Semiconductors

AN469

I²C/SMBus general purpose I/O expanders

Rev. 02 — 20 January 2005 Application note

Document information

Info	Content
Keywords	I2C-bus, I/O expander, GPIO, EEPROM
Abstract	One of the more common functions of the I²C-bus is to provide additional I/O port capability for a controller without increasing the number of pins on the controller package. Philips Semiconductors offers many different versions of I²C I/O port expanders. The following is a selection guide that provides the designer with an overview of the similarities and important differences of selecting the best port expander for the job.

PHILIPS

Revision history

Rev	Date	Description
2	20050120	Application note (9397 750 14523). Modifications: • The format of this application note has been redesigned to comply with the new presentation and information standard of Philips Semiconductors. • Title of application note changed from 'I²C I/O ports' to 'I²C/SMBus general purpose I/O expanders' • Newest I²C devices have been added (PCA9534, PCA9535, PCA9536, PCA9537, PCA9538, PCA9539) • Technical section added: device pinout, architecture overview, programming, $\overline{\text{RESET}}$, $\overline{\text{INT}}$, Frequently Asked Questions.
1	20010815	Application note; initial version.

Contact information

For additional information, please visit: **http://www.semiconductors.philips.com**

For sales office addresses, please send an email to: **sales.addresses@www.semiconductors.philips.com**

1. Introduction

1.1 Family overview

The Philips family of I²C/SMBus General Purpose parallel Input/Output (GPIO) devices provides a simple solution for application requiring more I/Os.

About 20 different devices can be chosen from two sub-families, commonly called ‘quasi bi-directional General Purpose I/Os’ and ‘totem-pole General Purpose I/Os’. Devices can be chosen with 4-bit, 8-bit or 16-bit width. Additional features (not available on all the devices) are active-LOW Interrupt output, active-LOW Reset input, programmable I²C address pins and low power consumption. Finally, some devices come with additional functions (EEPROM, DIP switch) providing integrated and price attractive combination solutions.

Quasi bi-directional GPIOs use a push-pull I/O port with an internal weak current-source pull-up to keep the port HIGH since the upper transistor is on for only $^{1}/_{2}$ clock cycle.

Totem-pole GPIOs use a configuration register that programs an I/O port as either an input or output.

Devices with a Reset input pin ($\overline{\text{RESET}}$) can be set to a known default state by a master device when, for instance, an ‘I²C-bus hung’ situation occurs, thus allowing the master to take control of the bus without having to cycle power to the device.

Devices with an Interrupt output pin ($\overline{\text{INT}}$) are able to provide an ‘input change’ status to a master device anytime an I/O used as an input changes its logic state.

Programmable I²C address pins allow more than one device in the same I²C-bus without any address conflicts.

Low power consumption GPIOs are attractive for portable applications or in general any application where current consumption is a key parameter.

The following is a selection guide that provides the designers and engineers with an overview of the similarities and important differences, allowing them to select the best port expander for the job. Even though important technical details are explained in this application note, the reader is encouraged to thoroughly review the data sheets for specific information on the device.

1.2 Applications

- Keypad and switch control
- ACPI power switch, relays, timer
- LED control
- Signal monitoring
- Sensors, fan control

2. GPIO devices

2.1 Quasi bi-directional GPIOs

Quasi bi-directional GPIO devices have the following common features:

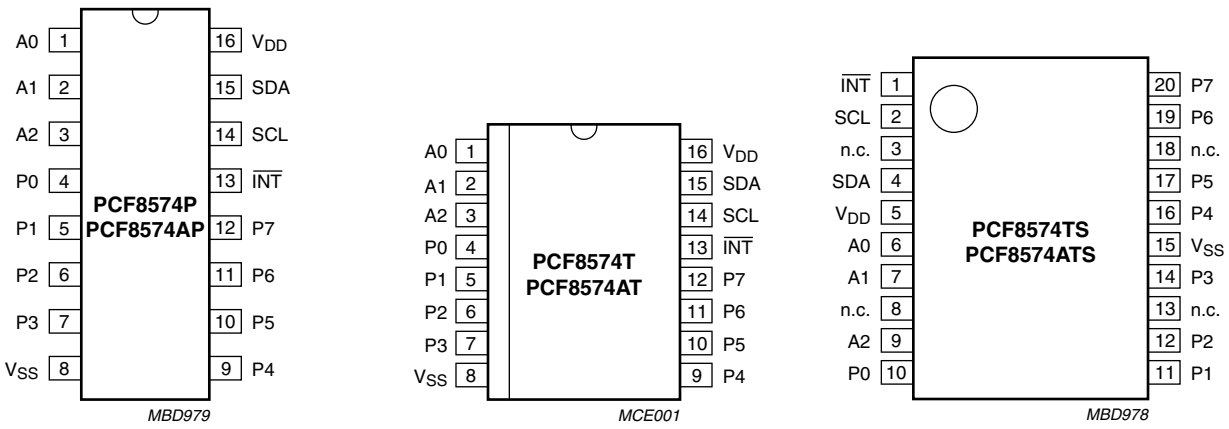
- I/O structure: quasi-bi-directional I/O port with an internal weak current-source pull-up. This architecture allows a good sink current capability (25 mA) but a limited source current capability (100 μA). This simple I/O structure allows the use of the I/O as input or output without the need of an internal configuration register. See Section 3 “Quasi bi-directional GPIO programming” for more details.
- I/O current drive capability:
  - sink capability = 25 mA
  - source capability = 100 μA
- Power-up state: devices power up with I/Os configured as inputs.

2.1.1 PCF8574 / PCF8574A

2.1.1.1 Device characteristics

- 8-bit GPIO
- Operating supply voltage 2.5 V to 6 V
- Support Standard-Mode (100 kHz) and Fast-Mode (400 kHz) I²C standards
- Active-LOW open-drain interrupt output
- 3 programmable I²C address pins allows 8 different addresses
- Maximum stand-by current of 10 μA
- Only difference between PCF8574 and PCF8574A is the fixed portion of the I²C address
- Package offering: DIP16, SO16 and SSOP20

2.1.1.2 Device pinout



a. DIP16 b. SO16 c. SSOP20

Fig 1. PCF8574/PCF8574A pinning

2.1.1.3 Ordering information

Table 1: Ordering information

Package	Container	PCF8574	PCF8574A
DIP16	Tube	PCF8574P	PCF8574AP
SO16	Tube	PCF8574T	PCF8574AT
	T & R	PCF8574T-T	PCF8574AT-T
SSOP20	Tube	PCF8574TS	PCF8574ATS
	T & R	PCF8574TS-T	PCF8574ATS-T

2.1.2 PCF8575 / PCF8575C

2.1.2.1 Device characteristics

- 16-bit GPIO
- Operating supply voltage 2.5 V to 5.5 V
- Support Standard-Mode (100 kHz) and Fast-Mode (400 kHz) I²C standards
- Active-LOW open-drain interrupt output
- 3 programmable I²C address pins allows 8 different addresses
- Maximum stand-by current of 10 μA
- Difference between PCF8575 and PCF8575C are:
  - No internal I/O current source (open-drain)
  - TTL input levels
- Package offering: SSOP24

2.1.2.2 Device pinout

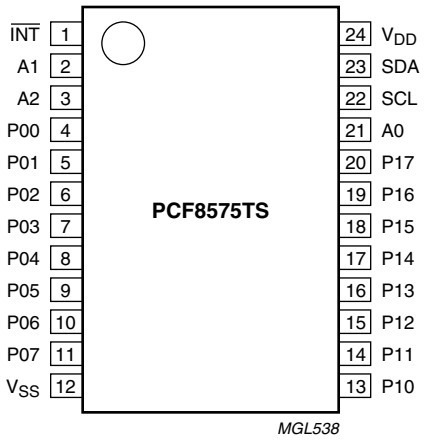


Fig 2. PCF8575/PCF8575C pinning

2.1.2.3 Ordering information

Table 2: Ordering information

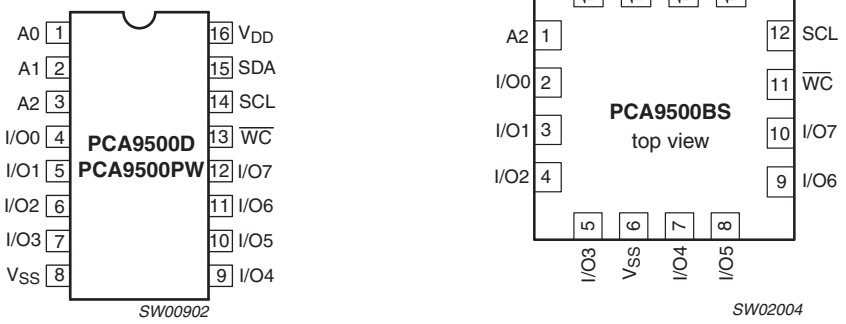
Package	Container	PCF8575	PCF8575C
SSOP24	Tube	PCF8575TS	PCF8575CTS
	T & R	PCF8575TS-T	PCF8575CTS-T

2.1.3 PCA9500

2.1.3.1 Device characteristics

- 8-bit GPIO
- 256 × 8-bit (2-kbit) EEPROM
- Operating supply voltage 2.5 V to 3.6 V
- 5 V tolerant I/Os
- Support Standard-Mode (100 kHz) and Fast-Mode (400 kHz) I²C standards
- Active-LOW Write Control pin
- 3 programmable I²C address pins allows 8 different addresses
- Software compatible with a PCF8574 GPIO + 24C02 EEPROM
- Package offering: SO16 (wide), TSSOP16, HVQFN16

2.1.3.2 Device pinout



a. SO16, TSSOP16 b. HVQFN16

Fig 3. PCA9500 pinning

2.1.3.3 Ordering information

Table 3: Ordering information

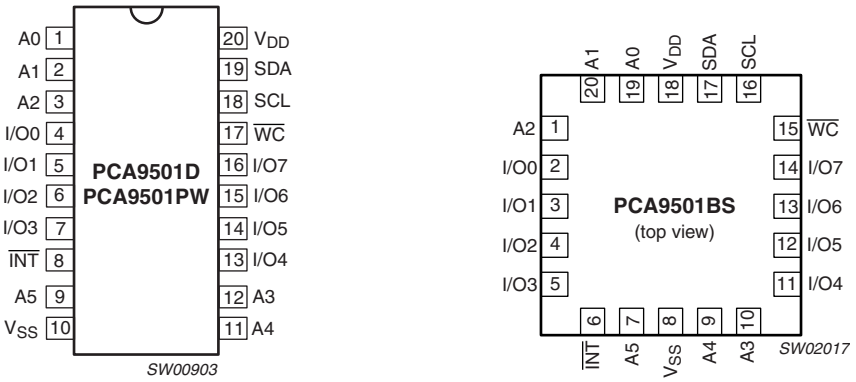
Package	Container	PCA9500
SO16 (wide)	Tube	PCA9500D
	T & R	PCA9500D-T
TSSOP16	Tube	PCA9500PW
	T & R	PCA9500PW-T
HVQFN16	T & R	PCA9500BS-T

2.1.4 PCA9501

2.1.4.1 Device characteristics

- 8-bit GPIO
- 256 × 8-bit (2-kbit) EEPROM
- Operating supply voltage 2.5 V to 3.6 V
- 5 V tolerant I/Os
- Support Standard-Mode (100 kHz) and Fast-Mode (400 kHz) I²C standards
- Active-LOW open-drain interrupt output
- Active-LOW Write Control pin
- Software compatible with a PCF8574 GPIO + 24C02 EEPROM
- 6 programmable I²C address pins allows 64 different addresses
- Package offering: SO20, TSSOP20, HVQFN20

2.1.4.2 Device pinout



a. SO20, TSSOP20 b. HVQFN20

Fig 4. PCA9501 pinning

2.1.4.3 Ordering information

Table 4: Ordering information

Package	Container	PCA9501
SO20	Tube	PCA9501D
	T & R	PCA9501D-T
TSSOP20	Tube	PCA9501PW
	T & R	PCA9501PW-T
HVQFN20	T & R	PCA9501BS-T

2.1.5 PCA9558

2.1.5.1 Device characteristics

- 8-bit GPIO
- 256 × 8-bit (2-kbit) EEPROM
- 5-bit multiplexed / 1-bit latched EEPROM DIP switch
- Operating supply voltage 3.0 V to 3.6 V
- 5 V tolerant I/Os
- Support Standard-Mode (100 kHz) and Fast-Mode (400 kHz) I²C standards
- 1 programmable I²C address pin allows 2 different addresses
- Package offering: TSSOP28

2.1.5.2 Device pinout

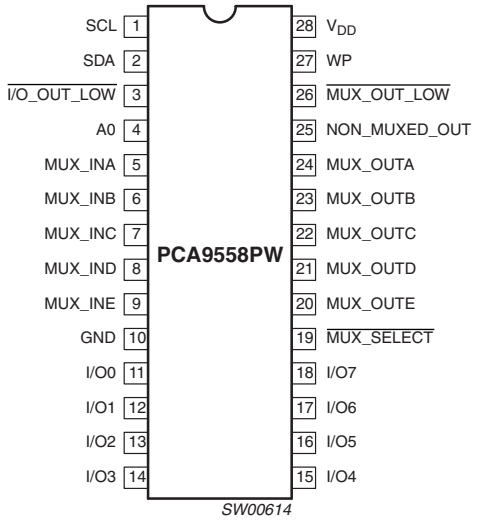


Fig 5. PCA9558 pinning

2.1.5.3 Ordering information

Table 5: Ordering information

Package	Container	PCA9558
TSSOP28	Tube	PCA9558PW
	T & R	PCA9558PW-T

2.2 Totem-pole GPIOs

Quasi bi-directional GPIO devices have the following common features:

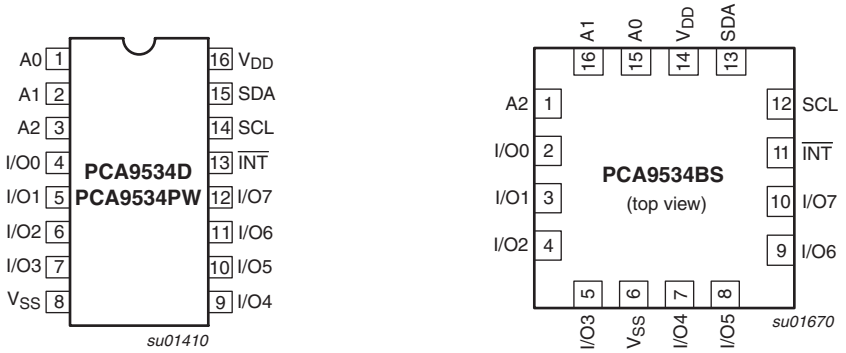
- I/O structure: Totem-pole (push-pull) architecture provides good sinking and sourcing capabilities. I/O configuration (Input or Output) is controlled by a Configuration Register programmable through the I²C-bus. An Output Register programs the pins configured as outputs to be HIGH (‘1’) or LOW (‘0’). A Polarity Inverter Register inverts the polarity of the logic level read in the Input Register. See Section 4 “Totem-pole GPIO programming” for more details.
- I/O current drive capability:
  - sink capability = 25 mA
  - source capability = 10 mA
- Power-up state: devices power up with I/Os configured as inputs
- Internal Power-On Reset (POR)

2.2.1 PCA9534

2.2.1.1 Device characteristics

- 8-bit GPIO
- Operating supply voltage 2.3 V to 5.5 V
- 5 V tolerant I/Os
- Support Standard-Mode (100 kHz) and Fast-Mode (400 kHz) I²C standards
- Active-LOW open-drain interrupt output
- 3 programmable I²C address pins allows 8 different addresses
- Maximum stand-by current of 1 μA
- Package offering: SO16, TSSOP16, HVQFN16

2.2.1.2 Device pinout



a. SO16, TSSOP16 b. HVQFN16

Fig 6. PCA9534 pinning

2.2.1.3 Ordering information

Table 6: Ordering information

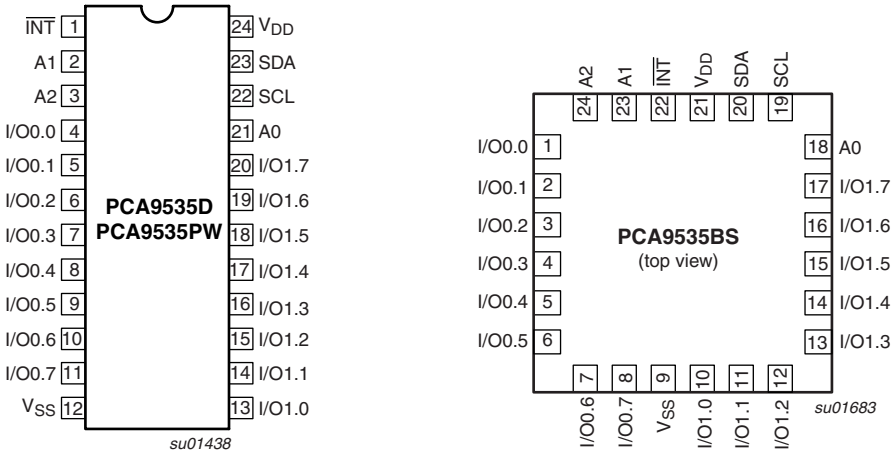
Package	Container	PCA9534
SO16	Tube	PCA9534D
	T & R	PCA9534D-T
TSSOP16	Tube	PCA9534PW
	T & R	PCA9534PW-T
HVQFN16	T & R	PCA9534BS-T

2.2.2 PCA9535

2.2.2.1 Device characteristics

- 16-bit GPIO
- Operating supply voltage 2.3 V to 5.5 V
- 5 V tolerant I/Os
- Support Standard-Mode (100 kHz) and Fast-Mode (400 kHz) I²C standards
- Active-LOW open-drain interrupt output
- 3 programmable I²C address pins allows 8 different addresses
- Maximum stand-by current of 1 μA
- Package offering: SO24, TSSOP24, HVQFN24

2.2.2.2 Device pinout



a. SO24, TSSOP24

b. HVQFN24

Fig 7. PCA9535 pinning

2.2.2.3 Ordering information

Table 7: Ordering information

Package	Container	PCA9535
SO24	Tube	PCA9535D
	T & R	PCA9535D-T
TSSOP24	Tube	PCA9535PW
	T & R	PCA9535PW-T
HVQFN24	T & R	PCA9535BS-T

2.2.3 PCA9536

2.2.3.1 Device characteristics

- 4-bit GPIO
- Operating supply voltage 2.3 V to 5.5 V
- Support Standard-Mode (100 kHz) and Fast-Mode (400 kHz) I²C standards
- Fixed I²C address
- Maximum stand-by current of 350 μA
- Package offering: SO8, TSSOP8

2.2.3.2 Device pinout

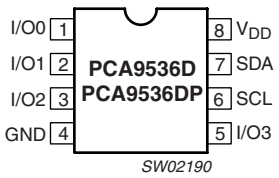


Fig 8. PCA9536 pinning

2.2.3.3 Ordering information

Table 8: Ordering information

Package	Container	PCA9536
SO8	Tube	PCA9536D
	T & R	PCA9536D-T
TSSOP8	T & R	PCA9536DP-T

2.2.4 PCA9537

2.2.4.1 Device characteristics

- 4-bit GPIO
- Operating supply voltage 2.3 V to 5.5 V
- Support Standard-Mode (100 kHz) and Fast-Mode (400 kHz) I²C standards
- Active-LOW open-drain interrupt output
- Active-LOW Reset input
- Fixed I²C address
- Maximum stand-by current of 1 μA
- Package offering: TSSOP10

2.2.4.2 Device pinout

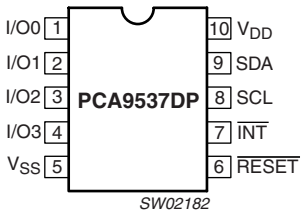


Fig 9. PCA9537 pinning

2.2.4.3 Ordering information

Table 9: Ordering information

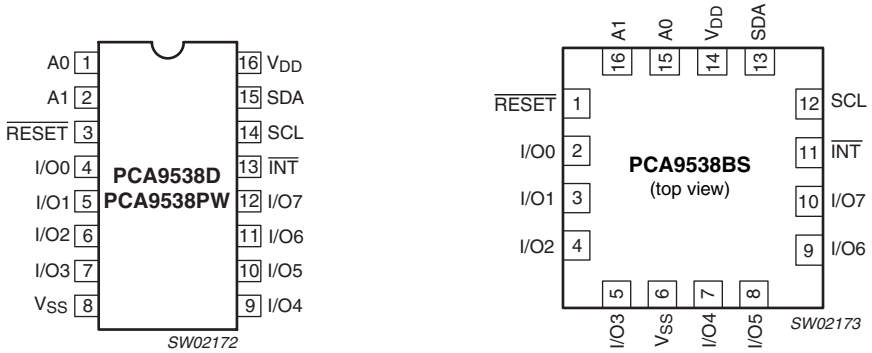
Package	Container	PCA9537
TSSOP10	T & R	PCA9537DP-T

2.2.5 PCA9538

2.2.5.1 Device characteristics

- 8-bit GPIO
- Operating supply voltage 2.3 V to 5.5 V
- Support Standard-Mode (100 kHz) and Fast-Mode (400 kHz) I²C standards
- Active-LOW open-drain interrupt output
- Active-LOW Reset input
- 2 programmable I²C address pins allows 4 different addresses
- Maximum stand-by current of 1 μA
- Package offering: SO16, TSSOP16, HVQFN16

2.2.5.2 Device pinout



a. SO16, TSSOP16

b. HVQFN16

Fig 10. PCA9538 pinning

2.2.5.3 Ordering information

Table 10: Ordering information

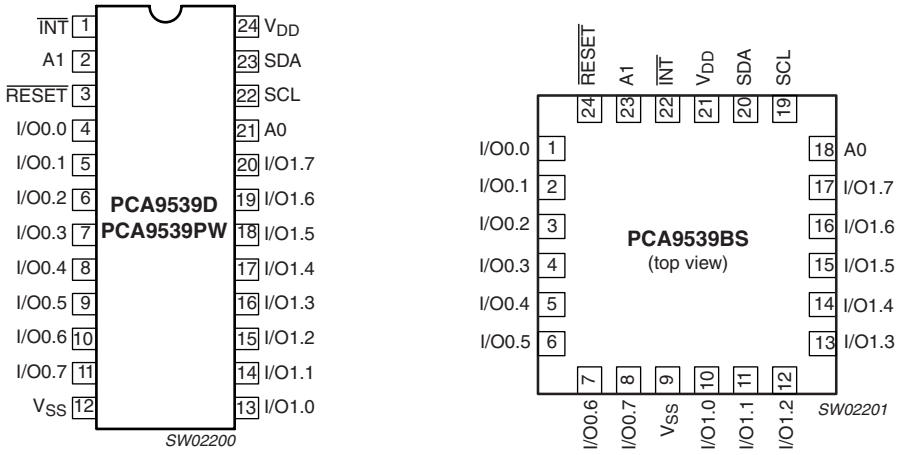
Package	Container	PCA9538
SO16	Tube	PCA9538D
	T & R	PCA9538D-T
TSSOP16	Tube	PCA9538PW
	T & R	PCA9538PW-T
HVQFN16	T & R	PCA9538BS-T

2.2.6 PCA9539

2.2.6.1 Device characteristics

- 16-bit GPIO
- Operating supply voltage 2.3 V to 5.5 V
- Support Standard-Mode (100 kHz) and Fast-Mode (400 kHz) I²C standards
- Active-LOW open-drain interrupt output
- Active-LOW Reset input
- 2 programmable I²C address pins allows 4 different addresses
- Maximum stand-by current of 1 μA
- Package offering: SO24, TSSOP24, HVQFN24

2.2.6.2 Device pinout



a. SO24, TSSOP24

b. HVQFN24

Fig 11. PCA9539 pinning

2.2.6.3 Ordering information

Table 11: Ordering information

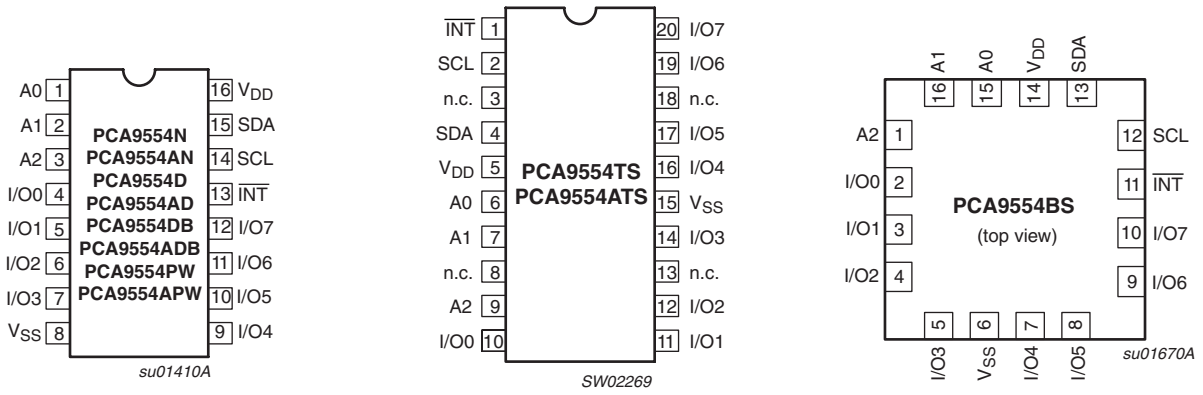
Package	Container	PCA9539
SO24	Tube	PCA9539D
	T & R	PCA9539D-T
TSSOP24	Tube	PCA9539PW
	T & R	PCA9539PW-T
HVQFN24	T & R	PCA9539BS-T

2.2.7 PCA9554 / PCA9554A

2.2.7.1 Device characteristics

- 8-bit GPIO
- Operating supply voltage 2.3 V to 5.5 V
- 5 V tolerant I/Os
- Support Standard-Mode (100 kHz) and Fast-Mode (400 kHz) I²C standards
- Active-LOW open-drain interrupt output
- 3 programmable I²C address pins allows 8 different addresses
- Maximum stand-by current of 700 μA
- Only difference between PCA9554 and PCA9554A is the fixed portion of the I²C address
- Package offering: DIP16, SO16, SSOP16, SSOP20, TSSOP16, HVQFN16

2.2.7.2 Device pinout



a. DIP16, SO16, SSOP16, TSSOP16

b. SSOP20

c. HVQFN16

Fig 12. PCA9554/PCA9554A pinning

2.2.7.3 Ordering information

Table 12: Ordering information

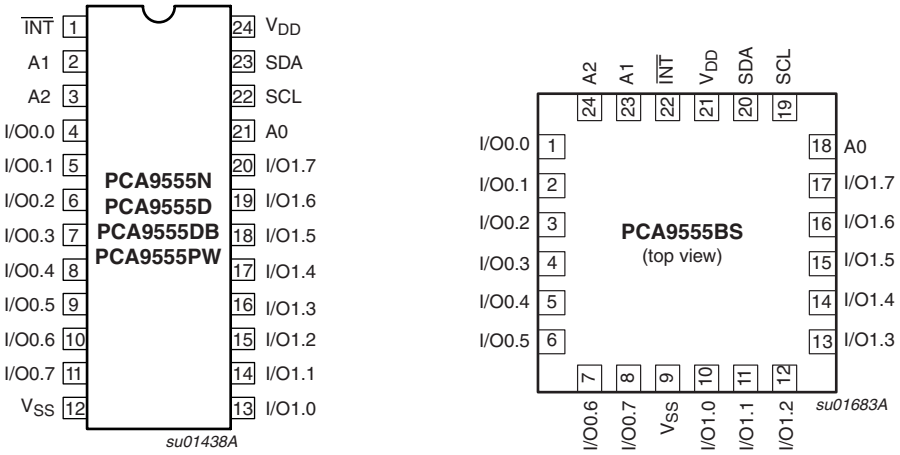
Package	Container	PCA9554	PCA9554A
DIP16	Tube	PCA9554N	PCA9554AN
SO16	Tube	PCA9554D	PCA9554AD
	T & R	PCA9554D-T	PCA9554AD-T
SSOP16	Tube	PCA9554DB	PCA9554ADB
	T & R	PCA9554DB-T	PCA9554ADB-T
SSOP20	Tube	PCA9554TS	PCA9554ATS
	T & R	PCA9554TS-T	PCA9554ATS-T
TSSOP16	Tube	PCA9554PW	PCA9554APW
	T & R	PCA9554PW-T	PCA9554APW-T
HVQFN16	T & R	PCA9554BS-T	PCA9554ABS-T

2.2.8 PCA9555

2.2.8.1 Device characteristics

- 16-bit GPIO
- Operating supply voltage 2.3 V to 5.5 V
- 5 V tolerant I/Os
- Support Standard-Mode (100 kHz) and Fast-Mode (400 kHz) I²C standards
- Active-LOW open-drain interrupt output
- 3 programmable I²C address pins allows 8 different addresses
- Maximum stand-by current of 1.5 mA
- Package offering: DIP24, SO24, SSOP24, TSSOP24, HVQFN24

2.2.8.2 Device pinout



a. DIP24, SO24, SSOP24, TSSOP24 b. HVQFN24

Fig 13. PCA9555 pinning

2.2.8.3 Ordering information

Table 13: Ordering information

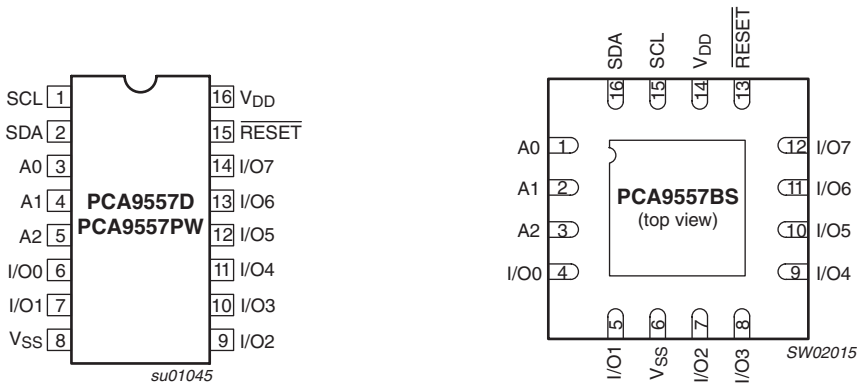
Package	Container	PCA9555
DIP24	Tube	PCA9555N
SO24	Tube	PCA9555D
	T & R	PCA9555D-T
SSOP24	Tube	PCA9555DB
	T & R	PCA9555DB-T
TSSOP24	Tube	PCA9555PW
	T & R	PCA9555PW-T
HVQFN24	Tube	PCA9555BS
	T & R	PCA9555BS-T

2.2.9 PCA9557

2.2.9.1 Device characteristics

- 8-bit GPIO: 7 totem-pole + 1 open-drain
- Operating supply voltage 2.5 V to 6 V
- Support Standard-Mode (100 kHz) and Fast-Mode (400 kHz) I²C standards
- Active-LOW Reset input
- 3 programmable I²C address pins allows 8 different addresses
- Maximum stand-by current of 1 μA
- Package offering: SO16, TSSOP16, HVQFN16

2.2.9.2 Device pinout



a. SO16, TSSOP16

b. HVQFN16

Fig 14. PCA9557 pinning

2.2.9.3 Ordering information

Table 14: Ordering information

Package	Container	PCA9557
SO16	Tube	PCA9557D
	T & R	PCA9557D-T
TSSOP16	Tube	PCA9557PW
	T & R	PCA9557PW-T
HVQFN16	Tube	PCA9557BS
	T & R	PCA9557BS-T

2.3 Quick features guide

2.3.1 Quasi bi-directional GPIOs

Table 15: Quasi bi-directional GPIOs selection table

Device	I/O	Address (#pins)	$\overline{INT}$	$\overline{RESET}$	V_{DD} (V)	Freq. (kHz)	I/O sink bit total (mA)	Source (mA)	Pull-ups	Packages	# pins	Misc.
PCF8574	8	0100xxx (3)	Yes	No	2.5 6.0	0 100	25 100	0.1	Weak	DIP, SO, SSOP	16	5 V tolerant I/Os
PCF8574A	8	0111xxx (3)	Yes	No	2.5 6.0	0 100	25 100	0.1	Weak	DIP, SO, SSOP	16	5 V tolerant I/Os
PCF8575	16	0100xxx (3)	Yes	No	2.5 5.4	0 400	25 100	0.1	Weak	SSOP	24	5 V tolerant I/Os
PCF8575C	16	0100xxx (3)	Yes	No	2.5 5.4	0 400	25 100	0.1	Weak	SSOP	24	5 V tolerant I/Os
PCA9500	8	0100xxx (3)	No	No	2.5 3.6	0 400	25 100	0.1	Weak	SO, TSSOP, HVQFN	16	5 V tolerant I/Os 2K EEPROM
PCA9501	8	0xxxxxx (6)	Yes	No	2.5 3.6	0 400	25 100	0.1	Weak	SO, TSSOP, HVQFN	20	5 V tolerant I/Os 2K EEPROM

2.3.2 Totem-pole GPIOs

Table 16: Totem-pole GPIOs selection table

Device	I/O	Address (#pins)	$\overline{INT}$	$\overline{RESET}$	V_{DD} (V)	Freq. (kHz)	I/O sink bit total (mA)	Source (mA)	Inter. pull- ups (kΩ)	Packages	# pins
PCA9534	8	0100xxx (3)	Yes	No	2.3 5.5	0 400	25 100	10	No	SO, TSSOP, HVQFN	16
PCA9535	16	0100xxx (3)	Yes	No	2.3 5.5	0 400	25 100	10	No	SO, TSSOP, HVQFN	24
PCA9536	4	1000001 (0)	No	No	2.3 5.5	0 400	25 100	10	100	SO, TSSOP	8
PCA9537	4	1001001 (0)	Yes	Yes	2.3 5.5	0 400	25 100	10	No	TSSOP	10
PCA9538	8	11100xx (2)	Yes	Yes	2.3 5.5	0 400	25 100	10	No	SO, TSSOP, HVQFN	16
PCA9539	16	11101xx (2)	Yes	Yes	2.3 5.5	0 400	25 100 / octal	10	No	SO, TSSOP, HVQFN	24
PCA9554	8	0100xxx (3)	Yes	No	2.3 5.5	0 400	25 100	10	100	DIP, SO, SSOP, TSSOP, HVQFN	16
										SSOP	20
PCA9554A	8	0111xx (3)	Yes	No	2.3 5.5	0 400	25 100	10	100	DIP, SO, SSOP, TSSOP, HVQFN	16
										SSOP	20

Table 16: Totem-pole GPIOs selection table *...continued*

Device	I/O	Address (#pins)	$\overline{INT}$	$\overline{RESET}$	V_{DD} (V)	Freq. (kHz)	I/O sink bit total (mA)	Source (mA)	Inter. pull- ups (kΩ)	Packages	# pins
PCA9555	16	0100xxx (3)	Yes	No	2.3 5.5	0 400	25 100 / octal	10	100	DIP, SO, SSOP, TSSOP, HVQFN	24
PCA9557	8	0011xxx (3)	No	Yes	2.3 5.5	0 400	25 100	10	100	SO, TSSOP, HVQFN	16
PCA9558	8	100111x (1)	No	Yes	3.0 3.6	10 400	4 32	0.1	weak	TSSOP	28

3. Quasi bi-directional GPIO programming

3.1 Input/Output structure overview

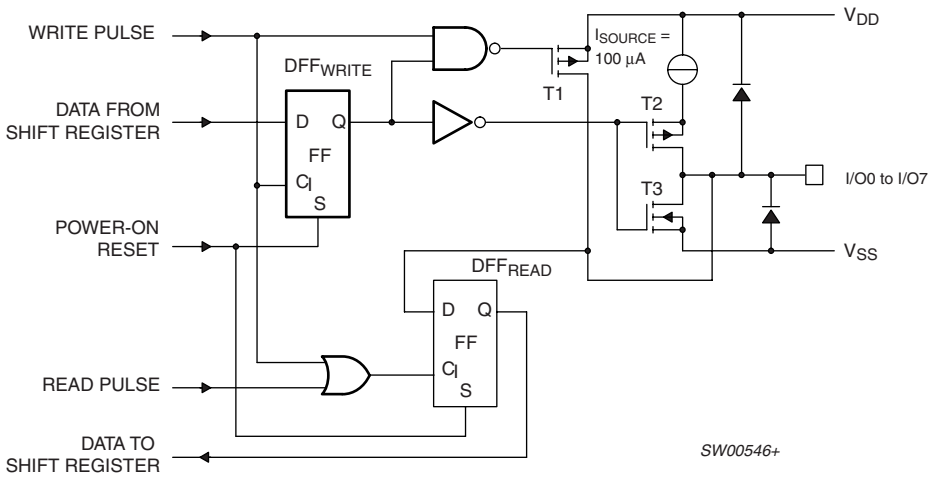


Fig 15. Quasi bi-directional I/O structure

The main components for this structure are the following:

- T1 and T2: transistor between the power supply and the I/O
- T3: transistor between the I/O and the ground
- I_{SOURCE}: 100 µA current source
- DFF_{WRITE}: Digital flip-flop used to program the I/O (input or output) and to program the output logic state
- DFF_{READ}: Digital flip-flop used to read the logic state of the I/O (Input or Output).

3.2 How it works (internal architecture)

At power-up, the output of DFF_{WRITE} is set to '1' (HIGH) via the Reset signal applied to its S pin. This causes T1 to be off (Write Pulse is at '0'), T2 to be on and T3 to be off. The 100 μA current source I_{SOURCE} acting like a weak pull-up resistor connects the I/O pin to V_{DD} and allows it to be used as an input and can then be forced to '0' (LOW, 'overriding the weak HIGH') or '1' (HIGH) by an external device.

When a Write sequence is initiated, data from the I²C-bus is written into DFF_{WRITE} and an internal Write Pulse is generated ('1' for half clock cycle) allowing the data to be stored into the flip-flop.

When a '1' is written to the DFF_{WRITE}, the I/O is automatically pulled up to V_{DD} through a current source, which is strong enough to be used as a '1' to control an external signal (used as output). An external resistor to V_{DD} can also be connected to the pin. The I/O is also ready to be used as an input and can be forced to '0' or '1' by an external device.

When a '0' is written to the DFF_{WRITE}, T1 and T2 are off while T3 is on, thus forcing the I/O to go to '0' (LOW). The I/O will stay in that state until a '1' is written to the device (see below).

When the I/O has been previously programmed with a '0' (I/O connected to ground) and then programmed to a '1', the following happens:

- The Write Pulse causes T1 to be on during the time it is equal to '1'. T2 is also on while T3 is off. During the time T1 is on, the I/O is physically connected to V_{DD}. T1 acts as a strong pull-up causing the pin to go immediately to V_{DD}, allowing fast rising edges into heavily loaded outputs.
- When the write sequence is finished (Write Pulse goes to '0' again), T1 is now off while T2 stays on. The I/O is now connected to V_{DD} through the current source I_{SOURCE}.

When a Read sequence is initiated, a internal Read Pulse is generated ('1' during a short time) and the logic level present at the I/O is stored into DFF_{READ} and sent to the master device requesting the read.

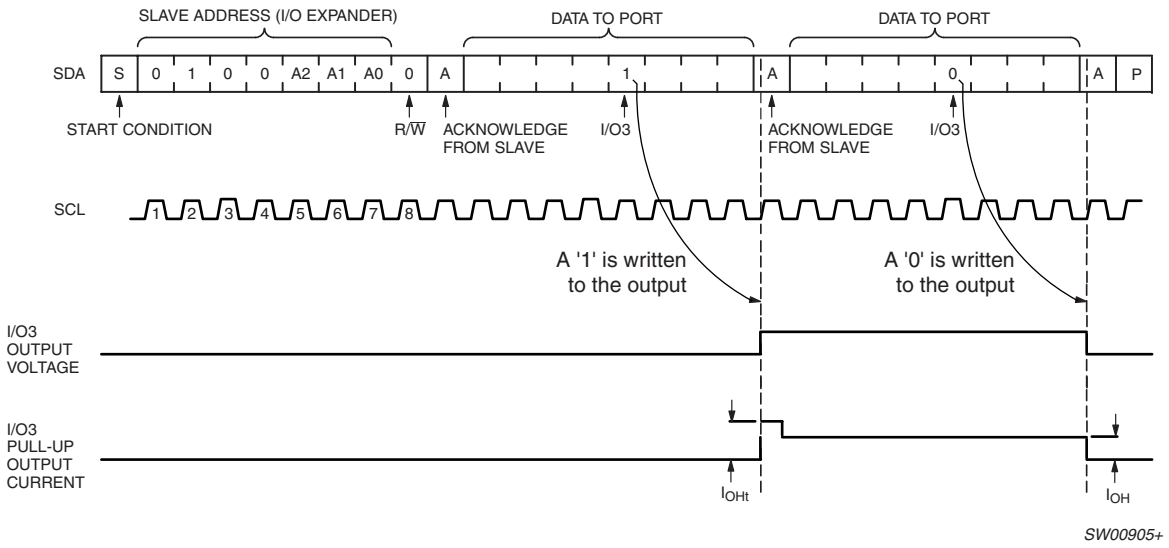


Fig 16. Quasi bi-directional I/O—voltage and current in the I/Os

3.3 Device programming

3.3.1 Write sequence

Device programming requires 2 bytes (8-bit device) or 3 bytes (16-bit device).

1. The first byte that follows the Start Command contains the I²C address of the device (7 MSBs) and the R/W bit (LSB) programmed to '0' (Write operation). Please refer to Table 15 "Quasi bi-directional GPIOs selection table" for the I²C address value.
2. The second byte is the data byte (and the third byte for a 16-bit device) and contains the logic values to configure the I/Os and their logic values:
   - Byte 1 written: bit 0 = P0 (LSB) to bit 7 = P7 (MSB) for an 8-bit device
   - Byte 1 written: bit 0 = P0.0 (LSB) to bit 7 = P0.7 (MSB) for a 16-bit device
   - Byte 2 written: bit 0 = P1.0 (LSB) to bit 7 = P1.7 (MSB) for a 16-bit device

   When '0' (LOW): the corresponding I/O is an output and is forced to '0' (25 mA sink capability)

   When '1' (HIGH): the corresponding I/O can be either an input or an output.
   - If input: an external signal can force a '0' or a '1' to that pin
   - If output: a '1' is applied to the output (100 μA source capability)
3. If more than one byte (8-bit device) or more than two bytes (16-bit device) of data are written, the previous programming will be overwritten.

Programming ends after a Stop Command has been issued by the master. The device goes to Idle mode.

Note that output logic levels are updated during the Acknowledge phase, after the byte containing the data has been sent by the master.

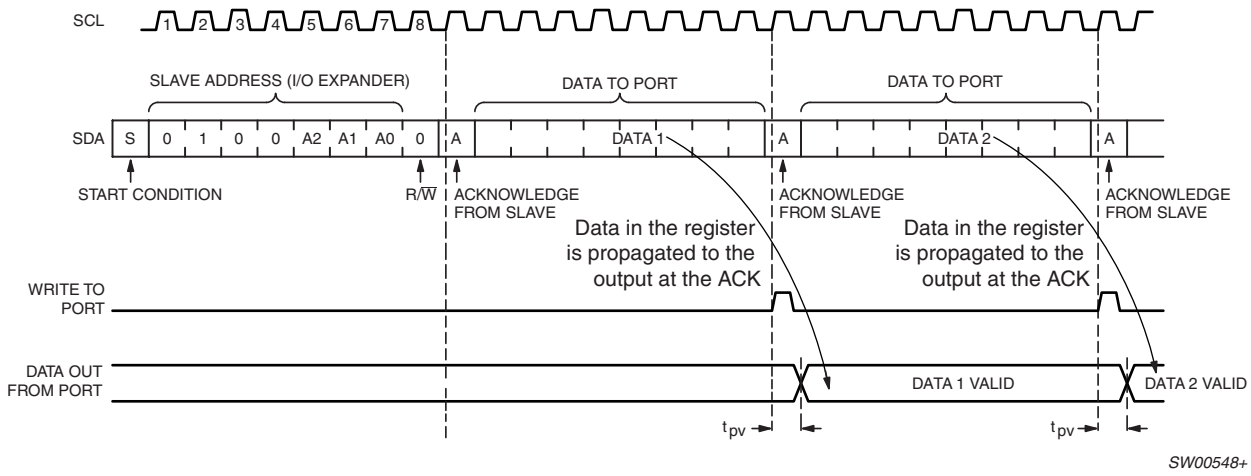


Fig 17. Quasi bi-directional I/O—Write sequence (8-bit device)

3.3.2 Read sequence

Input port reading sequence requires 2 bytes (8-bit device) or 3 bytes (16-bit device).

1. The first byte contains the I²C address of the device (7 MSBs) and the R/W bit (LSB) programmed to '1' (Read operation). Please refer to Table 15 "Quasi bi-directional GPIOs selection table" for the I²C address value.
2. The second byte is the data read from the device (and the third byte for a 16-bit device) and contains the values reflecting the logic states present on each I/O pin:
   - Byte 1 read: bit 0 = P0 (LSB) to bit 7 = P7 (MSB) for an 8-bit device
   - Byte 1 read: bit 0 = P0.0 (LSB) to bit 7 = P0.7 (MSB) for a 16-bit device
   - Byte 2 read: bit 0 = P1.0 (LSB) to bit 7 = P1.7 (MSB) for a 16-bit device

   When '0' (LOW): the corresponding I/O is at '0' (LOW)

   When '1' (HIGH): the corresponding I/O is at '1' (HIGH)
3. If more than one byte (8-bit device) or more than two bytes (16-bit device) of data are written, the same I/Os are read again.

A snapshot of the Port data is performed at the acknowledge. If Port data changes and then changes back to the original state before the next acknowledge, the momentary change will not be seen (read) by the master device.

Read sequence ends after a Stop Command has been issued by the master. The device goes to Idle mode. As specified by the I²C protocol, the master device does not acknowledge the last byte.

A read sequence also allows an Interrupt previously asserted to be deasserted. See Section 5 for more detail.

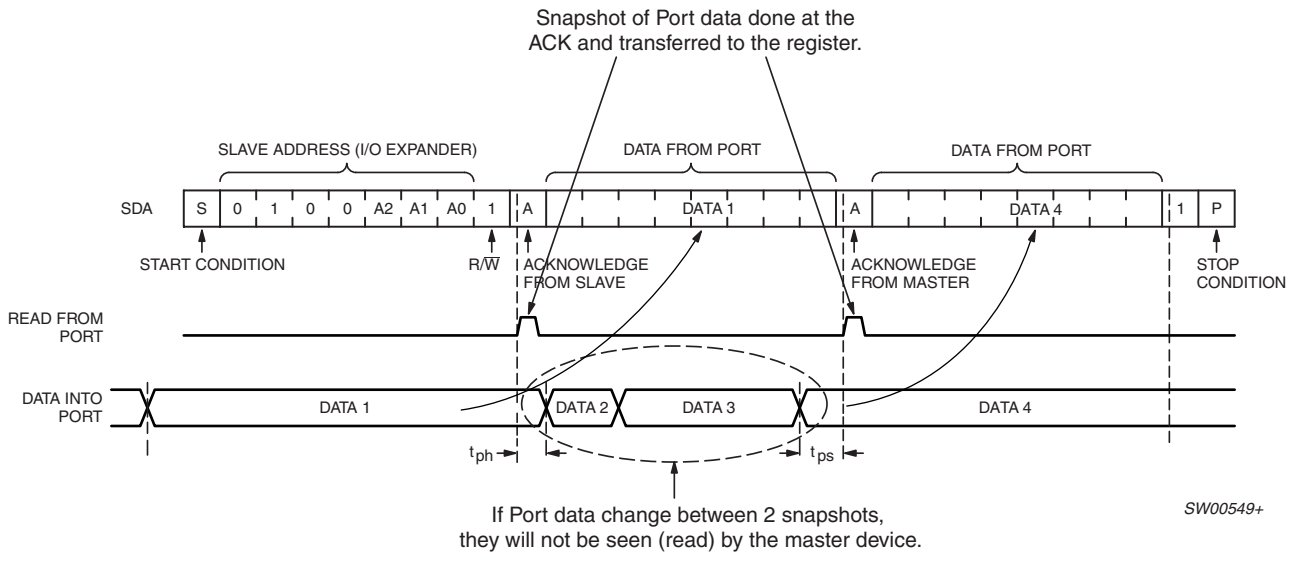
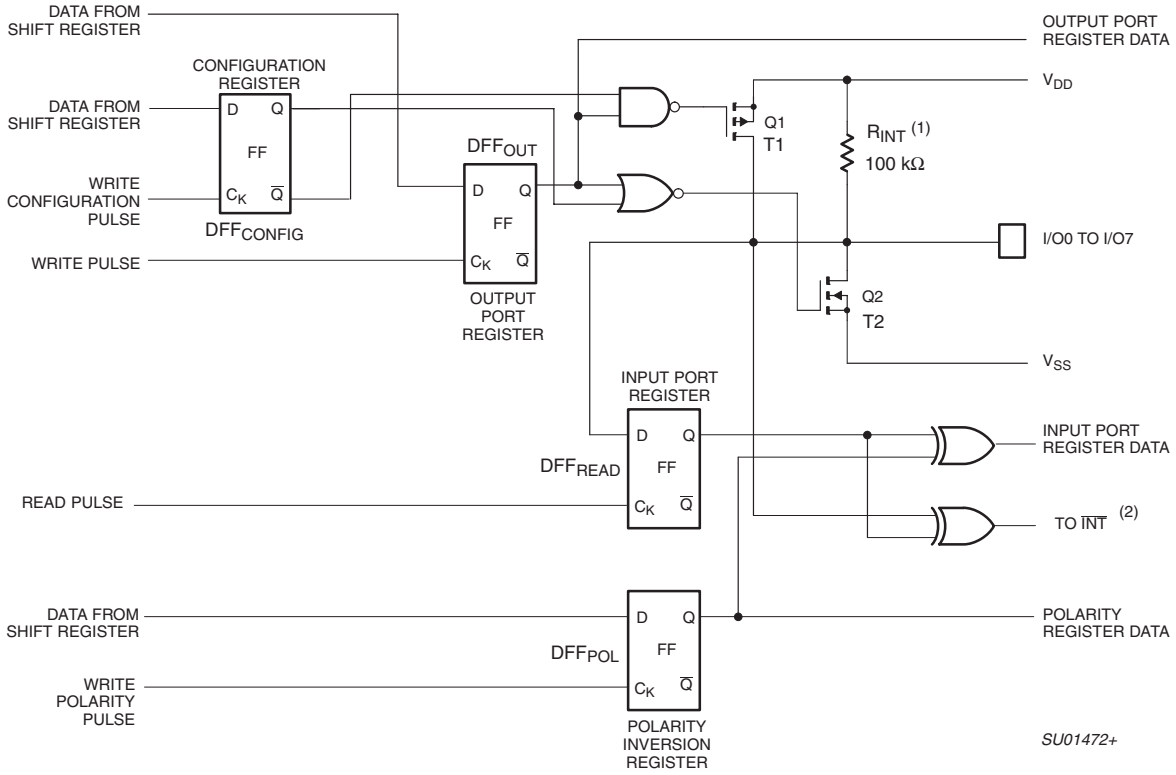


Fig 18. Quasi bi-directional I/O—Read sequence (8-bit device)

4. Totem-pole GPIO programming

4.1 Input/Output structure overview



(1) PCA9536, PCA9554, PCA9554A, PCA9555, PCA9557 only.

(2) Except PCA9536 and PCA9557.

Fig 19. Totem-pole I/O structure

The main components for this structure are the following:

- T1: transistor between the power supply and the I/O
- T2: transistor between the I/O and the ground
- R_{INT}: internal pull-up resistor (not applicable to all devices)
- DFF_{CONFIG}: Digital flip-flop used to configure the I/O as input or output
- DFF_{OUT}: Digital flip-flop used to program the logic states of the I/O when configured as output
- DFF_{READ}: Digital flip-flop used to read the logic state of the I/O (configured as Input or Output)
- DFF_{POL}: Digital flip-flop used to invert or not the polarity of the logic state that has been read.

4.2 How it works (internal architecture)

At power-up, or when a hardware reset is performed (see applicable devices in Table 16 "Totem-pole GPIOs selection table"), the outputs of DFF_{CONFIG} and DFF_{OUT} are set to '1' via a Reset signal applied to its S pin (not shown in Figure 19). This causes both T1 and T2 to be off.

- Devices with R_{INT}: the I/O pin is connected to V_{DD}
- Devices without R_{INT}: the I/O pin is in high-impedance

In both cases, all the I/Os are configured as inputs and can then be forced to '0' (LOW, 'overriding the weak HIGH' for devices with R_{INT}) or '1' (HIGH) by an external device.

To be used as outputs, the **Configuration Register** (a single register for the 4-bit and 8-bit devices, two for the 16-bit devices) needs to be programmed by a master device.

- 0: the corresponding I/O is configured as output
- 1: the corresponding I/O is configured as input

When a Configuration Register Write sequence is initiated by a master device, data from the I²C-bus are written into DFF_{CONFIG} and an internal Write Configuration Pulse is generated ('1' during a short time) allowing the data to be stored into the flip-flop.

Once the I/O has been configured as an output through the Configuration Register, the pin can be forced to '0' or '1' through the **Output Register** (a single register for the 4-bit and 8-bit devices, two for the 16-bit devices).

- 0: the corresponding output is forced to '0': T1 is off, T2 is on
- 1: the corresponding output is forced to '1': T1 is on, T2 is off

When an Output Register Write sequence is initiated by a master device, data from the I²C-bus is written into DFF_{OUT} and an internal Write Pulse is generated ('1' during a short time) allowing the data to be stored into the flip-flop.

Note that the information in the Output Register is relevant for I/Os that have been configured as outputs. When the corresponding I/O is an input, the logic value present in the Output Register is a 'Don't Care'.

The **Polarity Inversion Register** (a single register for the 4-bit and 8-bit devices, two for the 16-bit devices) allows polarity inversion of the data that is in the Input Port Register (a single register for the 4-bit and 8-bit devices, two for the 16-bit devices) that contains logic level values present at the corresponding pin (input or output).

- 0: the corresponding value in the Input port data polarity is retained
- 1: the corresponding value in the Input port data polarity is inverted

When a Polarity Inversion Register Write sequence is initiated by a master device, data from the I²C-bus is written into DFF_{POL} and an internal Write Polarity Pulse is generated ('1' during a short time) allowing the data to be stored into the flip-flop.

When a Read sequence is initiated, a internal Read Pulse is generated ('1' during a short time) and the logic level present at the I/O is stored into DFF_{READ}, then inverted or not based upon the Polarity Inversion Register values and sent to the master device requesting the read.

Table 17: Power-up and Reset (when applicable) default values

Command byte default value is 0x00.

Register	Default value (Hex)	Note
Input Port 0	0xFF	8-bit and 16-bit devices with internal pull-up resistors
	[1]	8-bit and 16-bit devices without internal pull-up resistors
Input Port 1	0xFF	Only 16-bit devices with internal pull-up resistors
	[1]	Only 16-bit devices without internal pull-up resistors
Output Port 0	0xFF	8-bit and 16-bit devices
Output Port 1	0xFF	Only 16-bit devices
Polarity Inversion 0	0x00	8-bit and 16-bit devices
Polarity Inversion 1	0x00	Only 16-bit devices
Configuration 0	0xFF	8-bit and 16-bit devices
Configuration 1	0xFF	Only 16-bit devices

[1] Since the I/Os are floating (no pull-up resistors), default values are determined by the external environment.

A programming example is shown in Figure 20.

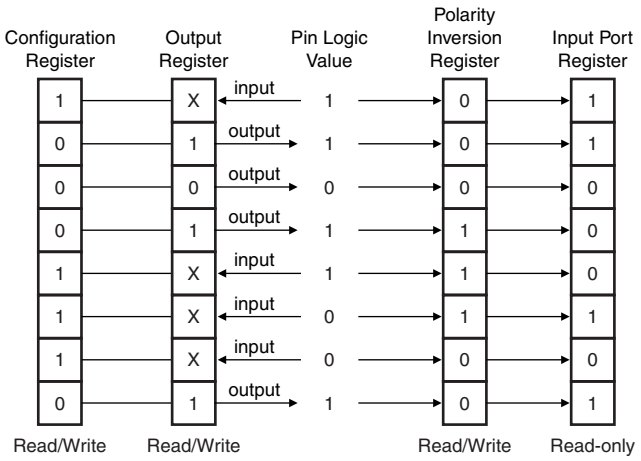


Fig 20. Register programming example

4.3 Device programming

4.3.1 Write sequence

Device programming is performed in 3 bytes for a 8-bit device and in 4 bytes for a 16-bit device:

1. The first byte that follows the Start Command contains the I²C address of the device (7 MSBs) and the R/W bit (LSB) programmed to '0' (Write operation). Please refer to Table 16 "Totem-pole GPIOs selection table" for the I²C address value.
2. The second byte is the Command byte and contains the pointer (location) of the register that needs to be written. Byte values and corresponding addressed register are defined in Table 18 "Command byte values".
3. The third byte is the data byte (and the fourth byte for a 16-bit device) and contains the data that needs to be written to the register corresponding to the Command value that has been written in the previous byte (Output, Configuration or Polarity Inversion). For the 16-bit devices, the pointer is automatically incremented to the next register of the same category (Port 0 to Port 1 and Port 1 to Port 0).
   - Byte 1 written: bit 0 = bit for P0 (LSB) to bit 7 = bit for P7 (MSB) for the 8-bit devices
   - Byte 1 written: bit 0 = bit for P0.0 (LSB) to bit 7 = bit for P0.7 (MSB) for the 16-bit devices
   - Byte 2 written: bit 0 = bit for P1.0 (LSB) to bit 7 = bit for P1.7 (MSB) for the 16-bit devices

   A Write to the Input Register will have no effect since this is a Read-Only register.

   When writing to the output Port register, output logic levels (I/Os configured as outputs) are updated during the Acknowledge phase, after the byte containing the data has been sent by the master.
4. If more than one byte (8-bit device) or more than two bytes (16-bit device) of data are written, the previous programming will be overwritten.

Programming ends after a Stop Command has been issued by the master. The device goes to Idle mode.

Table 18: Command byte values

PCA9534, PCA9536, PCA9537, PCA9538, PCA9554/54A, PCA9557		PCA9535, PCA9539, PCA9555		PCA9558	
Command	**Register**	**Command**	**Register**	**Command**	**Register**
0x00	Input Port	0x00	Input Port 0	0x07	Input Port
0x01	Output Port	0x01	Input Port 1	0x08	Output Port
0x02	Polarity Inversion	0x02	Output Port 0	0x09	Polarity Inversion
0x03	Configuration	0x03	Output Port 1	0x0A	Configuration
Other command values not allowed		0x04	Polarity Inversion 0	Other command values used for EEPROM, DIP switch or reserved. See PCA9558 data sheet for more detail.	
		0x05	Polarity Inversion 1		
		0x06	Configuration 0		
		0x07	Configuration 1		
		Other command values not allowed			

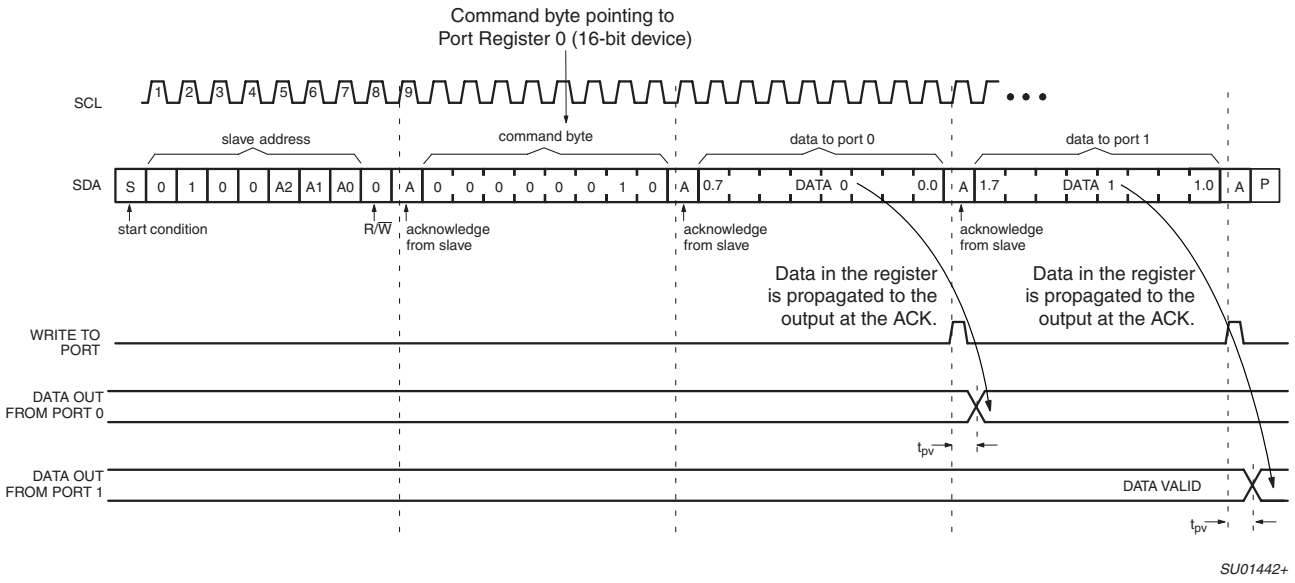


Fig 21. Totem-pole I/O—Write sequence (16-bit device)

4.3.2 Read sequence

Input port reading sequence is performed in 2 sub-sequences:

1. The master initiates a Write sequence in order to determine which register needs to be read.
2. After sending a Re-Start command (or a Stop command followed by a Start command), the master initiates the Read sequence of the register that needs to be read.

The entire reading sequence is performed in 4 bytes for a 8-bit device and in 5 bytes for a 16-bit device:

1. The first byte that follows the Start Command contains the I²C address of the device (7 MSBs) and the R/W bit (LSB) programmed to '0' (Write operation). Please refer to Table 16 "Totem-pole GPIOs selection table" for the I²C address value.
2. The second byte is the Command byte as described in Table 18 "Command byte values".
3. Once the master has sent the Command byte, it has to generate either a Re-Start command or a Stop command followed by a Start command in order to send the next byte that contains the I²C address of the device (7 MSBs, same as the one described in step 1) and the R/W bit (LSB) programmed to '1' (Read operation).
4. The fourth byte is the data read from the device (and the fifth byte for a 16-bit device) and contains the value of the register that needs to be read:
   - Byte 1 read: bit 0 = bit for P0 (LSB) to bit 7 = bit for P7 (MSB) for an 8-bit device
   - Byte 1 read: bit 0 = bit for P0.0 (LSB) to bit 7 = bit for P0.7 (MSB) for a 16-bit device
   - Byte 2 read: bit 0 = bit for P1.0 (LSB) to bit 7 = bit for P1.7 (MSB) for a 16-bit device

When a read of the Output Port, Configuration or Polarity Inversion registers is requested, data that is read is either previously programmed values or default power/reset values if no programming has been performed yet.

When a read of the Input Port register is performed:

– bit at '0' (LOW): the corresponding I/O is at '0' (LOW)

– bit at '1' (HIGH): the corresponding I/O is at '1' (HIGH)

5. If more than one byte (8-bit device) or more than two bytes (16-bit device) of data are written, the same I/Os are read again.

Snapshot of the Port data is performed at the acknowledge. If Port data changes and then changes back to the original state before the next acknowledge, it will not be seen (read) by the master device.

Read sequence ends after a Stop Command has been issued by the master. The device goes to Idle mode. As specified by the I²C protocol, the master device does not acknowledge the last byte.

A read sequence also allows an Interrupt previously asserted to be deasserted. See Section 5 for more detail.

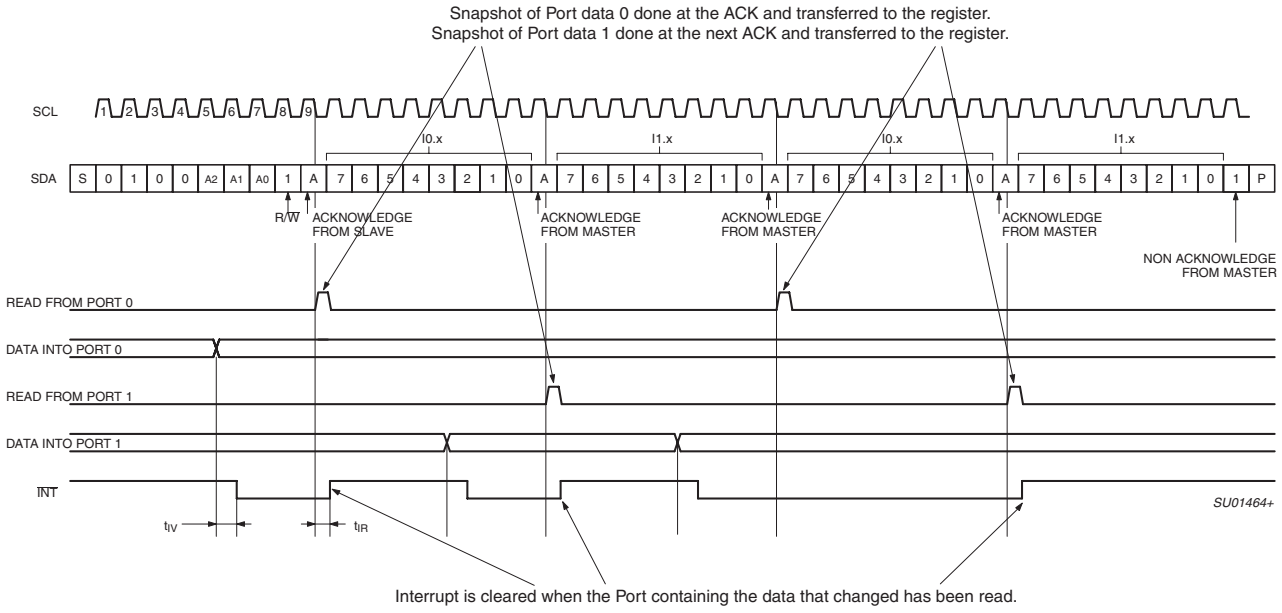


Fig 22. Totem-pole I/O—Read sequence (16-bit device)

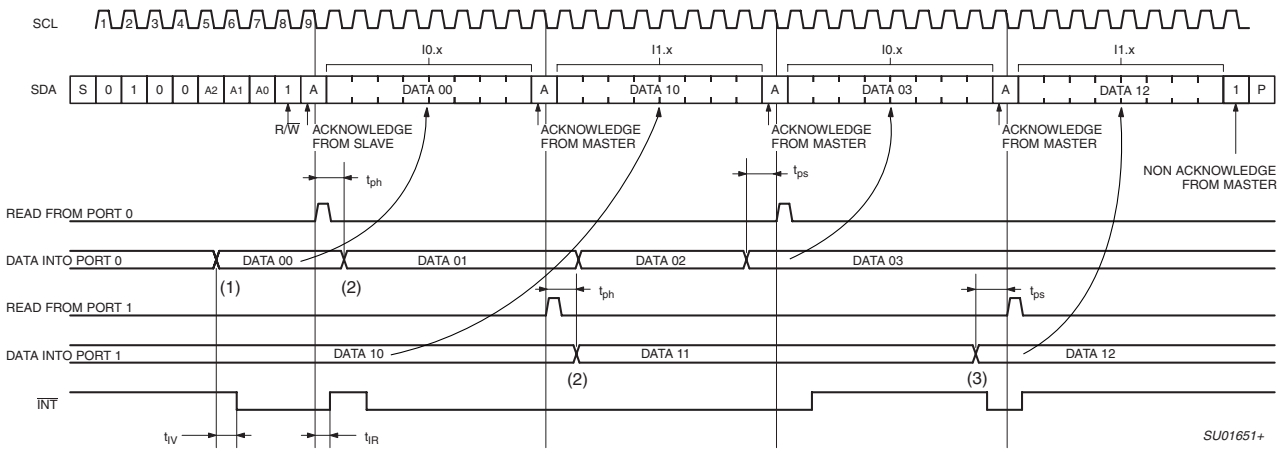
5. Active-LOW Interrupt ($\overline{INT}$)

In this paragraph, $\overline{INT}$ signal asserted refers to the signal being LOW and deasserted refers to the same signal being HIGH.

Active-LOW Interrupt output (see list of devices with that feature in Table 15 and Table 16) is generally connected to a master device. It is asserted each time an I/O configured as an input changes state.

The main things to keep in mind when using this feature are the following:

- Only an external change causes $\overline{INT}$ signal to be asserted
  - Quasi bi-directional I/O: a Write sequence to the device does not generate an Interrupt
  - Totem-pole: I/O: the I/O must be configured as an input to be able to generate an Interrupt
- Input values are not latched:
  - $\overline{INT}$ stays asserted when one input (or more) changes state
  - $\overline{INT}$ is deasserted as soon as the input that generated the Interrupt goes back to its previous logic value (assuming this input was the only one that changed state and generated the Interrupt)
  - $\overline{INT}$ is deasserted after a Read sequence (see below)
- Once $\overline{INT}$ has been asserted, the master device initiates a Read sequence in order to determine which input(s) generated the Interrupt condition (see Section 3.3.2 and Section 4.3.2 for more information about the Read sequence). During the Read sequence, $\overline{INT}$ stays asserted until the last register containing the input that caused the Interrupt to occur has been read. For example:
  - If P0.2 and P0.4 changed, $\overline{INT}$ is deasserted immediately after Input Port 0 has been read (assuming no other input changed in between)
  - If P0.2 and P1.4 changed, $\overline{INT}$ is deasserted only after Input Port 0 and Port 1 have been read (assuming no other input changed in between)
- $\overline{INT}$ stays asserted after a complete Read sequence if a new input(s) changes state during the Read sequence.



(1) Interrupt was generated by a signal from Port 0 (Data 00).

Interrupt cleared after reading Port 0.

Data read from Input Register - contains the input port that asserted the Interrupt (Data 00).

(2) Interrupt was generated by a signal from Port 0 (Data 01) and later (when Interrupt still asserted) and additional change occurred in Port 1 (Data 11).

Interrupt is then cleared after reading Port 1 and Port 0.

Data read from Input Register 1 does not contain the input port that changed (Data 11) because the snapshot occurred before the change event. Data 11 has not been read by the master device.

Data read from Input Register 0 does not necessarily contain the input port that changed: Data 01 was the initial data, then it changed to Data 02, and later to Data 03. Data read is Data 03, which is the value in the port at the snapshot moment.

Data 01 and Data 02 have not been read by the master.

Data 11 has not been read by the master.

(3) Interrupt was generated by a signal from Port 1.

Interrupt cleared after reading Port 1.

Data read in Input Register 1 contains the input port that asserted the Interrupt (Data 12).

Fig 23. Interrupt sequence (16-bit device)

6. Active-LOW Reset ($\overline{\text{RESET}}$) and Power-On Reset (POR)

The active-LOW Reset input (see list of devices with that feature in Table 15 and Table 16) and the Power-on reset (present on all the devices) allow the device to be initialized in a known default state at power-up or at anytime the master device wants to reset the slave device.

When power is applied to V_{DD}, the internal Power-on reset holds the I/O Expander in a reset condition until V_{DD} has reached V_{POR}. At that point, the reset condition is released and the registers and state machine will initialize to their default states. Thereafter, V_{DD} must be lowered below 0.2 V to reset the device. For a power reset cycle, V_{DD} must be lowered below 0.2 V and then restored to the operating voltage. The voltage must be held at that level for at least 200 µs before ramping up again to ensure proper initialization of all the internal registers and nodes.

A reset can be accomplished by holding the $\overline{\text{RESET}}$ pin LOW for a minimum of t_W defined in the device data sheet (see AC or dynamic characteristics table). The registers and I²C state machine will be held in their default state until the $\overline{\text{RESET}}$ input is once again HIGH. This input requires a pull-up resistor to V_{DD} if no active connection is used.

7. Programmable address pins

7.1 PCA9500 and PCA9501

Address input pins have internal 300 kΩ pull-up resistors and can be left floating when a HIGH logic level (‘1’) is needed. Address pin must be connected to Ground if a LOW logic level (‘0’) is needed.

7.2 Other devices

Address input pins do not have internal pull-up resistors. They are static CMOS logic inputs and there is then no static power dissipation as long as the input is connected to Ground (LOW) or V_{DD} (HIGH). There are no input pull-up resistors that would dissipate power when the inputs are grounded. If the inputs are held at an intermediate voltage between V_{DD} and Ground there will be a power dissipation. So either a HIGH (V_{DD}) or a LOW (Ground) will have no power but floating or intermediate voltages will result in power dissipation and undetermined address.

8. Software considerations in using totem-pole GPIOs in place of quasi bi-directional GPIOs

8.1 16-bit devices: PCA9555 / PCF8575

Because the PCA9555 and the PCF8575 are pin compatible and have the same I²C address, they can often be used in the same board design, however the software to use these devices differ a bit.

Software changes described below also applies to:

- The PCA9535, which is pin-to-pin compatible with the PCA9555 without the internal pull-up resistors.
- The PCA9539, which is software compatible with the PCA9555 but with a different footprint due to the additional $\overline{\text{RESET}}$ pin and different fixed I²C address.

At power-up, both devices look very similar to the target application. The PCF8575 powers up with all I/O active-HIGH, and so can immediately be used as inputs. The PCA9555 also powers up with all of the I/O pins configured as inputs. How you read or write to the individual device is slightly different as shown in Table 19.

Table 19: Software differences for the 16-bit devices

Operation	PCF8575	PCA9555
Write to the device	ST 40 00 00 SP (3 bytes)	ST 40 02 00 00 SP (4 bytes)
Read to the device	ST 41 xx xx SP (3 bytes)	ST 40 00 ST 41xx xx SP (5 bytes)

Where ST = Start, SP = Stop, and xx is data from device. In the write example, all outputs are programmed LOW (‘0’ to all ports in this example).

The PCF8575 requires that you send a total of 3 bytes to either read or write to the device. When writing to the PCF8575, the master sends a Start bit followed by the device address, with the least significant bit set to ‘0’. The following two bytes will set the condition of the output ports and then followed by a Stop bit. A total of 3 bytes are required.

To read from the PCF8575, all of the ports must be set to logic 1. The master then sets the last bit of the byte containing the address to logic 1. The next two bytes will contain the status of the input ports. A total of 3 bytes are used.

In the PCA9555, data is also transmitted by sending the device address and setting the least significant bit to logic 0. But the next byte after the address will be a command byte. Internal to PCA9555 are eight octal registers configured to operate in 4 registered pairs. The four pairs are Input ports (commands 0x00 and 0x01), output ports (commands 0x02 and 0x03), polarity inversion ports (command 0x04 and 0x05), and configuration ports commands 0x06 and 0x07). After sending data to one register, the next byte will automatically be sent to the other register pair. There is no limitation to the number of data bytes that can be sent in one write transmission.

Table 19 then shows a write sequence for the PCA9555. Note that the master generates a Start bit followed by the address with the least significant bit set to ‘0’. The next byte is the command to write to the output register (command 0x02). The following two bytes then write the output (in this case, the data is 00). A total of 4 bytes are used.

To be able to use the current source feature of the PCA9555, the master must first configure the port or ports as an output because the power-up default condition for the PCA9555 is that the ports are all set as inputs. For example, to program all of the ports as outputs, the master first generates a Start bit followed by the device address with the least significant bit set to ‘0’. The following byte addresses the configuration port (0x06) and the next two bytes set all of the ports to outputs (00 00). The command sequence would be ST 40 06 00 00 SP. The PCA9555 is now set as a 16-bit output port and can sink up to 25 mA active-LOW or source up to 10 mA active-HIGH. Caution must be used if the user plans on using the port pin as both an input and output. If the port pin is programmed to an active-HIGH and the pin is driven by an external signal LOW, excessive current could flow because of the conflict.

To read from the PCA9555 requires that you set the command register to 0x00 (input register) before you actually read the data. First the address is sent with the least significant byte set to ‘0’ followed by the command byte 0x00. A restart condition is then sent by the master followed by the read address with the least significant bit set to ‘1’. The following two bytes is the port data. A total of 5 bytes are used.

It should be noted that the next time input port data is needed, the input port will still be accessed until a new command byte has been transmitted, so only two bytes will be needed if no change to the command register is made. Although the PCA9555 appears to require a little more overhead to complete a transaction, it does have the advantage of being able to access either the ‘true’ or the ‘complement’ of each of its data registers independently. Also, since there is no limit to the number of reads in a single transmission, the master can continuously monitor alternate ports until a Stop bit is sent.

8.2 8-bit devices

Because the PCA9554 and the PCF8574 are pin compatible and have the same I²C address, they can often be used in the same board design, however the software to use these devices differ a bit.

Software changes described below also applies to:

- The PCA9554A, which is the same device as the PCA9554 with a different I²C address (same address as PCF8574A)
- The PCA9534, which is pin-to-pin compatible with the PCA9554 without the internal pull-up resistors
- The PCA9538, which is software compatible with the PCA9554 but with a different footprint due to the additional $\overline{\text{RESET}}$ pin and different fixed I²C address.

At power-up, both devices look very similar to the target application. The PCF8574 powers up with all I/O active-HIGH, and so can immediately be used as inputs. The PCA9554 also powers up with all of the I/O pins configured as inputs. How you read or write to the individual device is slightly different as shown in Table 20.

Table 20: Software differences for the 8-bit devices

Operation	PCF8574	PCA9554
Write to the device	ST 40 00 SP (2 bytes)	ST 40 01 00 SP (3 bytes)
Read to the device	ST 41 xx SP (2 bytes)	ST 40 00 ST 41xx SP (4 bytes)

Where ST = Start, SP = Stop, and xx is data from device. In the write example, all outputs are programmed LOW. (‘0’ to all ports in this example).

The PCF8574 requires that you send a total of 2 bytes to either read or write to the device. When writing to the PCF8574 the master sends a Start bit, followed by the device address with the least significant bit set to ‘0’. The following byte will set the condition of the output port and then followed by a Stop bit. A total of 2 bytes are required.

To read from the PCF8574, all of the ports must be set to logic 1. The master then sets the last bit of the byte containing the address to logic 1.

The next byte will contain the status of the input port. A total of 2 bytes are used.

In the PCA9554, data is also transmitted by sending the device address and setting the least significant bit to logic 0. But the next byte after the address will be a command byte. Internal to PCA9554 are 4 octal registers. The four registers are Input (commands 0x00), output (commands 0x01), polarity inversion (command 0x02), and configuration 0x03).

Table 20 then shows a write sequence for the PCA9554. Note that the master generates a Start bit followed by the address with the least significant bit set to ‘0’. The next byte (0x01) then tells the device that it should write the following byte to the output port and the last byte is the data to be written to the port.

To be able to use the current source feature of the PCA9554, the master must first configure the port as an output because the power-up default condition for the PCA9554 is that the ports are all set as inputs. For example, to program all of the ports as outputs, the master first generates a Start bit followed by the device address with the least significant bit set to ‘0’. The following byte addresses the configuration port (0x04) and the next byte set all of the ports to outputs (00). The command sequence would be

ST 40 06 00 SP. The PCA9554 is now set as an 8-bit output port and can sink up to 25 mA active-LOW or source up to 10 mA active-HIGH. Caution must be used if the user plans on using the port pin as both an input and output. If the port pin is programmed to an active-HIGH and the pin is driven by an external signal LOW, excessive current could flow because of the conflict.

To read from the PCA9554, requires that you set the command register to 0x00 (input register) before you actually read the data. First the address is sent with the least significant byte set to '0' followed by the command byte 0x00. A Restart condition is then sent by the master followed by the read address with the least significant bit set to '1'. The following byte is the port data. A total of 4 bytes are used.

It shout be noted that the next time input port data is needed, the input port will still be accessed until a new command byte has been transmitted, so only two bytes will be needed if no change to the command register is made. By using the command register to address the polarity inversion register (command 0x03), the PCA9554 can be programmed to either read or write the 'true' or 'complement' of the actual value at the port pin. The command port structure for the PCA9554 is also found in the PCA9556 and PCA9557.

9. Frequently asked questions

9.1 Power cycle and Reset

1. **Question:** We plan on using one of our PCA95xx I/O expander without a Reset pin due to limited amount of pins available on our master device. We will then power cycle the I/O expander each time something goes wrong in order to initialize the device. I then would like to know the exact reset scheme in order to perform a valid reset.

   Answer: At power-up, the devices are held in reset mode until V_{DD} goes higher than V_{POR} (see exact value defined in the DC (Static) characteristics table, typically from 1.25 V to 1.65 V depending on the device used). Once V_{DD} goes higher than V_{POR}, reset mode is left and the part becomes ready to be addressed by a master quickly (after about 20 µs for the PCA9554 for example). To reset the device from a power-on state, V_{DD} must be lowered to about 0.2 V above ground and must be held at that level for at least 200 µs before ramping up again.

2. **Question:** Our I/O card 3.3 V power supply will be ramped up very slowly, in the order of 100 ms to 300 ms to 3.3 V during hot-plug. During this period, the I²C-bus may be toggling. We don't care about data on the bus during ramp-up, but we don't want any damage to the PCA9555. Will the PCA9555 reset circuit able to handle the slow V_{CC} ramping and reset the chip correctly and will it sustain any damage or latch-up?

   Answer: Reset state is kept until the supply voltage reaches a value where this condition is finally released (1.25 V to 1.65 V). The fact that the supply voltage ramps up slowly or fast does not matter. Once the voltage value is higher than the threshold, the chip will be in a known state (Reset condition). The devices will not generate a latch-up condition (damaging the device) when the I²C-bus toggles while the device is not powered-up or in the power-up phase.

3. **Question:** What is the I/O condition just after Power-off is for the PCA9555?

   Answer: All I/Os are high-impedance when the part is unpowered.

4. **Question:** In the data sheet of PCF8575 it is mentioned that the Port output is HIGH after power-on. In the data sheet of PCF8575C it is mentioned that the Port output will be in 3-state condition after power-on. We found that there is no difference in the block diagram and the simplified schematic diagram of each I/O. What is the condition of the outputs after power-on?

   Answer: There is in fact a small difference in the schematic diagrams for PCF8575 and PCF8575C. The PCF8575C lacks the weak pull-up in the form of the 100 μA current source in the output I/Os. This is the reason that PCF8575's I/Os remain HIGH after power-on and after being written HIGH. Without this current source, PCF8575C's I/Os will be in high-impedance state after the initial strong pull-up transistor is OFF (it is ON during the acknowledge phase).

5. **Question:** I'm using the PCA9501 and I would like to know if there is a problem by powering down the PCA9501 when one of the I/Os is still driven by another device that is not powered down. Do you think any potential problems to the device?

   Answer: There is no problem using the device in this configuration as it can be seen in the simulation waveforms (Figure 24). The graph is the output pin (P7) that is held at 3.3 V, V_{DD} that is ramped from 0 V to 3.3 V and back to 0 V, and the current from the power-supply hooked up to the pin and internal V_{DD}. The scale is in the nA range. As you can see in the graph, the current is less than 5 nA. The current spike of 200 nA is due to the over-voltage protection switching from the internal voltage to the external pin voltage (P7) and is only temporary.

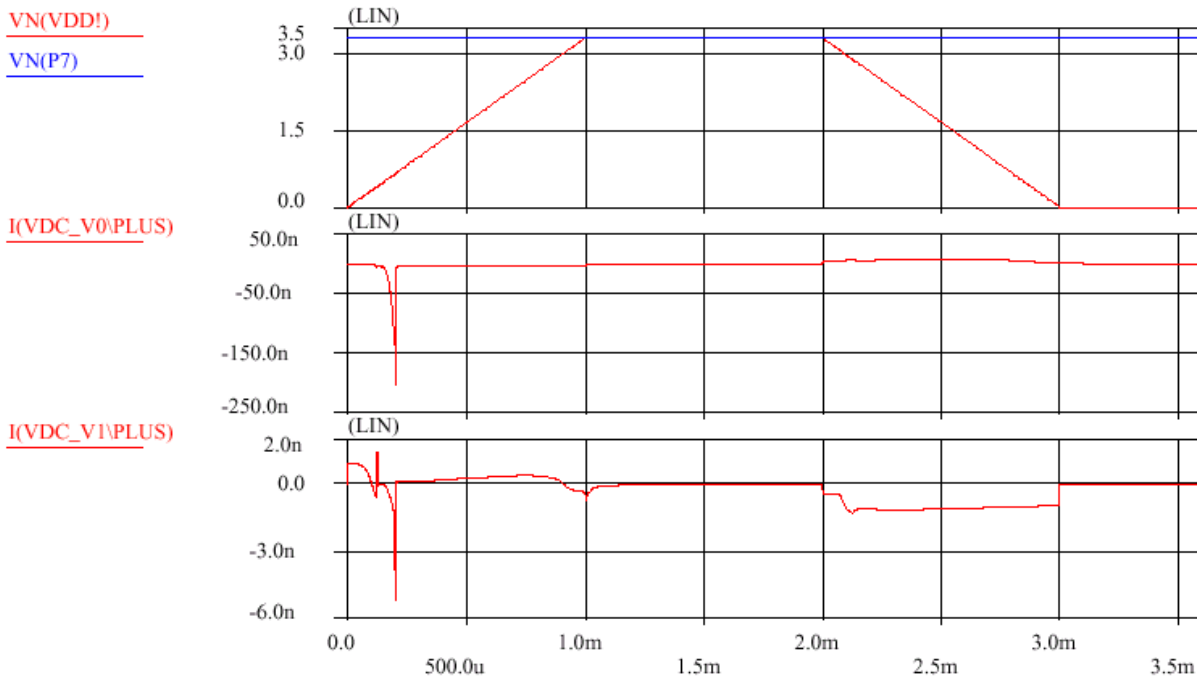


Fig 24. PCA9501 - I/O current during power cycle

6. **Question:** How long does the PCA9501 stay in a reset state after the power supply reaches 3.3 V?

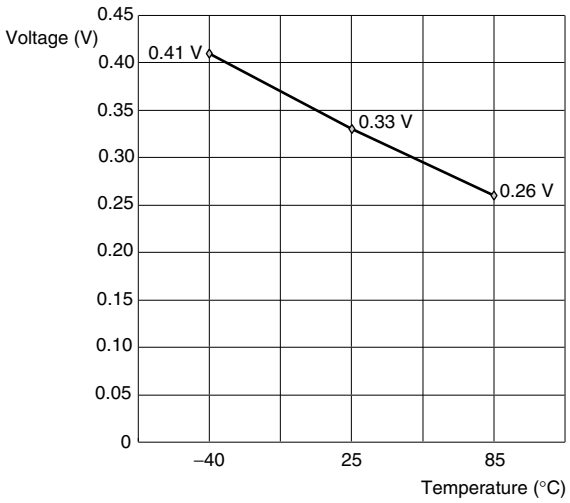
   Answer: The Power-On Reset (POR) of PCA9501 is active for as long as it takes the power supply—coming from 0 V—to reach V_{POR}, about 2.4 V. After passing this level, it will be actively in reset for only an additional 300 ns, after which it will be ready to receive a Start condition.

7. **Question:** I am using the PCA9535 and I would like to have more data on the exact power-down reset voltage value in order to be sure how low V_{DD} needs to go and generate a valid power-on reset.

   Answer: Characterization result for this parameter is provided in Figure 25.

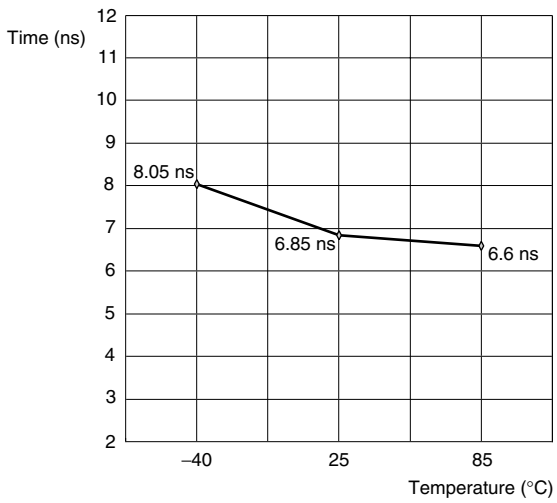
8. **Question:** I am using the PCA9535 and I would like to have more data on the maximum spike length the power supply (V_{DD}) is allowed to have. When this spike occurs, we assume that it goes lower than the power-down reset voltage (see Question 7).

   Answer: Characterization result for this parameter is provided in Figure 26.



Data in the registers set to default value.

Fig 25. Power-down value to initiate a reset



No power-on reset performed.

Fig 26. Maximum allowable time for V_{DD} = 0 V and data in registers is retained

9.2 Communication with the devices

1. **Question:** I want to use a PCA9555 in a 5 V supply voltage environment and have it communicating with a 3.3 V master device. Do I need voltage level translators for the I²C signals to allow proper communication?

   Answer: No, the PCA9555 can communicate with a 3.3 V master device while its own power supply is at 5 V with reduced upper-level noise margins.

2. **Question:** If multiple I/Os are configured as outputs, is there any guarantee that modifying the state on one output bit will not affect the non-modified outputs? For example, with all I/Os set to output and the current content of the output register as 00101100 (0x2C), we write 00101010 (0x2A) to the output register. Bit 2 and Bit 1 change state. Are the non-modified bits guaranteed not glitch? The typical application diagram would seem to imply that the outputs would be glitch-free. Is there any information available as to this operational characteristic?

   Answer: All GPIOs will not glitch as selected output states are changed HIGH and LOW. Outputs loaded with the same data (no change in particular output state) will not cause any glitch or transient to happen. Only outputs with different logic levels will cause a change to happen.

3. **Question:** We have a generic I²C controller interface in our ASIC that can perform 4 byte read & write transactions only. We use it to control a PCA9554 device. What happens to the slave device if our I²C controller sent out 4 bytes instead of the 3 bytes described in the data sheet (I²C address, Command Register, Data Bye)? Does the device truncate the last 3 bytes or just write over until the Stop bit?

   Answer: Depending on the command byte (01h, 02h, 03h), the state of the register associated with the command byte (output, polarity, configuration) will be immediately updated at the acknowledge. This means that if the new byte is different from the previous one, then the register value will change at the acknowledge (and then the output change might change as well depending on the previous byte value). This can be prevented by sending the same data as many times as it is required. Since the device has glitch-free outputs, programming the same value several times will not cause any transient state change.

4. **Question:** I'm using a PCA9555 where a SMBus/I²C master regularly reads the I/O expander. During this reading operation, the master may be reset. There is then no longer a master driving SCL (clock) while the target expander was driving SDA LOW. From this situation, there is then no way for the master to take control of the bus. Does the expander somehow time-out or just continue to drive SDA LOW waiting for more clocks? Unfortunately, the target expander doesn't get reset because it is essentially a remote device.

   Answer: There is no time-out feature in the PCA955X devices. The I/O expander will wait for more clock signals until it is done with the transmission. In this situation, after the Reset, the master needs to send 9 clock pulses to reconfigure the device still in the Slave-Transmitter mode so that the master can then send a Stop or a Re-Start command in order to restore the bus to an idle condition.

5. **Question:** I am using only some of the 16 I/Os of the PCA9555 (all of them are programmed as outputs to make the software easier). Do a '0' or '1' in the output registers for the unused outputs make a difference in term of total current consumption?

   Answer: Yes, due to the internal pull-up resistor, an I/O configured as output and set to '0' will increase the total current consumption by about 69 mA per bit. It is then better to set it to '1' instead.

9.3 Device characteristics, connectivity, external components

1. **Question:** Do PCA9554 and PCA9555 have a internal diode on SDA and SCL?

   Answer: PCA95xx I²C parts do not have a diode between SDA/SCL and V_{DD}.

2. **Question:** Do the I/O Expanders address select pins need pull-up or pull-down resistors, or can they be tied directly to V_{DD} or GND? I notice that in the 'Device Address' section of the new data sheets, it says that internal resistors are not used in order to save power, but if resistors are required on the board, then that power is needed anyway.

   Answer: Most of the devices have up to three address pins and the pins have to be held HIGH or LOW to select '1' or '0' that sets the I²C address. It is good engineering practice to use a resistor when they are pulled HIGH to limit the current through the device should something happen, but it is not required as the address pins don't use any current. The pin shall be tied to V_{DD} or GND (directly or through a resistor). It cannot be left floating. Note that the PCA9500/01 do have internal pull-up resistors for the address pins. If the address bit is HIGH, then the pin can be a 'no connect'; if the address bit is LOW, then the pin needs to be tied to GND.

   When a device like the PCA9500 with address pins with internal pull-up resistors is tied to GND, then there is a continuous current stream through the internal pull-up resistor from V_{CC} to GND. We want to avoid this since if the device is used in battery power applications it is an unneeded battery drain. Since the I²C devices can be used in both battery and AC-powered applications, we do not include the internal pull-up on the address pin and the customer needs to tie them HIGH or LOW. The PCA9500/01 were initially designed for non battery-powered applications, thus explaining the presence of the pull-up resistors in these devices.

3. **Question:** How weak should the external pull-up for the PCF8574 Quasi bi-directional I/O Expander be?

   Answer: There are no particular recommendations regarding the pull-up resistors on the I/O pins. There is an internal 100 μA current source in the PCF8574 I/O, so external pull-up on the I/O pins is not required. If external pull-up resistors are used, they need to be sized less than the available external driver sinking current capability

4. **Question:** On the HVQFN16 package for PCA9554BS, there is metal plate under the IC. Where should this metal plate be connected to? (GND? or floating?)

   Answer: The metal plate on the bottom of the HVQFN devices is for heat sink purposes. It **must** be connected to ground to provide an effective heat dissipation. If left floating, reduced heat dissipation will be provided by the metal plate.

5. **Question:** I noticed in the PCA9554 data sheet that the standby current I_{stbl} is 550 µA to 700 µA, I_{stbh} is 0.25 µA to 1 µA, and I_{DD} is 104 µA to 175 µA. Why is I_{stbl} much bigger value than I_{stbh} and I_{DD}?

   Answer: The PCA9554 has internal 100 kΩ pull-up resistors between the I/O and V_{DD}. When those I/Os are connected to the ground, there is a current—I_{stbl}—flowing, thus explaining a higher value:

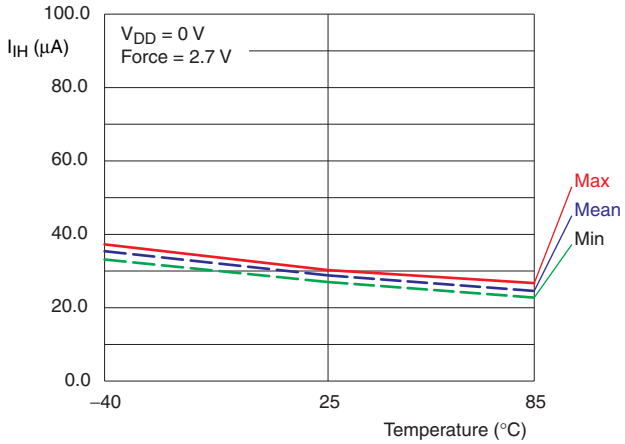
   $$8 \times \left(\frac{5.5\ \text{V}}{100\ \text{k}\Omega} \right) = 440\ \text{m}$$

   This current does not exist when the same nodes are connected to V_{DD} to measure I_{stbh}. If current consumption is a concern, use the PCA9534 instead. It is the same device but the internal pull-ups are removed and both I_{stbl} and I_{stbh} are the same.

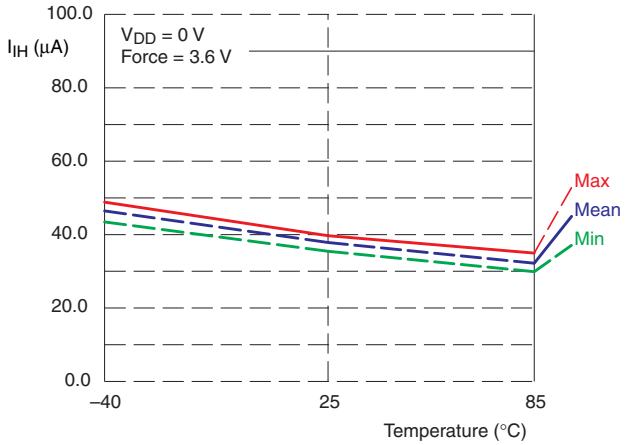
6. **Question:** I am using both PCA9535 and PCA9555 devices in my application and I would like to know what is the input leakage on the I/Os when an external voltage is forced to the pin while the power supply's V_{DD} is equal to 0 V?

   Answer: The PCA9535 leakage current is in the 5 nA to 10 nA over all the temperature range and an input voltage from 2 V to 5.5 V.

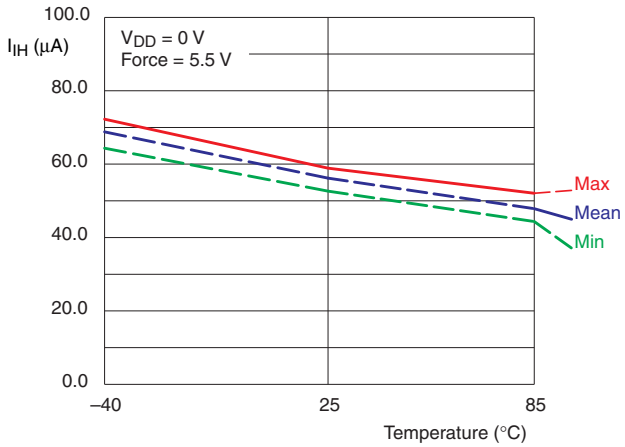
   The PCA9555 leakage current is different due to the internal pull-up resistors. Please refer to Figure 27 for more detail.



Temperature	–40 °C	25 °C	85 °C
Max	37.26 µA	30.26 µA	26.72 µA
Mean	35.40 µA	28.82 µA	24.53 µA
StdDev	0.972 µA	0.777 µA	0.811 µA
Min	33.13 µA	27.01 µA	22.76 µA



Temperature	–40 °C	25 °C	85 °C
Max	48.83 µA	39.67 µA	34.99 µA
Mean	46.45 µA	37.85 µA	32.21 µA
StdDev	1.272 µA	1.040 µA	1.069 µA
Min	43.45 µA	35.42 µA	29.88 µA



Temperature	–40 °C	25 °C	85 °C
Max	72.23 µA	58.90 µA	52.03 µA
Mean	68.78 µA	56.18 µA	47.83 µA
StdDev	1.922 µA	1.565 µA	1.597 µA
Min	64.34 µA	52.62 µA	44.34 µA

Fig 27. PCA9535 I/O leakage when V_{DD} = 0 V

10. Disclaimers

Life support — These products are not designed for use in life support appliances, devices, or systems where malfunction of these products can reasonably be expected to result in personal injury. Philips Semiconductors customers using or selling these products for use in such applications do so at their own risk and agree to fully indemnify Philips Semiconductors for any damages resulting from such application.

Right to make changes — Philips Semiconductors reserves the right to make changes in the products - including circuits, standard cells, and/or software - described or contained herein in order to improve design and/or performance. When the product is in full production (status 'Production'), relevant changes will be communicated via a Customer Product/Process Change Notification (CPCN). Philips Semiconductors assumes no responsibility or liability for the use of any of these products, conveys no licence or title under any patent, copyright, or mask work right to these products, and makes no representations or warranties that these products are free from patent, copyright, or mask work right infringement, unless otherwise specified.

11. Contents

Date of release: 20 January 2005
Document number: 9397 750 14523

Published in The Netherlands