes	controller status	00h	1
				0	0	0	1	CTRLINTMSK	R/W	yes	master interrupt mask	00h	1
				0	0	1	0	-	R	no	reserved	00h	
				0	0	1	1	-	R	no	reserved	00h	
				0	1	0	0	-	R	no	reserved	00h	
				0	1	0	1	-	R	no	reserved	00h	
				0	1	1	0	DEVICE_ID	R	no	device ID	61h	
				0	1	1	1	CTRLPRESET	R/W	yes	master parallel reset	00h	1
				1	1	1	1	CTRLRDY[4]	R	no	controller ready register	FFh	1

[1] Except TP and TE. Changing polarity of TP while TE is active will cause a false trigger.

[2] The transaction count (TRANCONFIG[0]) can be written to during the idle period between sequences.

[3] Refer to Section 7.3.2 “Buffer size” for channel memory allocation.

[4] Controller ready = FFh immediately after POR or after a hardware reset or global reset. It will clear (00h) once the initialization routine is done.

7.5.1 Channel registers

7.5.1.1 STATUS0_[n] — Transaction status registers

STATUS0_[n] is an 8-bit × 64 read-only register that provides status information for a given transaction. Only the 5 lower bits are used; the top bits will always read 0. When bits [4:2] are set, a channel interrupt is requested (the $\overline{\text{INT}}$ pin is asserted LOW). A read to STATUS0_[n] register will clear its status. To clear all the STATUS0_[n] registers, a byte-by-byte read of all STATUS0_[n] registers is required. The controller will auto-clear the STATUS0_[n] registers at each START of a sequence when FRAMECNT = 1 and only at the first START when FRAMECNT ≠ 1.

Each register byte can be accessed by direct addressing so that the host can choose to read the status on one or more individual transactions without having to read all 64 status bytes.

Table 4. STATUS0_[n] - Transaction status code register bit description

Bit	Symbol	Description
7:5	ST[7:5]	always reads 000
4	RSN	Read slave NACK. When HIGH, a NACK was received after a slave address was transmitted on the serial bus on a read transaction. An interrupt will be requested.
3	WSN	Write slave NACK. When HIGH, a NACK was received after a slave address was transmitted on the serial bus on a write transaction. An interrupt will be requested.
2	WDN	Write data NACK. When HIGH, a NACK was received for a data byte during a write transaction on the serial bus. An interrupt will be requested.
1	TA	Transaction active. When 1, the transaction is currently active on the serial bus. No interrupt is requested.
0	TR	Transaction ready. When 1, a transaction is loaded in the buffer and waiting to be executed. No interrupt is requested.

Remark: When STATUS0_[n] = 00h, no interrupt is requested and the transaction is in the Done/Idle state.

During program execution, the TR and TA bits behave as follows:

Example, we are to transfer 3 transactions in a sequence. All initialization is completed (loading of SLA, TRANCONFIG, DATA) and device is ready for serial transfer.

Before the STA bit is set, the STATUS0_[n] register will contain:

STATUS0_[0] = 0

STATUS0_[1] = 0

STATUS0_[2] = 0

STATUS0_[3] = 0

:

After STA is set:

STATUS0_[0] = 2

STATUS0_[1] = 1

STATUS0_[2] = 1

STATUS0_[3] = 0

:

Since there is no timing requirement in setting the STA bit after the initialization, the device will update the first status when the STA bit is set and will always go from 0 to 2 (Idle to Transaction active).

7.5.1.2 CONTROL — Control register

CONTROL is an 8-bit register. The STO bit is affected by the bus controller hardware: it is cleared when a STOP condition is present on the I^{2}C-bus.

Table 5. CONTROL - Control register bit description
Address: Channel 0 = C0h.
*Legend: * reset value*

Bit	Symbol	Access	Value	Description
7	STOSEQ	R/W		Stop sequence bit.
			1	When the STOSEQ bit is set while the channel is active, a STOP condition will be transmitted immediately following the end of the current sequence being transferred on the I^{2}C-bus. No further buffered transactions will be carried out and the channel will return to the idle state. Normal error reporting will occur up until the last bit. When a STOP condition is detected on the bus, the hardware clears the STOSEQ flag.
			0*	When STOSEQ is reset, no action will be taken.
6	STA	R/W		The START flag.
			1	When the STA bit is set to begin a sequence, the bus controller hardware checks the status of the I^{2}C-bus and generates a START condition if the bus is free. If the bus is not idle, then $\overline{\text{INT}}$ will go LOW and the CHSTATUS register will contain a bus error code (either DAE or CLE will be set). The STA bit may be set only at a valid idle state. The controller will reset the bit under the following conditions: • A sequence is done and FRAMECNT = 1. • A sequence loop is done and FRAMECNT > 1. • The STOSEQ bit is set, FRAMECNT = 0, and the current sequence is done. • The STOSEQ bit is set, FRAMECNT > 1, and the current sequence is done. • The STO bit is set and the current byte transaction is done. This bit cannot be set if the CHEN bit is 0.
			0*	When the STA bit is reset, no START condition will be generated.
5	STO	R/W		The STOP flag.
			1	When the STO bit is set while the channel is active, a STOP condition will be transmitted immediately following the current data or slave address byte being transferred on the I^{2}C-bus. If a read is in progress, a NACK will be generated before the STOP. No further buffered transactions will be carried out and the channel will return to the idle state. Normal error reporting will occur up until the last bit. When a STOP condition is detected on the bus, the hardware clears the STO flag.
			0*	When the STO bit is reset, no action will be taken.

Table 5. CONTROL - Control register bit description *…continued*
Address: Channel 0 = C0h.
*Legend: * reset value*

Bit	Symbol	Access	Value	Description
4	TP	R/W		Trigger polarity bit. Cannot be changed while channel is active.
			1	Trigger will be detected on a falling edge.
			0*	Trigger will be detected on a rising edge.
3	TE	R/W		Trigger Enable (TE) bit controls the trigger input used for frame refresh. TE cannot be changed while channel is active. When the trigger input is enabled, the trigger will override the contents of the FRAMECNT register and will start triggering when STA bit is set. Thereafter, when a trigger tick is detected, the controller will issue a START command and the stored sequence will be transferred on the serial bus.
			1	When TE = 1, the sequence is controlled by the Trigger input.
			0*	When TE = 0, the trigger inputs are ignored.
2	BPTRRST	W	1	Resets auto increment pointers for BYTECOUNT. Reads back as 0.
1	AIPTRRST	W	1	Resets auto increment pointers for SLATABLE and TRANCONFIG. The DATA register auto-increment pointer will be set to the value that corresponds to TRANSEL and TRANOFS registers. Reads back as 0. **Remark:** To reset the data pointer, write 00h to TRANSEL.
0	-	W	0	Reserved. User must write 0 to this bit.

Remark: Due to a small latency between setting the STA bit and the ability to detect a trigger pulse, if the STA bit is set simultaneously to an incoming trigger pulse, the pulse will be ignored and the controller will wait for the next trigger to send the START.

If the STO or STOSEQ bit are set at anytime while the STA bit is 0, then no action will be taken and the write to these bits is ignored.

Remark: STO has priority over STOSEQ.

Table 6. CONTROL register bits STA, STO, STOSEQ operation/behavior

Channel state (initialization steps)	Next write action by host					Results
	FRAMECNT	TE	STA	STO	STOSEQ	
Idle (reset, TRANCONFIG, SLATABLE, DATA, STA = 0)	1	0	0	X	X	No action.
	1	0	1	X	X	START transmitted on serial bus followed by sequence stored in buffer.
Active (reset, load TRANCONFIG, SLATABLE, DATA, STA = 1	1	0	X	0	X	No change; cannot write STA while active.
	1	0	X	1	X	When the STO bit is set, two actions are possible: 1. If the transaction is a read, a STOP is sent after the first read byte (NACK sent) and the byte count is updated. 2. If the transaction is a write, a STOP is sent after the end of ACK cycle of the current byte and BYTECNT is updated. The SD bits will be set.
REFRATE Loop idle (reset, load TRANCONFIG, SLATABLE, DATA STA = 1)[1]	≠ 1	0	0	X	X	No action.
	≠ 1	0	X	0	1	Channel will go immediately to the inactive state and SD and FLD bits will be set.[2]
	≠ 1	0	X	1	X	Channel will go immediately to the inactive state and SD and FLD bits will be set.[2]
REFRATE Loop active (reset, load, TRANCONFIG, SLATABLE, DATA, STA = 1)	≠ 1	0	X	0	0	No action.
	≠ 1	0	X	0	1	STOP at end of current frame. The SD and FLD bits will be set.
	≠ 1	0	X	1	X	When the STO bit is set, two actions are possible: 1. If the transaction is a read, a STOP is sent after the first read byte (NACK sent) and the byte count is updated. 2. If the transaction is a write, a STOP is sent after the end of ACK cycle of the current byte and BYTECNT is updated. The SD and FLD bits will be set.
Trigger Loop Idle (reset, load TRANCONFIG, SLATABLE, DATA, STA = 1)	X	1	0	X	X	No action.
	X	1	X	0	1	STOP at end of current frame. The SD and FLD bits will be set.
	X	1	X	1	X	When the STO bit is set, two actions are possible: 1. If the transaction is a read, a STOP is sent after the first read byte (NACK sent) and the byte count is updated. 2. If the transaction is a write, a STOP is sent after the end of ACK cycle of the current byte and the BYTECNT is updated. The SD and FLD bits will be set.

Table 6. CONTROL register bits STA, STO, STOSEQ operation/behavior *…continued*

Channel state (initialization steps)	Next write action by host					Results
	FRAMECNT	TE	STA	STO	STOSEQ	
Trigger Loop active (reset, load TRANCONFIG, SLATABLE, DATA, STA = 1)	X	1	X	0	0	No action.
	X	1	X	0	1	Channel will go immediately to the inactive state and SD and FLD bits will be set.[2]
	X	1	X	1	X	Channel will go immediately to the inactive state and SD and FLD bits will be set.[2]

[1] Loop Idle is defined as the time elapsed from a STOP to the START of the next sequence while STA = 1.

[2] Channel Active is defined by the CTRLSTATUS[3] bit.

7.5.1.3 CHSTATUS — Channel status register

CHSTATUS is an 8-bit read-only register that provides status information for the serial channel. Some of these status bits are error codes that cannot be masked ($\overline{\text{NMI}}$) by the INTMSK register and need attention from the host. All these status drive the $\overline{\text{INT}}$ pin active LOW. To clear the individual channel interrupt request, you must read the CHSTATUS register. The BE interrupt is cleared by reading the CTRLSTATUS register.

After the CHSTATUS register is cleared, only new errors or status updates will cause the CHSTATUS bits to be set.

Table 7. CHSTATUS - Channel and buffer status codes register bit description
Address: Channel 0 = C1h.

Bit	Symbol	Description
7	SD	Sequence Done. The sequence loaded in the buffer was sent and STOP issued on the serial bus.
6	FLD	Frame Loop Done. The FRAMECNT value has been reached. A STOP has been issued on the bus.
5	WE	Write Error detected in transaction. An SLA NACK or data NACK was detected in a write transaction of the sequence.
4	RE	Read Error detected in transaction. An SLA NACK was detected in a read transaction of the sequence.
3	DAE	Bus error, SDA stuck LOW.
2	CLE	Bus error, SCL stuck LOW.
1	SSE	Bus error, illegal START or STOP detected.
0	FE	Frame Error detected. The time required to send the sequence exceeds refresh rate programmed to the REFRATE register or the time between trigger ticks.

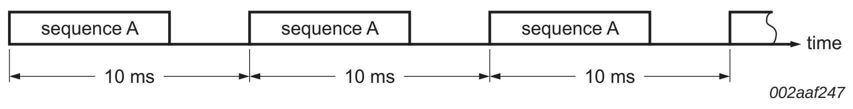
The DAE, CLE and SSE bits correspond to bus error states, and the FE bit corresponds to host programming errors.

DAE - SDA error bit: This bit indicates that the SDA line is stuck LOW when the PCA9661 is trying to send a START condition.

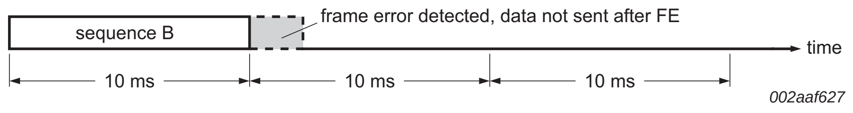
CLE - SCL error bit: This bit indicates that the SCL line is stuck LOW.

SSE - illegal START/STOP detected bit: This bit indicates that a bus error has occurred during a serial transfer. A bus error is caused when a START or STOP condition occurs at an illegal position in the format frame. Examples of such illegal positions are during the serial transfer of an address byte, a data byte, or an acknowledge bit. A bus error may also be caused when external interference disturbs the internal PCA9661 signals.

FE - Frame Error bit: This bit indicates that the time required to send the sequence exceeds the refresh rate programmed in the REFRATE register or the time between trigger ticks. Solving frame errors include programming longer refresh rates, speeding up the bus frequency, shortening the amount of bytes sent/received in the sequence, or increasing the time between trigger ticks. If the frame error is masked by the FEMSK, the device will continue to transmit transactions until the end of the sequence without re-starting the sequence even if new triggers are detected. The total number of sequences transmitted will be the number stored in the FRAMECNT register. Once a complete sequence is transmitted, a new sequence will initiate when a subsequent trigger appears. The FE flag will be held HIGH and sequences will still be transmitted unless CHSTATUS is read. If the frame error is unmasked, the sequence will be aborted at the next logical stopping point (i.e., for a read transaction a NACK will be sent), a STOP transmitted and an interrupt will be generated. Since the controller terminates the sequence in a controlled mechanism, there may be a 2-byte delay if a frame error (FE) is detected during a read transaction. The FE bit is set after the STOP is detected on the bus.

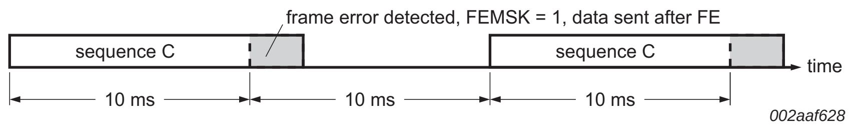


a. Sequence fully executed within the period programmed in REFRATE register



This condition causes a frame error and the FE bit to be set.

b. Sequence exceeds period programmed in REFRATE register, FEMSK = 0



c. Sequence exceeds period programmed in REFRATE register, FEMSK = 1

Fig 3. Frame Error detection

Table 8. Error detection operation/behavior

Channel state	AR (MODE register)	Error detected (CHSTATUS			Next Action
		DAE	CLE	SSE	
Active or idle	X	0	0	1	Interrupt set, if a transaction is active it will be immediately aborted and no further action taken by controller. Host to re-initialize bus (i.e., force a bus recovery), reset slaves, or take other appropriate recovery action. After bus is recovered, host to re-start transaction.
Active or idle, time-out enabled, and clock line is LOW	X	0	1	0	Interrupt set, active transaction will be immediately aborted and no further action taken by controller. No bus recovery possible by bus-controller. Host to recover bus by resetting slaves or system. After bus is recovered, host to re-start transaction.
Active and at a START or repeated-START condition	1	0	0	0	Interrupt not set, active transaction will be immediately aborted and a bus recovery will be attempted by the bus-controller. If successful, a start will be issued automatically and the serial transfer will continue normally at the location of the failed transaction. No host action is required.
	1	1	0	0	Interrupt set, an auto-recovery was attempted and failed. Active transaction will be immediately aborted and the bus-controller determines bus recovery actions, for example setting the BR bit or resetting the slaves.
	0	1	0	0	Interrupt set, active transaction will be immediately aborted and no bus recovery will be attempted by the bus-controller. Host may attempt a bus recovery by setting the BR bit or determine other bus recovery action.

7.5.1.4 INTMSK — Interrupt mask register

Through the INTMSK register, there is the option to manage which states generate an interrupt, allowing more control from the host on the transaction. The interrupt mask applies to all transactions on the channel. A bit set to 1 indicates that the mask is active. The INTMSK register default is all interrupts are un-masked (00h).

Table 9. INTMSK - Interrupt mask register bit description
Address: Channel 0 = C2h.

Bit	Symbol	Description
7	SDMSK	Sequence Done Mask. The end of sequence interrupt will not be generated.
6	FLDMSK	Frame loop done mask. A frame loop done interrupt will not be generated. The controller will enter the idle state.
5	WEMSK	Write Error Mask. An SLA NACK or data NACK interrupt will not be generated and the controller will skip the remaining write data in the transaction and continue with the START of the next transaction in the sequence.
4	REMSK	Read Error detected in transaction. An SLA NACK interrupt will not be generated and the controller will skip the read transaction and continue with the START of the next transaction in the sequence.

Table 9. INTMSK - Interrupt mask register bit description *…continued*
Address: Channel 0 = C2h.

Bit	Symbol	Description
3:1	-	reserved
0	FEMSK	Frame Error Mask. A frame error interrupt will not be generated. **Remark:** Use caution and good judgement when using this mask. Unexpected/erratic behavior may result in the slave devices.

7.5.1.5 SLATABLE — Slave address table register

SLATABLE is an 8-bit × 64 register set that makes up a table that stores the slave address for each transaction in the sequence. The table is loaded by using an auto-increment pointer that is not user-accessible. To reset the pointer, the AIPTRRST bit must be set in the CONTROL register. The slave addresses in the SLATABLE register are stored with a zero-based (N – 1) index. The first slave address occupies the 00h position.

Remark: Slave address entries greater than the transaction count are not part of the sequence. TRANCONFIG[0] contains the transaction count that will be included in the sequence.

Table 10. SLATABLE - Slave address table register bit description
Address: Channel 0 = C3h.

Bit	Symbol	Description
7:1	SLATABLE[7:1]	Slave address.
0	SLATABLE[0]	When 1, a read transaction is requested. When 0, a write transaction is requested.

Table 11. Example of SLATABLE registers

Transaction	Slave address
00h	10h
01h	12h
02h	28h
03h	40h
04h	14h
:	:
3Fh	36h

7.5.1.6 TRANCONFIG — Transaction configuration register

The TRANCONFIG register is an 8-bit × 65 register set that makes up a table that contains the number of transactions that will be executed in a sequence and the number of data bytes involved in the transaction.

The first byte of the register is the Transaction Count register. The remaining 64 registers are the Transaction Length registers.

Table 12. TRANCONFIG, byte 0 - Transaction configuration register bit description
Address: Channel 0 = C4h.

Bit	Symbol	Description
7:0		Number of transactions in the sequence. Maximum is 40h.

Table 13. TRANCONFIG, byte 1 to 40h - Transaction configuration register bit description

Bit	Symbol	Description
7:0		Number of bytes per transaction in the sequence. Maximum is FFh.

Table 14. Example of TRANCONFIG register loaded

Register	Value	Description
Transaction count	10h	16 transactions = 16 slave addresses in the SLATABLE
Transaction length 00h	0Ah	10 byte transaction
Transaction length 01h	12h	18 byte transaction
Transaction length 02h	28h	40 byte transaction
Transaction length 03h	40h	64 byte transaction
:	:	:
Transaction length 3Fh	12h	18 byte transaction

Remark: Even if the Transaction length (TRANCONFIG[1:40h]) and the SLATABLE([0:3Fh]) are fully initialized, only the specified number of transactions in the Transaction count (TRANCONFIG[0]) will be part of the sequence.

If the Transaction count is 0, then there will be no activity on the serial bus if the STA bit is set. In addition, there will be no interrupts generated or status updated. The controller will simply reset the CONTROL.STA bit without performing any transactions.

If the Transaction length is 0, a read transaction will be skipped and a write transaction will send the slave address plus write bit (SLA+W) on the serial bus with no data bytes.

7.5.1.7 DATA — I²C-bus Data register

DATA is an 8-bit read/write, auto-increment register. It is the interface port to the channel buffer. When accessing the buffer, the host writes a byte of serial data to be transmitted or reads bytes that have just been received at this location. The host can read from the DATA at any time and can only write to this 8-bit register while the channel is idle.

Remark: Reading the DATA when the serial interface is active may return outdated or erroneous data.

The host can read or write data up to the amount of memory space allotted to the channel. The location at which the data is accessed is stored in the TRANSEL and TRANOFS register (both default at 00h).

To return to the data location pointed by the contents of the TRANSEL and TRANOFS register after read or write access to the DATA register, set the AIPTRRST (auto-increment pointer reset) bit in the control register.

To return to the first DATA register location in the buffer set the TRANSEL to 00h.

Table 15. DATA - Data register bit description
Address: Channel 0 = C5h.

Bit	Symbol	Description
7:0	D[7:0]	Eight bits to be transmitted or just received. A logic 1 in DATA corresponds to a HIGH level on the I²C-bus. A logic 0 corresponds to a LOW level on the bus.

7.5.1.8 TRANSEL — Transaction data buffer select register

The TRANSEL register is used to select the pointer to a specific transaction in the DATA buffer. This allows the user to update the data of a specific slave without having to re-write the entire data buffer or to read back the stored serial data from a read transaction. The value of this register is the slave address position in the SLATABLE register. The TRANSEL register is zero-based (N – 1) register.

For example, if a change to the 22nd slave address data is required, the host would set the TRANSEL register to 15h. This register can be used in conjunction with the TRANSOFS register to access a specific byte in the data buffer. The host would then proceed to write the new data to the DATA register. The auto-increment feature continues to operate from this new position in the DATA register.

Setting TRANSEL to an uninitialized TRANCONFIG entry may cause a request to read/write data outside the data buffer. If this occurs, the BE bit in the CTRLSTATUS register will be set to a logic 1. Write data will be ignored and read data will be invalid.

When a new transaction is selected by programming the TRANSEL registers, the TRANSOFS register will automatically be reset to 00h.

Remark: When updating the data buffer, if the number of bytes to be updated or read exceeds the number of bytes that were specified in the TRANCONFIG register, the auto-increment will go over the transaction boundary into the next transaction stored in the buffer.

Remark: To reset the DATA pointer, write 00h to the TRANSEL register.

Table 16. TRANSEL - Transaction data buffer select register bit description
Address: Channel 0 = C6h.

Bit	Symbol	Description
7	-	Reserved.
6	-	Reserved.
5:0	TRANSEL[5:0]	Slave address position in the SLATABLE. The maximum number is 3Fh.

7.5.1.9 TRANOFS — Transaction data buffer byte select register

In conjunction with the TRANSEL register, the TRANOFS register is used to select the pointer to a specific byte in a transaction in the data buffer. This allows the user to read or re-write a specific data byte of a specific slave without having to read/re-write the entire data buffer. The TRANOFS register is zero-based (N – 1), so the maximum bytes this register will point to is 256.

For example, if the tenth byte in the 40th slave address data is required, the host would set the TRANSEL register to 27h and the TRANSOFS register to 09h. The host would then proceed with a read to the DATA register.

Setting TRANOFS to a byte offset outside of the data buffer will cause the BE bit in the CTRLSTATUS register will be set to a logic 1. Write data will be ignored and read data will be invalid.

Remark: The number of bytes to be updated or read should not exceed the number of bytes that were specified in the TRANCONFIG register. Doing so will cause the auto-increment to go over the transaction boundary into the next transaction stored in the buffer.

Table 17. TRANOFS - Transaction data buffer byte select register bit description
Address: Channel 0 = C7h.

Bit	Symbol	Description
7:0	TRANOFS[7:0]	Byte index for the specified transaction buffer in TRANSEL.

7.5.1.10 BYTECOUNT — Transmitted and received byte count register

The BYTECOUNT register stores the number of bytes that have been sent or received. The count is continuously updated, therefore the BYTECOUNT is a real time reporting of transmitted and received bytes. This is a read-only register. The BYTECOUNT includes only the bytes that have been ACKed in a write transaction and all bytes received in a read transaction including in transactions where the WEMSK or REMSK are enabled and part or complete transactions have been skipped (see Figure 9). The BYTECOUNT register is cleared at the START of every sequence.

Table 18. BYTECOUNT, byte 0 - Transaction configuration register bit description
Address: Channel 0 = C8h.

Bit	Symbol	Description
7:0	BYTECOUNT[7:0]	Number of bytes sent/received per transaction in the sequence. Maximum is FFh.

7.5.1.11 FRAMECNT — Frame count register

Table 19. FRAMECNT - Frame count register bit description
Address: Channel 0 = C9h.

Bit	Symbol	Description
7:0	FRAMECNT[7:0]	Bit 7 to bit 0 indicate the number of times buffered commands are to be re-transmitted. Default is 01h.

This register is a read/write register. The contents of this register holds the programmed value by the host and is not a real-time count of frames sent on the serial bus.

If the FRAMECNT is 00h, the sequence stored in the buffer will loop continuously. A STOP will be sent at the end of each sequence.

If the FRAMECNT is 01h, it is defined as the default state and the sequence stored in the buffer will be sent once and a STOP will be sent at the end of the sequence.

If the FRAMECNT is greater than 01h, the sequence stored in the buffer will loop FRAMECNT times and a STOP will be sent at the end of each sequence.

Remark: The FRAMECNT can only be set to loop on the sequence stored in the buffer.

7.5.1.12 REFRATE — Refresh rate register

The REFRATE register defines the time period between each sequence start when REFRATE looping is enabled (FRAMECNT $\neq$ 1, and TE = 0).

The refresh period defined by REFRATE should always be programmed to be greater than the time it takes for the sequence to be transferred on the I^{2}C-bus. If the REFRATE values is too small, the frame error (FE) bit will be set and an interrupt will be requested.

Table 20. REFRATE - Refresh rate register bit description
Address: Channel 0 = CAh.

Bit	Symbol	Description
7:0	REFRATE[7:0]	Bit 7 to bit 0 indicate the sequence refresh period. The resolution is 100 μs. The default value is 00h, the timer is disabled, and the sequences will be sent back-to-back if the FRAMECNT is = 0 or FRAMECNT is > 1.

Remark: If the FRAMECNT is 1, then the refresh rate function will be disabled.

7.5.1.13 SCLL, SCLH — Clock rate registers

Table 21. SCLL - Clock Rate Low register bit description (Standard-mode, Fast-mode, Fast-mode Plus)
Address: Channel 0 = CBh.

Bit	Symbol	Description
7:0	L[7:0]	Eight bits defining the LOW state of SCL. Default: 94 (5Eh).

Table 22. SCLH - Clock Rate High register bit description (Standard-mode, Fast-mode, Fast-mode Plus)
Address: Channel 0 = CCh.

Bit	Symbol	Description
7:0	H[7:0]	Eight bits defining the HIGH state of SCL. Default: 63 (3Fh).

The clock rate register for the Standard-mode, Fast-mode, and Fast-mode Plus (Fm+) is controlled by the SCLL and SCLH registers. They define the data rate for the serial bus of the PCA9661. The actual frequency on the serial bus is determined by t_{HIGH} (time where SCL is HIGH), t_{LOW} (time where SCL is LOW), t_r (rise time), and t_f (fall time) values. Writing illegal values into the SCLL and SCLH registers will cause the part to operate at the maximum channel frequency.

For Standard, Fast, and Fast-mode Plus, t_{HIGH} and t_{LOW} are calculated based on the values that are programmed into SCLH and SCLL registers and the PLL clock frequency. t_r and t_f are system/application dependent.

Remark: The MODE register needs to be programmed before programming the SCLL and SCLH registers in order to know which I^{2}C-bus mode is selected. See Section 7.5.1.14 "MODE — I^{2}C-bus mode register" for more detail.

Fast-mode Plus (Fm+) is the default selected mode at power-up or after reset.

The clock is derived from the internal PLL frequency which is set at 156 MHz (13 × OSC clock). Given a 1 % accuracy on the internal clock, the worst case T_{PLL} is

$$\frac{1}{12.12\ MHz \times 13} = \frac{1}{157.56\ MHz} = 6.347\ ns\ .$$

Calculating clock settings for Standard, Fast, and Fast-mode Plus:

$$TOTAL\_SCLLH = \frac{1}{T_{PLL} \times freq} / scale\ factor \qquad (1)$$

The scale factor is set by the MODE register and used in the TOTAL_SCLLH calculation. The scale factor is 8 for Standard-mode, 4 for Fast-mode, and 1 for Fast-mode Plus.

The SCLL and SCLH can be found by:

$$SCLL = 0.6 \times TOTAL\_SCLLH \qquad (2)$$

$$SCLH = 0.4 \times TOTAL\_SCLLH \qquad (3)$$

Remark: The contributions for the rise time (t_r) and fall time (t_f) are adjusted internally by hardware to match the desired frequency. If an invalid number is written to SCLL or SCLH such that it violates the specification, then the controller will adjust the bus frequency to the allowable SCLL and SCLH minimums.

Sample resulting SCL frequencies:

Table 23. SCL calculation scale factor

I²C-bus mode	Frequency	Scale factor
Standard	100 kHz	8
Fast	400 kHz	4
Fast-mode Plus	1000 kHz	1

Table 24. Typical SCL frequencies

Data shown under following conditions:
Pull-up resistor R_{PU} = 500 Ω; bus capacitance C_b = ~170 pF.

Desired frequency (kHz)	Actual frequency (kHz)	SCLL	SCLH
Standard-mode (Sm)			
100	99.3	116	79
90	90.0	129	87
80	80.0	145	98
70	69.5	168	112
60	59.7	194	132
50	50.0	233	156

Table 24. Typical SCL frequencies *…continued*

Data shown under following conditions:

Pull-up resistor R_{PU} = 500 Ω; bus capacitance C_b = ~170 pF.

Desired frequency (kHz)	Actual frequency (kHz)	SCLL	SCLH
Fast-mode (Fm)			
400	398.4	58	39
350	348.7	66	45
300	298.2	78	52
250	250.2	93	62
200	198.0	117	79
150	150.1	155	104
100	100.0	233	156
Fast-mode Plus (Fm+)			
1000	999.0	90	63
900	900.0	100	70
800	798.3	113	79
700	698.5	130	90
600	599.9	152	105
500	499.5	183	126
400	399.7	229	158

Remark: The correct MODE setting should be programmed based on desired frequency since the bus controller will internally select the appropriate t_r and t_f for the selected mode. The minimum I²C-bus frequency is 50 kHz.

Remark: The actual SCL frequency will be affected by the PLL frequency and the bus load. The controller will adjust the SCL timing by monitoring the rise time on the SCL line and bring the output frequency as close to the programmed value as possible without violating the I²C-bus specification for minimum clock HIGH and LOW timing.

7.5.1.14 MODE — I^{2}C-bus mode register

MODE is a read/write register. It contains the control bits that select the bus recovery options, and the correct timing parameters. Timing parameters involved with AC[1:0] are t_{BUF}, $t_{HD;STA}$, $t_{SU;STA}$, $t_{SU;STO}$, t_{HIGH}, t_{LOW}. The auto recovery and bus recovery bits are contained in this register. They control the bus recovery sequence as defined in Section 8.5.1 "I^{2}C-bus obstructed by a LOW level on SDA (DAE)".

Table 25. MODE - I^{2}C-bus mode register bit description
Address: Channel 0 = CDh.

Bit	Symbol	Description
7	CHEN	Channel Enable bit. R/W. 0: Channel is disabled, SCL and SDA high-impedance, USDA and USCL driven HIGH. All registers are accessible for setup and configuration, however a sequence cannot be started if the CHEN bit is 0 (STA cannot be set). 1 (default): Channel is enabled.
6	-	Reserved.
5	BR	Bus Recovery. When BR is set to 1, the bus controller will attempt a bus recovery by sending 9 clock pulses on the bus. Once the bus recovery is complete, the controller will reset the bit to 0. This bit is not intended to generate random or asynchronous 9 clock pulses on the bus. This function is performed automatically when the AR bit is 1.
4	AR	Auto Recovery. When AR = 1 (default), the bus controller will automatically attempt to recover the bus as described in Section 8.5.1 "I^{2}C-bus obstructed by a LOW level on SDA (DAE)". When AR = 0, the bus controller will abort the current transaction and generate an error code by setting the DAE bit in the CHSTATUS register and pulling the $\overline{\text{INT}}$ pin LOW.
3:2	-	Reserved.
1:0	AC[1:0]	I^{2}C-bus mode selection to ensure proper timing parameters (see Table 26 and Table 37). AC[1:0] = 00: Standard-mode AC parameters selected. AC[1:0] = 01: Fast-mode AC parameters selected. AC[1:0] = 10 (default): Fast-mode Plus AC parameters selected. AC[1:0] = 11: Reserved.

Remark: CHEN bit value must be changed only when the I^{2}C-bus is idle.

Remark: Any change in the AC[1:0] bits (Fast-mode to Standard-mode, for example) may cause the HIGH and LOW timings of SCL to be violated. It is then required to program the SCLL and SCLH registers with values in accordance with the selected mode.

Table 26. I2C-bus mode selection example

I2C-bus frequency (kHz)[1]	Scale factor	AC[1:0]	Mode
100	8	00	Standard
400	4	01	Fast
1000	1	10	Fast-mode Plus
-	-	11	reserved

[1] Using the formula $f_{SCL} = \frac{1}{T_{PLL}[(SCLL + SCLH) \times sf] + t_r + t_f}$

7.5.1.15 TIMEOUT — Time-out register

TIMEOUT is an 8-bit read/write register. It is used to determine the maximum time that SCL is allowed to be in a LOW logic state before a CLE interrupt is generated.

When the I2C-bus interface is operating, TIMEOUT is loaded in the time-out counter at every LOW SCL transition.

Table 27. TIMEOUT - Time-out register bit description
Address: Channel 0 = CEh.

Bit	Symbol	Description

7	TE	Time-out enable/disable
		TE = 1: Time-out function enabled
		TE = 0: Time-out function disabled

6:0	TO[6:0]	Time-out value. The time-out period = (TIMEOUT[6:0] + 1) $\times$ 200 μs. The time-out value may vary some, and is an approximate value.

The Time-out register can be used in the following cases:

- When the bus controller wants to send a START condition and the SCL line is held LOW by some other device. Then the bus controller waits a time period equivalent to the time-out value for the SCL to be released. In case it is not released, the bus controller concludes that there is a bus error, sets the CLE bit in the CHSTATUS register, generates an interrupt signal and releases the SCL and SDA lines.
- The time-out feature starts every time the SCL goes LOW. If SCL stays LOW for a time period equal to or greater than the time-out value, the bus controller concludes there is a bus error and behaves in the manner described above. When the I2C-bus interface is operating, TIMEOUT is loaded in the time-out counter at every SCL transition. See Section 8.7 “Global reset” for more information.

7.5.1.16 PRESET — I^{2}C-bus channel parallel software reset register

Table 28. PRESET - I^{2}C-bus channel parallel software reset register bit description
Address: Channel 0 = CFh.

Bit	Symbol	Description
7:0	PRESET[7:0]	Read/Write register used during an I^{2}C-bus channel parallel reset command.

PRESET is an 8-bit write-only register. Programming the PRESET register allows the user to reset the PCA9661 channel under software control. The software reset is achieved by writing two consecutive bytes to this register. The first byte must be A5h while the second byte must be 5Ah. The writes must be consecutive and the values must match A5h and 5Ah. If this sequence is not followed as described, the reset is aborted.

The PRESET resets state-machines, registers, and buffer pointers to the default values, zeroes the TRANCONFIG, SLATABLE, BYTECOUNT, and DATA arrays of the respective channel and will not reset the entire chip. The parallel bus remains active while a software reset is active. The user can read the PRESET register to determine when the reset has completed, PRESET returns all 1s when the reset is active and all 0s when complete.

7.5.2 Global registers

7.5.2.1 CTRLSTATUS — Controller status register

The CTRLSTATUS register reports the status of the controller, including the interrupts generated by the parallel bus. There are three status bits. When CTRLSTATUS contains 00h, it indicates the idle state and therefore no serial interrupts are requested. The content of this register is continuously updated during the operation of the controller.

The lower 3 bits represent the channels that have an interrupt request pending. To clear the individual channel interrupt request, you must read the CHSTATUS register. Bits [5:3] indicate if a channel is currently active or if it is in the idle state.

Table 29. CTRLSTATUS - Interrupt status register bit description
Address: F0h.

Bit	Symbol	Description
7	BE	Buffer Error. A buffer error such as overflow has been detected.
6:4	-	reserved
3	CH0ACT	Channel 0 is active.
2:1	-	reserved
0	CH0INTP	Channel 0 interrupt pending.

Remark: A global reset will reset all channels and configuration settings.

BE - Buffer Error bit: This bit indicates that a buffer error has been detected. For example, a buffer overflow due to the host programming too many bytes will set this bit. A software or hardware reset is necessary to recover from a buffer error.

The buffer error may occur when a data location is being read or written to that has not previously been configured by the TRANCONFIG register. The buffer error can occur on a parallel data write or read beyond the buffer capacity, or setting the TRANSEL and TRANOFS pointers beyond the buffer boundary.

When the DATA register is loaded with data that goes beyond the capacity of the buffer, the bytes that go over the buffer size will be ignored and a Buffer Error (BE) will be generated.

Special case: The BE interrupt is cleared by reading the CTRLSTATUS register. All other interrupts are cleared by reading the respective CHSTATUS register.

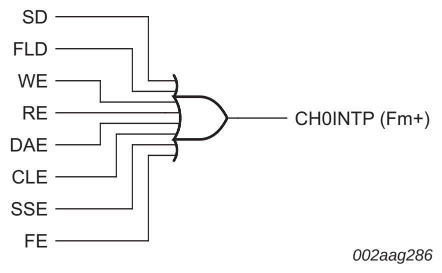


Fig 4. PCA9661 status reporting logic

See Table 7 for channel status.

7.5.2.2 CTRLINTMSK — Control Interrupt mask register

The CTRLINTMSK masks all interrupts generated by the masked channel. This allows the host MCU to complete other operations before servicing the interrupt without being interrupted by the same channel.

Table 30. CTRLINTMSK - Control interrupt mask register bit description

Address: F1h.

Bit	Symbol	Description
7	BEMSK	Buffer Error Mask. A buffer error interrupt will not be generated. **Remark:** Use caution and good judgement when using this mask. Unexpected/erratic behavior may result in the slave devices.
6:1	-	reserved
0	CH0MSK	When this bit is set to 1, all interrupts for the channel will be masked and the $\overline{\text{INT}}$ pin will not be pulled LOW.

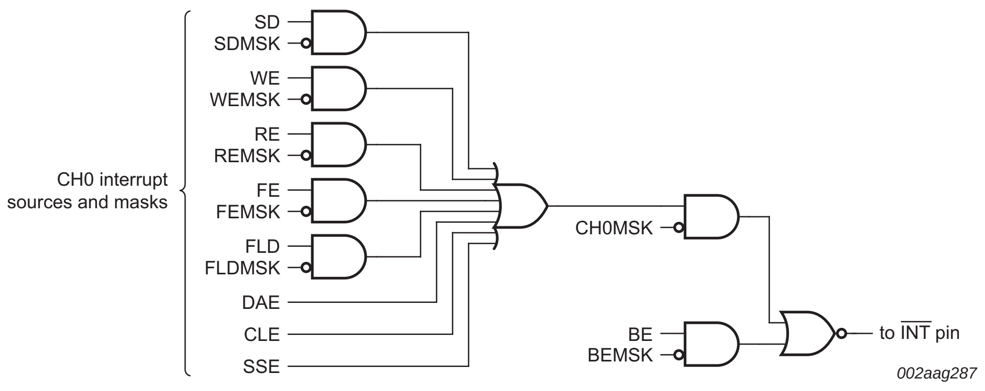


Fig 5. PCA9661 interrupt logic

See Table 9 for interrupt mask.

7.5.2.3 DEVICE_ID — Device ID

The DEVICE_ID register stores the bus controller part number so it can be identified on the parallel bus.

Table 31. DEVICE_ID - Device ID register bit description
Address: F6h.

Bit	Symbol	Description
7	U/A	Selects PCU or PCA device. 1 = PCU96xx 0 = PCA96xx
6:0	BCD	BCD (Binary Coded Decimal) code of the ending 2 digits for ID. Range is 00h to 79h. The code for the PCA9661 is 61h.

7.5.2.4 CTRLPRESET — Parallel software reset register

Table 32. CTRLPRESET - Parallel software reset register bit description
Address: F7h.

Bit	Symbol	Description
7:0	CTRLPRESET[7:0]	Write-only register used during a device parallel reset command.

CTRLPRESET is an 8-bit write-only register. Programming the CTRLPRESET register allows the user to reset the PCA9661 under software control. The software reset is achieved by writing two consecutive bytes to this register. The first byte must be A5h while the second byte must be 5Ah. The writes must be consecutive and the values must match A5h and 5Ah. If this sequence is not followed as described, the reset is aborted.

7.5.2.5 CTRLRDY — Controller ready register

Table 33. CTRLRDY - Controller ready register bit description
Address: FFh.

Bit	Symbol	Description
7:0	CTRLRDY[7:0]	Read-only register indicates the internal state of the controller. FFh indicates the controller is initializing, 00h indicates controller is in normal operating mode.

CTRLRDY (address FFh) is an 8-bit read-only register. It indicates the internal state of the controller. When the register is FFh, the controller is in the initialization state. The initialization state will be entered at power-up, after a hardware reset, or after a global software reset.

The oscillator and the PLL will be initialized only after a Power-On Reset (POR), a hardware reset, or a global software reset (CTRLPRESET).

When the register is 00h, the controller is in the normal operating mode.

Access while the controller is initializing requires $\overline{CE}$ pin follow the $\overline{RD}$ pin transitions to update the state of the controller that is read back. After controller is ready, the $\overline{CE}$ pin can be held LOW while $\overline{RD}$ and $\overline{WR}$ pins transition. See Figure 6, Figure 7 and Figure 8.

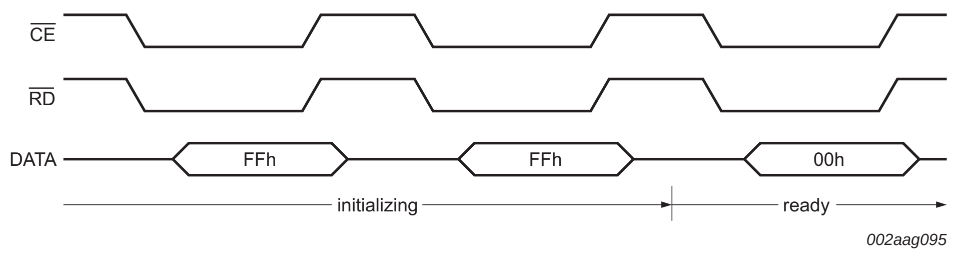


Fig 6. During initialization, $\overline{CE}$ must transition with $\overline{RD}$ at each read operation

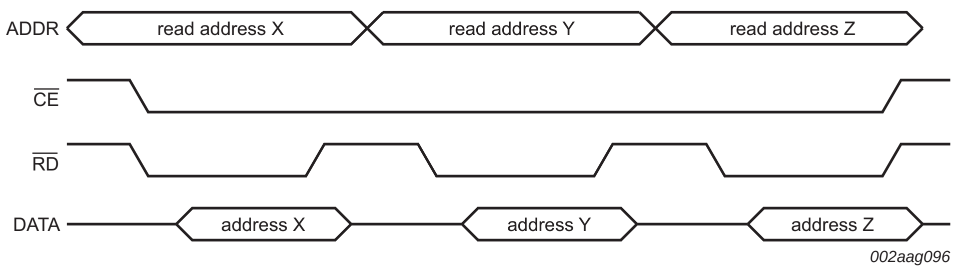


Fig 7. During normal operation, $\overline{CE}$ may remain LOW while $\overline{RD}$ transitions during multiple reads

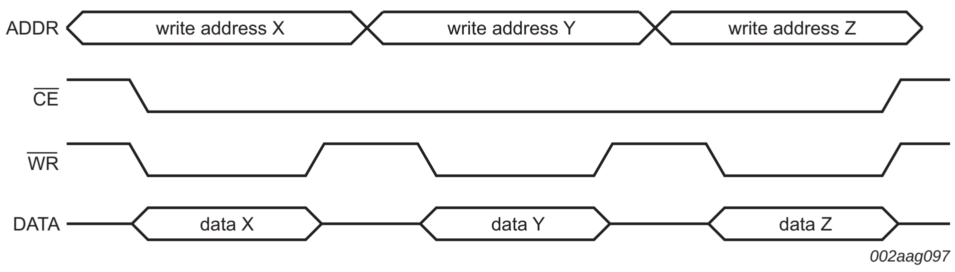


Fig 8. During normal operation, $\overline{CE}$ may remain LOW while $\overline{WR}$ transitions during multiple writes

8. PCA9661 operation

The PCA9661 is designed to efficiently transmit and receive large amounts of data on a single master bus. There are three major components that compose the architecture of the I^{2}C-bus controller that interact with each other to provide a high throughput and a high level of automation when it conducts transactions:

- Slave address table: specifies the address of the slaves on the bus and the direction (read or write).
- Transaction configuration: specifies the size of the transaction.
- Data buffer: contains the data to be transmitted or received from the slave.

These three components are integrated in the PCA9661 to build a sequence. A sequence is a set of read or write transactions and the minimum sequence size is one read or write transaction. Several transactions can be stored in one sequence and be executed without the intervention of the host controller (CPU) through loop control and using the built-in refresh rate timers.

The PCA9661 executes transactions in the order they were loaded into the buffer without interrupting the host. Once the end of a sequence is reached, the Sequence Done (SD) bit will be asserted in the CHSTATUS register and the controller will request an interrupt, if SDMSK = 0. At this point, the host can reload the buffer with a new sequence or resend the one that is currently loaded in the buffer.

When a sequence is in progress, no interrupts are generated unless there is an error when a transaction is conducted. The host will only receive an interrupt when the sequence is done. The PCA9661 will dynamically shift between being a Master Transmitter or a Master Receiver according to the direction bits specified in the SLATABLE. The host has the ability to retrieve stored serial data as soon as a read transaction is done, while the controller carries on the remaining transactions in the sequence.

8.1 Sequence execution

Sequences can have transactions of two types:

- Write transactions, where the PCA9661 will behave as a Master Transmitter
- Read transactions, where the PCA9661 will behave as a Master Receiver

Data transfers in each direction are shown in Figure 9. This figure contains the following abbreviations:

S — START condition

SLA — 7-bit slave address

R — Read bit (HIGH level at SDA)

W — Write bit (LOW level at SDA)

A — Acknowledge bit (LOW level at SDA)

$\overline{\text{A}}$ — Not acknowledge bit (HIGH level at SDA)

Data — 8-bit data byte

P — STOP condition

In Figure 9, circles are used to indicate when a bit is set in the CHSTATUS register. A channel interrupt is not requested when CHSTATUS = 00h and the $\overline{\text{INT}}$ pin is not asserted when the interrupt is masked (see Section 7.5.2.2).

For a successful sequence execution, all three components mentioned above must exist in the memory and must be correctly set up. There are not safeguards against programming incorrect transaction sizes, data buffer lengths, or direction bits. If the transaction length is set to 00h, then only the slave address with direction bit will be transmitted.

Once the host has configured the serial port and programmed the TRANCONFIG (number of slaves and bytes per slave), the SLATABLE (slave addresses), TRANSEL (transaction data buffer selection) and the TRANOFS (byte offset selection) and loaded the serial data into the DATA buffer, the sequence is ready to be transmitted.

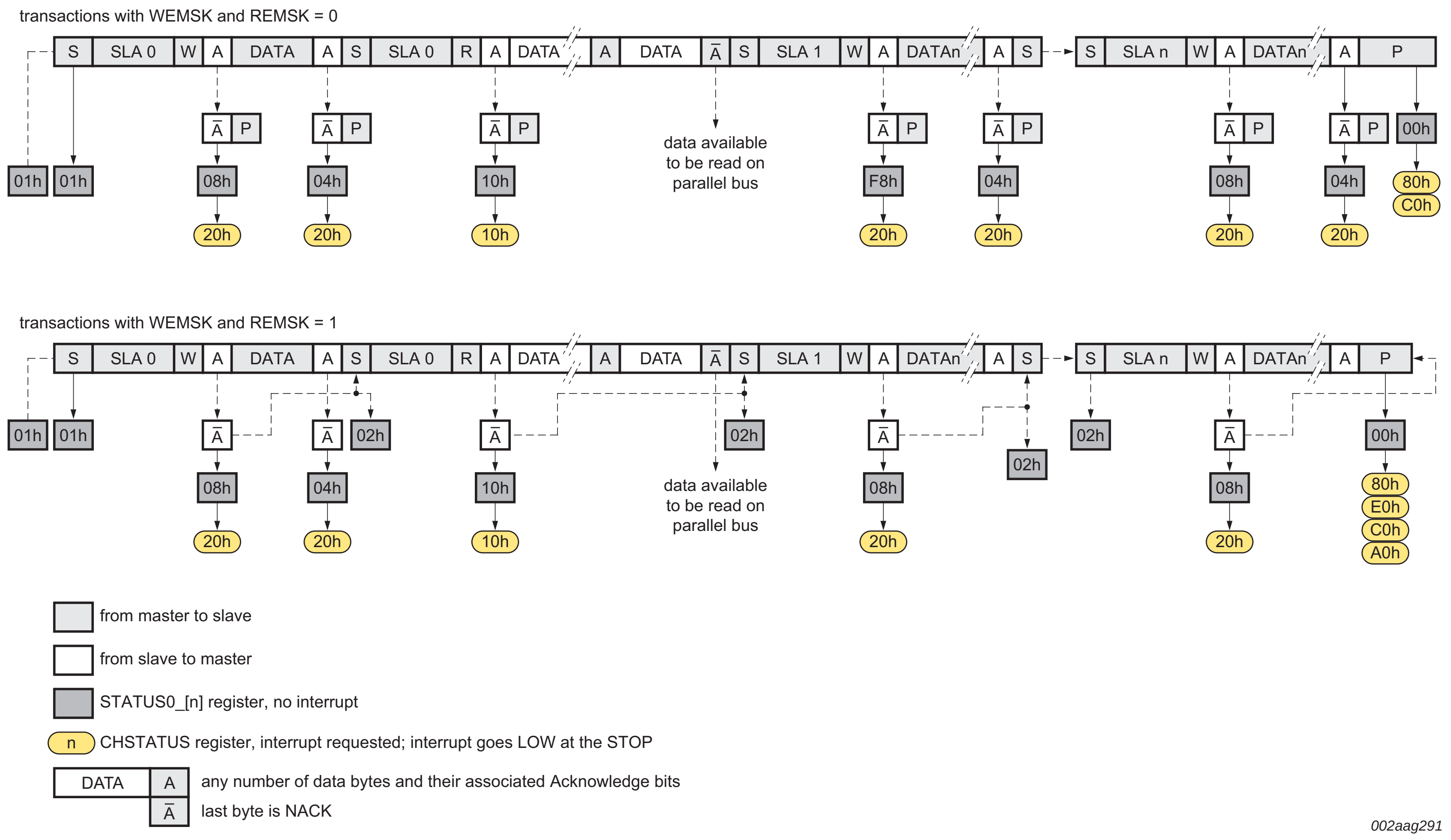
To send the sequence, the host will set the STA bit in the CONTROL register and the controller will immediately send a START on the serial bus. Then, the transactions will be carried out in the order they appear in the SLATABLE, each being separated by a ReSTART command.

If the interrupts are unmasked, the serial transfer will be conducted without generating interrupts in between transactions. Once all transactions are successfully completed, the controller will generate a STOP, the Sequence Done bit (SD) will be set in the CHSTATUS and an interrupt will be generated.

When the interrupts are unmasked, a NACK on slave address or data (in a write cycle) will terminate the serial transfer, generate a STOP, and the $\overline{\text{INT}}$ pin will be asserted. The host can read the CTRLSTATUS (Controller status register) to determine which channel generated the interrupt, then it can read the CHSTATUS register of the channel and the STATUS0_[n] to determine which slave address caused the error.

If the interrupts WEMSK and REMSK are set, then a NACK on slave address or data (in a write cycle) will **not** terminate the serial transfer, the error will be stored in the STATUS0_[n] register and the serial transfer will continue with the next transaction in the sequence. Once all transactions are completed, the controller will generate a STOP and the Sequence Done bit (SD) and other error bits (WE or RE) will be set in the CHSTATUS and an interrupt will be generated.

If the host wants to poll the PCA9661, it can mask all registers including the SD bit and read the CTRLSTATUS, CHSTATUS, STATUS0_[n], and/or the CONTROL registers to determine the state of the controller.


transactions with WEMSK and REMSK = 0
S
SLA 0
W
A
DATA
R
Ā
P
SLA 1
DATAn
SLA n
01h
08h
04h
10h
F8h
00h
20h
80h
C0h
data available to be read on parallel bus
transactions with WEMSK and REMSK = 1
02h
E0h
A0h
from master to slave
from slave to master
STATUS0_[n] register, no interrupt
n
CHSTATUS register, interrupt requested; interrupt goes LOW at the STOP
any number of data bytes and their associated Acknowledge bits
last byte is NACK
002aag291

Example CHSTATUS codes:

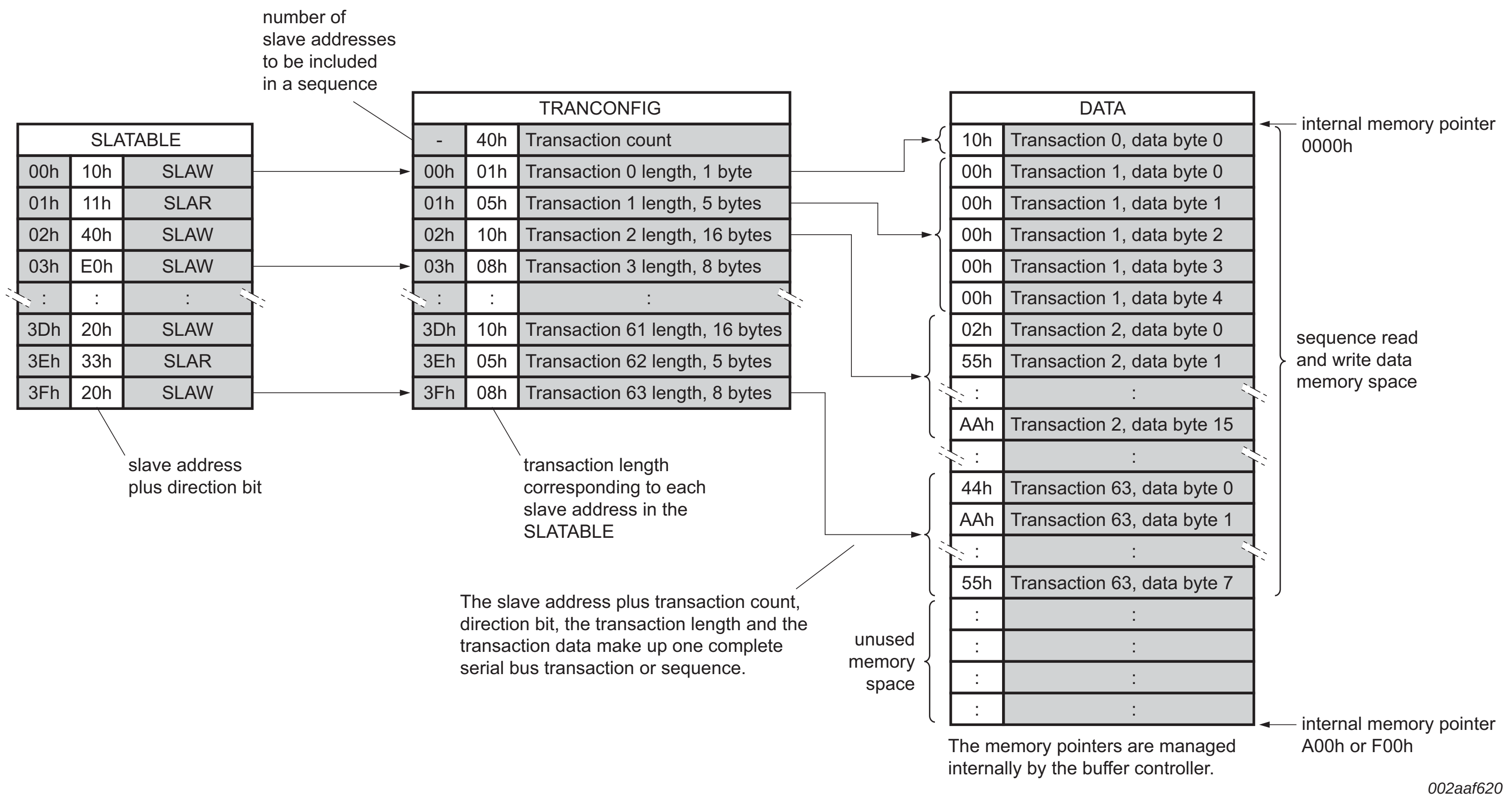
80h: sequence done with no errors

C0h: frame loop and sequence done with no errors

A0h: sequence done with a write error

D0h: frame loop and sequence done with a read error

Fig 9. PCA9661 I^{2}C status codes



SLATABLE		
00h	10h	SLAW
01h	11h	SLAR
02h	40h	SLAW
03h	E0h	SLAW
:	:	:
3Dh	20h	SLAW
3Eh	33h	SLAR
3Fh	20h	SLAW

TRANCONFIG		
-	40h	Transaction count
00h	01h	Transaction 0 length, 1 byte
01h	05h	Transaction 1 length, 5 bytes
02h	10h	Transaction 2 length, 16 bytes
03h	08h	Transaction 3 length, 8 bytes
:	:	:
3Dh	10h	Transaction 61 length, 16 bytes
3Eh	05h	Transaction 62 length, 5 bytes
3Fh	08h	Transaction 63 length, 8 bytes

DATA	
10h	Transaction 0, data byte 0
00h	Transaction 1, data byte 0
00h	Transaction 1, data byte 1
00h	Transaction 1, data byte 2
00h	Transaction 1, data byte 3
00h	Transaction 1, data byte 4
02h	Transaction 2, data byte 0
55h	Transaction 2, data byte 1
:	:
AAh	Transaction 2, data byte 15
:	:
44h	Transaction 63, data byte 0
AAh	Transaction 63, data byte 1
:	:
55h	Transaction 63, data byte 7
:	:
:	:
:	:
:	:

number of slave addresses to be included in a sequence
slave address plus direction bit
transaction length corresponding to each slave address in the SLATABLE
The slave address plus transaction count, direction bit, the transaction length and the transaction data make up one complete serial bus transaction or sequence.
unused memory space
internal memory pointer 0000h
sequence read and write data memory space
internal memory pointer A00h or F00h
The memory pointers are managed internally by the buffer controller.
002aaf620

Status and configuration registers are not shown.

Shaded areas are comments/indexes that are not user-accessible.

Fig 10. PCA9661 sequence block diagram; sample sequence loaded

8.2 Read transactions

Many I^{2}C-bus slave devices need a command or register offset to setup a read operation. In this case, a read transaction is actually a multi-part transaction consisting of a write transaction followed by a read transaction. This is done by setting the transactions in that order when programming the sequence.

If no write is required prior to a read, then the read transaction can be placed in any location of the sequence. Once the read transaction is completed (i.e., the TR bit is cleared to 0) the data is immediately available for the host to retrieve it on the parallel bus.

8.3 Stopping a sequence

If the host needs to stop the execution of a sequence, it should set the STO bit in the CONTROL register. For write transactions, the host will issue a STOP after the acknowledge cycle of the current byte being transferred on the serial bus. For read transactions, if the host sets the STO bit while an address + read bit (SLA+R) is sent, the controller will complete the read of one byte by sending 9 clocks and a NACK on the ninth clock before sending the STOP condition. If the host sets the STO bit while a read transaction is in progress, the current byte will be NACKed before sending a STOP condition. No interrupts will be generated and all the status registers will be up to date. The Sequence Done bit (SD) will be set to indicate to the host that the STOP condition was completed and the bus is idle. The Sequence Done and the Frame Loop Done will be set if the channel is in Loop mode (FRAMECNT $\neq$ 1) and a STO or STOSEQ bit is set.

If the host issues a STOP (by setting the STO) in the middle of a sequence followed by a START (by setting the STA), then the controller will re-send the sequence from the beginning, not from the point where the sequence was last stopped.

8.4 Looping a sequence

A sequence can be set to automatically loop several times using the FRAMECNT and one of the following:

- The REFRATE register. The REFRATE register contains the value of the refresh rate which is timing required between the START of two sequences. The refresh rate is derived from the internal clock of the bus controller. If the REFRATE is programmed to 00h, the sequences will be looped back-to-back.
- Trigger enable (TE) bit. When TE is set, the refresh rate is controlled by the external trigger input and the contents of the REFRATE registers is ignored. There is no maximum timing requirement for the trigger interval.

The FRAMECNT register sets the number of times the sequence will be repeated. A frame is defined as a sequence associated with its respective refresh rate. As described above, the frame refresh rate is determined by the REFRATE register or an external trigger source.

During looping, there is no host intervention required and all status and error reporting remains active. The SD (Sequence Done) bit can be masked to avoid getting interrupted each time a frame is completed while the other error reporting bits remain unmasked. In this manner, normal transactions can run without host intervention and errors will be reported at the STOP of the current byte where the error occurred.

Once the FRAMECNT values is reached, the FLD bit in the CHSTATUS register is set and no further transactions will be executed and the channel will go to the idle state. The FLD interrupt can be masked with the FLDMSK bit in the CTRLINTMSK register. The host can poll the CTRLSTATUS register to check if the channel is active (looping) or if it is idle.

For indefinite or long term looping the host can do the following:

1. A sequence can be set to loop indefinitely by setting the FRAMECNT register to 00h. Each frame will be sent out following the REFRATE settings or the Trigger input if the TE bit is set. To end the Loop mode, the host sets the STO or STOSEQ bits in the CONTROL register.
2. A frame will be sent out continuously and back-to-back if FRAMECNT and REFRATE are set to 00h. To end the Loop mode, the hose sets the STO or STOSEQ bits in the CONTROL register.

8.4.1 Looping with REFRATE control

When using the REFRATE register (TE bit is 0) the refresh rate timing is controlled internally. Once the STA bit is set, the START command will be immediately sent on the serial bus followed by the sequence. Thereafter, the controller will issue a START command followed by the stored sequence every time the REFRATE value is reached. It is important to program enough time in the REFRATE to allow a complete sequence to reach the Sequence Done state. If the refresh rate is not long enough, the Frame Error (FE) bit will be set and an interrupt will be generated. The FE bit is maskable, however, masking the FE bit may yield undesired results on the serial interface. If the FE bit is masked, the Loop mode will continue to operate and the FE flag will remain set. To exit the Loop mode, the STO or the STOSEQ bit should be set.

8.4.2 Looping with Trigger control

The PCA9661 has one trigger input. The trigger enable (TE) bit in the CONTROL register is used to control the use of external triggering. Once enabled, the trigger will override the contents of the REFRATE register, and will start triggering when the STA bit is set. Therefore, a significant time delay can occur between setting the STA bit and the detection of a trigger. When a trigger edge is detected, the controller will issue a START command and the stored sequence will be transferred on the serial bus. The trigger will control the timing of the frame, therefore, enough time should be allowed by the trigger to allow the sequence to reach the Sequence Done state.

If a trigger edge is detected while a sequence is actively being transmitted on the bus, the Frame Error (FE) bit will be set and an interrupt will be generated. The FE bit is maskable, however, masking the FE bit may yield undesired results on the serial interface. If the FE bit is masked, the Loop mode will continue to operate and the FE flag will remain set. The polarity of the trigger edge detect is controlled by the TP bit in the CONTROL register. To exit the Trigger mode, the STO or the STOSEQ bit should be set.

8.5 Bus errors (Fm+ channel only)

Bus errors are a rare occurrence in a well designed I²C-bus system. The PCA9661 has a robust error detection mechanism that detects hang-ups such as if SDA or SCL is pulled LOW by an external source, or if an illegal START or STOP condition appears on the bus.

8.5.1 I²C-bus obstructed by a LOW level on SDA (DAE)

An I²C-bus hang-up occurs if SDA is pulled LOW by an uncontrolled source (e.g., a slave device out of bit synchronization). If the SDA line is obstructed by another device on the bus, the problem can be solved by transmitting additional clock pulses on the SCL line (see Figure 11). The SDA stuck fault detection is only active during a START or repeated-START condition.

When the error is detected, if the auto-recovery bit is set (AR = 1), the PCA9661 sends out nine clock pulses followed by the STOP condition (see Figure 11). If the SDA line is released by the slave pulling it LOW, a normal START condition is transmitted by the PCA9661, the TA bit is set in the STATUS0_[n] register and the serial transfer continues. If the SDA line is not released by the slave pulling it LOW, then the PCA9661 concludes that there is a bus error, sets the DAE bit in the CHSTATUS register, generates an interrupt signal, and releases the SCL and SDA lines.

If the auto-recovery bit is reset (AR = 0) during error detection, the PCA9661 loads the bus error (sets the DAE bit in the CHSTATUS register), generates an interrupt signal, and releases the SCL and SDA lines. After the host reads the status register, it can force a bus recovery sequence by setting the bus recovery bit to 1 (BR = 1). The PCA9661 will transmit additional clock pulses on the SCL line and the host must re-start the transmission by setting the STA bit.

If a repeated START condition is transmitted while SDA is obstructed (pulled LOW), the PCA9661 performs the same action as described above. In each case, the TA bit is set after a successful START condition is transmitted and normal serial transfer continues. Note that the host is not involved in solving these bus hang-up problems when the auto-recovery bit is set (AR = 1).

When a host is unable to recover the bus by having the AR bit set or forcing a bus recovery sequence by setting the bus recovery by setting the BR, then it may be necessary to reset the slaves or the system.

Remark: If the AR bit is set and an SDA stuck LOW is detected, the transaction will continue normally after an auto-recovery from the failed location in the sequence. If the AR bit is zero and a manual bus recovery is performed, the transaction will be re-started from the beginning of the sequence.

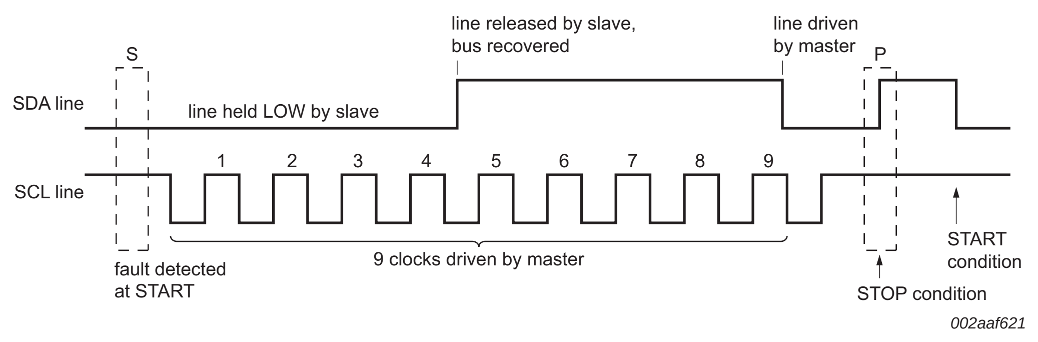


Fig 11. Recovering from a bus obstruction caused by a LOW level on SDA (AR = 1)

8.5.2 I^{2}C-bus obstructed by a LOW level on SCL (CLE)

An I^{2}C-bus hang-up occurs if SDA or SCL is pulled LOW by an uncontrolled source. If the SCL line is obstructed (pulled LOW) by a device on the bus, no further serial transfer is possible, and the PCA9661 cannot resolve this type of problem. When this occurs, the problem must be resolved by the device that is pulling the SCL bus line LOW. To resolve this type of a problem, resetting the slaves or the system may be required.

When the SCL line stays LOW for a period equal to the time-out value, the PCA9661 concludes that this is a bus error and behaves in a manner described in Section 7.5.1.15 "TIMEOUT — Time-out register".

The bus recovery function (setting the BR bit) will not have any effect on an SCL stuck LOW error.

8.5.3 Illegal START or STOP (SSE)

The illegal START or STOP detection is active immediately after the CTRLRDY register is set to 00h at device start-up. The SSE condition will be monitored and detected at any time the bus controller is not the one initiating the transition.

An SSE occurs when a START or STOP condition is present at an illegal position. Examples of illegal positions are during the serial transfer of an address byte, a data or an acknowledge bit.

When an SSE condition is detected, the PCA9661 releases the SDA and SCL lines, sets the interrupt flag, and sets the SSE bit in the channel status register (CHSTATUS).

8.6 Power-on reset

When power is applied to V_{DD}, an internal Power-On Reset holds the PCA9661 in a reset condition until V_{DD} has reached V_{POR}. At this point, the reset condition is released and the PCA9661 goes to the power-up initialization phase where the following operations are performed:

1. The oscillator and PLL will be re-initialized.
2. Internal register initialization is performed.
3. The memory space will be zeroed out.

The complete power-up initialization phase takes t_{rst} to be performed. During this time, writes to the PCA9661 through the parallel port are ignored. However, the parallel port can be read. This allows the device connected to the parallel port of the PCA9661 to poll the CTRLRDY register.

8.7 Global reset

Reset of the PCA9661 to its default state can be performed in 2 different ways:

- By holding the $\overline{\text{RESET}}$ pin LOW for a minimum of $t_{w(rst)}$.
- By using the Parallel Software Reset sequence as described in Figure 12. The host must write to the CTRLPRESET register of the target channel in two successive parallel bus writes to the bus controller. The first byte is A5h and the second byte is 5Ah.

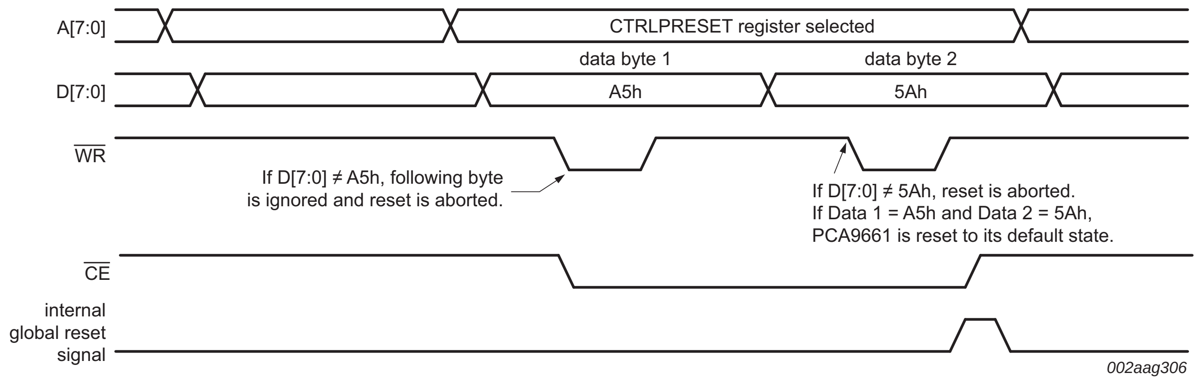

A[7:0]
CTRLPRESET register selected
data byte 1
data byte 2
D[7:0]
A5h
5Ah
$\overline{\text{WR}}$
If D[7:0] ≠ A5h, following byte is ignored and reset is aborted.
If D[7:0] ≠ 5Ah, reset is aborted.
If Data 1 = A5h and Data 2 = 5Ah, PCA9661 is reset to its default state.
$\overline{\text{CE}}$
internal global reset signal
002aag306

Fig 12. Parallel Software Reset sequence

The $\overline{\text{RESET}}$ hardware pin and the global software reset function behave the same as the power-on reset. A complete power-up initialization phase will be performed as defined in Section 8.6. The $\overline{\text{RESET}}$ pin has an internal pull-up resistor (through a series diode) to guarantee proper operation of the device. This pin should not be left floating and should always be driven.

8.8 Channel reset

In addition to the above chip reset options, each channel can be individually reset by programming the PRESET register for that channel as described in Figure 13. The channel will reset to its default power-up state. The host must write to the PRESET register of the target channel in two successive parallel bus writes to the bus controller. The first byte is A5h and the second byte is 5Ah.

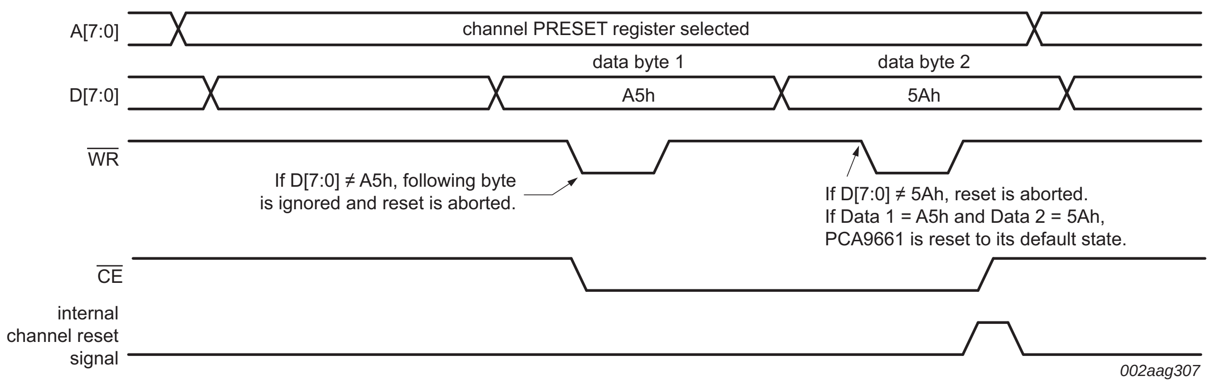
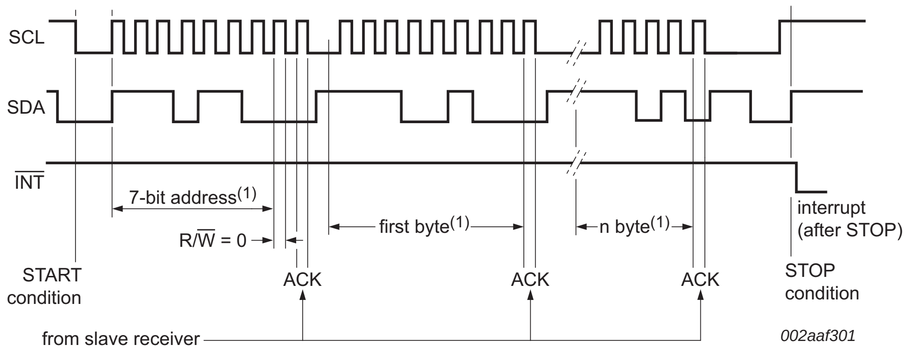

A[7:0]
channel PRESET register selected
D[7:0]
data byte 1
A5h
data byte 2
5Ah
WR
If D[7:0] ≠ A5h, following byte is ignored and reset is aborted.
If D[7:0] ≠ 5Ah, reset is aborted. If Data 1 = A5h and Data 2 = 5Ah, PCA9661 is reset to its default state.
CE
internal channel reset signal
002aag307

Fig 13. I^{2}C-bus Channel Parallel Software Reset sequence

8.9 I^{2}C-bus timing diagrams

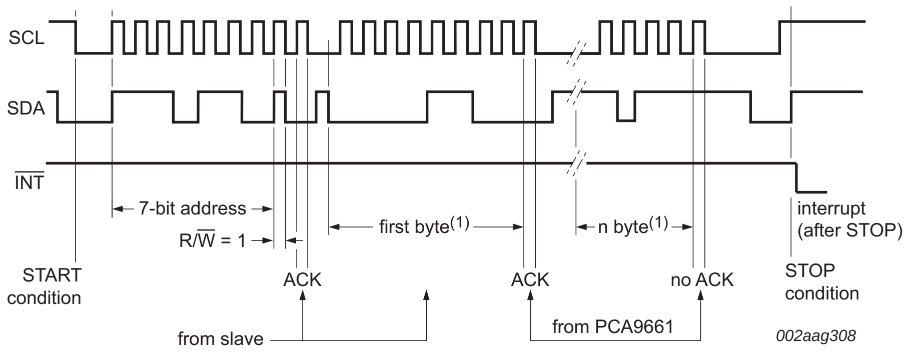
The diagrams Figure 14 and Figure 15 illustrate typical timing diagrams for the PCA9661.



PCA9661 writes data to slave.

(1) 7-bit address + R/W = 0 byte and number of bytes sent = value programmed in Transaction length register in TRANCONFIG register.

Fig 14. Bus timing diagram; write transactions



PCA9661 reads data from slave.

(1) Number of bytes received = value programmed in the Transaction length register in TRANCONFIG.

Fig 15. Bus timing diagram; read transactions

9. Characteristics of the I^{2}C-bus

The I^{2}C-bus is for 2-way, 2-line communication between different ICs or modules. The two lines are a serial data line (SDA) and a serial clock line (SCL). Both lines must be connected to a positive supply via a pull-up resistor when connected to the output stages of a device. Data transfer may be initiated only when the bus is not busy.

9.1 Bit transfer

One data bit is transferred during each clock pulse. The data on the SDA line must remain stable during the HIGH period of the clock pulse as changes in the data line at this time will be interpreted as control signals (see Figure 16).

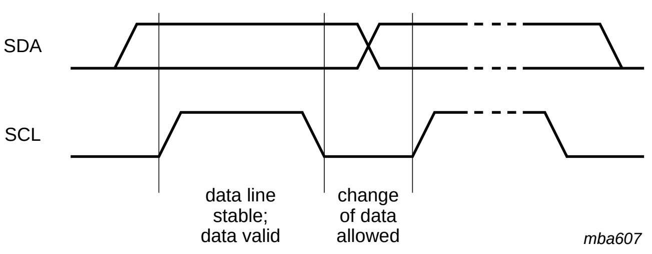


Fig 16. Bit transfer

9.1.1 START and STOP conditions

Both data and clock lines remain HIGH when the bus is not busy. A HIGH-to-LOW transition of the data line while the clock is HIGH is defined as the START condition (S). A LOW-to-HIGH transition of the data line while the clock is HIGH is defined as the STOP condition (P) (see Figure 17).

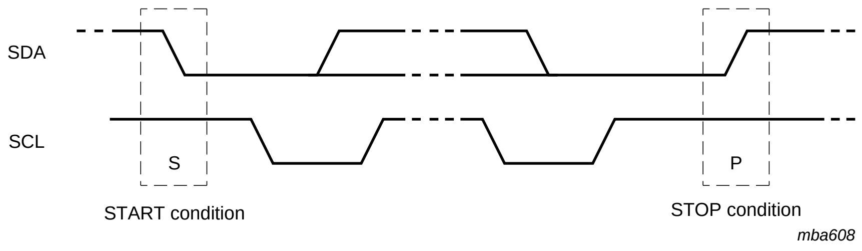


Fig 17. Definition of START and STOP conditions

9.2 System configuration

A device generating a message is a ‘transmitter’; a device receiving is the ‘receiver’. The device that controls the message is the ‘master’ and the devices which are controlled by the master are the ‘slaves’ (see Figure 18).

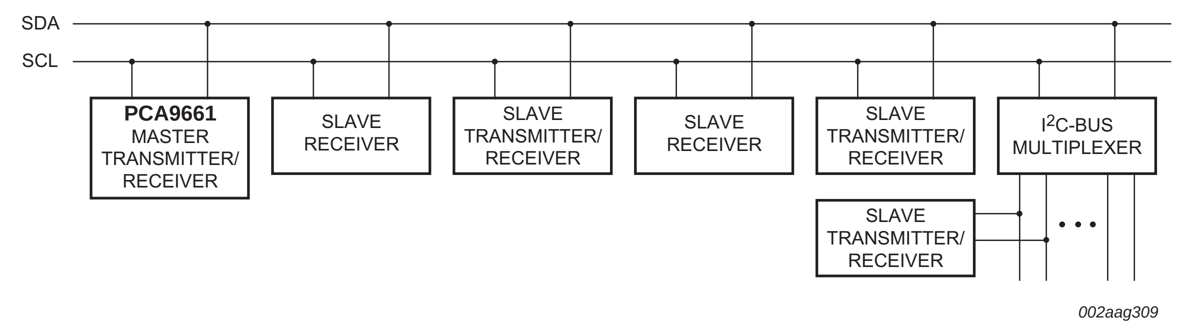


Fig 18. System configuration

9.3 Acknowledge

The number of data bytes transferred between the START and the STOP conditions from transmitter to receiver is not limited. Each byte of eight bits is followed by one acknowledge bit. The acknowledge bit is a HIGH level put on the bus by the transmitter, whereas the master generates an extra acknowledge related clock pulse.

A slave receiver which is addressed must generate an acknowledge after the reception of each byte. Also a master must generate an acknowledge after the reception of each byte that has been clocked out of the slave transmitter. The device that acknowledges has to pull down the SDA line during the acknowledge clock pulse, so that the SDA line is stable LOW during the HIGH period of the acknowledge related clock pulse; set-up and hold times must be taken into account.

A master receiver must signal an end of data to the transmitter by not generating an acknowledge on the last byte that has been clocked out of the slave. In this event, the transmitter must leave the data line HIGH to enable the master to generate a STOP condition.

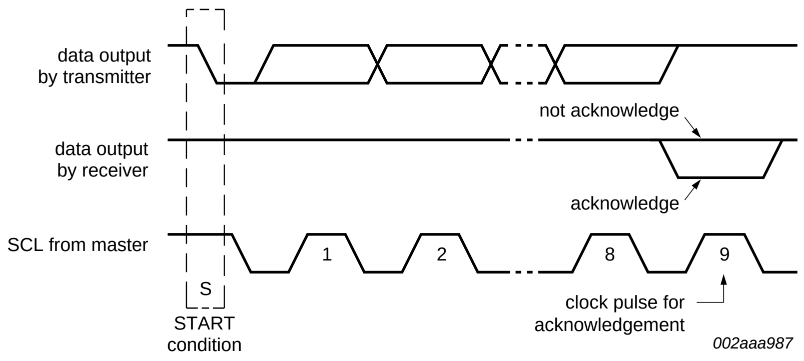


Fig 19. Acknowledgement on the I²C-bus

10. JTAG port

The PCA9661 has a JTAG IEEE 1149.1 compliant port. All signals (TDI, TMS, TCK, $\overline{\text{TRST}}$ and TDO) are accessible. Only EXTEST functions are enabled, for example to conduct board-level continuity tests. Device debug/emulation functionality such as INTEST commands are not supported. The JTAG port is used for boundary scan testing (i.e., opens/shorts) during PCB manufacturing.

The following EXTEST JTAG instructions are supported:

- BYPASS
- EXTEST
- IDCODE
- SAMPLE
- PRELOAD
- CLAMP
- HIGHZ

If the JTAG boundary scan is not being used, then the JTAG pins **must** be held in the following states:

- TDI, TCK, TMS: V_{DD}
- $\overline{\text{TRST}}$: V_{SS}

11. Application design-in information

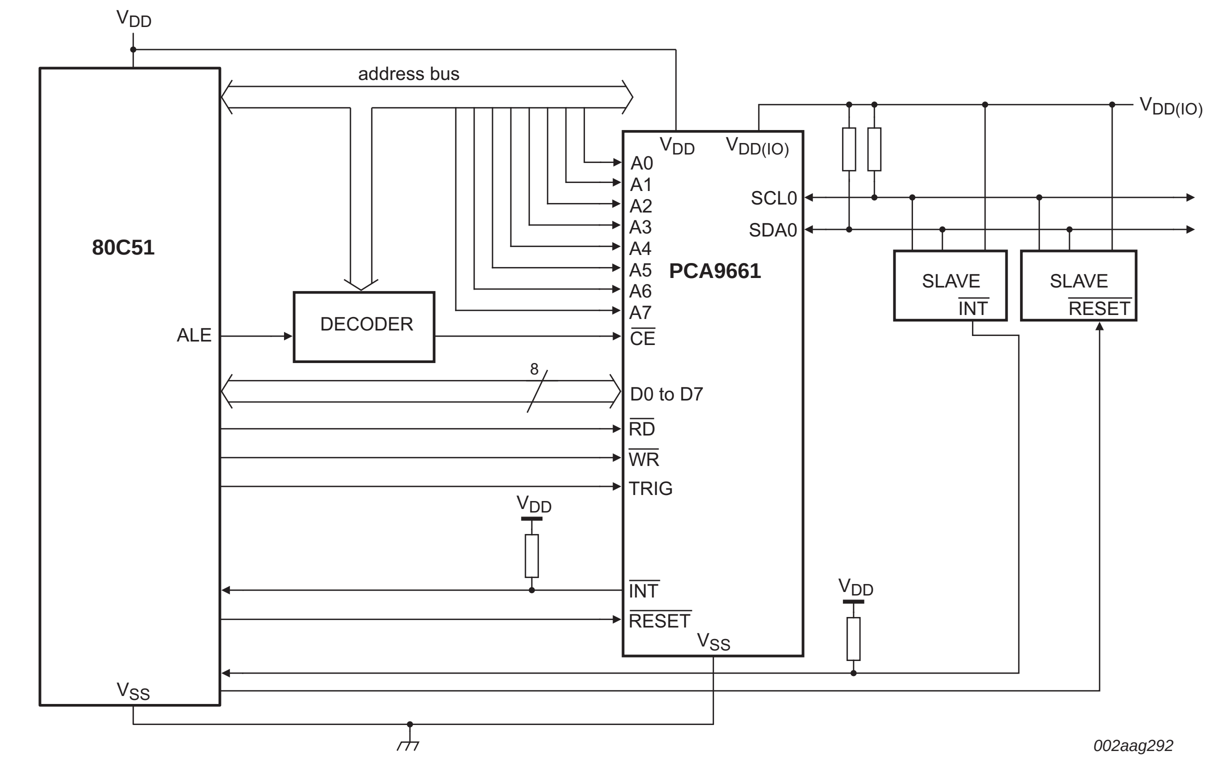


Fig 20. Application diagram using the 80C51

11.1 Specific applications

The PCA9661 is a parallel bus to I²C-bus controller that is designed to allow 'smart' devices to interface with I²C-bus or SMBus components, where the 'smart' device does not have an integrated I²C-bus port and the designer does not want to 'bit-bang' the I²C-bus port. The PCA9661 can also be used to add more I²C-bus ports to 'smart' devices, provide a higher frequency, lower voltage migration path for the PCF8584, PCA9564 and PCA9665 and convert 8 bits of parallel data to a serial bus to avoid running multiple traces across the printed-circuit board.

11.2 Add I²C-bus port

As shown in Figure 21, the PCA9661 converts 8-bits of parallel data into a single master capable I²C-bus port for microcontrollers, microprocessors, custom ASICs, DSPs, etc., that need to interface with I²C-bus or SMBus components.

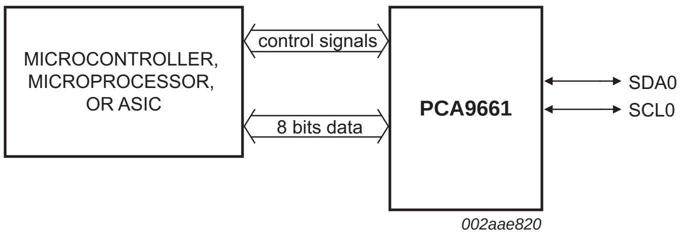


Fig 21. Adding I²C-bus port application

11.3 Add additional I²C-bus ports

The PCA9661 can be used to convert 8-bit parallel data into additional single master capable I²C-bus port as shown in Figure 22. It is used if the microcontroller, microprocessor, custom ASIC, DSP, etc., already have an I²C-bus port but need one or more additional I²C-bus ports to interface with more I²C-bus or SMBus components or components that cannot be located on the same bus (e.g., 100 kHz and 400 kHz slaves on different buses so that each bus can operate at its maximum potential).

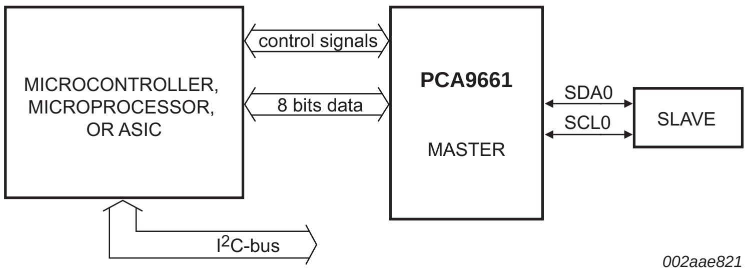


Fig 22. Adding additional I²C-bus ports application

12. Limiting values

Table 34. Limiting values

In accordance with the Absolute Maximum Rating System (IEC 60134).

Symbol	Parameter	Conditions	Min	Max	Unit
V_{DD}	supply voltage		–0.3	+4.6	V
$V_{DD(IO)}$	input/output supply voltage	power supply reference for I^2C-bus I/O pins	–0.3	+7.0	V
V_I	input voltage	parallel bus interface	–0.3	+4.6	V
		I^2C-bus pins [1]	–0.3	+7.0	V
I_I	input current	any input	–10	+10	mA
I_O	output current	any output	–10	+10	mA
I_{OSH}	HIGH-level short-circuit output current	I/O D0 to D7	-	106	mA
I_{OSL}	LOW-level short-circuit output current	I/O D0 to D7	-	110	mA
P_{tot}	total power dissipation		-	300	mW
P/out	power dissipation per output		-	50	mW
T_{stg}	storage temperature		–65	+150	°C
T_{amb}	ambient temperature	operating	–40	+85	°C

[1] 5.5 V steady state voltage tolerance on inputs and outputs is valid only when the supply voltage is present. 4.6 V steady state voltage tolerance on inputs and outputs when no supply voltage is present.

13. Static characteristics

Table 35. Static characteristics

V_{DD} = 3.0 V to 3.6 V; T_{amb} = –40 °C to +85 °C; unless otherwise specified.

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
Supply						
V_{DD}	supply voltage	monotonic supply during power-up and power-down with a ramp time (t_{ramp}): 5 μs < t_r < 20 ms (5 % $V_{DD(min)}$ to 95 % $V_{DD(min)}$)	3.0	-	3.6	V
$V_{DD(PLL)}$	PLL supply voltage	power supply for PLL bias circuit	3.0	-	3.6	V
$V_{DD(IO)}$	input/output supply voltage	power supply reference for I^2C-bus I/O pins	3.0	-	5.5	V
I_{DD}	supply current	operating mode; no load	-	15	25	mA
$I_{DD(IO)}$	input/output supply current	$V_{DD(IO)}$ = 5.5 V; V_{DD} = 3.6 V; I/O not switching	-	-	1	mA
V_{POR}	power-on reset voltage	LOW to HIGH	-	2.75	-	V
		HIGH to LOW	-	2.60	-	V
Inputs $\overline{WR}$, $\overline{RD}$, A0 to A7, $\overline{CE}$, TRIG						
V_{IL}	LOW-level input voltage		0	-	$0.3V_{DD}$	V
V_{IH}	HIGH-level input voltage	[1]	$0.7V_{DD}$	-	3.6	V
V_{hys}	hysteresis voltage		$0.1V_{DD}$	-	-	V
I_L	leakage current	input; V_I = 0 V or 3.6 V	–1	-	+1	μA
C_i	input capacitance	V_I = V_{SS} or V_{DD}	-	2.0	4.5	pF

Table 35. Static characteristics *…continued*
V_{DD} = 3.0 V to 3.6 V; T_{amb} = −40 °C to +85 °C; unless otherwise specified.

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
Input RESET						
V_{IL}	LOW-level input voltage		0	-	$0.3V_{DD}$	V
V_{IH}	HIGH-level input voltage	[1]	$0.7V_{DD}$	-	3.6	V
V_{hys}	hysteresis voltage		$0.1V_{DD}$	-	-	V
I_L	leakage current	input; V_I = 0 V or 3.6 V	−1	-	+75	μA
C_i	input capacitance	$V_I = V_{SS}$ or V_{DD}	-	2.0	5	pF
Inputs/outputs D0 to D7						
V_{IL}	LOW-level input voltage		0	-	$0.3V_{DD}$	V
V_{IH}	HIGH-level input voltage		$0.7V_{DD}$	-	3.6	V
I_{OH}	HIGH-level output current	$V_{OH} = V_{DD(IO)} - 0.4$ V	3.2	-	-	mA
I_{OL}	LOW-level output current	V_{OL} = 0.4 V	2.0	-	-	mA
I_L	leakage current	input; V_I = 0 V or 5.5 V	−1	-	+1	μA
C_{io}	input/output capacitance	$V_I = V_{SS}$ or V_{DD}	-	2.8	4	pF
USDA and USCL						
I_{OL}	LOW-level output current	V_{OL} = 0.4 V	5	-	-	mA
I_{OH}	HIGH-level output current	$V_{OH} = V_{DD(IO)} - 0.4$ V	4.8	-	-	mA
C_{io}	input/output capacitance	$V_I = V_{SS}$ or $V_{DD(IO)}$	-	5.6	7	pF
R_{ON}	ON resistance		-	50	-	Ω
I_L	leakage current	V_{DD} = 3.6 V	−1	-	+1	μA
		V_{DD} = 5.5 V	−10	-	+10	μA
SDA0 and SCL0						
V_{IL}	LOW-level input voltage		0	-	$0.3V_{DD(IO)}$	V
V_{IH}	HIGH-level input voltage	[1]	$0.7V_{DD(IO)}$	-	5.5	V
I_L	leakage current	input/output; V_I = 0 V or 3.6 V	−75	-	+1	μA
		input/output; V_I = 0 V or 5.5 V	−75	-	+1	μA
I_{OL}	LOW-level output current	V_{OL} = 0.4 V	30	-	-	mA
C_{io}	input/output capacitance	$V_I = V_{SS}$ or $V_{DD(IO)}$	-	5.6	7	pF
Output INT						
I_{OL}	LOW-level output current	V_{OL} = 0.4 V	6.0	-	-	mA
I_L	leakage current	V_O = 0 V or 3.6 V	−1	-	+75	μA
C_o	output capacitance	$V_I = V_{SS}$ or V_{DD}	-	3.8	5.5	pF

[1] 5.5 V steady state voltage tolerance on inputs and outputs is valid only when the supply voltage is present. 4.6 V steady state voltage tolerance on inputs and outputs when no supply voltage is present.

14. Dynamic characteristics

Table 36. Dynamic characteristics (3.3 volt)[1][2][3]

V_{DD} = 3.3 V ± 0.3 V; T_{amb} = −40 °C to +85 °C; unless otherwise specified.

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
Initialization timing						
$t_{init(po)}$	power-on initialization time	$V_{DD} \geq 3.0$ V	-	-	650	µs
t_{init}	initialization time	channel initialization time from Channel Software Reset	-	-	70	µs
		controller initialization time from POR, $\overline{\text{RESET}}$, or Global Software Reset inactive	-	-	650	µs
$\overline{\text{RESET}}$ timing						
$t_{w(rst)}$	reset pulse width		4	-	-	µs
t_{rst}	reset time	[4][5]	1.5	-	-	µs
$\overline{\text{INT}}$ timing						
$t_{as(int)}$	interrupt assert time		-	-	500	ns
$t_{das(int)}$	interrupt de-assert time		-	-	100	ns
TRIG timing						
$t_{w(trig)}$	trigger pulse width	HIGH or LOW	100	-	-	ns
Bus timing (see Figure 23 and Figure 25)						
$t_{su(A)}$	address set-up time	to $\overline{\text{RD}}$, $\overline{\text{WR}}$ LOW	0	-	-	ns
$t_{h(A)}$	address hold time	from $\overline{\text{RD}}$, $\overline{\text{WR}}$ LOW	14	-	-	ns
$t_{su(CE\_N)}$	$\overline{\text{CE}}$ set-up time	to $\overline{\text{RD}}$, $\overline{\text{WR}}$ LOW	0	-	-	ns
$t_{h(CE\_N)}$	$\overline{\text{CE}}$ hold time	from $\overline{\text{RD}}$, $\overline{\text{WR}}$ LOW	0	-	-	ns
$t_{w(RDL)}$	$\overline{\text{RD}}$ LOW pulse width		40	-	-	ns
$t_{w(WRL)}$	$\overline{\text{WR}}$ LOW pulse width		40	-	-	ns
$t_{d(DV)}$	data valid delay time	after $\overline{\text{RD}}$ and $\overline{\text{CE}}$ LOW	-	-	45	ns
$t_{d(QZ)}$	data output float delay time	after $\overline{\text{RD}}$ or $\overline{\text{CE}}$ HIGH	-	-	7	ns
$t_{su(Q)}$	data output set-up time	before $\overline{\text{WR}}$ HIGH	5	-	-	ns
$t_{h(Q)}$	data output hold time	after $\overline{\text{WR}}$ HIGH	2	-	-	ns
$t_{w(RDH)}$	$\overline{\text{RD}}$ HIGH pulse width		40	-	-	ns
$t_{w(WRH)}$	$\overline{\text{WR}}$ HIGH pulse width		40	-	-	ns

[1] Parameters are valid over specified temperature and voltage range.

[2] All voltage measurements are referenced to ground (V_{SS}). For testing, all inputs swing between 0 V and 3.0 V with a transition time of 5 ns maximum. All time measurements are referenced at input voltages of 1.5 V and output voltages shown in Figure 23 and Figure 25.

[3] Test conditions for outputs: C_L = 50 pF; R_L = 500 Ω, except open-drain outputs.
Test conditions for open-drain outputs: C_L = 50 pF; R_L = 1 kΩ pull-up to V_{DD}.

[4] Resetting the device while actively communicating on the bus may cause glitches or an errant STOP condition.

[5] Upon reset, the full delay will be the sum of t_{rst} and the RC time constant of the SDA and SCL bus.

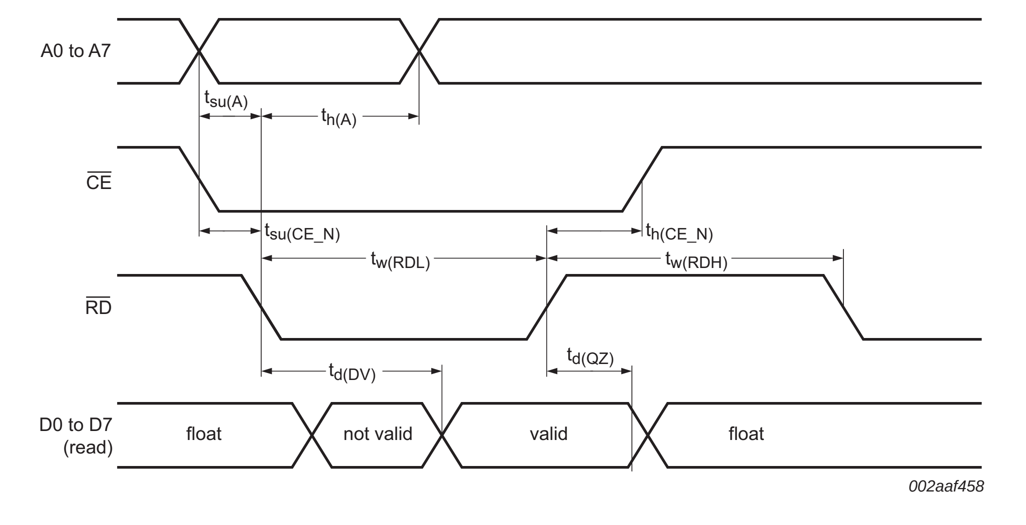


Fig 23. Bus timing (read cycle)

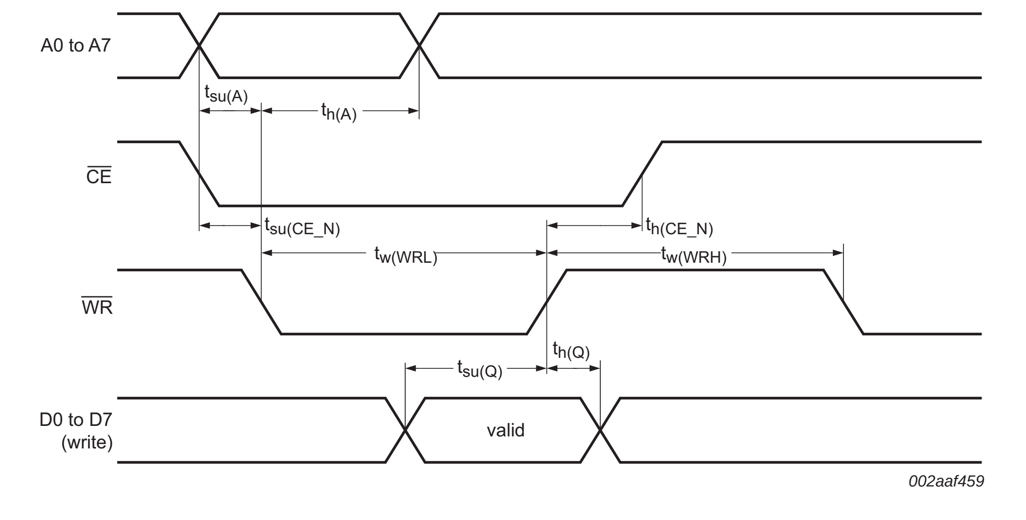
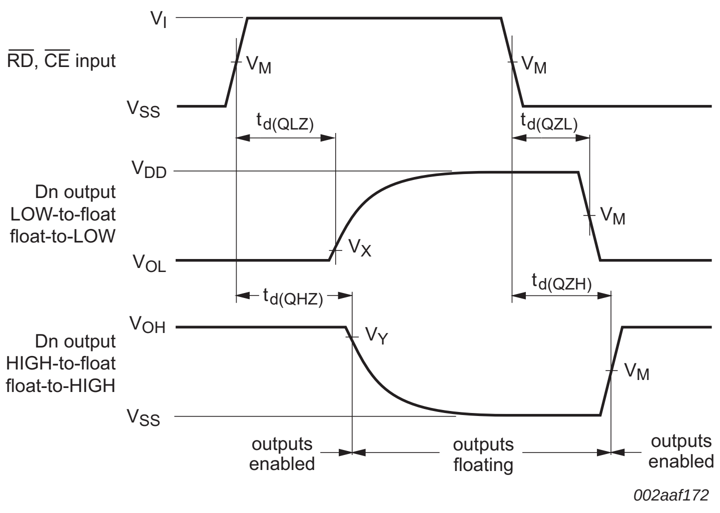


Fig 24. Parallel bus timing (write cycle)



$V_M = 1.5$ V

$V_X = V_{OL} + 0.2$ V

$V_Y = V_{OH} - 0.2$ V

V_{OL} and V_{OH} are typical output voltage drops that occur with the output load.

Fig 25. Data timing

Table 37. I^{2}C-bus frequency and timing specifications

All the timing limits are valid within the operating supply voltage and ambient temperature range; V_{DD} = 2.5 V ± 0.2 V and 3.3 V ± 0.3 V; T_{amb} = −40 °C to +85 °C; and refer to V_{IL} and V_{IH} with an input voltage of V_{SS} to V_{DD}.

Symbol	Parameter	Conditions		Standard-mode I^{2}C-bus		Fast-mode I^{2}C-bus		Fast-mode Plus I^{2}C-bus		Unit
				Min	Max	Min	Max	Min	Max	
f_{SCL}	SCL clock frequency		[1]	0	100	0	400	0	1000	kHz
t_{BUF}	bus free time between a STOP and START condition			4.7	-	1.3	-	0.5	-	µs
$t_{HD;STA}$	hold time (repeated) START condition			4.0	-	0.6	-	0.26	-	µs
$t_{SU;STA}$	set-up time for a repeated START condition			4.7	-	0.6	-	0.26	-	µs
$t_{SU;STO}$	set-up time for STOP condition			4.0	-	0.6	-	0.26	-	µs
$t_{HD;DAT}$	data hold time			0	-	0	-	0	-	ns
$t_{VD;ACK}$	data valid acknowledge time		[2]	0.1	3.45	0.1	0.9	0.1	0.45	µs
$t_{VD;DAT}$	data valid time		[3]	100	-	100	-	100	-	ns
$t_{SU;DAT}$	data set-up time			100	-	100	-	100	-	ns
t_{LOW}	LOW period of the SCL clock			4.7	-	1.3	-	0.5	-	µs
t_{HIGH}	HIGH period of the SCL clock			4.0	-	0.6	-	0.26	-	µs
t_f	fall time of both SDA and SCL signals		[4][5]	-	300	20 + 0.1C_b[6]	300	-	120	ns
t_r	rise time of both SDA and SCL signals			-	1000	20 + 0.1C_b[6]	300	-	120	ns
t_{SP}	pulse width of spikes that must be suppressed by the input filter		[7]	-	50	-	50	-	50	ns

[1] Minimum SCL clock frequency is limited by the bus time-out feature, generates a CLE error if the SCL is held LOW for the TIMEOUT period.

[2] $t_{VD;ACK}$ = time for Acknowledgement signal from SCL LOW to SDA (out) LOW.

[3] $t_{VD;DAT}$ = minimum time for SDA data out to be valid following SCL LOW.

[4] A master device must internally provide a hold time of at least 300 ns for the SDA signal (refer to the V_{IL} of the SCL signal) in order to bridge the undefined region SCL's falling edge.

[5] The maximum t_f for the SDA and SCL bus lines is specified at 300 ns. The maximum fall time for the SDA output stage t_f is specified at 250 ns. This allows series protection resistors to be connected between the SDA and the SCL pins and the SDA/SCL bus lines without exceeding the maximum specified t_f.

[6] C_b = total capacitance of one bus line in pF.

[7] Input filters on the SDA0 and SCL0 inputs suppress noise spikes less than 50 ns.

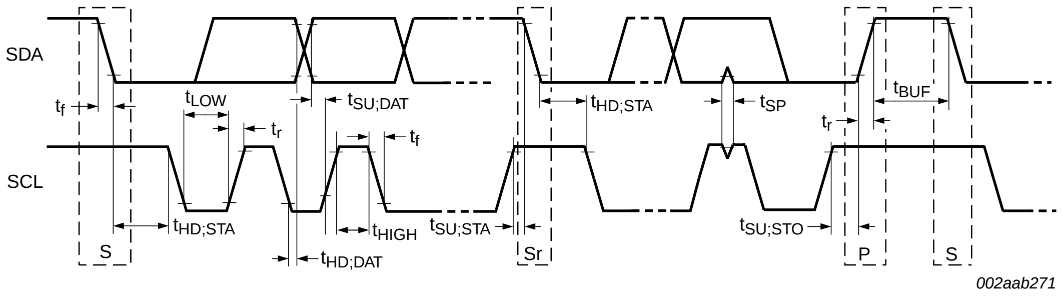
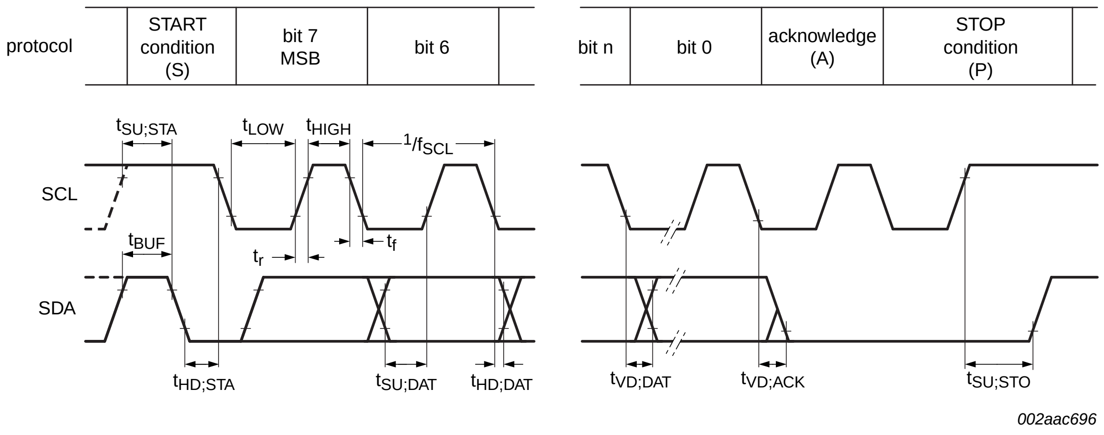


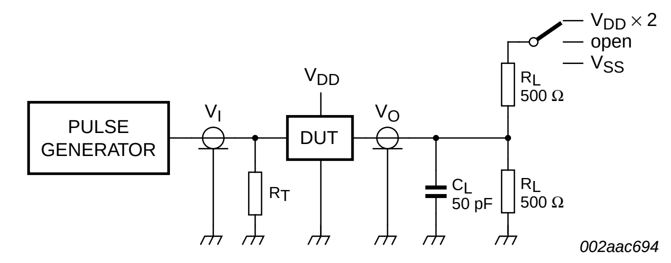
Fig 26. Definition of timing on the I^{2}C-bus



Rise and fall times refer to V_{IL} and V_{IH}.

Fig 27. I^{2}C-bus timing diagram

15. Test information



Test data are given in Table 38.

R_L = load resistance.

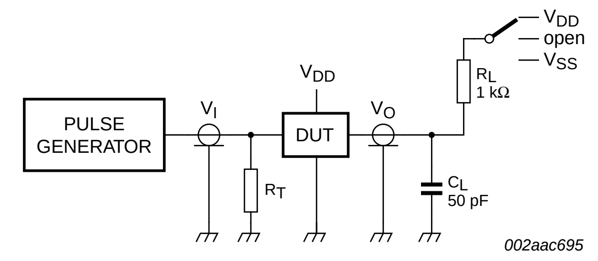
C_L = load capacitance includes jig and probe capacitance.

R_T = termination resistance should be equal to the output impedance Z_O of the pulse generators.

Fig 28. Test circuitry for switching times

Table 38. Test data

Test	Conditions	Load		S1
		C_L	R_L	
$t_{d(DV)}$, $t_{d(QZ)}$	Dn outputs active LOW	50 pF	500 Ω	$V_{DD} \times 2$
	Dn outputs active HIGH	50 pF	500 Ω	open



Test data are given in Table 39.

R_L = load resistance.

C_L = load capacitance includes jig and probe capacitance.

R_T = termination resistance should be equal to the output impedance Z_O of the pulse generators.

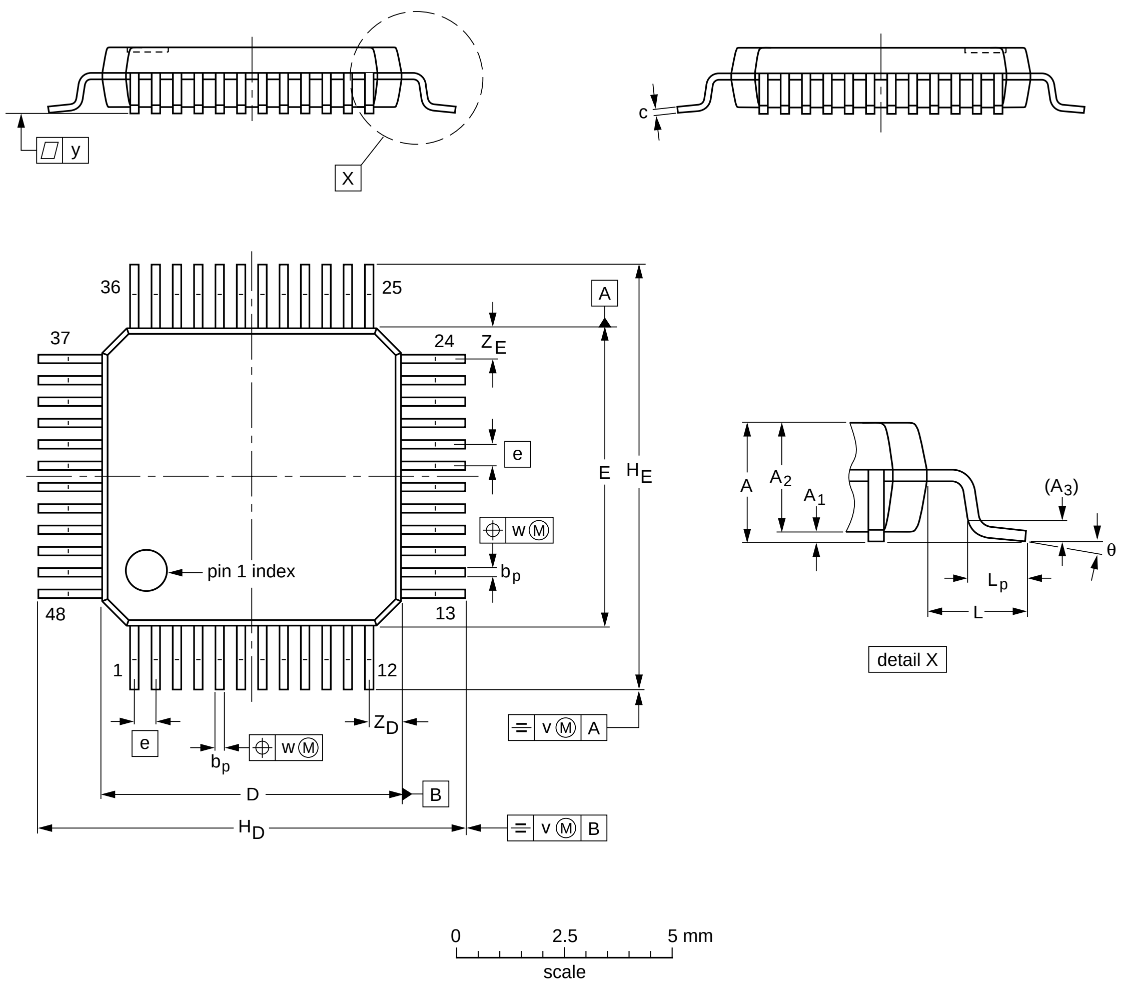
Fig 29. Test circuitry for open-drain switching times

Table 39. Test data

Test	Load		S1
	C_L	R_L	
$t_{as(int)}$	50 pF	1 kΩ	V_{DD}
$t_{das(int)}$	50 pF	1 kΩ	V_{DD}

16. Package outline

LQFP48: plastic low profile quad flat package; 48 leads; body 7 x 7 x 1.4 mm **SOT313-2**



DIMENSIONS (mm are the original dimensions)

UNIT	A max.	A_1	A_2	A_3	b_p	c	$D^{(1)}$	$E^{(1)}$	e	H_D	H_E	L	L_p	v	w	y	$Z_D^{(1)}$	$Z_E^{(1)}$	θ
mm	1.6	0.20 0.05	1.45 1.35	0.25	0.27 0.17	0.18 0.12	7.1 6.9	7.1 6.9	0.5	9.15 8.85	9.15 8.85	1	0.75 0.45	0.2	0.12	0.1	0.95 0.55	0.95 0.55	7^o 0^o

Note

1. Plastic or metal protrusions of 0.25 mm maximum per side are not included.

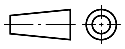
OUTLINE VERSION	REFERENCES				EUROPEAN PROJECTION	ISSUE DATE
	IEC	JEDEC	JEITA			
SOT313-2	136E05	MS-026				~~00-01-19~~ 03-02-25

Fig 30. Package outline SOT313-2 (LQFP48)

17. Handling information

All input and output pins are protected against ElectroStatic Discharge (ESD) under normal handling. When handling ensure that the appropriate precautions are taken as described in *JESD625-A* or equivalent standards.

18. Soldering of SMD packages

This text provides a very brief insight into a complex technology. A more in-depth account of soldering ICs can be found in Application Note *AN10365 “Surface mount reflow soldering description”*.

18.1 Introduction to soldering

Soldering is one of the most common methods through which packages are attached to Printed Circuit Boards (PCBs), to form electrical circuits. The soldered joint provides both the mechanical and the electrical connection. There is no single soldering method that is ideal for all IC packages. Wave soldering is often preferred when through-hole and Surface Mount Devices (SMDs) are mixed on one printed wiring board; however, it is not suitable for fine pitch SMDs. Reflow soldering is ideal for the small pitches and high densities that come with increased miniaturization.

18.2 Wave and reflow soldering

Wave soldering is a joining technology in which the joints are made by solder coming from a standing wave of liquid solder. The wave soldering process is suitable for the following:

- Through-hole components
- Leaded or leadless SMDs, which are glued to the surface of the printed circuit board

Not all SMDs can be wave soldered. Packages with solder balls, and some leadless packages which have solder lands underneath the body, cannot be wave soldered. Also, leaded SMDs with leads having a pitch smaller than ~0.6 mm cannot be wave soldered, due to an increased probability of bridging.

The reflow soldering process involves applying solder paste to a board, followed by component placement and exposure to a temperature profile. Leaded packages, packages with solder balls, and leadless packages are all reflow solderable.

Key characteristics in both wave and reflow soldering are:

- Board specifications, including the board finish, solder masks and vias
- Package footprints, including solder thieves and orientation
- The moisture sensitivity level of the packages
- Package placement
- Inspection and repair
- Lead-free soldering versus SnPb soldering

18.3 Wave soldering

Key characteristics in wave soldering are:

- Process issues, such as application of adhesive and flux, clinching of leads, board transport, the solder wave parameters, and the time during which components are exposed to the wave
- Solder bath specifications, including temperature and impurities

18.4 Reflow soldering

Key characteristics in reflow soldering are:

- Lead-free versus SnPb soldering; note that a lead-free reflow process usually leads to higher minimum peak temperatures (see Figure 31) than a SnPb process, thus reducing the process window
- Solder paste printing issues including smearing, release, and adjusting the process window for a mix of large and small components on one board
- Reflow temperature profile; this profile includes preheat, reflow (in which the board is heated to the peak temperature) and cooling down. It is imperative that the peak temperature is high enough for the solder to make reliable solder joints (a solder paste characteristic). In addition, the peak temperature must be low enough that the packages and/or boards are not damaged. The peak temperature of the package depends on package thickness and volume and is classified in accordance with Table 40 and 41

Table 40. SnPb eutectic process (from J-STD-020C)

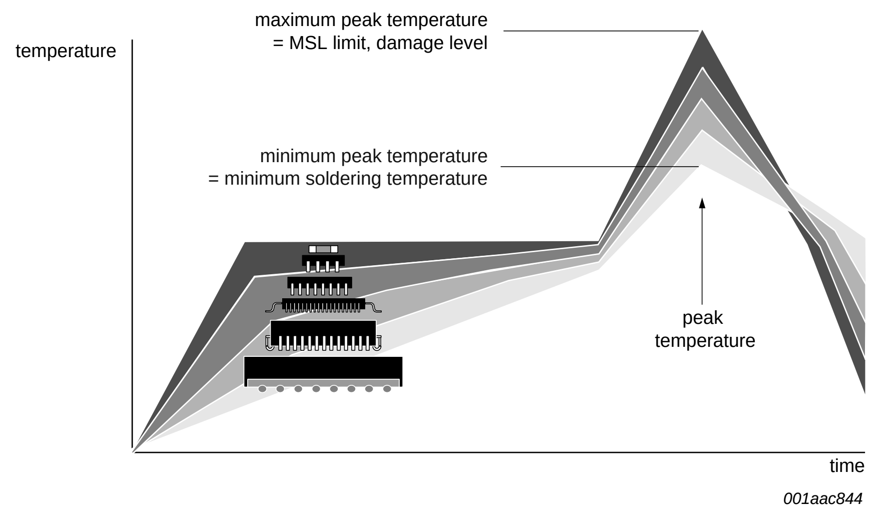
Package thickness (mm)	Package reflow temperature (°C)	
	Volume (mm^3)	
	< 350	≥ 350
< 2.5	235	220
≥ 2.5	220	220

Table 41. Lead-free process (from J-STD-020C)

Package thickness (mm)	Package reflow temperature (°C)		
	Volume (mm^3)		
	< 350	350 to 2000	> 2000
< 1.6	260	260	260
1.6 to 2.5	260	250	245
> 2.5	250	245	245

Moisture sensitivity precautions, as indicated on the packing, must be respected at all times.

Studies have shown that small packages reach higher temperatures during reflow soldering, see Figure 31.



MSL: Moisture Sensitivity Level

Fig 31. Temperature profiles for large and small components

For further information on temperature profiles, refer to Application Note *AN10365 “Surface mount reflow soldering description”*.

19. Abbreviations

Table 42. Abbreviations

Acronym	Description
ASIC	Application Specific Integrated Circuit
CDM	Charged-Device Model
CPU	Central Processing Unit
DSP	Digital Signal Processor
DUT	Device Under Test
ESD	ElectroStatic Discharge
Fm+	Fast-mode Plus
HBM	Human Body Model
I^{2}C-bus	Inter-Integrated Circuit bus
I/O	Input/Output
JTAG	Joint Test Action Group
LED	Light Emitting Diode
PCB	Printed-Circuit Board
PLL	Phase-Locked Loop
SMBus	System Management Bus

20. Revision history

Table 43. Revision history

Document ID	Release date	Data sheet status	Change notice	Supersedes
PCA9661 v.1	20110804	Product data sheet	-	-

21. Legal information

21.1 Data sheet status

Document status[1][2]	Product status[3]	Definition
Objective [short] data sheet	Development	This document contains data from the objective specification for product development.
Preliminary [short] data sheet	Qualification	This document contains data from the preliminary specification.
Product [short] data sheet	Production	This document contains the product specification.

[1] Please consult the most recently issued document before initiating or completing a design.

[2] The term ‘short data sheet’ is explained in section “Definitions”.

[3] The product status of device(s) described in this document may have changed since this document was published and may differ in case of multiple devices. The latest product status information is available on the Internet at URL http://www.nxp.com.

21.2 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.

Short data sheet — A short data sheet is an extract from a full data sheet with the same product type number(s) and title. A short data sheet is intended for quick reference only and should not be relied upon to contain detailed and full information. For detailed and full information see the relevant full data sheet, which is available on request via the local NXP Semiconductors sales office. In case of any inconsistency or conflict with the short data sheet, the full data sheet shall prevail.

Product specification — The information and data provided in a Product data sheet shall define the specification of the product as agreed between NXP Semiconductors and its customer, unless NXP Semiconductors and customer have explicitly agreed otherwise in writing. In no event however, shall an agreement be valid in which the NXP Semiconductors product is deemed to offer functions and qualities beyond those described in the Product data sheet.

21.3 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.

Limiting values — Stress above one or more limiting values (as defined in the Absolute Maximum Ratings System of IEC 60134) will cause permanent damage to the device. Limiting values are stress ratings only and (proper) operation of the device at these or any other conditions above those given in the Recommended operating conditions section (if present) or the Characteristics sections of this document is not warranted. Constant or repeated exposure to limiting values will permanently and irreversibly affect the quality and reliability of the device.

Terms and conditions of commercial sale — NXP Semiconductors products are sold subject to the general terms and conditions of commercial sale, as published at http://www.nxp.com/profile/terms, unless otherwise agreed in a valid written individual agreement. In case an individual agreement is concluded only the terms and conditions of the respective agreement shall apply. NXP Semiconductors hereby expressly objects to applying the customer’s general terms and conditions with regard to the purchase of NXP Semiconductors products by customer.

No offer to sell or license — Nothing in this document may be interpreted or construed as an offer to sell products that is open for acceptance or the grant, conveyance or implication of any license under any copyrights, patents or other industrial or intellectual property rights.

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 national authorities.

21.4 Trademarks

22. Contact information

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

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

23. Contents

continued >>

Date of release: 4 August 2011
Document identifier: PCA9661