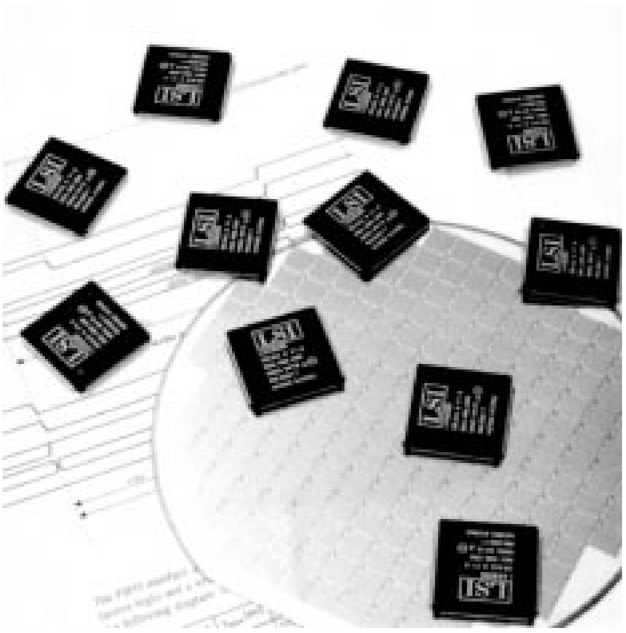
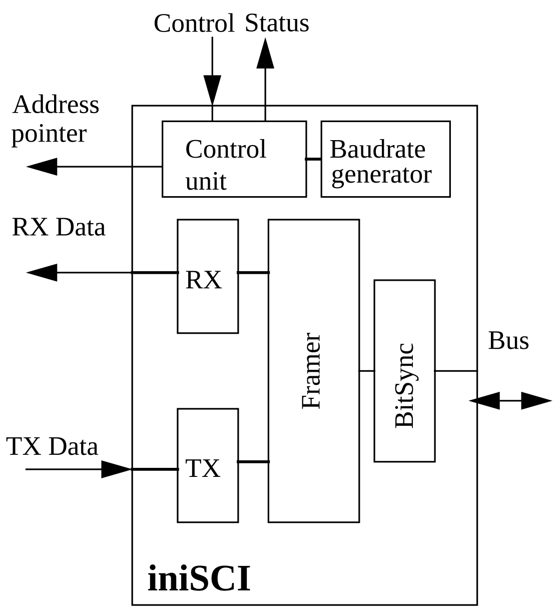
iniSCI Master IIC compatible data sheet



Features:

- I2C-Compatible
- Single Master
- Programmable Baud Rate Generator (390 - 100k bps with 1us Clock Enable)
- 2 MHz Clock Required for 100k bps
- Automatic Incremented Address Pointer
- Message Acknowledgement
- Customizable for Special Requirements
- Structured Model Description (SD)
- Synchronous, Synthesisable VHDL Model

Blockdiagramm:



INICORE - the reliable Core and System Provider. We provide high quality IP, design expertise and leading edge silicon to the industry.

The **iniSCI** core is a synthesizable, flexible, and structured VHDL implementation of a single Master serial controller interface (SCI) that uses a two-wire bus for communicating between integrated circuits or standard peripherals like smart LCDs and keypads. The core contains the entire physical and data link layers, allowing it to handle bus timing and frame generation/extraction, and thus reducing overhead from the system application. A flexible parallel interface is used for on-chip data transfer, facilitating integration of the iniSCI core to the rest of the system.
iniSCI is structured into a BitSync bus interface module and receiver, transmitter, and framer modules. This modular structure facilitates an understanding of the core's functionality, thus simplifying customization.
INICORE created the structured VHDL I2C compatible model for simulation and synthesis on any target technology.
INICORE provides also iniSCI as a core concept, where documentation, VHDL models, testbenches and samples allow you to build your own system with our application proved design parts inside.

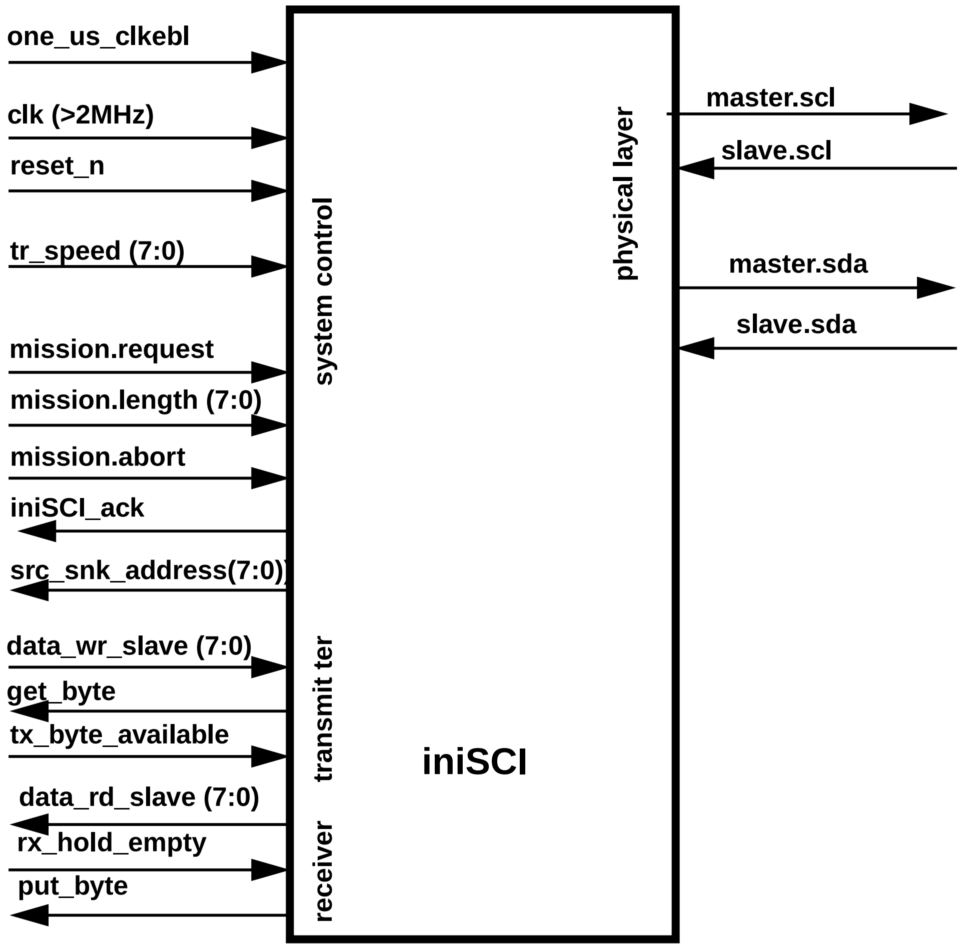
INICORE

US Sales Office:
INICORE INC.
5600 Mowry School Road, Suite 180,
Newark, CA 94560
Tel: 510 445 1529 Fax: 510 656 0995
E-mail: ask_us@inicore.com
Web: www.inicore.com

INICORE AG
Mattenstrasse 6a,CH-2555 Brügg, Switzerland
Tel: ++41 32 374 32 00, Fax: ++41 32 374 32 01
E-mail: ask_us@inicore.ch
Web: www.inicore.ch

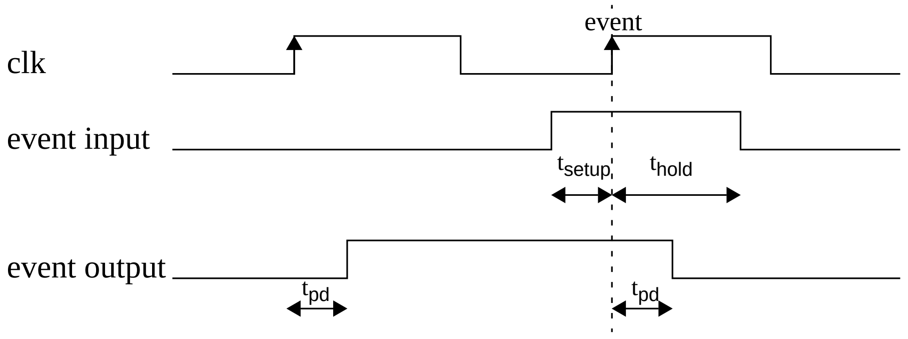
Ref.Nr.: 401-DS-12 / 03/2000

1 Overview



1.1 Event communication

For communicating events, the iniSCI core uses or produces always active ‘1’ pulses, which are activated for only one clk cycle (ex. one_us_clkebl). In the inactive state, they remain low with respect to the rising clk edge, so glitches may occur. For communicating over clock domains, these events must be synchronized first!



The parameters t_{setup}, t_{hold} and t_{pd} are technology dependent and must be determined according to the choose technology.

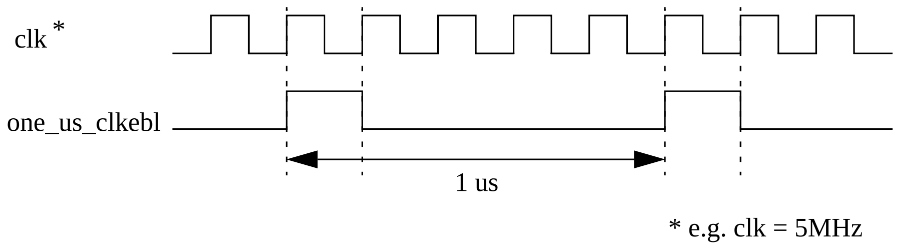
2 IO description

The following part lists the input and output ports of the iniSCI core and gives a short overview of their functionality.

2.1 General inputs

These pins are used to clock and initialize the whole iniSCI core. There are no other clocks in this core.

pin name	type	description
clk	in	System clock for the whole iniSCI
one_us_clkebl	in	Clock enable pulse (stable for one clock cycle) each 1 us.
reset_n	in	Asynchronous system reset, active low

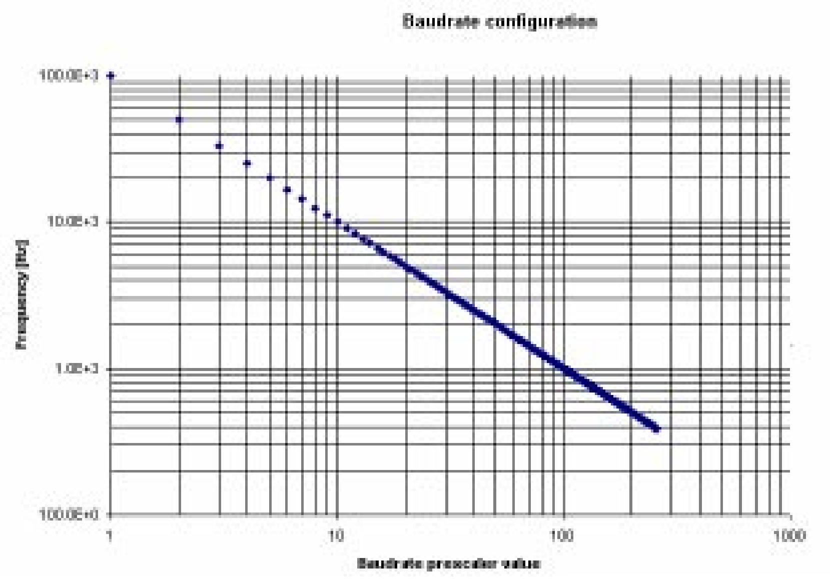


2.2 Configuration

The iniSCI has only one configuration register. The speed configuration register (abbreviation V) is 8 bit wide.
The configuration value (V) can be calculated using the following equation:

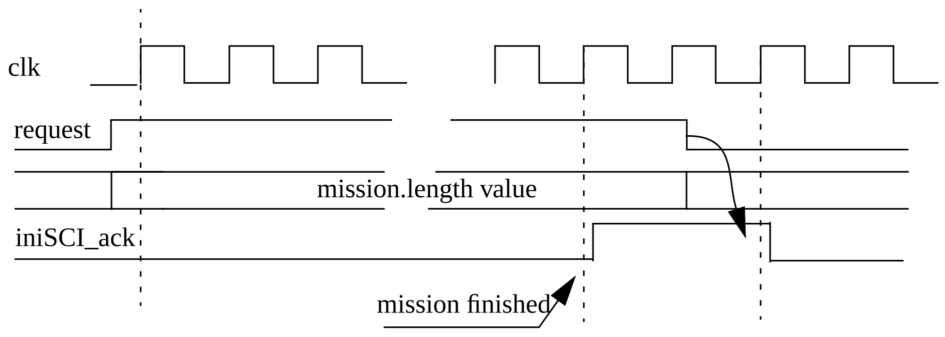
Bd = 1 / (1 us * V*10) If V = 255 then Bd = 392 Bd (min. speed)
If V = 1 then Bd = 100 kBd (max. speed of IIC protocol).

pin name	type	description
tr_speed [7:0]	in	Baudrate prescaler value (range: 1 to 255)



2.3 Mission control

The mission control signals are used to start, stop the transmisison of a frame and to control its length.



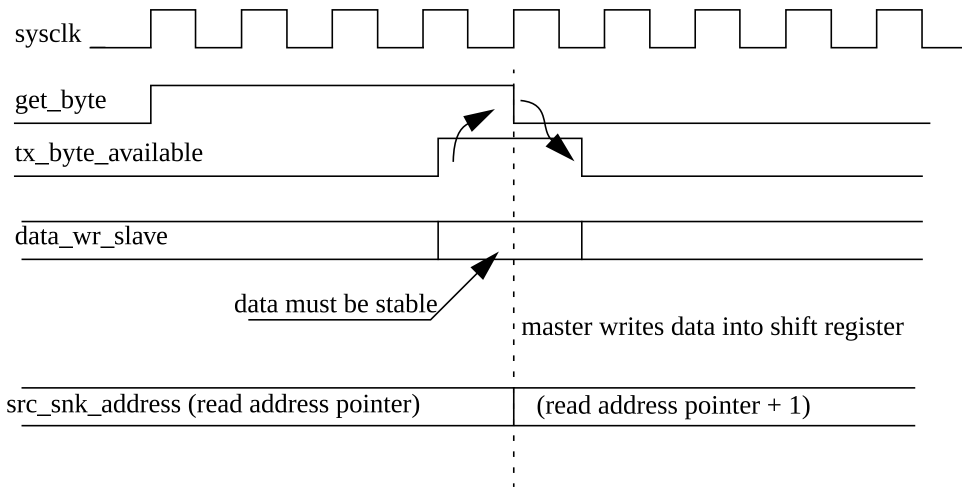
pin name	**type**	**description**
mission.length [7:0]	in	Length of the frame to be transmitted or received. The **missionlength may not be 0!** while a request is asserted. Example: If mission.length is 1 then only the controlbyte (Address + RdWrN Bit) of the I2C protocol is sent. If the missionlength is 2 then controlbyte is sent and one I2C data byte is transmitted or received.
mission.request	in	‘1’: starts the I2C read/write session ‘0’: - <see read/write slave waveforms >
mission.abort	in	‘1’: abort the frame at the end of current byte. ‘0’: no abort <see abort waveforms >
iniSCI_ack	out	‘1’: signals a ‘mission complete’ or ‘abort done’ . <see also abort handling> It returns to zero after abort and/or request are reseted (abort and request must be zero). <see read/write slave waveforms >

2.4 Transmitter interface

For transmitting data, a parallel event controlled interface is used. It is an efficient way to embed the iniSCI into systems as well as connecting simple or complex specific interfaces to it.

pin name	type	description
data_wr_slave [7:0]	in	8 bit data to be transmitted. The data must be valid when tx_byte_available is active. The first TX byte is the slave address (7bit) and the Rd/WrN bit (LSB). It is called the control byte.
get_byte	out	‘1’: the I2C master requests data from the user. It returns to ‘0’ when tx_byte_available acknownledges with ‘1’.
tx_byte_ available	in	‘1’: acknowledges a get_byte from the master and says that the data byte is stable. It has the effect that the master writes the data into the shift register.
src_snk address	out	read and write address pointer If iniSCI master is transmitting (writing to slave) then the src_snk_address works as a read pointer. It starts with “00000000”. If iniSCI master is receiving data from slave then the src_snk_address works as a write pointer. It starts with “00000001”. This way the control byte will not be overwritten when shared memory is used. Note: In case that no slave is responding, the master resends the control byte automatically. The control byte has to be located at address “00000000”.

The following diagram shows a typical case:


sysclk
get_byte
tx_byte_available
data_wr_slave
data must be stable
master writes data into shift register
src_snk_address (read address pointer)
(read address pointer + 1)

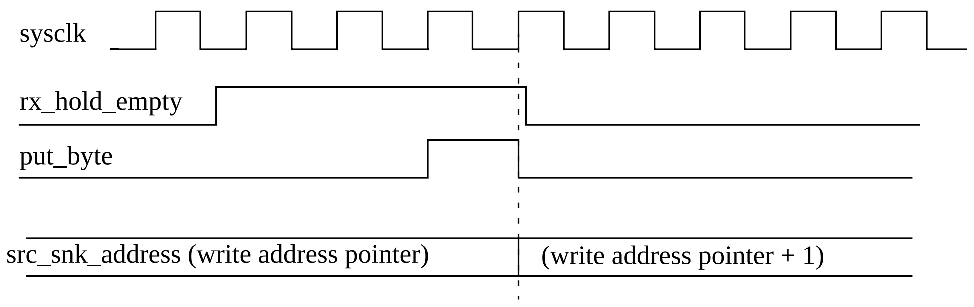
The transmitter path stores the incoming byte in the shift register by means of the tx_byte_available signal and starts the transmitting activity. get_byte goes low after the tx_byte_available is high.

2.5 Receiver interface

For receiving data, the same type of interface is used as in the transmitter path.

pin name	type	description
data_rd_slave [7:0]	in	8 bit data received from the slave.
put_byte	out	Write enable for an external hold register. If rx_hold_empty is ‘0’ the put_byte is not generated.
rx_hold_empty	in	‘1’: then a external hold register is ready to be written with the new data received from slave. ‘0’: the hold register is full. Master may not write received data.
src_snk address	out	read and write address pointer If iniSCI master is transmitting (writing to slave) then is the src_snk_address the read pointer. It starts with “00000000”. If iniSCI master is receiving data from slave then is the src_snk_address a write pointer. It starts with “00000001”, so the control byte will not be overwritten when shared memory is used. Note: In case that no slave is responding, the master resends the control byte. For this reason must the control byte be located in address “00000000”.

The following diagram shows a typical case:

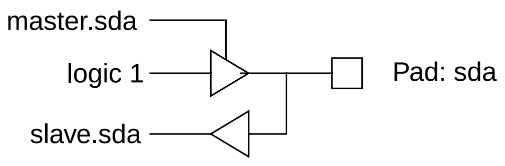

sysclk
rx_hold_empty
put_byte
src_snk_address (write address pointer)
(write address pointer + 1)

2.6 Serial bus

These are the signals to the IIC bus side and should be connected to a open collector/drain buffer.

pin name	type	description
slave.scl	in	serial clock input
master.scl	out	serial clock output
slave.sda	in	serial data input
master.sda	out	serial data output

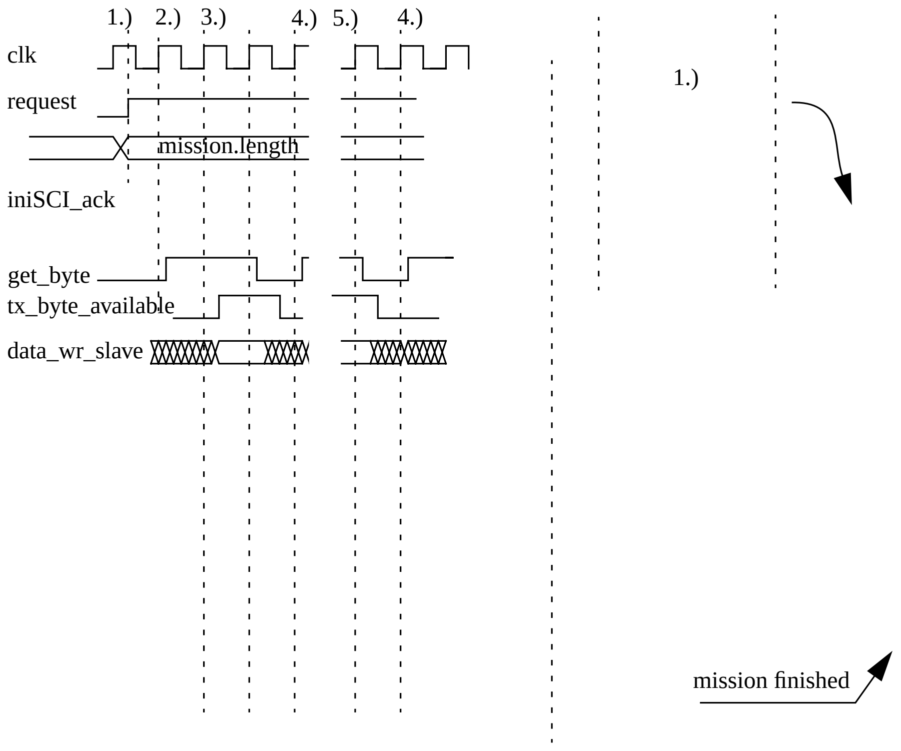
Instead of an open collector/drain buffer a tristateable bi-buffer can be used:



3 Waveforms

3.1 Write slave

User site signals:



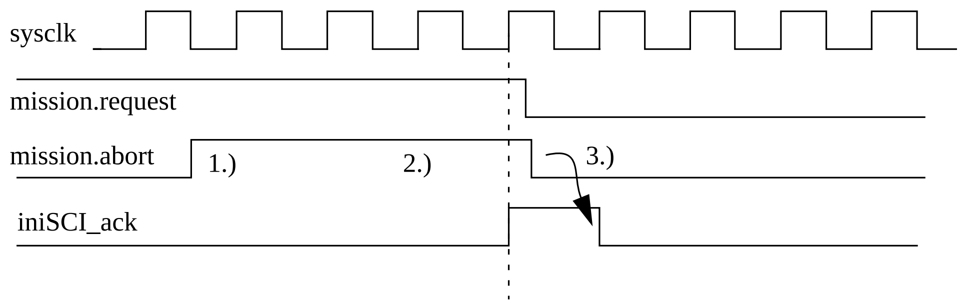
1.) Set mission length and request. Lenght must be stable during transmission.
2.) iniSCI master sets get_byte signal and waits for the command byte.
3.) When valid command byte set tx_byte_available.
4.) iniSCI master sets get_byte signal and waits for the (n-th)data byte.
5.) When valid (n-th)data byte set tx_byte_available.
Point 4. & 5. are repeated until the entire frame is sent.
6.) iniSCI master acknowledges the transfer by asserting the iniSCI_ack and deasserts it again when the request signal is low (handshake).

3.2 Read slave

4 Special event handling

4.1 Abort transmission

1.) The abort may be set when ever you want, but the master stops only at full byte time. That means that it will not stop in a current byte transmission. The stop occurs at the end of the transmitting byte.
2.) The abort signal must rest asserted until the iniSCI_ack acknowledges the abort.
3.) After resetting the abort signal, the iniSCI will be deasserted.



4.2 No slave acknowledge

In case that no slave is responding, the master resends the slave controlbyte until an abort is generated or a slave acknowledge occurs. The slave controlbyte contains the 7 bit of the slave address and the RdWrN bit. The master reads each time this byte before sending it. This byte must be in address location “0000000” (Address pointer = 0).

The external logic must have a time out to stop the transmission if no slave is responding. For this case the logic must check the iniSCI_ack. If no iniSCI_ack occurs before time out, the logic may generate an abort to stop the master.

4.3 Write data not available

The iniSCI master will hold the serial clock line (scl) low until the next data is available.