Chipsmall
深圳市津利帝科技有限公司
Geelead Technology Limited
Chipsmall
采购部

PCU9669

Parallel bus to 1 channel Fm+ and 2 channel UFm I²C-bus controller

Rev. 2 — 1 July 2011 Product data sheet

1. General description

The PCU9669 is an advanced single master mode I²C-bus controller. It is a fourth generation bus controller designed for data intensive I²C-bus data transfers. It has three independent I²C-bus channels, one of them with data rates up to 1 Mbits/s using the Fast-mode Plus (Fm+) open-drain topology and two with a much larger transmit only transfer rate of up to 5 Mbits/s using the new Ultra Fast-mode (UFm) bus with push-pull topology. Each channel has a generous 4352 byte data buffer which makes the PCU9669 the ideal companion to any CPU that needs to transmit and receive large amounts of serial data with minimal interruptions.

The PCU9669 is a 8-bit parallel-bus to I²C-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 I²C-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 I²C-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 I²C-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 I²C-bus protocol converter and interface
- 5 Mbit/s unidirectional data transfer on Ultra Fast-mode (UFm) channel (push-pull driver)
- 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
- Individual 4352-byte buffers for the Fm+ and UFm channels for a total of 13056 bytes of buffer space
- 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 (Fm+ only)
- 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
PCU9669B	PCU9669	LQFP48	plastic low profile quad flat package; 48 leads; body 7 × 7 × 1.4 mm	SOT313-2

5. Block diagram

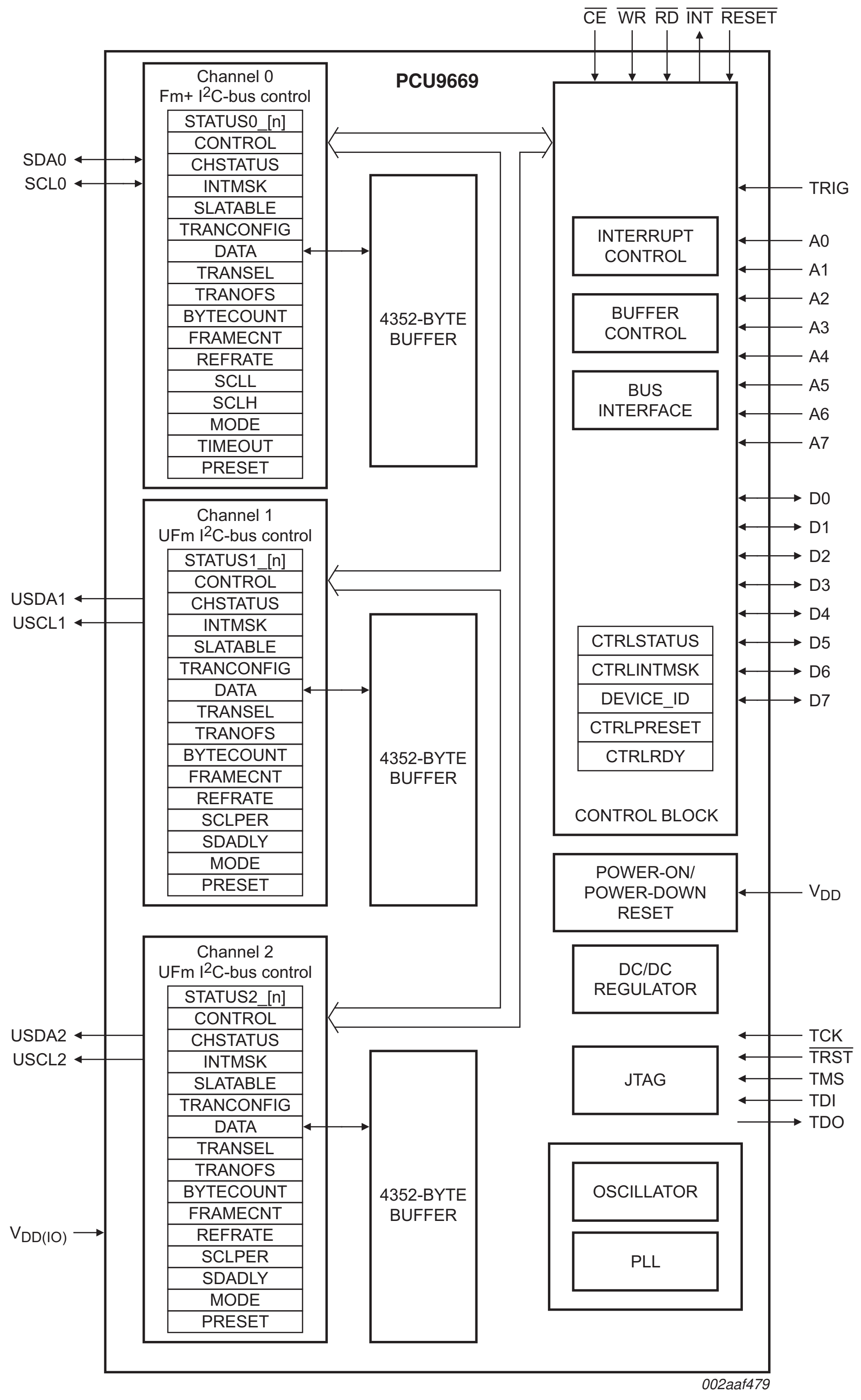


Fig 1. Block diagram

6. Pinning information

6.1 Pinning

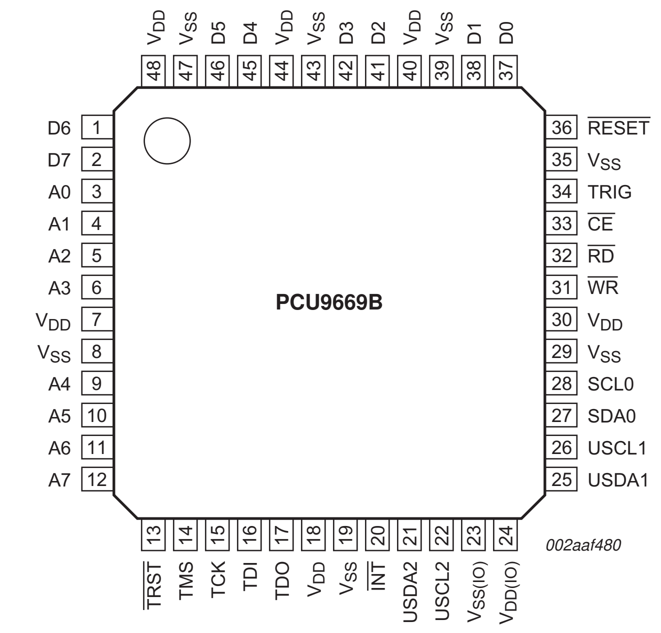


Fig 2. Pin configuration for LQFP48

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
$\overline{\text{TRST}}$	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.).
$\overline{\text{INT}}$	20	O	**Interrupt request:** Active LOW, open-drain, output. This pin requires a pull-up device.
USDA2	21	O	**Channel 2 Ultra Fast-mode I²C-bus serial data output.** Push-pull drive. No pull-up device is needed.
USCL2	22	O	**Channel 2 Ultra Fast-mode I²C-bus serial clock output.** Push-pull drive. No pull-up device is needed.
USDA1	25	O	**Channel 1 Ultra Fast-mode I²C-bus serial data output.** Push-pull drive. No pull-up device is needed.
USCL1	26	O	**Channel 1 Ultra Fast-mode I²C-bus serial clock output.** Push-pull drive. No pull-up device is needed.
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.
$\overline{\text{WR}}$	31	I	**Write strobe:** When LOW and $\overline{\text{CE}}$ is also LOW, the content of the data bus is loaded into the addressed register. Data are latched on the rising edge of $\overline{\text{WR}}$. $\overline{\text{CE}}$ may remain LOW or transition with $\overline{\text{WR}}$.
$\overline{\text{RD}}$	32	I	**Read strobe:** When LOW and $\overline{\text{CE}}$ 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 $\overline{\text{RD}}$. Data lines are driven when $\overline{\text{RD}}$ and $\overline{\text{CE}}$ are LOW. $\overline{\text{CE}}$ may transition with $\overline{\text{RD}}$.
$\overline{\text{CE}}$	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 $\overline{\text{WR}}$, $\overline{\text{RD}}$ and A0 to A7 inputs. When HIGH, places the D0 to D7 lines in the 3-state condition. During the initialization period, $\overline{\text{CE}}$ must transition with $\overline{\text{RD}}$ until controller is ready.
TRIG	34	I	**Trigger input:** provides the trigger to start a new frame.
$\overline{\text{RESET}}$	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. Sets the voltage reference point for V_{IL}/V_{IH} and the output drive rail for the UFm channel.
$V_{SS(IO)}$	23	power	**I/O supply ground.** Can be tied to V_{SS}.
V_{DD}	7, 18, 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, 19, 29, 35, 39, 43, 47	power	**Supply ground.** All V_{SS} pins should be connected together externally.

7. Functional description

7.1 General

The PCU9669 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 PCU9669 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 PCU9669 channels have individual 4352-byte data buffers (see Section 7.3.2 “Buffer sizes”) that allow 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 PCU9669 perform it without the intervention of the host each time a requested transaction is performed. The host can then perform other tasks while the PCU9669 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 sizes”.)

After sending all the commands and data it wanted to the I^{2}C-bus controller, the host could either continue to program data for other channels or write to the CONTROL register to begin data transmission on the current 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], STATUS1_[n], STATUS2_[n] — Transaction status registers”).

To program data for another channel, that channel is selected and data programmed as described above. One or more channels can be busy with serial transmission while additional parallel-bus data is sent to the buffer of an idle channel.

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 sizes

The PCU9669 channels have individual buffers assigned to them. The contents of the buffers should only be modified during channel idle states.

The memory allocation is 4352 bytes per channel.

The buffer sizes represent 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 STATUSx_[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 STATUSx_[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 STATUSx_[n] register (where 'n' is the buffer number of the slave) or the CHSTATUS or CTRLSTATUS registers.

7.5 Registers

The PCU9669 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 STATUSx_[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. PCU9669 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 registers													
0	0	channel 0 transaction number (hex)						STATUS0_[n]	R	no	individual transaction status (direct address)	00h	64
0	1	channel 1 transaction number (hex)						STATUS1_[n]	R	no	individual transaction status (direct address) ([7:2] = 0 in UFm)	00h	64
1	0	channel 2 transaction number (hex)						STATUS2_[n]	R	no	individual transaction status (direct address) ([7:2] = 0 in UFm)	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. PCU9669 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)
Channel 1 (UFm) registers													
1	1	0	1	0	0	0	0	CONTROL	R/W	yes[1]	channel 1 control ([7] = 1)	00h	1
				0	0	0	1	CHSTATUS	R	no	channel 1 status ([5:1] = 0 in UFm)	00h	1
				0	0	1	0	INTMSK	R/W	yes	channel 1 interrupt mask ([5:1] = don't care)	00h	1
				0	0	1	1	SLATABLE	R/W	no	channel 1 slave address table (auto-increment)	00h	64
				0	1	0	0	TRANCONFIG	R/W	yes, for TRANCOUNT[2]	channel 1 transaction configuration (auto-increment)	00h	65
				0	1	0	1	DATA	R/W	yes	channel 1 data (auto-increment)	00h	bufsize[3]
				0	1	1	0	TRANSEL	R/W	yes	channel 1 transaction data buffer select	00h	1
				0	1	1	1	TRANOFS	R/W	yes	channel 1 transaction data buffer byte offset	00h	1
				1	0	0	0	BYTECOUNT	R	no	channel 1 transmitted byte count (auto-increment)	00h	64
				1	0	0	1	FRAMECNT	R/W	no	channel 1 frame count	01h	1
				1	0	1	0	REFRATE	R/W	no	channel 1 frame refresh rate	00h	1
				1	0	1	1	SCLPER	R/W	no	channel 1 clock period	20h	1
				1	1	0	0	SDADLY	R/W	no	channel 1 SDA delay	08h	1
				1	1	0	1	MODE[4]	R/W	no	channel 1 mode	83h	1
				1	1	1	0	-	-	-	reserved	00h	1
				1	1	1	1	PRESET	R/W	yes	channel 1 parallel reset	00h	1

Table 3. PCU9669 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)
Channel 2 (UFm) registers													
1	1	1	0	0	0	0	0	CONTROL	R/W	yes[1]	channel 2 control ([7] = 1)	00h	1
				0	0	0	1	CHSTATUS	R	no	channel 2 status ([5:1] = 0 in UFm)	00h	1
				0	0	1	0	INTMSK	R/W	yes	channel 2 interrupt mask ([5:1] = don't care)	00h	1
				0	0	1	1	SLATABLE	R/W	no	channel 2 slave address table (auto-increment)	00h	64
				0	1	0	0	TRANCONFIG	R/W	yes, for TRANCOUNT[2]	channel 2 transaction configuration (auto-increment)	00h	65
				0	1	0	1	DATA	R/W	yes	channel 2 data (auto-increment)	00h	bufsize[3]
				0	1	1	0	TRANSEL	R/W	yes	channel 2 transaction data buffer select	00h	1
				0	1	1	1	TRANOFS	R/W	yes	channel 2 transaction data buffer byte offset	00h	1
				1	0	0	0	BYTECOUNT	R	no	channel 2 transmitted byte count (auto-increment)	00h	64
				1	0	0	1	FRAMECNT	R/W	no	channel 2 frame count	01h	1
				1	0	1	0	REFRATE	R/W	no	channel 2 frame refresh rate	00h	1
				1	0	1	1	SCLPER	R/W	no	channel 2 clock period	20h	1
				1	1	0	0	SDADLY	R/W	no	channel 2 SDA delay	08h	1
				1	1	0	1	MODE[4]	R/W	no	channel 2 mode	83h	1
				1	1	1	0	-	-	no	reserved	00h	1
				1	1	1	1	PRESET	R/W	yes	channel 2 parallel reset	00h	1

Table 3. PCU9669 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	08h	
				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	E9h	
				0	1	1	1	CTRLPRESET	R/W	yes	master parallel reset	00h	1
				1	1	1	1	CTRLRDY[5]	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 sizes” for channel memory allocation.

[4] Unused bits in the UFm register set will return 0b when read and writes will be ignored.

[5] 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], STATUS1_[n], STATUS2_[n] — Transaction status registers

STATUS0_[n], STATUS1_[n], and STATUS2_[n] are 8-bit × 64 read-only registers that provide 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 STATUSx_[n] register will clear its status. To clear all the STATUSx_[n] registers, a byte-by-byte read of all STATUSx_[n] registers is required. The controller will auto-clear the STATUSx_[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. STATUSx_[n] - Transaction status code register bit description

Bit	Symbol	Description
7:5	ST[7:5]	always reads 000
4	RSN[1]	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[1]	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[1]	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.

[1] Does not apply to the UFm channel.

Remark: When STATUSx_[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 STATUSx_[n] register will contain:

STATUSx_[0] = 0

STATUSx_[1] = 0

STATUSx_[2] = 0

STATUSx_[3] = 0

:

After STA is set:

STATUSx_[0] = 2

STATUSx_[1] = 1

STATUSx_[2] = 1

STATUSx_[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; Channel 1 = D0h; Channel 2 = E0h.

*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 (does not apply to the UFm channel). 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; Channel 1 = D0h; Channel 2 = E0h.

*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[5:3] bits.

7.5.1.3 CHSTATUS — Channel status register

CHSTATUS is an 8-bit read-only register that provides status information for a given 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; Channel 1 = D1h; Channel 2 = E1h.

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[1]	Write Error detected in transaction. An SLA NACK or data NACK was detected in a write transaction of the sequence.
4	RE[1]	Read Error detected in transaction. An SLA NACK was detected in a read transaction of the sequence.
3	DAE[1]	Bus error, SDA stuck LOW.
2	CLE[1]	Bus error, SCL stuck LOW.
1	SSE[1]	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.

[1] Does not apply to UFm channel. Always read as logic 0.

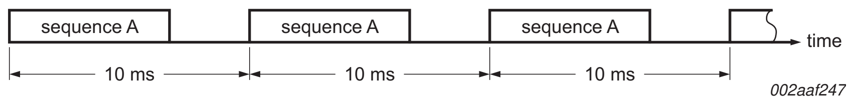
The DAE, CLE and SSE bits correspond to bus error states, and the FE bit corresponds to host programming errors.

DAE - SDA error bit: This bit indicates that the SDA line is stuck LOW when the PCU9669 is trying to send a START condition.

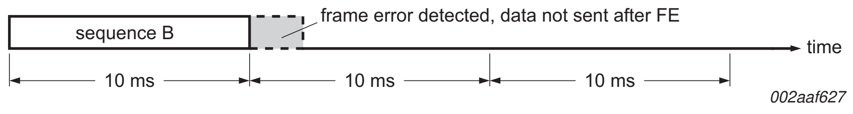
CLE - SCL error bit: This bit indicates that the SCL line is stuck LOW.

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 PCU9669 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.

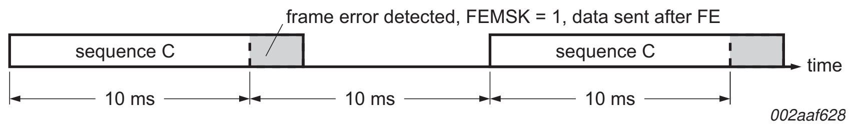


a. Sequence fully executed within the period programmed in REFRATE register



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 in a given 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; Channel 1 = D2h; Channel 2 = E2h.

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[1]	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[1]	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; Channel 1 = D2h; Channel 2 = E2h.

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.

[1] Does not apply to UFm channel.

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; Channel 1 = D3h; Channel 2 = E3h.

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; Channel 1 = D4h; Channel 2 = E4h.

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; Channel 1 = D5h; Channel 2 = E5h.

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 I^{2}C-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; Channel 1 = D6h; Channel 2 = E6h.

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; Channel 1 = D7h; Channel 2 = E7h.

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; Channel 1 = D8h; Channel 2 = E8h.

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; Channel 1 = D9h; Channel 2 = E9h.

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; Channel 1 = DAh; Channel 2 = EAh.

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 and SCLPER, SDADLY — 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 and the for the Ultra Fast-mode channel by the SCLPER and SDADLY registers. They define the data rate for the serial bus of the PCU9669. 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 or SCLPER registers will cause the part to operate at the respective 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. For UFm mode, the clock is a fixed 50 % duty cycle defined by the SCLPER. In both cases 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 I²C-bus mode is selected. See Section 7.5.1.14 "MODE — I²C-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:

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

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:

$$SCLL = 0.6 \times TOTAL\_SCLLH \tag{2}$$

$$SCLH = 0.4 \times TOTAL\_SCLLH \tag{3}$$

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