Interface with OID decoder "SN9P701"

1. Hardware configuration

◆ OID decoder (SN9P701-00X) communicates with DSP/MCU by a 4-wire cable, Fig.1 . SDIO in 4-wire cable is pulled high by a external pull-up resistor.

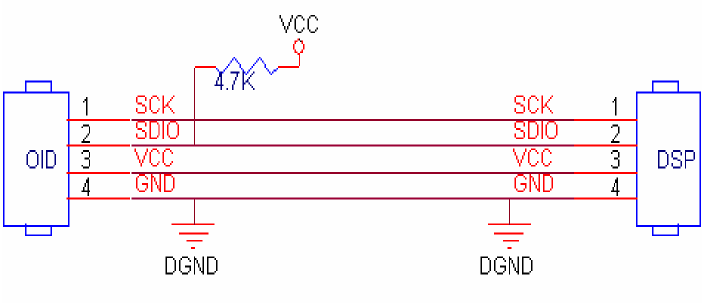


Fig.1 4-Wire Cable between OID and DSP

◆ SCK and SDIO on 4-wire cable are used to transfer data between OID Decoder and DSP. The transmission interface is called the **2-Wire Interface**. Features of SCK and SDIO are shown in Table.1.

	SCK	**SDIO**
Usage	Serial Clock	Serial Data
Direction	DSP → OID	DSP ↔ OID
Pull High	No	Yes
Default State	Low	Pulled High

Table.1 Features of SCK and SDIO

◆ Data transferred between OID and DSP are commands, indexes, and status of OID. Features of data are shown in Table.2.

	OID → DSP	**OID ← DSP**
Length	23 bits	8 bits
Usage	Indexes / Commands / Status of OID	Commands

Table.2 Data between OID and DSP

2. 2-Wire Interface

General Description

2-Wire Interface is used between OID decoder and DSP. Only 2 lines are used to transfer data through this protocol. Data are transferred from MSb to LSb serially on SDIO. DSP plays the **Master** and OID decoder plays the **Slave** of 2-Wire Interface.

Master & Slave

◆ Master is the leading part in 2-Wire Interface. Only Master can do following actions in 2-Wire Interface :

■ Initiate a transfer cycle

■ Generate clock signals on SCK

■ Decide the direction of a transfer cycle

◆ Slave is the passive part in 2-Wire Interface. It should be sensitive to the action of Master. If there are data ready to send to Master, Slave can send a **transfer request** (pulling low SDIO line) to inform Master.

Conditions

◆ Default : SCK is kept low by Master, and SDIO is released and pulled high by external pull-up resister.

◆ Start : Master initiates a transfer cycle by changing SCK from low to high.

◆ End : If Master keeps low on SCK over 64µs+20% (~1024*1.2 Clocks @ 16MHz), Slave supposes that the transfer cycle is finished. Please refer to Fig.2.

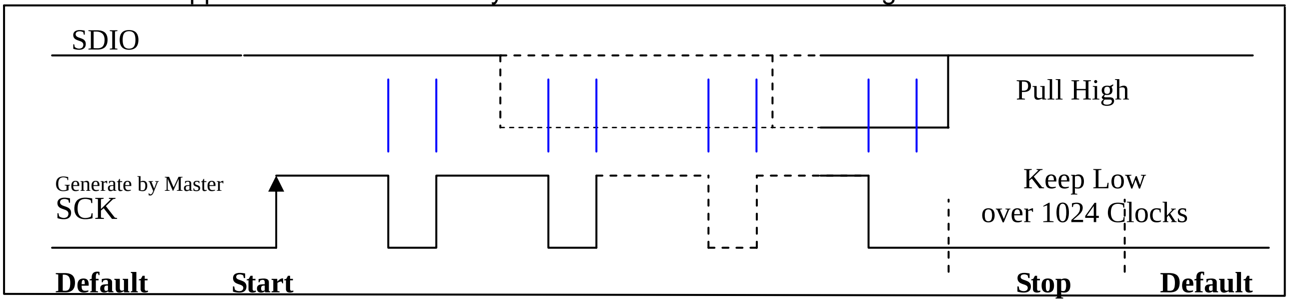


Fig.2 Conditions of 2-Wire Interface

【NOTE】

When Master keeps high on SCK, Slave will wait for another high to low edge on SCK. Slave will keep waiting the negative edge on SCK, if Master keeps high on SCK for a long time.

Read & Write Bits

Bits of data are transferred from MSb to LSb on SDIO serially. Level change on SDIO can only occur when SCK is high. As SCK is low, level change on SDIO is forbidden. In another words, it's available to capture bits on SDIO when SCK is low, and it's able to put bits on SDIO when SCK is high. Level changes on SDIO should occur right after positive edge on SCK line. Please refer to Fig.3. In order to communicate successfully, Master must obey the following rule to control SCK.

- Master must keep SCK high over 2μs (32 Clocks @16MHz) for level changing on SDIO
- Master must keep SCK low over 2μs (32 Clocks @16MHz) but shorter than 64μs-20% (1024*0.8 Clocks @16MHz) for bit capture on SDIO.

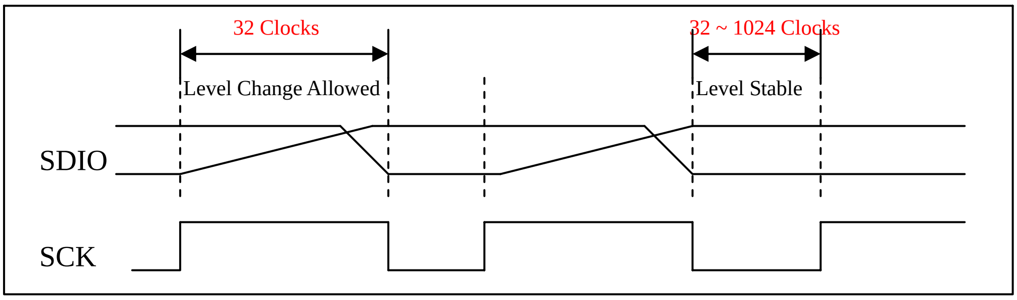


Fig.3 Valid data

Read Cycle

- Master read data from Slave side (Slave transmits data to master). Please refer to Fig.4.
- There are 23 bits transferred on SDIO.
  - R/W control bit: SDIO is kept low by Master to indicate the read cycle
  - Bit_0 ~ Bit_22 : Data, generated by Slave
- ~~SDIO is released by Master and pulled high by external pull-up resister after the bit_22 captured.~~

Read Cycle					
Line	**Part**	**Start**	**R/W Bit**	**Bit_0 ~ Bit_22**	**Stop**
SDIO	Master	Z	L	Z	Z
	Slave	L	Z	23-bit Data	Z

Before Read cycle, Slave generates transfer request (pulling low SDIO line) to inform Master. After Master generates start condition and placing read control bit on SDIO line, Slave begins to place and keep each bit of serial data on SDIO line after every positive edge generated by Master on SCK line. After transmitting, SCK should be kept low by Master as a stop condition and SDIO line is released by Slave and pulled high by external pull-up resistor.

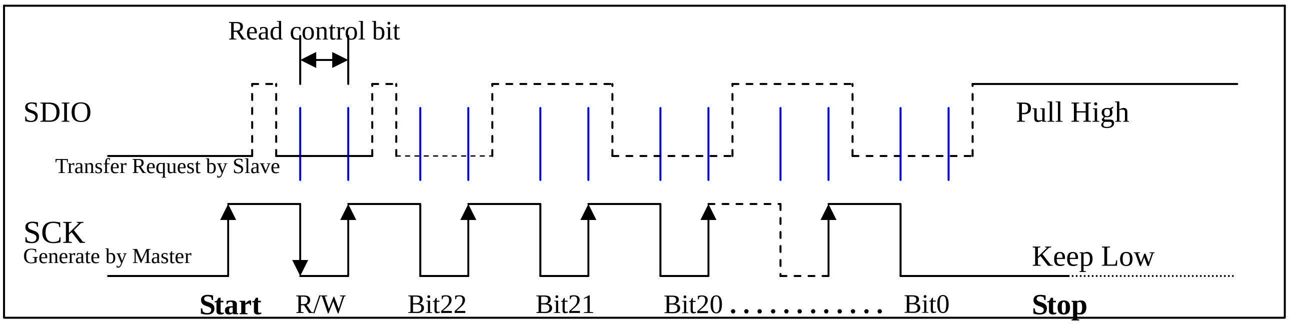


Fig.4 Read Cycle

【NOTICE】

In both read and write cycle, Slave will release SDIO when Master generates a "Start" rising edge on SCK. If Master pulls low SDIO after Slave releasing it, there might be a pulse-like waveform on SDIO at the 1st clock of the read cycle. For the same reason, there might be another pulse-like waveform on SDIO at the 2nd clock of the read cycle because Slave takes control of SDIO after Master releasing it.

Write Cycle

- Master transmits data to Slave. Please refer to Fig.5.
- There are 8 bits transferred on SDIO.
  - R/W control bit : SDIO is pulled high by Master to symbol the write cycle
  - Bit_0 ~ Bit_7 : Data, generated by Master
- ~~SDIO is pulled high by external pull-up resister after the bit_7 captured.~~

Write Cycle					
Line	**Part**	**Start**	**R/W Bit**	**Bit_0 ~ Bit_7**	**Stop**
SDIO	Master	Z	H	8-bit Data	H
	Slave	L / Z	Z	Z	Z

During Write cycle, master generates positive edge on SCK line and places Write control bit on SDIO line. Then master transmits each bit of 1 byte data to slave on SDIO. And slave starts to receive data after each negative edge on SCK line following Write control bit on SDIO line. After transmitting, SCK line keeps low as a stop condition and SDIO line is pulled high by pull-up resistor.

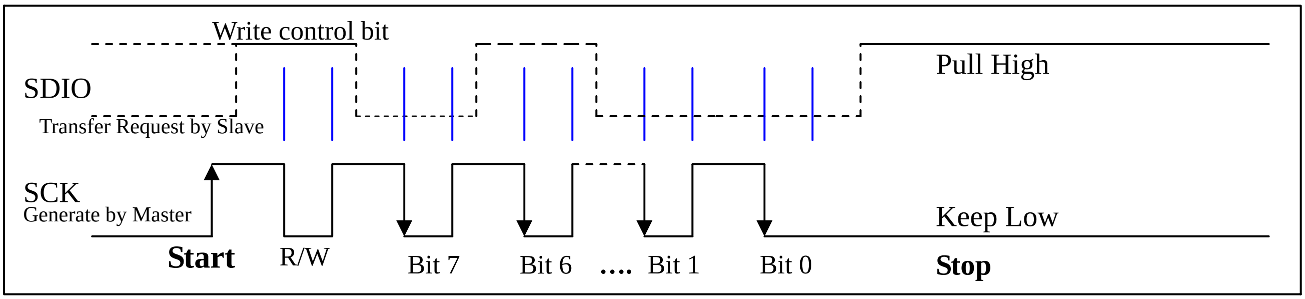


Fig.5 Write Cycle

【NOTICE】

SDIO should be checked before every Write cycle. If SDIO is low (transfer request) before Write cycle, Read cycle should be done before Write cycle.

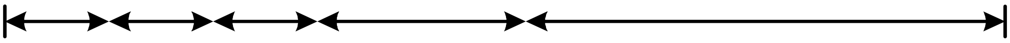
3. Data Format

Data From OID

◆ **Serial data packet from OID decoder is shown as below :**

(MSB first when transmitting data)

Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	. . . .	Bit0



Bit17 ~ Bit0：Index number

Bit18：Reserved,

Bit19：Reserved,

Bit20：**Battery status** (1: battery high，0:battery low)

Bit21：**Command／index** (1: command, 0: index)

Bit22：**Reserved bit** (1:OID，0: reserved)

- Bit17 and bit0 are reserved to show index number whenever a new index detected.
- Bit20 is used to show battery status.
  SN9P701 checks battery status every 10 sec and reports it in the Bit20 of data packet to DSP/MCU. If there is an index happened to change in the same time, then the field of Bit17~Bit0 of the data packet will be filled by "the new index number", else it will be replaced by "DontCare".
- Bit21 is used to indicate the data packet is an index or a command.
- Bit22 is used to indicate the data packe is a OID data or just a undefined number.
  For example, if OID index 0 is sent to DSP with battery high, and key_0 and key_1 are released, the received data will be 0x500000.

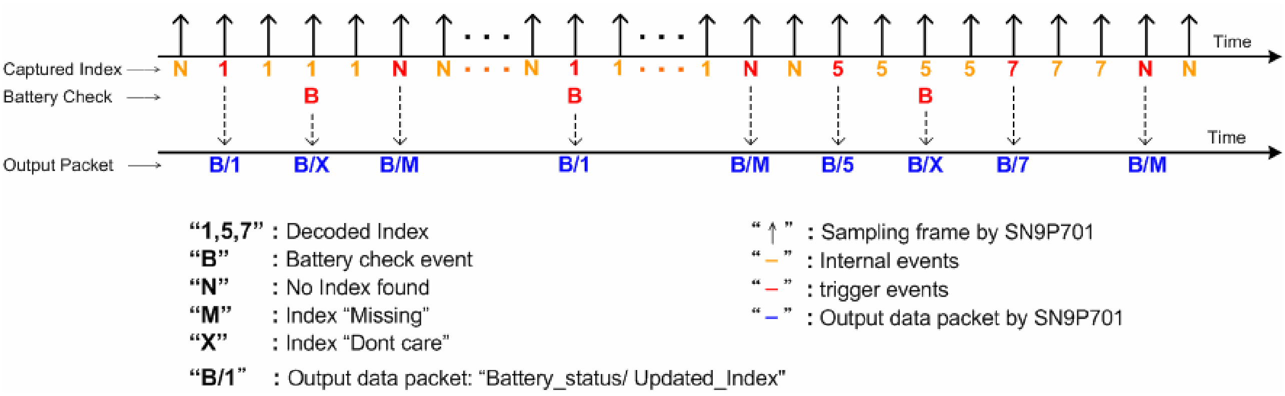
◆ **The time to send data to DSP/MCU**

OID decoder keeps sampling image and decodes it into digits (index) after power on. Every decoded result will be compared with previous one. If they are different, new result will be reported to DSP/MCU via data packet.

OID decoder sends data to DSP/MCU when one of following events happened.

1. New index has been detected： "No index" → "index found" or "index_A" → "index_B"
2. Index missing： "index found" → "No index found"
3. Decoder power on/off , reset
4. Internal battery_status check (every 10 sec)

Fig.6 Timing diagram for sending data packet to DSP/MCU


Time
Captured Index
N 1 1 1 1 N N · · · N 1 1 · · · 1 N N 5 5 5 5 7 7 7 N N
Battery Check
B B B
Time
Output Packet
B/1 B/X B/M B/1 B/M B/5 B/X B/7 B/M
“1,5,7” : Decoded Index
“B” : Battery check event
“N” : No Index found
“M” : Index “Missing”
“X” : Index “Dont care”
“B/1” : Output data packet: “Battery_status/ Updated_Index”
“↑” : Sampling frame by SN9P701
“–” : Internal events
“–” : trigger events
“–” : Output data packet by SN9P701

- **Index Type (Bit17~Bit0)**

Until now SN9P701-00x supports index range from **0x000000 to 0x00ffff.** There are totally 65,536 indexes. The useful index range is divided into following types:

OID_IndexType_Normal	0x00000 ~ 0x3ffef
Reserved for internal use	0x3fff0~0x3fffa
OID_IndexType_Normal _DontCare	0x3fffb
OID_IndexType_Normal_Missing	0x3fffc ~ 0x3ffff

- OID_IndexType_Normal (0x000000 ~ 0x03ffef)

  Index in the range is according to the dot pattern that retrieved from captured image. Currently only 0x000000 ~ 0x00ffff are supported by SN9P701 and OID EZ compiler.

  Index range 0x010000 ~ 0x03ffef are reserved for extended usage.

  Note:

  OID Authoring tool and EZ compiler defines number 0x00fcff to 0x00fdfe as page code. Number 0x00fdff to 0x00fffe is reserved for system. Number 0x00ffff is idle code.

- OID_IndexType_DontCare (0x03fffb)

  This “DontCare” index is used to inform DSP/MCU, that the index contained in this data packet is useless. Normally “DontCare” will be filled in Bit17~Bit0 to replace a normal index when the following condition happens:

  1. Key_status is changed and no newer index is found.

  2. Battery is checked and no newer index is found

  For example, if OID index DontCare is sent to DSP with battery high, and key_0 and key_1 are released, the received data will be 0x503fffb.

- OID_IndexType_Missing (0x03fffc ~ 0x03ffff)

  The indexes are used to inform DSP/MCU the captured image can’t be recognized. User can ignore it because it happens only when OID pen is moved out of the paper, or decoder can’t recognize current picture.

- Reserved for internal usage (0x3fff0 ~ 0x3fffa)

  Those indexes are reserved by Sonix compiler. Please don’t define it to make application.

◆ Commands From SN9P701 To DSP/MCU

OIDCmd_PowerOn	0x60fff8
OIDCmd_PowerDown	0x60fff7
OIDCmd_SystemReset	0x60fff1

- OIDCmd_PowerOn = 0x60fff8

  The command is used to inform DSP that OID decoder has already turned on.

- OIDCmd_PowerDown = 0x60fff7

  The command is used to inform DSP that OID decoder is going to sleep.

- OIDCmd_SystemReset = 0x60fff1

  The command is used to inform DSP that OID decoder has something wrong and has re-initiated itself. DSP shall make setup again to OID decoder if necessary.

Commands to SN9P701

Serial data to SN9P701 are 8-bits commands only which are used to set up functions of decoder.

◆ **Serial data format to SN9P701 is shown as below :** (MSB first when transmitting data)

Bit7	Bit6	. . . .	Bit1	Bit0

◆ **User to OID command**

UserCmd_PowerDownOID	0x56
UserCmd_AutoSleepFunEnable	0xa0
UserCmd_AutoSleepFunDisable	0xa3
UserCmd_TriggerToClearAutoSleepTimer	0xa6
UserCmd_ClearAutoSleepTimerIfOIDDetect	0xac
UserCmd_NonClearAutoSleepTimerIfOIDDetect	0x50
UserCmd_CheckOIDStatus	0x30

- UserCmd_PowerDownOID

  Power down OID decoder by DSP/MCU.

- UserCmd_AutoSleepFunEnable (Default setting in OID decoder)

Enable auto-sleep function of OID decoder by DSP/MCU. The default setting of pen is “Enable” and internal auto-sleep timer is set as 5 minutes.

- UserCmd_AutoSleepFunDisable

  Disable auto-sleep function of OID decoder by DSP/MCU.

- UserCmd_TriggerToClearAutoSleepTimmer

  Clear auto-sleep timer by DSP/MCU.

- UserCmd_ClearAutoSleepTimmerIfOIDDetect (Default setting in OID decoder)

  Auto-sleep timer will be cleared when an index is read.

- UserCmd_NonClearAutoSleepTimmerIfOIDDetect

  Auto-sleep timer will not be cleared when an index is read.

- UserCmd_CheckOIDStatus

  DSP/MCU can send this command to get OID decoder’s status. OID decoder will reply a data packet after receiving the command.

  If OID decoder happens to send an index after receiving this command, it will send only current data packet to DSP/MCU as a reply. Otherwise a “DontCare” data packet will be sent to DSP/MCU. Both of above two conditions, the key and battery status will be included in the packet.

4. Control Flow

- **Turn On OID from DSP/MCU:**
  1. Pull SCK high and hold more than 20ms will force SN9P701 hardware to wake up.
     (SCK must be held high over 20ms and less than 2sec)
  2. Pull SCK low to make SN9P701 to enter normal operating.
  3. Receive OIDCmd_PowerDown from SN9P701 after pulling SCK low.
  4. OID decoder will power off directly if the data (OIDCmd_PowerDown) isn’t taken by DSP/MCU within 2 sec.

- **DSP reads data from OID decoder after turned on :**
  1. SN9P701 pulls SOID low to indicate DSP/MCU that there is a data packet to send.
  2. Data packet should be taken by DSP/MCU ASAP. If it has not been taken within 300ms, SN9P701 will remove the data and go to decode new image.

- **Turn off OID from DSP/MCU:**
  1. Send UserCmd_PowerDownOID to inform OID reader to power off.
  2. OID reader replies OIDCmd_PowerDown after receiving UserCmd_PowerDownOID.
  3. OID reader will power off right after the OIDCmd_PowerDown is taken by DSP/MCU.
  4. If OIDCmd_PowerDown has not been taken within 300ms, SN9P701 will power off directly, too.

- **Turn on OID decoder from decoder side**：(please refer to reference design)
  1. Press key_0 on pen side and hold more than 65ms will force SN9P701 hardware to wake up
  2. SN9P701waits until key_0 has been released, then sends OIDCmd_PowerDown to DSP/MCU.
  3. SN9P701 will still power off even if OIDCmd_PowerDown is not taken within 300ms

- **Turn off OID decoder from decoder side**
  1. Press key_0 on pen side and hold more than 65ms to inform SN9P701 to power off.
  2. SN9P701 sends OIDCmd_PowerDown to inform DSP/MCU after key_0 has been released.
  3. OID reader powers off right after the OIDCmd_PowerDown is taken by DSP/MCU.
  4. SN9P701 will still power off even if OIDCmd_PowerDown is not taken within 300ms.