Introduction

Smart Card ISO-7816 Library

For

8,16 & 32 bit PIC Microcontrollers

The Smart Card library for PIC microcontrollers support ISO 7816-3 and ISO 7816-4 standard protocols. It allows the PIC microcontroller to communicate with smart cards compatible with these protocols. The library supports both T=0 and T=1 smart card protocols.

The library comprises of PIC18/PIC24/dsPIC33F/PIC32 UART driver and T0/T1 protocol source code meeting ISO 7816-3 standard. An example high level demo application code is also provided to help the user port the smart card library to different hardware boards and different microcontrollers of PIC family.

This document assumes that the reader is familiar with ISO 7816-3 standards and T=0/T=1 protocols.
SW License Agreement

MICROCHIP IS WILLING TO LICENSE THE ACCOMPANYING SOFTWARE AND DOCUMENTATION TO YOU ONLY ON THE CONDITION THAT YOU ACCEPT ALL OF THE FOLLOWING TERMS. TO ACCEPT THE TERMS OF THIS LICENSE, CLICK "I ACCEPT" AND PROCEED WITH THE DOWNLOAD OR INSTALL. IF YOU DO NOT ACCEPT THESE LICENSE TERMS, CLICK "I DO NOT ACCEPT," AND DO NOT DOWNLOAD OR INSTALL THIS SOFTWARE.

NON-EXCLUSIVE SOFTWARE LICENSE AGREEMENT

This Nonexclusive Software License Agreement ("Agreement") is a contract between you, your heirs, successors and assigns ("Licensee") and Microchip Technology Incorporated, a Delaware corporation, with a principal place of business at 2355 W. Chandler Blvd., Chandler, AZ 85224-6199, and its subsidiary, Microchip Technology (Barbados) II Incorporated (collectively, "Microchip") for the accompanying Microchip software including, but not limited to, Graphics Library Software, IrDA Stack Software, MCHPFSUSB Stack Software, Memory Disk Drive File System Software, mTouch(TM) Capacitive Library Software, Smart Card Library Software, TCP/IP Stack Software, MiWi(TM) DE Software, Security Package Software, and/or any PC programs and any updates thereto (collectively, the "Software"), and accompanying documentation, including images and any other graphic resources provided by Microchip ("Documentation").
1. Definitions. As used in this Agreement, the following capitalized terms will have the meanings defined below:

a. "Microchip Products" means Microchip microcontrollers and Microchip digital signal controllers.

b. "Licensee Products" means Licensee products that use or incorporate Microchip Products.

c. "Object Code" means the Software computer programming code that is in binary form (including related documentation, if any), and error corrections, improvements, modifications, and updates.

d. "Source Code" means the Software computer programming code that may be printed out or displayed in human readable form (including related programmer comments and documentation, if any), and error corrections, improvements, modifications, and updates.

e. "Third Party" means Licensee’s agents, representatives, consultants, clients, customers, or contract manufacturers.

f. "Third Party Products" means Third Party products that use or incorporate Microchip Products.

2. Software License Grant. Microchip grants strictly to Licensee a non-exclusive, non-transferable, worldwide license to:

a. use the Software in connection with Licensee Products and/or Third Party Products;

b. if Source Code is provided, modify the Software; provided that Licensee clearly notifies Third Parties regarding the source of such modifications;

c. distribute the Software to Third Parties for use in Third Party Products, so long as such Third Party agrees to be bound by this
Agreement (in writing or by "click to accept") and this Agreement accompanies such distribution;

d. sublicense to a Third Party to use the Software, so long as such Third Party agrees to be bound by this Agreement (in writing or by "click to accept");

e. with respect to the TCP/IP Stack Software, Licensee may port the ENC28J60.c, ENC28J60.h, ENCX24J600.c, and ENCX24J600.h driver source files to a non-Microchip Product used in conjunction with a Microchip ethernet controller;

f. with respect to the MiWi (TM) DE Software, Licensee may only exercise its rights when the Software is embedded on a Microchip Product and used with a Microchip radio frequency transceiver or UBEC UZ2400 radio frequency transceiver which are integrated into Licensee Products or Third Party Products.

For purposes of clarity, Licensee may NOT embed the Software on a non-Microchip Product, except as described in this Section.

3. Documentation License Grant. Microchip grants strictly to Licensee a non-exclusive, non-transferable, worldwide license to use the Documentation in support of Licensee's authorized use of the Software

4. Third Party Requirements. Licensee acknowledges that it is Licensee's responsibility to comply with any third party license terms or requirements applicable to the use of such third party software, specifications, systems, or tools. This includes, by way of example but not as a limitation, any standards setting organizations requirements and, particularly with respect to the Security Package Software, local encryption laws and requirements. Microchip is not responsible and will not be held responsible in any manner for Licensee’s failure to comply with such applicable terms or requirements.
5. Open Source Components. Notwithstanding the license grant in Section 1 above, Licensee further acknowledges that certain components of the Software may be covered by so-called "open source" software licenses ("Open Source Components"). Open Source Components means any software licenses approved as open source licenses by the Open Source Initiative or any substantially similar licenses, including without limitation any license that, as a condition of distribution of the software licensed under such license, requires that the distributor make the software available in source code format. To the extent required by the licenses covering Open Source Components, the terms of such license will apply in lieu of the terms of this Agreement. To the extent the terms of the licenses applicable to Open Source Components prohibit any of the restrictions in this Agreement with respect to such Open Source Components, such restrictions will not apply to such Open Source Component.

6. Licensee Obligations. Licensee will not: (a) engage in unauthorized use, modification, disclosure or distribution of Software or Documentation, or its derivatives; (b) use all or any portion of the Software, Documentation, or its derivatives except in conjunction with Microchip Products, Licensee Products or Third Party Products; or (c) reverse engineer (by disassembly, decompilation or otherwise) Software or any portion thereof. Licensee may not remove or alter any Microchip copyright or other proprietary rights notice posted in any portion of the Software or Documentation. Licensee will defend, indemnify and hold Microchip and its subsidiaries harmless from and against any and all claims, costs, damages, expenses (including reasonable attorney's fees), liabilities, and losses, including without limitation: (x) any claims directly or indirectly arising from or related to the use, modification, disclosure or distribution of the Software, Documentation, or any intellectual property rights related thereto; (y) the use, sale and distribution of Licensee Products or Third Party Products; and (z) breach of this
7. Confidentiality. Licensee agrees that the Software (including but not limited to the Source Code, Object Code and library files) and its derivatives, Documentation and underlying inventions, algorithms, know-how and ideas relating to the Software and the Documentation are proprietary information belonging to Microchip and its licensors ("Proprietary Information"). Except as expressly and unambiguously allowed herein, Licensee will hold in confidence and not use or disclose any Proprietary Information and will similarly bind its employees and Third Party(ies) in writing. Proprietary Information will not include information that: (i) is in or enters the public domain without breach of this Agreement and through no fault of the receiving party; (ii) the receiving party was legally in possession of prior to receiving it; (iii) the receiving party can demonstrate was developed by the receiving party independently and without use of or reference to the disclosing party’s Proprietary Information; or (iv) the receiving party receives from a third party without restriction on disclosure. If Licensee is required to disclose Proprietary Information by law, court order, or government agency, License will give Microchip prompt notice of such requirement in order to allow Microchip to object or limit such disclosure. Licensee agrees that the provisions of this Agreement regarding unauthorized use and nondisclosure of the Software, Documentation and related Proprietary Rights are necessary to protect the legitimate business interests of Microchip and its licensors and that monetary damage alone cannot adequately compensate Microchip or its licensors if such provisions are violated. Licensee, therefore, agrees that if Microchip alleges that Licensee or Third Party has breached or violated such provision then Microchip will have the right to injunctive relief, without the requirement for the posting of a bond, in addition to all other remedies at law or in equity.
8. Ownership of Proprietary Rights. Microchip and its licensors retain all right, title and interest in and to the Software and Documentation including, but not limited to all patent, copyright, trade secret and other intellectual property rights in the Software, Documentation, and underlying technology and all copies and derivative works thereof (by whomever produced). Licensee and Third Party use of such modifications and derivatives is limited to the license rights described in this Agreement.

9. Termination of Agreement. Without prejudice to any other rights, this Agreement terminates immediately, without notice by Microchip, upon a failure by Licensee or Third Party to comply with any provision of this Agreement. Upon termination, Licensee and Third Party will immediately stop using the Software, Documentation, and derivatives thereof, and immediately destroy all such copies.

10. Warranty Disclaimers. THE SOFTWARE AND DOCUMENTATION ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF MERCHANTABILITY, TITLE, NON-INFRINGEMENT AND FITNESS FOR A PARTICULAR PURPOSE. MICROCHIP AND ITS LICENSORS ASSUME NO RESPONSIBILITY FOR THE ACCURACY, RELIABILITY OR APPLICATION OF THE SOFTWARE OR DOCUMENTATION. MICROCHIP AND ITS LICENSORS DO NOT WARRANT THAT THE SOFTWARE WILL MEET REQUIREMENTS OF LICENSEE OR THIRD PARTY, BE UNINTERRUPTED OR ERROR-FREE. MICROCHIP AND ITS LICENSORS HAVE NO OBLIGATION TO CORRECT ANY DEFECTS IN THE SOFTWARE.

11. Limited Liability. IN NO EVENT WILL MICROCHIP OR ITS LICENSORS BE LIABLE OR OBLIGATED UNDER ANY LEGAL OR EQUITABLE THEORY FOR ANY DIRECT OR INDIRECT
DAMAGES OR EXPENSES INCLUDING BUT NOT LIMITED TO INCIDENTAL, SPECIAL, INDIRECT, PUNITIVE OR CONSEQUENTIAL DAMAGES, LOST PROFITS OR LOST DATA, COST OF PROCUREMENT OF SUBSTITUTE GOODS, TECHNOLOGY, SERVICES, OR ANY CLAIMS BY THIRD PARTIES (INCLUDING BUT NOT LIMITED TO ANY DEFENSE THEREOF), OR OTHER SIMILAR COSTS. The aggregate and cumulative liability of Microchip and its licensors for damages hereunder will in no event exceed $1000 or the amount Licensee paid Microchip for the Software and Documentation, whichever is greater. Licensee acknowledges that the foregoing limitations are reasonable and an essential part of this Agreement.

12. General. THIS AGREEMENT WILL BE GOVERNED BY AND CONSTRUED UNDER THE LAWS OF THE STATE OF ARIZONA AND THE UNITED STATES WITHOUT REGARD TO CONFLICTS OF LAWS PROVISIONS. Licensee agrees that any disputes arising out of or related to this Agreement, Software or Documentation will be brought exclusively in either the U.S. District Court for the District of Arizona, Phoenix Division, or the Superior Court of Arizona located in Maricopa County, Arizona. This Agreement will constitute the entire agreement between the parties with respect to the subject matter hereof. It will not be modified except by a written agreement signed by an authorized representative of Microchip. If any provision of this Agreement will be held by a court of competent jurisdiction to be illegal, invalid or unenforceable, that provision will be limited or eliminated to the minimum extent necessary so that this Agreement will otherwise remain in full force and effect and enforceable. No waiver of any breach of any provision of this Agreement will constitute a waiver of any prior, concurrent or subsequent breach of the same or any other provisions hereof, and no waiver will be effective unless made in writing and signed by an authorized representative of the waiving party. Licensee agrees to comply with all import and export laws and restrictions
and regulations of the Department of Commerce or other United States or foreign agency or authority. The indemnities, obligations of confidentiality, and limitations on liability described herein, and any right of action for breach of this Agreement prior to termination, will survive any termination of this Agreement. Any prohibited assignment will be null and void. Use, duplication or disclosure by the United States Government is subject to restrictions set forth in subparagraphs (a) through (d) of the Commercial Computer-Restricted Rights clause of FAR 52.227-19 when applicable, or in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013, and in similar clauses in the NASA FAR Supplement. Contractor/manufacturer is Microchip Technology Inc., 2355 W. Chandler Blvd., Chandler, AZ 85224-6199.

If Licensee has any questions about this Agreement, please write to Microchip Technology Inc., 2355 W. Chandler Blvd., Chandler, AZ 85224-6199 USA. ATTN: Marketing.

Copyright (c) 2012 Microchip Technology Inc. All rights reserved.

License Rev. No. 05-012412

SW License Agreement
# Release Notes

## Resource Usage (v1.02.8)

- [Resource Usage for PIC18](#)
- [Resource Usage for PIC24F](#)
- [Resource Usage - PIC24H](#)
- [Resource Usage - dsPIC33F](#)
- [Resource Usage - PIC32](#)

## Peripherals

<table>
<thead>
<tr>
<th>Type/Use</th>
<th>Specific/Configurable</th>
<th>Polled/Interrupt</th>
<th>Limitations</th>
</tr>
</thead>
<tbody>
<tr>
<td>UART</td>
<td>Select via Programming, Tx and Rx Signals</td>
<td>Polled</td>
<td>None</td>
</tr>
<tr>
<td>Card Power Output</td>
<td>Select via <code>#define</code> in SCpic18.h or SCpic24.h or SCpic32.h or SCdspic33f</td>
<td>Polled</td>
<td>Be able to source sufficient current to power the Smartcard</td>
</tr>
<tr>
<td>Card Reset Output</td>
<td>Select via <code>#define</code> in SCpic18.h or SCpic24.h or SCpic32.h or SCdspic33f</td>
<td>Polled</td>
<td>Totem pole or Open Drain with pullup</td>
</tr>
<tr>
<td>Card Present Input</td>
<td>Select via <code>#define</code> in SCpic18.h or SCpic24.h or SCpic32.h or SCdspic33f</td>
<td>Polled</td>
<td>Input with Pullup</td>
</tr>
<tr>
<td>Clock Output</td>
<td>REFO Output</td>
<td>n/a</td>
<td>Clock Output to</td>
</tr>
</tbody>
</table>
Card should be close to 4MHz (3.57MHz for exact Baud Rate, but not required)

### Topics

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Resource Usage - PIC18</strong></td>
<td>These tables specify the program memory, execution speed, RAM usage, and build requirements for PIC18 devices.</td>
</tr>
<tr>
<td><strong>Resource Usage - PIC24F</strong></td>
<td>These tables specify the program memory, execution speed, RAM usage, and build requirements PIC24F devices.</td>
</tr>
<tr>
<td><strong>Resource Usage - PIC24H</strong></td>
<td>These tables specify the program memory, execution speed, RAM usage, and build requirements PIC24H devices.</td>
</tr>
<tr>
<td><strong>Resource Usage - dsPIC33F</strong></td>
<td>These tables specify the program memory, execution speed, RAM usage, and build requirements dsPIC33F devices.</td>
</tr>
<tr>
<td><strong>Resource Usage - PIC32</strong></td>
<td>These tables specify the program memory, execution speed, RAM usage, and build requirements PIC32 devices.</td>
</tr>
</tbody>
</table>
These tables specify the program memory, execution speed, RAM usage, and build requirements for PIC18 devices.

### Program Memory (bytes)

<table>
<thead>
<tr>
<th>Module</th>
<th>Optimization Level</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smartcard Library (T=0)</td>
<td>None</td>
<td>4K</td>
</tr>
<tr>
<td>Smartcard Library (T=0)</td>
<td>Enable All</td>
<td>2.8K</td>
</tr>
<tr>
<td>Smartcard Library (T=0 &amp; T=1)</td>
<td>None</td>
<td>6.8K</td>
</tr>
<tr>
<td>Smartcard Library (T=0 &amp; T=1)</td>
<td>Enable All</td>
<td>4.9K</td>
</tr>
</tbody>
</table>

### RAM Usage (bytes)

<table>
<thead>
<tr>
<th>Module/Layer</th>
<th>Global</th>
<th>Stack</th>
<th>Heap</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smartcard Library (T=0)</td>
<td>300</td>
<td>Not available</td>
<td>None</td>
</tr>
<tr>
<td>Smartcard Library (T=0 &amp; T=1)</td>
<td>330</td>
<td>Not available</td>
<td>None</td>
</tr>
</tbody>
</table>

### Build Requirements

None

[Release Notes > Resource Usage - PIC18](#)
Resource Usage - PIC24F

These tables specify the program memory, execution speed, RAM usage, and build requirements PIC24F devices.

Program Memory

<table>
<thead>
<tr>
<th>Module</th>
<th>Global</th>
<th>Stack</th>
<th>Heap</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smartcard Library (T=0)</td>
<td>None</td>
<td>2.7K</td>
<td></td>
</tr>
<tr>
<td>Smartcard Library (T=0)</td>
<td>-O1</td>
<td>1.9K</td>
<td></td>
</tr>
<tr>
<td>Smartcard Library (T=0)</td>
<td>-Os</td>
<td>1.7K</td>
<td></td>
</tr>
<tr>
<td>Smartcard Library (T=0 &amp; T=1)</td>
<td>None</td>
<td>4.7K</td>
<td></td>
</tr>
<tr>
<td>Smartcard Library (T=0 &amp; T=1)</td>
<td>-O1</td>
<td>3.2K</td>
<td></td>
</tr>
<tr>
<td>Smartcard Library (T=0 &amp; T=1)</td>
<td>-Os</td>
<td>2.9K</td>
<td></td>
</tr>
</tbody>
</table>

RAM Usage (bytes)

<table>
<thead>
<tr>
<th>Module/Layer</th>
<th>Global</th>
<th>Stack</th>
<th>Heap</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smartcard Library (T=0)</td>
<td>300</td>
<td>Not available</td>
<td>None</td>
</tr>
<tr>
<td>Smartcard Library (T=0 &amp; T=1)</td>
<td>330</td>
<td>Not available</td>
<td>None</td>
</tr>
</tbody>
</table>

Build Requirements

None

Release Notes > Resource Usage - PIC24F
These tables specify the program memory, execution speed, RAM usage, and build requirements PIC24H devices.

### Program Memory

<table>
<thead>
<tr>
<th>Module</th>
<th>Global</th>
<th>Stack</th>
<th>Heap</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smartcard Library (T=0)</td>
<td>None</td>
<td>2.7K</td>
<td></td>
</tr>
<tr>
<td>Smartcard Library (T=0)</td>
<td>-O1</td>
<td>2K</td>
<td></td>
</tr>
<tr>
<td>Smartcard Library (T=0)</td>
<td>-Os</td>
<td>1.7K</td>
<td></td>
</tr>
<tr>
<td>Smartcard Library (T=0 &amp; T=1)</td>
<td>None</td>
<td>4.7K</td>
<td></td>
</tr>
<tr>
<td>Smartcard Library (T=0 &amp; T=1)</td>
<td>-O1</td>
<td>3.2K</td>
<td></td>
</tr>
<tr>
<td>Smartcard Library (T=0 &amp; T=1)</td>
<td>-Os</td>
<td>2.9K</td>
<td></td>
</tr>
</tbody>
</table>

### RAM Usage (bytes)

<table>
<thead>
<tr>
<th>Module/Layer</th>
<th>Global</th>
<th>Stack</th>
<th>Heap</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smartcard Library (T=0)</td>
<td>300</td>
<td>Not available</td>
<td>None</td>
</tr>
<tr>
<td>Smartcard Library (T=0 &amp; T=1)</td>
<td>330</td>
<td>Not available</td>
<td>None</td>
</tr>
</tbody>
</table>

### Build Requirements

None
Resource Usage - dsPIC33F

These tables specify the program memory, execution speed, RAM usage, and build requirements dsPIC33F devices.

Program Memory

<table>
<thead>
<tr>
<th>Module</th>
<th>Options</th>
<th>Program Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smartcard Library (T=0)</td>
<td>None</td>
<td>2.7K</td>
</tr>
<tr>
<td>Smartcard Library (T=0)</td>
<td>-O1</td>
<td>2K</td>
</tr>
<tr>
<td>Smartcard Library (T=0)</td>
<td>-Os</td>
<td>1.7K</td>
</tr>
<tr>
<td>Smartcard Library (T=0 &amp; T=1)</td>
<td>None</td>
<td>4.7K</td>
</tr>
<tr>
<td>Smartcard Library (T=0 &amp; T=1)</td>
<td>-O1</td>
<td>3.2K</td>
</tr>
<tr>
<td>Smartcard Library (T=0 &amp; T=1)</td>
<td>-Os</td>
<td>2.9K</td>
</tr>
</tbody>
</table>

RAM Usage (bytes)

<table>
<thead>
<tr>
<th>Module/Layer</th>
<th>Global</th>
<th>Stack</th>
<th>Heap</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smartcard Library (T=0)</td>
<td>300</td>
<td>Not available</td>
<td>None</td>
</tr>
<tr>
<td>Smartcard Library (T=0 &amp; T=1)</td>
<td>330</td>
<td>Not available</td>
<td>None</td>
</tr>
</tbody>
</table>

Build Requirements

None

Release Notes > Resource Usage - dsPIC33F
Resource Usage - PIC32

These tables specify the program memory, execution speed, RAM usage, and build requirements PIC32 devices.

### Program Memory

<table>
<thead>
<tr>
<th>Module</th>
<th>Memory Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smartcard Library (T=0)</td>
<td>None</td>
</tr>
<tr>
<td>Smartcard Library (T=0)</td>
<td>-O1</td>
</tr>
<tr>
<td>Smartcard Library (T=0)</td>
<td>-Os</td>
</tr>
<tr>
<td>Smartcard Library (T=0 &amp; T=1)</td>
<td>None</td>
</tr>
<tr>
<td>Smartcard Library (T=0 &amp; T=1)</td>
<td>-O1</td>
</tr>
<tr>
<td>Smartcard Library (T=0 &amp; T=1)</td>
<td>-Os</td>
</tr>
</tbody>
</table>

### RAM Usage (bytes)

<table>
<thead>
<tr>
<th>Module/Layer</th>
<th>Global</th>
<th>Stack</th>
<th>Heap</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smartcard Library (T=0)</td>
<td>300</td>
<td>Not available</td>
<td>None</td>
</tr>
<tr>
<td>Smartcard Library (T=0 &amp; T=1)</td>
<td>330</td>
<td>Not available</td>
<td>None</td>
</tr>
</tbody>
</table>

### Build Requirements

None

Release Notes > Resource Usage - PIC32
Two communication protocols that are generally used for contact type smart card communications are:

- $T = 0$ (asynchronous half duplex character transmission)
- $T = 1$ (asynchronous half duplex block transmission)

The data transfers between the card and the terminal(smart card reader) happens on the single wire I/O line.

Following the initial reset of the card after insertion, the card responds with a series of characters called the Answer to Reset, or ATR. This series of characters establishes the initial communication details, including the specific protocol, bit timing, and data transfer details for all subsequent communications. While subsequent data transfers can change certain communications parameters, the ATR establishes initial communications conditions.

The Clock Signal for Baud rate generation is provided to the card by the reader (terminal). The Smartcard default baudrate divider is 372, which produce 9600 bps when a clock signal of 3.57MHz is supplied to the card. Most Smartcards allow higher clock rates, so a simple 4MHz clock can be easily used. Using a 4MHz clock, the default baudrate comes out to be 10752 bps. The PICs UART is appropriately configured by the library, so the communication can be setup using the higher baudrate settings.

The Smartcard 7816-3 communications requires a 0.5 stop bit. This is important for the Receiver, as it must pull the I/O line low before the middle of the stop bit (10.5 bit time from start edge) in order to indicate error condition to the Transmitter. The receiver pulls the line low for 1 to 2 bit time (etu). The transmitter checks the I/O line at the end of stop bit, or 11 etu. If the transmitter detects the line low, it retransmit the previous data byte after at
The uart peripheral in PIC micros sets Rx Ready and Transmitter Empty flags to true at 0.5 stop bit, which allows the implementation of the 7816-3 error detection and retransmission protocol possible.

### Topics

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Library Architecture</strong></td>
<td>The Smartcard Library has a modular design with separate files for the high level library code and the low level driver for UART for implementing the ISO7816-3/4 protocol.</td>
</tr>
</tbody>
</table>
How the Library Works

The current release of smart card library supports PIC18, PIC24F, dsPIC33F, PIC24H & PIC32MX microcontrollers. The smart card library provides the API necessary to communicate with the ISO7816-3/4 compliant Smartcard. The sequence of the API calls is as given below. SClib.h contains all the API's that are required by the main application to communicate with the smart card. The current release of smart card library supports both T=0 and T=1 protocol.

```c
//Initialize smart card stack
SC_Initialize();

// Wait until the card is inserted in the slot
while( !SC_CardPresent() )

// After detecting the card, turn on...
more
```
Library Architecture

The Smartcard Library has a modular design with separate files for the high level library code and the low level driver for UART for implementing the ISO7816-3/4 protocol.

<table>
<thead>
<tr>
<th>Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>Microchip Smart Card Library - ISO 7816-4 Protocol</td>
</tr>
<tr>
<td>PIC UART Driver – ISO 7816-3 Protocol</td>
</tr>
<tr>
<td>PIC UART and Port I/O</td>
</tr>
</tbody>
</table>

Smartcard Library Overview > Library Architecture
How the Library Works

The current release of smart card library supports PIC18, PIC24F, dsPIC33F, PIC24H & PIC32MX microcontrollers. The smart card library provides the API necessary to communicate with the ISO7816-3/4 compliant Smartcard. The sequence of the API calls is as given below. SClib.h contains all the API's that are required by the main application to communicate with the smart card. The current release of smart card library supports both T=0 and T=1 protocol.

\[
// Initialize smart card stack
SC_Initialize();
\]

\[
// Wait until the card is inserted in the slot
while( !SC_CardPresent() )
\]

\[
// After detecting the card, turn on the power to the card and process Answer-to-Reset
if( !SC_PowerOnATR() )
\]

\[
// Do protocol & parameter selection. Configure the desired baud rate
if( !SC_DoPPS(ppsString) )
\]
//Execute Card Commands
//If T=1 card is inserted in the slot, execute T=1 commands
if(SC_T1ProtocolType())
{
  if(!SC_TransactT1(&prologueField,apduData,&cardResponse))
}
//If T=0 card is inserted in the slot, execute T=0 commands
else if(SC_T0ProtocolType())
{
  if(!SC_TransactT0(&cardCommand,&cardResponse,apduData))
}
...
...
// Shut Down the Card when there is nothing to do with it
SC_Shutdown();
...

Note:

1) For T=1 protocol "prologueField" buffer should contain the prologue field(NAD,PCB,LENGTH) that needs to be sent to Smart Card. Once the transaction is completed between the card & the micro, response from the card is stored in "cardResponse" buffer."apduData" points to the data buffer of the command as
well as data response from the card.

2) For T=0 protocol "cardCommand" buffer should contain the command that needs to be sent to the Smart Card. Once the transaction is completed between the card & the micro, response from the card is stored in "cardResponse" buffer. "apduData" points to the data buffer of command as well as data response from the card.
Getting Started - Smart Card Demo

This demo shows how the smart card library for PIC microcontroller is used to communicate a smart card using $T = 0$ & $T = 1$ protocols. The demo has to be run in the debug mode of MPLAB IDE.

Topics

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required Hardware</td>
<td>To run this demo application, you will need one of the following sets of hardware:</td>
</tr>
<tr>
<td>Configuring the Hardware</td>
<td>This section describes how to set up the various configurations of hardware to run this demo.</td>
</tr>
<tr>
<td>Firmware</td>
<td>To run this project, you will need to load the corresponding firmware into the devices.</td>
</tr>
<tr>
<td></td>
<td>The source code for this demo is available in the “&lt;Microchip Solutions\Smart Card Demo” directory. In this directory you will find all of the user level source and header files, linker file as well as project file for each of the hardware platforms. Find the project (*.mcp) file that corresponds to the hardware platform you wish to test. Compile and program the demo code into the hardware platform. For more help on how to compile and program projects, please refer to the MPLAB® IDE help... more</td>
</tr>
<tr>
<td>Running the Demo</td>
<td>This demo uses the selected hardware platform as a Smart card reader. The demo has to be run in the debug mode of MPLAB IDE. Please refer &quot;Configuring the Hardware&quot; section for the bench setup connections.</td>
</tr>
</tbody>
</table>
Smart Card consists of 8 pins namely:-

I/O: Input or Output for serial data to the integrated circuit inside the card.
VPP: Programming voltage input (optional use by the card).
GND: Ground (reference voltage).
CLK: Clocking or timing signal.
RST: Reset Signal to the Card.
VCC: Power supply input (optional use by the card).

Communication between the interfacing device and smart card is done... more
### Required Hardware

To run this demo application, you will need one of the following sets of hardware:

#### Topics

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| **Configuration 1: PIC18 Explorer Board** | 1. PIC18 Explorer Board (Microchip part number DM183032)  
2. SC (Smart/Sim Card) PICTail Board  
3. And one of the following PIMs  
   1. PIC18F87J50 Plug-In-Module (PIM) (Microchip part number MA180021)  
   2. PIC18F46J50 Full Speed USB Demo Board (Microchip part number MA180024) |
| **Configuration 2: Explorer 16 Board** | 1. Explorer 16 (Microchip part number DM240001)  
2. SC (Smart/Sim Card) PICTail Board  
3. And one of the following PIMs  
   1. PIC24FJ256GB110 Plug-In-Module (PIM) (Microchip part number MA240014),  
   2. PIC32MX795F512L Plug-In-Module (PIM) (Microchip part number MA320003),  
   3. dsPIC33FJ128MC710,  
   4. PIC24HJ256GP610 |
<table>
<thead>
<tr>
<th>Configuration 3: Low Pin Count USB Development Kit</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Low Pin Count USB Development Kit with PICKit 2 Debugger/Programmer (Microchip part number DV164126) or without Debugger/Programmer (Microchip part number DM164127).</td>
</tr>
<tr>
<td>2. SC (Smart/Sim Card) PICTail Board</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Configuration 4: PICDEM FS USB Board</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. PICDEM FS USB Board (Microchip part number DM163025)</td>
</tr>
<tr>
<td>2. SC (Smart/Sim Card) PICTail Board</td>
</tr>
</tbody>
</table>

**Getting Started - Smart Card Demo > Required Hardware**
Configuration 1: PIC18 Explorer Board

1. PIC18 Explorer Board (Microchip part number DM183032)
2. SC (Smart/Sim Card) PICTail Board
3. And one of the following PIMs
   1. PIC18F87J50 Plug-In-Module (PIM) (Microchip part number MA180021)
   2. PIC18F46J50 Full Speed USB Demo Board (Microchip part number MA180024)
Configuration 2: Explorer 16 Board

1. Explorer 16 (Microchip part number DM240001)
2. SC (Smart/Sim Card) PICTail Board
3. And one of the following PIMs
   1. PIC24FJ256GB110 Plug-In-Module (PIM) (Microchip part number MA240014),
   2. PIC32MX795F512L Plug-In-Module (PIM) (Microchip part number MA320003),
   3. dsPIC33FJ128MC710,
   4. PIC24HJ256GP610
Configuration 3: Low Pin Count USB Development Kit

1. Low Pin Count USB Development Kit with PICKit 2 Debugger/Programmer (Microchip part number DV164126) or without Debugger/Programmer (Microchip part number DM164127).
2. SC (Smart/Sim Card) PICTail Board
Configuration 4: PICDEM FS USB Board

1. PICDEM FS USB Board (Microchip part number DM163025)
2. SC (Smart/Sim Card) PICTail Board

Getting Started - Smart Card Demo > Required Hardware > Configuration 4: PICDEM FS USB Board
**Configuring the Hardware:**

This section describes how to set up the various configurations of hardware to run this demo.

### Topics

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Configuration using PIC18 Explorer Board</strong></td>
<td>1. Before inserting PIC18F87J50 PIM or PIC18F46J50 PIM in the PIC18 Explorer board, insure that the processor selector switch (S4) is in the “ICE” position as seen in the image below. Failure to so will result in difficulties in getting the PIC18F87J50/PIC18F46J50 PIM to sit properly on the PIC18 Explorer.</td>
</tr>
</tbody>
</table>

2. Before inserting PIC18F87J50/PIC18F46J50 PIM into the PIC18 Explorer board, remove all the attached cables from both the boards. Be careful while inserting the PIM into PIC18 board. Insure that no pins are bent or damaged during the process. Also insure that the PIM is not shifted in any direction and that... [more](#)
1. Before attaching the PIM to the Explorer 16 board, insure that the processor selector switch (S2) is in the “PIM” position as seen in the image below.

2. Short the J7 jumper to the “PIC24” setting.

3. Be careful while inserting the PIC24FJ256GB110.
PIM or any other appropriate PIM into Exp 16 board. Insure that no pins are bent or damaged during the process. Also insure that the PIM is not shifted in any direction and that all of the headers are properly aligned.

4. Short JP1 to SRC1 (i.e. RD1) or SRC2 (i.e. RB15) based upon the smart card clock pin configured...

### Configuration using PIC18F14K50 + LPC Board

Ensure that JP1 of SC PICTail card & J12 of LPC board are left open. One side of J4 port pins of the SC PICTail card matches with the J11 port of LPC board. Insert the matching side of J4 port of SC PICTail board into the J11 port of LPC board. Make sure that the Smart Card Connector is facing towards the LPC board. Insert the Smart Card in SC PICTail board. Apart from the above guidelines, couple of below steps has to be followed to make the demo work:-

1. Short Tx & Rx line of the UART (i.e....)

### Configuration using PICDEM FS USB Board

1. If using the PICDEM FS USB Demo Board, no hardware related configuration or jumper setting changes should be necessary. The demo board need only be programmed with appropriate firmware.

2. Don’t short the jumper at J11 port.

3. Insert the J2 port of SC (Smart/Sim Card) PICTail
card into J3 port of PICDEM FSUSB board as per the pin configuration. Insert the Smart Card in SC PICTail board.
1. Before inserting PIC18F87J50 PIM or PIC18F46J50 PIM in the PIC18 Explorer board, insure that the processor selector switch (S4) is in the “ICE” position as seen in the image below. Failure to so will result in difficulties in getting the PIC18F87J50/PIC18F46J50 PIM to sit properly on the PIC18 Explorer.

2. Before inserting PIC18F87J50/PIC18F46J50 PIM into the PIC18 Explorer board, remove all the attached cables from both the boards. Be careful while inserting the PIM into PIC18 board. Insure that no pins are bent or damaged during the process. Also insure that the PIM is not shifted in any direction and that all of the headers are properly aligned.

3. Insert the J4 port pins of SC (Smart/Sim Card) PICTail Board in the J3 port of PIC18 Explorer board. Make sure that the Smart Card Connector is facing towards the PIC18 Explorer board. Insert the Smart Card in SC PICTail Board.
Configuring the Hardware:

Configuration using PIC18 Explorer Board
Configuration using Explorer 16 Board

1. Before attaching the PIM to the Explorer 16 board, insure that the processor selector switch (S2) is in the “PIM” position as seen in the image below.

2. Short the J7 jumper to the “PIC24” setting
3. Be careful while inserting the PIC24FJ256GB110 PIM or any other appropriate PIM into Exp 16 board. Insure that no pins are bent or damaged during the process. Also insure that the PIM is not shifted in any direction and that all of the headers are properly aligned.

4. Short JP1 to SRC1 (i.e. RD1) or SRC2 (i.e. RB15) based upon the smart card clock pin configured in the firmware: Example: - Short JP1 to SRC1 while using PIC24FJ256GB110 demo and Short JP2 to SRC2 while using PIC32MX795F512L demo.
5. Insert the J2 slot of SC (Smart/Sim Card) PICTail card into J5 port of Explorer 16 board. Make sure that the Smart Card Connector is facing towards the Explorer 16 board. Insert the Smart Card in SC PICTail board.
Configuration using PIC18F14K50 + LPC Board

Ensure that JP1 of SC PICTail card & J12 of LPC board are left open. One side of J4 port pins of the SC PICTail card matches with the J11 port of LPC board. Insert the matching side of J4 port of SC PICTail board into the J11 port of LPC board. Make sure that the Smart Card Connector is facing towards the LPC board. Insert the Smart Card in SC PICTail board.

Apart from the above guidelines, couple of below steps has to be followed to make the demo work:-

1. Short Tx & Rx line of the UART (i.e. short pin 1 & pin 6 of J13 port using a wire in the LPC board) and connect it to I/O pin of SC PICTail board.
2. Connect RB6 (i.e. pin 5 of J13 port in LPC board) to “Card Present” signal pin of SC PICTail board as shown below.
Configuring the Hardware:

Configuration using PIC18F14K50 + LPC Board
Configuration using PICDEM FS USB Board

1. If using the PICDEM FS USB Demo Board, no hardware related configuration or jumper setting changes should be necessary. The demo board need only be programmed with appropriate firmware.

2. Don't short the jumper at J11 port.
3. Insert the J2 port of SC (Smart/Sim Card) PICTail card into J3 port of PICDEM FSUSB board as per the pin configuration. Insert the Smart Card in SC PICTail board.

Getting Started - Smart Card Demo > Configuring the Hardware: > Configuration using PICDEM FS USB Board
Firmware

To run this project, you will need to load the corresponding firmware into the devices.

The source code for this demo is available in the “<Microchip Solutions\Smart Card Demo” directory. In this directory you will find all of the user level source and header files, linker file as well as project file for each of the hardware platforms. Find the project (*.mcp) file that corresponds to the hardware platform you wish to test. Compile and program the demo code into the hardware platform. For more help on how to compile and program projects, please refer to the MPLAB® IDE help available through the help menu of MPLAB (Help->Topics…->MPLAB IDE).
Running the Demo

This demo uses the selected hardware platform as a Smart card reader. The demo has to be run in the debug mode of MPLAB IDE. Please refer "Configuring the Hardware" section for the bench setup connections.

Smart Card consists of 8 pins namely:-

- **VCC**: Input or Output for serial data to the integrated circuit inside the card.
- **VPP**: Programming voltage input (optional use by the card).
- **GND**: Ground (reference voltage).
- **CLK**: Clocking or timing signal.
- **RST**: Reset Signal to the Card.
- **VCC**: Power supply input (optional use by the card).

Communication between the interfacing device and smart card is done as per the following steps:-

1. Insertion of the smart card in the slot.
2. Detection of the smart card insertion by the microcontroller (interfacing device).
3. Microcontroller does the cold reset of the smart card.
4. Answer to Reset (ATR) response by the card.
5. PPS exchange (if smart card supports it).
6. Execution of the transaction(s) between the card & the interfacing device.
7. Removal of the smart card from the slot.
8. Detection of the smart card removal by the microcontroller.
9. Deactivation of the contacts.

Contact type smart card communication protocols that are generally used are:-

- $T = 0$ asynchronous half duplex character transmission.
- $T = 1$ asynchronous half duplex block transmission.

The data transfers between the card and the terminal happens on the single wire I/O line. The smart card library supports both $T=0$ & $T=1$ protocol.

**Example code for $T=0$ cards:**

The demo executes the card commands namely SUBMIT CODE, SELECT FILE, READ RECORD & WRITE RECORD. The command list can be extended further as per the project requirement.

**Example code for $T=1$ cards:**

The demo executes the "Get CPLC (Card Production Life Cycle) data" command for $T=1$ java card. The command list can be extended further as per the smart card manual and the project requirement.

The demo waits in the while(1) loop until the smart card is inserted in the smart card connector slot. Once the card is inserted in the slot, 'Cold Reset' and 'PPS' (Protocol & Parameter Selection) has to be performed to the smart card running MPLAB.
project in debug mode. If the user has inserted T=0 card in the slot, then “SC_TransactT0” function is called & the result of the executed command from the smart card is stored in “apduData”. If the user has inserted T=1 card in the slot, then “SC_TransactT1” function is called & the result of the executed command from the smart card is stored in “apduData”.

Variable “cardResponse” stores the status codes & the length of the received data from the smart card.

**Note:** After initially being reset by the card reader, the smart card responds with a string of characters known as the Answer to Reset, or ATR. These characters consist of an initial character, TS, followed by a maximum of 32 additional characters. Together, these characters provide information to the card reader about how to communicate with the card for the remainder of the session. If the card reader wants to modify the data transmission parameters in the smart card, then it must perform PPS in accordance with ISO/IEC 7816-3 before the transmission protocol is actually used.

For more details about smart card communication using PIC microcontrollers, please refer the application note [AN1370](#)
The current smartcard software library supports 8,16 & 32 bit PIC microcontrollers. The port pins connection b/w the micro & smart card is defined in "sc_config.h" file. The demo uses the signal connections between the smart card & PIC microcontroller port pins as per the below table:-

<table>
<thead>
<tr>
<th>Signal Name</th>
<th>PIC18F46J50</th>
<th>PIC18F87J50</th>
<th>PIC18F4550</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIM_CARD_DET</td>
<td>RB1</td>
<td>RB1</td>
<td>RB1</td>
</tr>
<tr>
<td>SMART_CLK</td>
<td>RB2</td>
<td>RC2</td>
<td>RC2</td>
</tr>
<tr>
<td>SMART_I/O</td>
<td>RC6,RC7</td>
<td>RC6,RC7</td>
<td>RC6,RC7</td>
</tr>
<tr>
<td>SMART_RST</td>
<td>RB4</td>
<td>RB4</td>
<td>RB4</td>
</tr>
<tr>
<td>SMART_CARD_DET</td>
<td>RB3</td>
<td>RB3</td>
<td>RB3</td>
</tr>
<tr>
<td>SMART_VCC</td>
<td>RB0</td>
<td>RB0</td>
<td>RB0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Signal Name</th>
<th>PIC24FJ256GB110</th>
<th>PIC32MX795F512L</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIM_CARD_DET</td>
<td>RB1</td>
<td>RB1</td>
</tr>
<tr>
<td>SMART_CLK</td>
<td>RB15</td>
<td>RD1</td>
</tr>
<tr>
<td>SMART_I/O</td>
<td>RC4,RF2</td>
<td>RF2,RF8</td>
</tr>
<tr>
<td>SMART_RST</td>
<td>RE8</td>
<td>RE8</td>
</tr>
<tr>
<td>SMART_CARD_DET</td>
<td>RB0</td>
<td>RB0</td>
</tr>
<tr>
<td>SMART_VCC</td>
<td>RB9</td>
<td>RB9</td>
</tr>
</tbody>
</table>

“SIM_CARD_DET”/“SMART_CARD_DET” signals indicate the presence of Smart Card/Sim Card to the microcontroller. Either of one between Smart Card & Sim Card has to be inserted in the
Smart Card PICTail board. If both the cards are inserted at a time in the PICTail card, then the demo won’t work successfully.

If the user wants to connect the smart card signals to different port pins of the micro, then the pin mapping in “sc_config.h” file needs to be modified.

Enabling the macro "SC_PROTO_T1" in “sc_config.h” file, will enable the smart card library to support both T=0 & T=1 cards. Disabling the macro "SC_PROTO_T1" in “sc_config.h” file, will enable the smart card library to support only T=0 cards.

Configuring the Library
This section lists the API provided by Microchip Smartcard Library

### Topics

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Functions</strong></td>
<td>The following table lists functions in this documentation.</td>
</tr>
<tr>
<td><strong>Types</strong></td>
<td>The following table lists types in this documentation.</td>
</tr>
<tr>
<td><strong>Macros</strong></td>
<td>The following table lists macros in this documentation.</td>
</tr>
<tr>
<td><strong>Files</strong></td>
<td>The following table lists files in this documentation.</td>
</tr>
<tr>
<td><strong>Variables</strong></td>
<td>The following table lists variables in this documentation.</td>
</tr>
</tbody>
</table>

### Files

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SClib.h</strong></td>
<td>FileName: SClib.h Dependencies: See INCLUDES section Processor: PIC18, PIC24 &amp; PIC32 Microcontrollers Hardware: This demo is natively intended to be used on Exp 16, LPC &amp; HPC Exp board. This demo can be modified for use on other hardware platforms. Complier: Microchip</td>
</tr>
</tbody>
</table>

C18 (for PIC18), C30 (for PIC24) & C30 (for PIC32) Company: Microchip Technology, Inc.

Software License Agreement:
The software supplied herewith by Microchip Technology Incorporated (the “Company”) for its PIC® Microcontroller is intended and supplied to you, the Company’s customer, for use solely and exclusively on Microchip PIC Microcontroller products. The software is owned by the Company and/or ...

more

## Functions

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SC_CardPresent</td>
<td>This macro checks if card is inserted in the socket</td>
</tr>
<tr>
<td>SC_DoPPS</td>
<td>This function does the PPS exchange with the smart card &amp; configures the baud rate of the PIC UART module as per the PPS response from the smart card.</td>
</tr>
<tr>
<td>SC_GetCardState</td>
<td>This function returns the current state of SmartCard</td>
</tr>
<tr>
<td>SC_Initialize</td>
<td>This function initializes the smart card library</td>
</tr>
<tr>
<td>SC_PowerOnATR</td>
<td>This function performs the power on sequence of the SmartCard and interprets the Answer-to-Reset data received from the card.</td>
</tr>
<tr>
<td>SC_Shutdown</td>
<td>This function Performs the Power Down</td>
</tr>
</tbody>
</table>
The sequence of the SmartCard

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SC_TransactT0</td>
<td>This function Sends/recieves the ISO 7816-4 compaliant APDU commands to the card.</td>
</tr>
<tr>
<td>SC_TransactT1</td>
<td>This function Sends/recieves the ISO 7816-4 compaliant T = 1 commands to the card.</td>
</tr>
</tbody>
</table>

### Macros

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SC_MCP_LIB</strong></td>
<td>Smart Card Library</td>
</tr>
<tr>
<td>SC_ABORT_RESPONSE</td>
<td>PCB byte for Abort Response of T1 Protocol</td>
</tr>
<tr>
<td>SC_AUTHENTICATE</td>
<td>Authenticate Command code to the Smart Card</td>
</tr>
<tr>
<td>SC_BWI</td>
<td>DEFAULT Value of BWI Indicator used in calculation of BWT for T=1 protocol</td>
</tr>
<tr>
<td>SC_CHANGE_PIN</td>
<td>Change Pin Command code to the Smart Card</td>
</tr>
<tr>
<td>SC_CLEAR_CARD</td>
<td>Clear Card Command code to the Smart Card</td>
</tr>
<tr>
<td>SC_CRC_TYPE_EDC</td>
<td>Cyclic Redundancy Check(CRC) type is used for EDC in Epilogue Field</td>
</tr>
<tr>
<td>Command Code</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>SC_CREDIT</td>
<td>Credit Command code to the Smart Card</td>
</tr>
<tr>
<td>SC_CWI</td>
<td>DEFAULT Value of CWI Indicator used in calculation of CWT for T=1 protocol</td>
</tr>
<tr>
<td>SC_DEBIT</td>
<td>Debit Command code to the Smart Card</td>
</tr>
<tr>
<td>SC_GET_RESPONSE</td>
<td>Get Response Command code to the Smart Card</td>
</tr>
<tr>
<td>SC_IFS_RESPONSE</td>
<td>PCB byte for IFS Response of T1 Protocol</td>
</tr>
<tr>
<td>SC_INQUIRE_ACCT</td>
<td>Inquire Account Command code to the Smart Card</td>
</tr>
<tr>
<td>SC_LRC_TYPE_EDC</td>
<td>Longitudinal Redundancy Check(LRC) type is used for EDC in Epilogue Field</td>
</tr>
<tr>
<td>SC_READ_RECORD</td>
<td>Read Record Command code to the Smart Card</td>
</tr>
<tr>
<td>SC_RESYNC_REQ</td>
<td>PCB byte for Resync Request of T1 Protocol</td>
</tr>
<tr>
<td>SC_REVOKE</td>
<td>Revoke Command code to the Smart Card</td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>--------------------------------------------------------------</td>
</tr>
<tr>
<td>SC_SELECT_FILE</td>
<td>Select File Command code to the Smart Card</td>
</tr>
<tr>
<td>SC_START_SESSION</td>
<td>Start Session Command code to the Smart Card</td>
</tr>
<tr>
<td>SC_STATE_CARD_ACTIVE</td>
<td>Card is powered and ATR received</td>
</tr>
<tr>
<td>SC_STATE_CARD_INACTIVE</td>
<td>Card present but not powered</td>
</tr>
<tr>
<td>SC_STATE_CARD_NOT_PRESENT</td>
<td>No Card Detected</td>
</tr>
<tr>
<td>SC_SUBMIT_CODE</td>
<td>Submit Code Command to the Smart Card</td>
</tr>
<tr>
<td>SC_T0ProtocolType</td>
<td>Returns '1' if T=0 protocol is supported &amp; Returns 0 otherwise</td>
</tr>
<tr>
<td>SC_T1ProtocolType</td>
<td>Returns '1' if T=1 protocol is supported &amp; Returns 0 otherwise</td>
</tr>
<tr>
<td>SC_TA1Present</td>
<td>Returns '1' if TA1 present &amp; Returns 0 otherwise</td>
</tr>
<tr>
<td>SC_TA2Present</td>
<td>Returns '1' if TA2 present &amp; Returns 0 otherwise</td>
</tr>
<tr>
<td>SC_TB1Present</td>
<td>Returns '1' if TB1 present &amp; Returns 0 otherwise</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td>SC_TB2Present</td>
<td>Returns '1' if TB2 present &amp; Returns 0 otherwise</td>
</tr>
<tr>
<td>SC_TC1Present</td>
<td>Returns '1' if TC1 present &amp; Returns 0 otherwise</td>
</tr>
<tr>
<td>SC_TC2Present</td>
<td>Returns '1' if TC2 present &amp; Returns 0 otherwise</td>
</tr>
<tr>
<td>SC_TD1Present</td>
<td>Returns '1' if TD1 present &amp; Returns 0 otherwise</td>
</tr>
<tr>
<td>SC_TD2Present</td>
<td>Returns '1' if TD2 present &amp; Returns 0 otherwise</td>
</tr>
<tr>
<td>SC_WAIT_TIME_EXT_RESPONSE</td>
<td>PCB byte for Wait Time Extension Response of T1 Protocol</td>
</tr>
<tr>
<td>SC_WI</td>
<td>DEFAULT Value of WI Indicator used in calculation of WWT for T=0 protocol</td>
</tr>
<tr>
<td>SC_WRITE_RECORD</td>
<td>Write Record Command code to the Smart Card</td>
</tr>
</tbody>
</table>

**Types**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SC_APDU_COMMAND</td>
<td>SmartCard APDU Command 7816-4</td>
</tr>
</tbody>
</table>
### Variables

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>scATR_HistoryBuffer</strong></td>
<td>Historical bytes sent by Smart Card</td>
</tr>
<tr>
<td><strong>scATR_HistoryLength</strong></td>
<td>Number of Historical bytes present</td>
</tr>
<tr>
<td><strong>scATRLength</strong></td>
<td>length of ATR data sent by smart card</td>
</tr>
<tr>
<td><strong>scCardATR</strong></td>
<td>ATR data sent by smartcard.</td>
</tr>
<tr>
<td><strong>scLastError</strong></td>
<td>Smart Card Error type is stored in this variable</td>
</tr>
<tr>
<td><strong>scPPSresponse</strong></td>
<td>PPS Response Bytes</td>
</tr>
<tr>
<td><strong>scPPSresponseLength</strong></td>
<td>Length of PPS Response</td>
</tr>
<tr>
<td><strong>scTA1</strong></td>
<td>TA1 determines the clock-rate conversion factor F &amp; bit-rate-adjustment factor D</td>
</tr>
<tr>
<td><strong>scTA2</strong></td>
<td>TA2 determines whether the smart card will operate in specific mode or negotiable mode following the ATR</td>
</tr>
<tr>
<td><strong>scTA3</strong></td>
<td>TA3 conveys the Information Field Size Integer (IFSI) for the smart card.</td>
</tr>
<tr>
<td><strong>scTB1</strong></td>
<td>TB1 conveys information on the smart card's programming voltage requirements.</td>
</tr>
<tr>
<td><strong>scTB2</strong></td>
<td>TB2 conveys PI2, which determines the value of programming voltage required by the smart card. The value of PI1 in TB1 is superceded when TB2 is present.</td>
</tr>
<tr>
<td><strong>scTB3</strong></td>
<td>TB3 indicates the value of the Character Waiting Time Integer (CWI) and Block Waiting Time Integer (BWI) used to compute the Character Waiting Time (CWT) and Block Waiting Time (BWT).</td>
</tr>
<tr>
<td><strong>scTC1</strong></td>
<td>TC1 determines the extra guard time to be added between consecutive characters sent to the smart card from the terminal.</td>
</tr>
<tr>
<td><strong>scTC2</strong></td>
<td>TC2 is specific to protocol type T=0. TC2 conveys work waiting-time integer (WI) that determines the maximum interval between the leading edge of the start bit of any character sent by the smart card and the leading edge of the start bit of the previous character sent either by the card or the reader.</td>
</tr>
<tr>
<td><strong>scTC3</strong></td>
<td>When TC3 is present, it indicates the type of block-error detection to be</td>
</tr>
</tbody>
</table>
used. When TC3 is not present, the default longitudinal redundancy check (LRC) is used.

<table>
<thead>
<tr>
<th></th>
<th>scTD1</th>
<th>TD1 indicates if any further interface bytes are to be transmitted, and if so, which protocol will be used.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>scTD2</td>
<td>The TD2 character has the same function as the TD1 character.</td>
</tr>
<tr>
<td></td>
<td>scTD3</td>
<td>TD3 indicates interface bytes similar to that of TD1 &amp; TD2</td>
</tr>
</tbody>
</table>

Library API

Microchip Smart Card Library 1.02.8 - [July 18, 2012]
Copyright © 2012 Microchip Technology, Inc. All rights reserved.
## Functions

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SC_CardPresent</strong></td>
<td>This macro checks if card is inserted in the socket</td>
</tr>
<tr>
<td><strong>SC_DoPPS</strong></td>
<td>This function does the PPS exchange with the smart card &amp; configures the baud rate of the PIC UART module as per the PPS response from the smart card.</td>
</tr>
<tr>
<td><strong>SC_GetCardState</strong></td>
<td>This function returns the current state of SmartCard</td>
</tr>
<tr>
<td><strong>SC_Initialize</strong></td>
<td>This function initializes the smart card library</td>
</tr>
<tr>
<td><strong>SC_PowerOnATR</strong></td>
<td>This function performs the power on sequence of the SmartCard and interprets the Answer-to-Reset data received from the card.</td>
</tr>
<tr>
<td><strong>SC_Shutdown</strong></td>
<td>This function Performs the Power Down sequence of the SmartCard</td>
</tr>
<tr>
<td><strong>SC_TransactT0</strong></td>
<td>This function Sends/recieves the ISO 7816-4 compliant APDU commands to the card.</td>
</tr>
<tr>
<td><strong>SC_TransactT1</strong></td>
<td>This function Sends/recieves the ISO 7816-4 compliant T = 1 commands to the card.</td>
</tr>
</tbody>
</table>
SC_CardPresent Function

C

BOOL SC_CardPresent();

Description

This macro checks if card is inserted in the socket

Preconditions

SC_Initialize() is called

Remarks

None

Library API > Functions > SC_CardPresent Function
SC_DoPPS Function

```c
BOOL SC_DoPPS(
    BYTE * ppsPtr
);
```

Description

This function does the PPS exchange with the smart card & configures the baud rate of the PIC UART module as per the PPS response from the smart card.

Preconditions

- **SC_PowerOnATR** was success

Remarks

This function is called when **SC_PowerOnATR()** returns TRUE.
**SC_GetCardState Function**

```c
int SC_GetCardState();
```

**Description**

This function returns the current state of SmartCard

**Preconditions**

`SC_Initialize` is called.

**Return Values**

<table>
<thead>
<tr>
<th>Return Values</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>SC_STATE_CARD_NOT_PRESENT</code></td>
<td>No Card Detected</td>
</tr>
<tr>
<td><code>SC_STATE_CARD_ACTIVE</code></td>
<td>Card is powered and ATR received</td>
</tr>
<tr>
<td><code>SC_STATE_CARD_INACTIVE</code></td>
<td>Card present but not powered</td>
</tr>
</tbody>
</table>

**Remarks**

None
SC_Initialize Function

```
void SC_Initialize();
```

**Description**

This function initializes the smart card library

**Preconditions**

None

**Remarks**

None
**SC_PowerOnATR Function**

```c
BOOL SC_PowerOnATR();
```

**Description**

This function performs the power on sequence of the SmartCard and interprets the Answer-to-Reset data received from the card.

**Preconditions**

- `SC_Initialize()` is called, and card is present

**Remarks**

None

**Library API** > **Functions** > **SC_PowerOnATR Function**
SC_Shutdown Function

```c
void SC_Shutdown();
```

Description

This function Performs the Power Down sequence of the SmartCard

Preconditions

- SC_Initialize is called.

Remarks

None
SC_TransactT0 Function

```c
BOOL SC_TransactT0(  
   SC_APDU_COMMAND* apduCommand,  
   SC_APDU_RESPONSE* apduResponse,  
   BYTE* apduDataBuffer  
);
```

**Description**

This function Sends/recieves the ISO 7816-4 complaiant APDU commands to the card.

**Preconditions**

SC_DoPPS was success or SC_DoPPS functionality not called

**Parameters**

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SC_APDU_COMMAND* apduCommand</td>
<td>Pointer to APDU Command Structure</td>
</tr>
<tr>
<td>SC_APDU_RESPONSE* pResp</td>
<td>Pointer to APDU Response structure</td>
</tr>
<tr>
<td>BYTE* pResp</td>
<td>Pointer to the Command/Response Data buffer</td>
</tr>
</tbody>
</table>

**Remarks**

In the APDU command structure, the LC field defines the number of data bytes to be transmitted to the card. This array can hold
max of 256 bytes, which can be redefined by the user. The LE field in APDU command defines the number of bytes expected to be received from the card. This array can hold max 256 bytes, which can be redefined by the user.
**SC_TransactT1 Function**

```c
BOOL SC_TransactT1(
    SC_T1_PROLOGUE_FIELD* pfield,
    BYTE* iField,
    SC_APDU_RESPONSE* apduResponse
);
```

**Description**

This function Sends/recieves the ISO 7816-4 compliant T = 1 commands to the card.

**Preconditions**

*SC_DoPPS* was success

**Parameters**

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SC_T1_PROLOGUE_FIELD* pfield</td>
<td>Pointer to Prologue Field</td>
</tr>
<tr>
<td>BYTE* iField</td>
<td>Pointer to the Information Field of Tx/Rx Data</td>
</tr>
<tr>
<td>SC_APDU_RESPONSE* apduResponse</td>
<td>Pointer to APDU Response structure</td>
</tr>
</tbody>
</table>

**Library API > Functions > SC_TransactT1 Function**
# Types

## Enumerations

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SC_ERROR</td>
<td>Smart Card error types</td>
</tr>
<tr>
<td>T1BLOCK_TYPE</td>
<td>Block types in T=1 protocol</td>
</tr>
</tbody>
</table>

## Structures

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SC_APDU_COMMAND</td>
<td>SmartCard APDU Command 7816-4</td>
</tr>
<tr>
<td>SC_APDU_RESPONSE</td>
<td>SmartCard APDU Response structure 7816-4</td>
</tr>
<tr>
<td>SC_T1_PROLOGUE_FIELD</td>
<td>Prologue Field for T=1 Protocol</td>
</tr>
</tbody>
</table>
SC_APDU_COMMAND Structure

```c
typedef struct {
    BYTE CLA;
    BYTE INS;
    BYTE P1;
    BYTE P2;
    BYTE LC;
    BYTE LE;
} SC_APDU_COMMAND;
```

Description

SmartCard APDU Command 7816-4

Members

<table>
<thead>
<tr>
<th>Members</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BYTE CLA;</td>
<td>Command class</td>
</tr>
<tr>
<td>BYTE INS;</td>
<td>Operation code</td>
</tr>
<tr>
<td>BYTE P1;</td>
<td>Selection Mode</td>
</tr>
<tr>
<td>BYTE P2;</td>
<td>Selection Option</td>
</tr>
<tr>
<td>BYTE LC;</td>
<td>Data length</td>
</tr>
<tr>
<td>BYTE LE;</td>
<td>Expected length of data to be returned</td>
</tr>
</tbody>
</table>
SC_APDU_RESPONSE Structure

C

typedef struct {
    WORD RXDATALEN;
    BYTE SW1;
    BYTE SW2;
} SC_APDU_RESPONSE;

Description

SmartCard APDU Response structure 7816-4

Members

<table>
<thead>
<tr>
<th>Members</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WORD RXDATALEN;</td>
<td>Received Data length from smart card (excluding SW1 and SW2 bytes)</td>
</tr>
<tr>
<td>BYTE SW1;</td>
<td>Status byte 1</td>
</tr>
<tr>
<td>BYTE SW2;</td>
<td>Status byte 2</td>
</tr>
</tbody>
</table>
### SC_ERROR Enumeration

```c
typedef enum {
    SC_ERR_NONE,
    SC_ERR_CARD_NOT_SUPPORTED,
    SC_ERR_BAR_OR_NO_ATR_RESPONSE,
    SC_ERR_CARD_NOT_PRESENT,
    SC_ERR_CARD_NO_RESPONSE,
    SC_ERR_RECEIVE_LRC,
    SC_ERR_RECEIVE_CRC,
    SC_CARD_VPP_ERR,
    SC_ERR_ATR_DATA,
    SC_ERR_RSV1
} SC_ERROR;
```

### Description

Smart Card error types

### Members

<table>
<thead>
<tr>
<th>Members</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SC_ERR_NONE</td>
<td>No Error</td>
</tr>
<tr>
<td>SC_ERR_CARD_NOT_SUPPORTED</td>
<td>Card Not Supported</td>
</tr>
<tr>
<td>SC_ERR_BAR_OR_NO_ATR_RESPONSE</td>
<td>No ATR Response from the card</td>
</tr>
<tr>
<td>SC_ERR_CARD_NOT_PRESENT</td>
<td>Card Not present in the slot</td>
</tr>
<tr>
<td>SC_ERR_CARD_NO_RESPONSE</td>
<td>No response from the card</td>
</tr>
<tr>
<td>SC_ERR_RECEIVE_LRC</td>
<td>LRC Error in the block received from the card</td>
</tr>
<tr>
<td>------------------------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td>SC_ERR_RECEIVE_CRC</td>
<td>CRC Error in the block received from the card</td>
</tr>
<tr>
<td>SC_CARD_VPP_ERR</td>
<td>VPP Error received from the card</td>
</tr>
<tr>
<td>SC_ERR_ATR_DATA</td>
<td>ERROR in ATR data received from the card</td>
</tr>
<tr>
<td>SC_ERR_RSV1</td>
<td>Smart Card Error 1 (Reserved) - can be used based upon the Application</td>
</tr>
</tbody>
</table>
SC_T1_PROLOGUE_FIELD Structure

C

typedef struct {
  BYTE NAD;
  BYTE PCB;
  BYTE LENGTH;
} SC_T1_PROLOGUE_FIELD;

Description

Prologue Field for T=1 Protocol

Members

<table>
<thead>
<tr>
<th>Members</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BYTE NAD;</td>
<td>Node Address</td>
</tr>
<tr>
<td>BYTE PCB;</td>
<td>Protocol Control Byte</td>
</tr>
<tr>
<td>BYTE LENGTH;</td>
<td>LENGTH of I-Field</td>
</tr>
</tbody>
</table>

Library API > Types > SC_T1_PROLOGUE_FIELD Structure
T1BLOCK_TYPE Enumeration

C

typedef enum {
    I_BLOCK,
    R_BLOCK,
    S_BLOCK,
    INVALID_BLOCK
} T1BLOCK_TYPE;

Description

Block types in T=1 protocol

Members

<table>
<thead>
<tr>
<th>Members</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>I_BLOCK</td>
<td>I Block</td>
</tr>
<tr>
<td>R_BLOCK</td>
<td>R Block</td>
</tr>
<tr>
<td>S_BLOCK</td>
<td>S Block</td>
</tr>
<tr>
<td>INVALID_BLOCK</td>
<td>INVALID BLOCK</td>
</tr>
</tbody>
</table>
## Macros

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SC_MCP_LIB</strong></td>
<td>Smart Card Library</td>
</tr>
<tr>
<td>SC_ABORT_RESPONSE</td>
<td>PCB byte for Abort Response of T1 Protocol</td>
</tr>
<tr>
<td>SC_AUTHENTICATE</td>
<td>Authenticate Command code to the Smart Card</td>
</tr>
<tr>
<td>SC_BWI</td>
<td>DEFAULT Value of BWI Indicator used in calculation of BWT for T=1 protocol</td>
</tr>
<tr>
<td>SC_CHANGE_PIN</td>
<td>Change Pin Command code to the Smart Card</td>
</tr>
<tr>
<td>SC_CLEAR_CARD</td>
<td>Clear Card Command code to the Smart Card</td>
</tr>
<tr>
<td>SC_CRC_TYPE_EDC</td>
<td>Cyclic Redundancy Check(CRC) type is used for EDC in Epilogue Field</td>
</tr>
<tr>
<td>SC CREDIT</td>
<td>Credit Command code to the Smart Card</td>
</tr>
<tr>
<td>SC_CWI</td>
<td>DEFAULT Value of CWI Indicator used in</td>
</tr>
<tr>
<td>Command Code</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>SC_DEBIT</td>
<td>Debit Command code to the Smart Card</td>
</tr>
<tr>
<td>SC_GET_RESPONSE</td>
<td>Get Response Command code to the Smart Card</td>
</tr>
<tr>
<td>SC_IFS_RESPONSE</td>
<td>PCB byte for IFS Response of T1 Protocol</td>
</tr>
<tr>
<td>SC_INQUIRE_ACCT</td>
<td>Inquire Account Command code to the Smart Card</td>
</tr>
<tr>
<td>SC_LRC_TYPE_EDC</td>
<td>Longitudinal Redundancy Check(LRC) type is used for EDC in Epilogue Field</td>
</tr>
<tr>
<td>SC_READ_RECORD</td>
<td>Read Record Command code to the Smart Card</td>
</tr>
<tr>
<td>SC_RESYNC_REQ</td>
<td>PCB byte for Resync Request of T1 Protocol</td>
</tr>
<tr>
<td>SC_REVOKE</td>
<td>Revoke Command code to the Smart Card</td>
</tr>
<tr>
<td>SC_SELECT_FILE</td>
<td>Select File Command code to the Smart Card</td>
</tr>
<tr>
<td>SC_START_SESSION</td>
<td>Start Session Command code to the Smart Card</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>SC_STATE_CARD_ACTIVE</td>
<td>Card is powered and ATR received</td>
</tr>
<tr>
<td>SC_STATE_CARD_INACTIVE</td>
<td>Card present but not powered</td>
</tr>
<tr>
<td>SC_STATE_CARD_NOT_PRESENT</td>
<td>No Card Detected</td>
</tr>
<tr>
<td>SC_SUBMIT_CODE</td>
<td>Submit Code Command to the Smart Card</td>
</tr>
<tr>
<td>SC_T0ProtocolType</td>
<td>Returns '1' if T=0 protocol is supported &amp; Returns 0 otherwise</td>
</tr>
<tr>
<td>SC_T1ProtocolType</td>
<td>Returns '1' if T=1 protocol is supported &amp; Returns 0 otherwise</td>
</tr>
<tr>
<td>SC_TA1Present</td>
<td>Returns '1' if TA1 present &amp; Returns 0 otherwise</td>
</tr>
<tr>
<td>SC_TA2Present</td>
<td>Returns '1' if TA2 present &amp; Returns 0 otherwise</td>
</tr>
<tr>
<td>SC_TB1Present</td>
<td>Returns '1' if TB1 present &amp; Returns 0 otherwise</td>
</tr>
<tr>
<td>SC_TB2Present</td>
<td>Returns '1' if TB2 present &amp; Returns 0 otherwise</td>
</tr>
<tr>
<td>SC_TC1Present</td>
<td>Returns '1' if TC1 present &amp; Returns 0 otherwise</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>SC_TC2Present</td>
<td>Returns '1' if TC2 present &amp; Returns 0 otherwise</td>
</tr>
<tr>
<td>SC_TD1Present</td>
<td>Returns '1' if TD1 present &amp; Returns 0 otherwise</td>
</tr>
<tr>
<td>SC_TD2Present</td>
<td>Returns '1' if TD2 present &amp; Returns 0 otherwise</td>
</tr>
<tr>
<td>SC_WAIT_TIME_EXT_RESPONSE</td>
<td>PCB byte for Wait Time Extension Response of T1 Protocol</td>
</tr>
<tr>
<td>SC_WI</td>
<td>DEFAULT Value of WI Indicator used in calculation of WWT for T=0 protocol</td>
</tr>
<tr>
<td>SC_WRITE_RECORD</td>
<td>Write Record Command code to the Smart Card</td>
</tr>
</tbody>
</table>
__SC_MCP_LIB__ Macro

C
#define __SC_MCP_LIB__

Description

Smart Card Library

Library API > Macros > __SC_MCP_LIB__ Macro
SC_ABORT_RESPONSE Macro

C

#define SC_ABORT_RESPONSE (BYTE)0xE2

Description

PCB byte for Abort Response of T1 Protocol

Library API > Macros > SC_ABORT_RESPONSE Macro
SC_AUTHENTICATE Macro

```c
#define SC_AUTHENTICATE 0x82
```

Description

Authenticate Command code to the Smart Card
**SC_BWI Macro**

C

```c
#define SC_BWI (BYTE)0x04
```

**Description**

DEFAULT Value of BWI Indicator used in calculation of BWT for T=1 protocol
SC_CHANGE_PIN Macro

```c
#define SC_CHANGE_PIN 0x24
```

Description

Change Pin Command code to the Smart Card
SC_CLEAR_CARD Macro

#define SC_CLEAR_CARD 0x30

Description

Clear Card Command code to the Smart Card

Library API > Macros > SC_CLEAR_CARD Macro
SC_CRC_TYPE_EDC Macro

```
#define SC_CRC_TYPE_EDC (BYTE)1
```

Description

Cyclic Redundancy Check (CRC) type is used for EDC in Epilogue Field
SC_CREDIT Macro

C

#define SC_CREDIT 0xE2

Description

Credit Command code to the Smart Card

Library API > Macros > SC_CREDIT Macro
SC_CWI Macro

```
C
#define SC_CWI (BYTE)13
```

**Description**

DEFAULT Value of CWI Indicator used in calculation of CWT for T=1 protocol
SC_DEBIT Macro

C

#define SC_DEBIT 0xE6

Description

Debit Command code to the Smart Card

Library API > Macros > SC_DEBIT Macro
SC_GET_RESPONSE Macro

C

```c
#define SC_GET_RESPONSE 0xC0
```

Description

Get Response Command code to the Smart Card
SC_IFS_RESPONSE Macro

C
#define SC_IFS_RESPONSE (BYTE)0xE1

Description

PCB byte for IFS Response of T1 Protocol

Library API > Macros > SC_IFS_RESPONSE Macro
SC_INQUIRE_ACCT Macro

```c
#define SC_INQUIRE_ACCT 0xE4
```

Description

Inquire Account Command code to the Smart Card
SC_LRC_TYPE_EDC Macro

C

#define SC_LRC_TYPE_EDC (BYTE)0

Description

Longitudnal Redundancy Check(LRC) type is used for EDC in Epilogue Field
SC_READ_RECORD Macro

C

#define SC_READ_RECORD 0xB2

Description

Read Record Command code to the Smart Card

Library API > Macros > SC_READ_RECORD Macro
**SC_RESYNC_REQ Macro**

```c
#define SC_RESYNC_REQ (BYTE)0xC0
```

**Description**

PCB byte for Resync Request of T1 Protocol

[Library API > Macros > SC_RESYNC_REQ Macro](#)
SC_REVOKE Macro

C

#define SC_REVOKE 0xE8

Description

Revoke Command code to the Smart Card

Library API > Macros > SC_REVOKE Macro
SC_SELECT_FILE Macro

```c
#define SC_SELECT_FILE 0xA4
```

Description

Select File Command code to the Smart Card

Library API > Macros > SC_SELECT_FILE Macro
SC_START_SESSION Macro

C

#define SC_START_SESSION 0x84

Description

Start Session Command code to the Smart Card

Library API > Macros > SC_START_SESSION Macro
SC_STATE_CARD_ACTIVE Macro

```c
#define SC_STATE_CARD_ACTIVE 20  // Card is powered and ATR received
```

**Description**

Card is powered and ATR received
SC_STATE_CARD_INACTIVE Macro

```c
#define SC_STATE_CARD_INACTIVE 30  // Card present
```

**Description**

Card present but not powered
SC_STATE_CARD_NOT_PRESENT Macro

```c
#define SC_STATE_CARD_NOT_PRESENT 10 // No Card Detected
```

**Description**

No Card Detected
SC_SUBMIT_CODE Macro

C

#define SC_SUBMIT_CODE 0x20

Description

Submit Code Command to the Smart Card

Library API > Macros > SC_SUBMIT_CODE Macro
SC_T0ProtocolType Macro

C

#define SC_T0ProtocolType (((scTD1 & 0x0F) == 0x00)?TRUE:FALSE)

Description

Returns '1' if T=0 protocol is supported & Returns 0 otherwise
SC_T1ProtocolType Macro

```c
#define SC_T1ProtocolType (((scTD1 & 0x0F) == 0x01)?TRUE:FALSE)
```

Description

Returns '1' if T=1 protocol is supported & Returns 0 otherwise
SC_TA1Present Macro

C
#define SC_TA1Present ((scCardATR[1] & 0x10)?TRUE:FALSE)

Description

Returns '1' if TA1 present & Returns 0 otherwise
**SC_TA2Present Macro**

```c
#define SC_TA2Present ((scTD1 & 0x10)?TRUE:FALSE)
```

**Description**

Returns '1' if TA2 present & Returns 0 otherwise
SC_TB1Present Macro

```c
#define SC_TB1Present (scCardATR[1] & 0x20)?TRUE:FALSE)
```

Description

Returns '1' if TB1 present & Returns 0 otherwise
### SC_TB2Present Macro

```c
#define SC_TB2Present ((scTD1 & 0x20)?TRUE:FALSE)
```

### Description

Returns '1' if TB2 present & Returns 0 otherwise
SC_TC1Present Macro

C

#define SC_TC1Present ((scCardATR[1] & 0x40) ? TRUE : FALSE)

Description

Returns '1' if TC1 present & Returns 0 otherwise
SC_TC2Present Macro

C
#define SC_TC2Present ((scTD1 & 0x40)?TRUE:FALSE)

Description

Returns '1' if TC2 present & Returns 0 otherwise
SC_TD1Present Macro

```c
#define SC_TD1Present ((scCardATR[1] & 0x80)?TRUE:FALSE)
```

### Description

Returns '1' if TD1 present & Returns 0 otherwise
SC_TD2Present Macro

```c
#define SC_TD2Present ((scTD1 & 0x80)?TRUE:FALSE)
```

Description

Returns '1' if TD2 present & Returns 0 otherwise

Library API > Macros > SC_TD2Present Macro
### SC_WAIT_TIME_EXT_RESPONSE Macro

<table>
<thead>
<tr>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>#define SC_WAIT_TIME_EXT_RESPONSE (BYTE)0xE3</td>
</tr>
</tbody>
</table>

**Description**

PCB byte for Wait Time Extension Response of T1 Protocol
**SC_WI Macro**

```c
#define SC_WI (BYTE)0x0A
```

**Description**

DEFAULT Value of WI Indicator used in calculation of WWT for T=0 protocol
SC_WRITE_RECORD Macro

```
C
#define SC_WRITE_RECORD 0xD2
```

Description

Write Record Command code to the Smart Card
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SClib.h</strong></td>
<td>FileName: SClib.h Dependencies: See INCLUDES section Processor: PIC18, PIC24 &amp; PIC32 Microcontrollers Hardware: This demo is natively intended to be used on Exp 16, LPC &amp; HPC Exp board. This demo can be modified for use on other hardware platforms. Compiler: Microchip C18 (for PIC18), C30 (for PIC24) &amp; C30 (for PIC32) Company: Microchip Technology, Inc. Software License Agreement: The software supplied herewith by Microchip Technology Incorporated (the “Company”) for its PIC® Microcontroller is intended and supplied to you, the Company's customer, for use solely and exclusively on Microchip PIC Microcontroller products. The software is owned by the Company and/or... [more]</td>
</tr>
</tbody>
</table>

**Library API > Files**

Microchip Smart Card Library 1.02.8 - [July 18, 2012]  
Copyright © 2012 Microchip Technology, Inc. All rights reserved.
FileName: SClib.h  Dependencies: See INCLUDES section
Processor: PIC18, PIC24 & PIC32 Microcontrollers
Hardware: This demo is natively intended to be used on Exp 16, LPC & HPC Exp board. This demo can be modified for use on other hardware platforms.
Compiler: Microchip C18 (for PIC18), C30 (for PIC24) & C30 (for PIC32)
Company: Microchip Technology, Inc.

Software License Agreement:

The software supplied herewith by Microchip Technology Incorporated (the “Company”) for its PIC® Microcontroller is intended and supplied to you, the Company’s customer, for use solely and exclusively on Microchip PIC Microcontroller products. The software is owned by the Company and/or its supplier, and is protected under applicable copyright laws. All rights are reserved. Any use in violation of the foregoing restrictions may subject the user to criminal sanctions under applicable laws, as well as to civil liability for the breach of the terms and conditions of this license.

THIS SOFTWARE IS PROVIDED IN AN “AS IS” CONDITION. NO WARRANTIES, WHETHER EXPRESS, IMPLIED OR STATUTORY, INCLUDING, BUT NOT LIMITED TO, IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE APPLY TO THIS SOFTWARE. THE COMPANY SHALL NOT, IN ANY CIRCUMSTANCES, BE LIABLE FOR SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES, FOR ANY REASON WHATSOEVER.
File Description:

Change History: Rev Description

--- ----------------------------------------
1.0 Initial release
1.01 Cleaned up unnecessary variables, supported T=1 protocol and improvements in T=0 functions following the coding standards
1.02 Modified PPS functionality API. Modified the code in more structured way.

Enumerations

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SC_ERROR</td>
<td>Smart Card error types</td>
</tr>
<tr>
<td>T1BLOCK_TYPE</td>
<td>Block types in T=1 protocol</td>
</tr>
</tbody>
</table>

Functions

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SC_CardPresent</td>
<td>This macro checks if card is inserted in the socket</td>
</tr>
<tr>
<td>SC_DoPPS</td>
<td>This function does the PPS exchange with the smart card &amp; configures the baud rate of the PIC UART module as per the PPS response from the smart card.</td>
</tr>
<tr>
<td>SC_GetCardState</td>
<td>This function returns the current state of SmartCard</td>
</tr>
<tr>
<td>SC_Initialize</td>
<td>This function initializes the smart card library</td>
</tr>
<tr>
<td>SC_PowerOnATR</td>
<td>This function performs the power on</td>
</tr>
</tbody>
</table>
sequence of the SmartCard and interprets the Answer-to-Reset data received from the card.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SC_Shutdown</strong></td>
<td>This function Performs the Power Down sequence of the SmartCard</td>
</tr>
<tr>
<td><strong>SC_TransactT0</strong></td>
<td>This function Sends/recieves the ISO 7816-4 compaliant APDU commands to the card.</td>
</tr>
<tr>
<td><strong>SC_TransactT1</strong></td>
<td>This function Sends/recieves the ISO 7816-4 compaliant T = 1 commands to the card.</td>
</tr>
</tbody>
</table>

### Macros

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SC_MCP_LIB</strong></td>
<td>Smart Card Library</td>
</tr>
<tr>
<td>SC_ABORT_RESPONSE</td>
<td>PCB byte for Abort Response of T1 Protocol</td>
</tr>
<tr>
<td>SC_AUTHENTICATE</td>
<td>Authenticate Command code to the Smart Card</td>
</tr>
<tr>
<td>SC_BWI</td>
<td>DEFAULT Value of BWI Indicator used in calculation of BWT for T=1 protocol</td>
</tr>
<tr>
<td>SC_CHANGE_PIN</td>
<td>Change Pin Command code to the Smart Card</td>
</tr>
<tr>
<td>SC_CLEAR_CARD</td>
<td>Clear Card Command code to the Smart Card</td>
</tr>
<tr>
<td>Command Code</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>SC_CRC_TYPE_EDC</td>
<td>Cyclic Redundancy Check (CRC) type is used for EDC in Epilogue Field</td>
</tr>
<tr>
<td>SC_CREDIT</td>
<td>Credit Command code to the Smart Card</td>
</tr>
<tr>
<td>SC_CWI</td>
<td>DEFAULT Value of CWI Indicator used in calculation of CWT for T=1 protocol</td>
</tr>
<tr>
<td>SC_DEBIT</td>
<td>Debit Command code to the Smart Card</td>
</tr>
<tr>
<td>SC_GET_RESPONSE</td>
<td>Get Response Command code to the Smart Card</td>
</tr>
<tr>
<td>SC_IFS_RESPONSE</td>
<td>PCB byte for IFS Response of T1 Protocol</td>
</tr>
<tr>
<td>SC_INQUIRE_ACCT</td>
<td>Inquire Account Command code to the Smart Card</td>
</tr>
<tr>
<td>SC_LRC_TYPE_EDC</td>
<td>Longitudinal Redundancy Check (LRC) type is used for EDC in Epilogue Field</td>
</tr>
<tr>
<td>SC_READ_RECORD</td>
<td>Read Record Command code to the Smart Card</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>SC_RESYNC_REQ</td>
<td>PCB byte for Resync Request of T1 Protocol</td>
</tr>
<tr>
<td>SC_REVOKE</td>
<td>Revoke Command code to the Smart Card</td>
</tr>
<tr>
<td>SC_SELECT_FILE</td>
<td>Select File Command code to the Smart Card</td>
</tr>
<tr>
<td>SC_START_SESSION</td>
<td>Start Session Command code to the Smart Card</td>
</tr>
<tr>
<td>SC_STATE_CARD_ACTIVE</td>
<td>Card is powered and ATR received</td>
</tr>
<tr>
<td>SC_STATE_CARD_INACTIVE</td>
<td>Card present but not powered</td>
</tr>
<tr>
<td>SC_STATE_CARD_NOT_PRESENT</td>
<td>No Card Detected</td>
</tr>
<tr>
<td>SC_SUBMIT_CODE</td>
<td>Submit Code Command to the Smart Card</td>
</tr>
<tr>
<td>SC_T0ProtocolType</td>
<td>Returns '1' if T=0 protocol is supported &amp; Returns 0 otherwise</td>
</tr>
<tr>
<td>SC_T1ProtocolType</td>
<td>Returns '1' if T=1 protocol is supported &amp; Returns 0 otherwise</td>
</tr>
<tr>
<td>SC_TA1Present</td>
<td>Returns '1' if TA1 present &amp; Returns 0 otherwise</td>
</tr>
<tr>
<td>SC_TA2Present</td>
<td>Returns '1' if TA2 present &amp; Returns 0 otherwise</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>SC_TB1Present</td>
<td>Returns '1' if TB1 present &amp; Returns 0 otherwise</td>
</tr>
<tr>
<td>SC_TB2Present</td>
<td>Returns '1' if TB2 present &amp; Returns 0 otherwise</td>
</tr>
<tr>
<td>SC_TC1Present</td>
<td>Returns '1' if TC1 present &amp; Returns 0 otherwise</td>
</tr>
<tr>
<td>SC_TC2Present</td>
<td>Returns '1' if TC2 present &amp; Returns 0 otherwise</td>
</tr>
<tr>
<td>SC_TD1Present</td>
<td>Returns '1' if TD1 present &amp; Returns 0 otherwise</td>
</tr>
<tr>
<td>SC_TD2Present</td>
<td>Returns '1' if TD2 present &amp; Returns 0 otherwise</td>
</tr>
<tr>
<td>SC_WAIT_TIME_EXT_RESPONSE</td>
<td>PCB byte for Wait Time Extension Response of T1 Protocol</td>
</tr>
<tr>
<td>SC_WI</td>
<td>DEFAULT Value of WI Indicator used in calculation of WWT for T=0 protocol</td>
</tr>
<tr>
<td>SC_WRITE_RECORD</td>
<td>Write Record Command code to the Smart Card</td>
</tr>
</tbody>
</table>

Structures
# Name | Description
--- | ---
SC_APDU_COMMAND | SmartCard APDU Command 7816-4
SC_APDU_RESPONSE | SmartCard APDU Response structure 7816-4
SC_T1_PROLOGUE_FIELD | Prologue Field for T=1 Protocol

## Variables

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scATR_HistoryBuffer</td>
<td>Historical bytes sent by Smart Card</td>
</tr>
<tr>
<td>scATR_HistoryLength</td>
<td>Number of Historical bytes present</td>
</tr>
<tr>
<td>scATRLength</td>
<td>length of ATR data sent by smart card</td>
</tr>
<tr>
<td>scCardATR</td>
<td>ATR data sent by smartcard.</td>
</tr>
<tr>
<td>scLastError</td>
<td>Smart Card Error type is stored in this variable</td>
</tr>
<tr>
<td>scPPSresponse</td>
<td>PPS Response Bytes</td>
</tr>
<tr>
<td>scPPSresponseLength</td>
<td>Length of PPS Response</td>
</tr>
<tr>
<td>scTA1</td>
<td>TA1 determines the clock-rate conversion factor F &amp; bit-rate-adjustment factor D</td>
</tr>
<tr>
<td>scTA2</td>
<td>TA2 determines whether the smart card will operate in specific mode or negotiable mode following the ATR</td>
</tr>
<tr>
<td><strong>scTA3</strong></td>
<td>TA3 conveys the Information Field Size Integer (IFSI) for the smart card.</td>
</tr>
<tr>
<td><strong>scTB1</strong></td>
<td>TB1 conveys information on the smart card's programming voltage requirements.</td>
</tr>
<tr>
<td><strong>scTB2</strong></td>
<td>TB2 conveys PI2, which determines the value of programming voltage required by the smart card. The value of PI1 in TB1 is superceded when TB2 is present.</td>
</tr>
<tr>
<td><strong>scTB3</strong></td>
<td>TB3 indicates the value of the Character Waiting Time Integer (CWI) and Block Waiting Time Integer (BWI) used to compute the Character Waiting Time (CWT) and Block Waiting Time (BWT).</td>
</tr>
<tr>
<td><strong>scTC1</strong></td>
<td>TC1 determines the extra guard time to be added between consecutive characters sent to the smart card from the terminal.</td>
</tr>
<tr>
<td><strong>scTC2</strong></td>
<td>TC2 is specific to protocol type T=0. TC2 conveys work waiting-time integer (WI) that determines the maximum interval between the leading edge of the start bit of any character sent by the smart card and the leading edge of the start bit of the previous character sent either by the card or the reader.</td>
</tr>
</tbody>
</table>
| **scTC3** | When TC3 is present, it indicates the type of block-error detection to be
used. When TC3 is not present, the default longitudinal redundancy check (LRC) is used.

<table>
<thead>
<tr>
<th></th>
<th>scTD1</th>
<th>TD1 indicates if any further interface bytes are to be transmitted, and if so, which protocol will be used.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>scTD2</td>
<td>The TD2 character has the same function as the TD1 character.</td>
</tr>
<tr>
<td></td>
<td>scTD3</td>
<td>TD3 indicates interface bytes similar to that of TD1 &amp; TD2</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>scATR_HistoryBuffer</td>
<td>Historical bytes sent by Smart Card</td>
<td></td>
</tr>
<tr>
<td>scATR_HistoryLength</td>
<td>Number of Historical bytes present</td>
<td></td>
</tr>
<tr>
<td>scATRLength</td>
<td>length of ATR data sent by smart card</td>
<td></td>
</tr>
<tr>
<td>scCardATR</td>
<td>ATR data sent by smartcard.</td>
<td></td>
</tr>
<tr>
<td>scLastError</td>
<td>Smart Card Error type is stored in this variable</td>
<td></td>
</tr>
<tr>
<td>scPPSResponse</td>
<td>PPS Response Bytes</td>
<td></td>
</tr>
<tr>
<td>scPPSResponseLength</td>
<td>Length of PPS Response</td>
<td></td>
</tr>
<tr>
<td>scTA1</td>
<td>TA1 determines the clock-rate conversion factor F &amp; bit-rate-adjustment factor D</td>
<td></td>
</tr>
<tr>
<td>scTA2</td>
<td>TA2 determines whether the smart card will operate in specific mode or negotiable mode following the ATR</td>
<td></td>
</tr>
<tr>
<td>scTA3</td>
<td>TA3 conveys the Information Field Size Integer (IFSI) for the smart card.</td>
<td></td>
</tr>
<tr>
<td>scTB1</td>
<td>TB1 conveys information on the smart card's programming voltage requirements.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>scTB2</td>
<td>TB2 conveys PI2, which determines the value of programming voltage required by the smart card. The value of PI1 in TB1 is superceded when TB2 is present</td>
</tr>
<tr>
<td>---</td>
<td>--------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>scTB3</td>
<td>TB3 indicates the value of the Character Waiting Time Integer (CWI) and Block Waiting Time Integer (BWI) used to compute the Character Waiting Time (CWT) and Block Waiting Time (BWT).</td>
</tr>
<tr>
<td></td>
<td>scTC1</td>
<td>TC1 determines the extra guard time to be added between consecutive characters sent to the smart card from the terminal.</td>
</tr>
<tr>
<td></td>
<td>scTC2</td>
<td>TC2 is specific to protocol type T=0. TC2 conveys work waiting-time integer (WI) that determines the maximum interval between the leading edge of the start bit of any character sent by the smart card and the leading edge of the start bit of the previous character sent either by the card or the reader</td>
</tr>
<tr>
<td></td>
<td>scTC3</td>
<td>When TC3 is present, it indicates the type of block-error detection to be used. When TC3 is not present, the default longitudinal redundancy check (LRC) is used.</td>
</tr>
<tr>
<td></td>
<td>scTD1</td>
<td>TD1 indicates if any further interface bytes are to be transmitted, and if so, which protocol will be used.</td>
</tr>
<tr>
<td>scTD2</td>
<td>The TD2 character has the same function as the TD1 character.</td>
<td></td>
</tr>
<tr>
<td>-------</td>
<td>-----------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>scTD3</td>
<td>TD3 indicates interface bytes similar to that of TD1 &amp; TD2</td>
<td></td>
</tr>
</tbody>
</table>
scATR_HistoryBuffer Variable

C
BYTE* scATR_HistoryBuffer;

Description

Historical bytes sent by Smart Card

Library API > Variables > scATR_HistoryBuffer Variable
scATR_HistoryLength Variable

C

BYTE scATR_HistoryLength;

Description

Number of Historical bytes present
scATRLength Variable

C

BYTE scATRLength;

Description

length of ATR data sent by smart card

Library API > Variables > scATRLength Variable
scCardATR Variable

```
C
BYTE scCardATR[];
```

**Description**

ATR data sent by smartcard.

**Library API > Variables > scCardATR Variable**
scLastError Variable

C

`SC_ERROR` `scLastError;`

Description

Smart Card Error type is stored in this variable
scPPSresponse Variable

C
BYTE scPPSresponse[7];

Description

PPS Response Bytes

Library API > Variables > scPPSresponse Variable
scPPSresponseLength Variable

C

BYTE scPPSresponseLength;

Description

Length of PPS Response

Library API > Variables > scPPSresponseLength Variable
scTA1 Variable

| C | BYTE scTA1; |

Description

TA1 determines the clock-rate conversion factor F & bit-rate-adjustment factor D
scTA2 Variable

C
BYTE scTA2;

Description

TA2 determines whether the smart card will operate in specific mode or negotiable mode following the ATR.
scTA3 Variable

```
C
BYTE scTA3;
```

**Description**

TA3 conveys the Information Field Size Integer (IFSI) for the smart card.

Library API > Variables > scTA3 Variable
**scTB1 Variable**

```c
BYTE scTB1;
```

**Description**

TB1 conveys information on the smart card's programming voltage requirements.
scTB2 Variable

<table>
<thead>
<tr>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>BYTE scTB2;</td>
</tr>
</tbody>
</table>

Description

TB2 conveys PI2, which determines the value of programming voltage required by the smart card. The value of PI1 in TB1 is superceded when TB2 is present.
scTB3 Variable

C
BYTE scTB3;

Description

TB3 indicates the value of the Character Waiting Time Integer (CWI) and Block Waiting Time Integer (BWI) used to compute the Character Waiting Time (CWT) and Block Waiting Time (BWT).
scTC1 Variable

| C BYTE scTC1; |

Description

TC1 determines the extra guard time to be added between consecutive characters sent to the smart card from the terminal.
scTC2 Variable

```
C
BYTE scTC2;
```

Description

TC2 is specific to protocol type T=0. TC2 conveys work waiting-time integer (WI) that determines the maximum interval between the leading edge of the start bit of any character sent by the smart card and the leading edge of the start bit of the previous character sent either by the card or the reader.
scTC3 Variable

C
BYTE scTC3;

Description

When TC3 is present, it indicates the type of block-error detection to be used. When TC3 is not present, the default longitudinal redundancy check (LRC) is used.
# scTD1 Variable

```c
BYTE scTD1;
```

## Description

TD1 indicates if any further interface bytes are to be transmitted, and if so, which protocol will be used.
scTD2 Variable

```c
BYTE scTD2;
```

Description

The TD2 character has the same function as the TD1 character.
scTD3 Variable

C
BYTE scTD3;

Description

TD3 indicates interface bytes similar to that of TD1 & TD2
Integrating with an Existing Application

It is easy to integrate the smart card library with the existing applications. The smart card library uses UART and 4 I/O port pins.

The pins used for the communication b/w the smart card & PIC microcontroller are given in Configuring the Library section. “sc_config.h” is the only file where the user has to modify to port the smart card stack to different PIC microcontrollers.

The API's that needs to be called by the main application are mentioned in SClib.h file. Please refer "How the Library Works" to know the usage of smart card library API's.
Revision History

This section describes in more detail the changes made between versions of the Smart Card Library stack. This section generally discusses only changes made to the core files (those found in the "Microchip Solutions\Microchip" folder). This section generally doesn't include changes to the demo projects unless those changes are important to know about. This section also doesn't encompass minor changes to the stack files such as arrangement or locations of definitions or any other organizational changes.

Topics

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| **v1.02.8** | 1. In SClib.c.:-
  ◦ "SC_TransactT0" function is modified to handle a 256 bytes read from smart card as per the "Case 2S" requirement of ISO 7816 specification.
  ◦ The assignment of "apduResponse->SW1" & "apduResponse->SW2" is modified in "SC_TransactT1" function
2. In SCpic24.c, SCpic18.c, SCpic32.c & SCdspic33f.c:-
  ◦ "SCdrv_InitUART" function is modified to switch on the power supply to the smart card during initialization phase. |
| **v1.02.6** | 1. In SClib.c.:-
  ◦ Changed the size of input/output parameters of static functions 'SC_UpdateCRC', 'SC_UpdateEDC' & 'SC_SendT1Block'. This |
fix is done to optimize the code.

- Modified the contents of 'SC_UpdateCRC' & 'SC_SendT1Block' function to suit the above change.
- Modified "SC_TransactT0" function, to transmit first byte as 0x00 when LC & LE bytes are 0x00.
- Changed the local variable 'edc' from 'WORD' type to 'unsigned short int' type (in static function :- 'SC_ReceiveT1Block')

2. In SCpic24.c, SCpic18.c, SCpic32.c & SCdspic33f.c:-
   - The variable 'delayLapsedFlag' is declared as 'volatile' type, as it is modified in the Interrupt Service Routine.

1. In SClib.c:-
   - The wait time was getting reinitialized to default value while communicating with smart card using T = 0 protocol. So deleted "t0WWTetu = 10752;" in "SC_TransactT0" function.
   - Modified the function "SC_SendT1Block" in such a way that EDC is transmitted more efficiently for LRC/CRC mode in T = 1 protocol.
   - Initialized local variable "txLength" to '0' in function "SC_TransactT1" to remove non-critical compiler warnings.

2. In sc_config.h
   - Removed the following unused file inclusions:-
     1. libpic30.h
     2. math.h
     3. delays.h
     4. plib.h
| v1.02.2 | 1. Modified the PPS functionality as per ISO 7816 standard.  
2. Fixed BWT (Block Wait Time) and WT (Wait Time) calculation issues.  
3. Removed recursive function calls and modified the code to make it well structured and organized.  
4. Modified "SCdrv_EnableDelayTimerIntr" and "SCdrv_SetDelayTimerCnt" macros to configure 16 bit timers (this macro is used to provide delays).  
5. "WaitMicroSec()" & "WaitMilliSec()" macros are removed from sc_config.h file.  
6. Moved timer interrupts (used by smart card stack) to ISO 7816 hardware driver files.  
7. Added "TIMER1_SINGLE_COUNT_MICRO SECONDS" and "TIMER0_SINGLE_COUNT_MICRO SECONDS" macros in sc_config.h file.  
8. WaitMicroSec() and WaitMilliSec() delay functions have been rewritten in the ISO 7816 driver files to provide accurate... [more] |
| v1.02 | Supported smart card library stack to PIC32, PIC24H and dsPIC33F devices. |
| v1.01 | The following list of variable names has been changed to follow a common coding standard across the smartcard library. |

**Revision History**

Microchip Smart Card Library 1.02.8 - [July 18, 2012]  
Copyright © 2012 Microchip Technology, Inc. All rights reserved.  
[Contents] [Index] [Reference] [Home]
1. In SClib.c:-
   - "SC_TransactT0" function is modified to handle a 256 bytes read from smart card as per the "Case 2S" requirement of ISO 7816 specification.
   - The assignment of "apduResponse->SW1" & "apduResponse->SW2" is modified in "SC_TransactT1" function

2. In SCpic24.c, SCpic18.c, SCpic32.c & SCdspic33f.c:-
   - "SCdrv_InitUART" function is modified to switch on the power supply to the smart card during initialization phase.
v1.02.6

1. In SClib.c:-
   - Changed the size of input/output parameters of static functions 'SC_UpdateCRC', 'SC_UpdateEDC' & 'SC_SendT1Block'. This fix is done to optimize the code.
   - Modified the contents of 'SC_UpdateCRC' & 'SC_SendT1Block' function to suit the above change.
   - Modified "SC_TransactT0" function, to transmit first byte as 0x00 when LC & LE bytes are 0x00.
   - Changed the local variable 'edc' from 'WORD' type to 'unsigned short int' type (in static function :- 'SC_ReceiveT1Block')

2. In SCpic24.c, SCpic18.c, SCpic32.c & SCdspic33f.c:-
   - The variable 'delayLapsedFlag' is declared as 'volatile' type, as it is modified in the Interrupt Service Routine.
1. In SClib.c:-
   ○ The wait time was getting reinitialized to default value while communicating with smart card using T = 0 protocol. So deleted "t0WWTetu = 10752;" in "SC_TransactT0" function.
   ○ Modified the function "SC_SendT1Block" in such a way that EDC is transmitted more efficiently for LRC/CRC mode in T = 1 protocol.
   ○ Initialized local variable "txLength" to '0' in function "SC_TransactT1" to remove non-critical compiler warnings.

2. In sc_config.h
   ○ Removed the following unused file inclusions:-
     1. libpic30.h
     2. math.h
     3. delays.h
     4. plib.h
v1.02.2

1. Modified the PPS functionality as per ISO 7816 standard.
2. Fixed BWT (Block Wait Time) and WT (Wait Time) calculation issues.
3. Removed recursive function calls and modified the code to make it well structured and organized.
4. Modified "SCdrv_EnableDelayTimerIntr" and "SCdrv_SetDelayTimerCnt" macros to configure 16 bit timers (this macro is used to provide delays).
5. "WaitMicroSec()" & "WaitMilliSec()" macros are removed from sc_config.h file.
6. Moved timer interrupts (used by smart card stack) to ISO 7816 hardware driver files.
7. Added "TIMER1_SINGLE_COUNT_MICRO_SECONDS" and "TIMER0_SINGLE_COUNT_MICRO_SECONDS" macros in sc_config.h file.
8. WaitMicroSec() and WaitMilliSec() delay functions have been rewritten in the ISO 7816 driver files to provide accurate delays.
9. The following PPS response variables have been added as part of the global memory.

<table>
<thead>
<tr>
<th>Names</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scPPSresponse[7]</td>
<td>PPS Response Bytes from smart card</td>
</tr>
<tr>
<td>scPPSresponseLength</td>
<td>Length of PPS Response</td>
</tr>
</tbody>
</table>

The prototype definition of function “SC_DoPPS()” has been changed to “SC_DoPPS( BYTE *ppsPtr )”. The input parameter for “SC_DoPPS” function is PPS request string. This feature enables the user to send the desired PPS request to the card.

Revision History > v1.02.2
v1.02

Supported smart card library stack to PIC32, PIC24H and dsPIC33F devices.

Revision History > v1.02
The following list of variable names has been changed to follow a common coding standard across the smartcard library.

<table>
<thead>
<tr>
<th>Changed From</th>
<th>Changed To</th>
</tr>
</thead>
<tbody>
<tr>
<td>SC_CardATR</td>
<td>scCardATR</td>
</tr>
<tr>
<td>SC_ATRLen</td>
<td>scATRLength</td>
</tr>
<tr>
<td>SC_LastError</td>
<td>scLastError</td>
</tr>
<tr>
<td>SC_TA1</td>
<td>scTA1</td>
</tr>
<tr>
<td>SC_TA2</td>
<td>scTA2</td>
</tr>
<tr>
<td>SC_TA3</td>
<td>scTA3</td>
</tr>
<tr>
<td>SC_TB1</td>
<td>scTB1</td>
</tr>
<tr>
<td>SC_TB2</td>
<td>scTB2</td>
</tr>
<tr>
<td>SC_TB3</td>
<td>scTB3</td>
</tr>
<tr>
<td>SC_TC1</td>
<td>scTC1</td>
</tr>
<tr>
<td>SC_TC2</td>
<td>scTC2</td>
</tr>
<tr>
<td>SC_TC3</td>
<td>scTC3</td>
</tr>
<tr>
<td>SC_TD1</td>
<td>scTD1</td>
</tr>
<tr>
<td>SC_TD2</td>
<td>scTD2</td>
</tr>
<tr>
<td>SC_TD3</td>
<td>scTD3</td>
</tr>
<tr>
<td>SC_ATR_HistBfr</td>
<td>scATR_HistoryBuffer</td>
</tr>
<tr>
<td>SC_ATR_HistLen</td>
<td>scATR_HistoryLength</td>
</tr>
</tbody>
</table>

The following list of type definitions has been changed to make
them more understandable.

<table>
<thead>
<tr>
<th>Changed From</th>
<th>Changed To</th>
</tr>
</thead>
<tbody>
<tr>
<td>SC_APDU_Cmd</td>
<td>SC_APDU_COMMAND</td>
</tr>
<tr>
<td>SC_APDU_Resp</td>
<td>SC_APDU_RESPONSE</td>
</tr>
</tbody>
</table>

The function name “SC_Transact” has been changed to “SC_TransactT0” to signify that this function handles only T=0 transactions with the smart card.

The function name “SC_TransactT1” has been added newly to signify that this function handles only T=1 transactions with the smart card. The application has to call “SC_TransactT0” or “SC_TransactT1” function depending upon the card inserted.

Revision History > v1.01
Contents

Introduction
SW License Agreement

Release Notes
Resource Usage - PIC18
Resource Usage - PIC24F
Resource Usage - PIC24H
Resource Usage - dsPIC33F
Resource Usage - PIC32

Smartcard Library Overview
Library Architecture
How the Library Works

Getting Started - Smart Card Demo

Required Hardware
Configuration 1: PIC18 Explorer Board
Configuration 2: Explorer 16 Board
Configuration 3: Low Pin Count USB Development Kit
Configuration 4: PICDEM FS USB Board

Configuring the Hardware:
Configuration using PIC18 Explorer Board
Configuration using Explorer 16 Board
Configuration using PIC18F14K50 + LPC Board
Configuration using PICDEM FS USB Board

Firmware
Running the Demo
Configuring the Library

Library API

Functions
SC_CardPresent Function
SC_DoPPS Function
SC_GetCardState Function
SC_Initialize Function
SC_PowerOnATR Function
SC_Shutdown Function
SC_TransactT0 Function
SC_TransactT1 Function

Types
SC_APDU_COMMAND Structure
SC_APDU_RESPONSE Structure
SC_ERROR Enumeration
SC_T1_PROLOGUE_FIELD Structure
T1BLOCK_TYPE Enumeration

Macros
__SC_MCP_LIB__ Macro
SC_ABORT_RESPONSE Macro
SC_AUTHENTICATE Macro
SC_BWI Macro
SC_CHANGE_PIN Macro
SC_CLEAR_CARD Macro
SC_CRC_TYPE_EDC Macro
SC_CREDIT Macro
SC_CWI Macro
SC_DEBIT Macro
SC_GET_RESPONSE Macro
SC_IFS_RESPONSE Macro
SC_INQUIRE_ACCT Macro
SC_LRC_TYPE_EDC Macro
SC_READ_RECORD Macro
SC_RESYNC_REQ Macro
SC_REVOKE Macro
SC_SELECT_FILE Macro
SC_START_SESSION Macro
SC_STATE_CARD_ACTIVE Macro
SC_STATE_CARD_INACTIVE Macro
SC_STATE_CARD_NOT_PRESENT Macro
SC_SUBMIT_CODE Macro
SC_T0ProtocolType Macro
SC_T1ProtocolType Macro
SC_TA1Present Macro
SC_TA2Present Macro
SC_TB1Present Macro
SC_TB2Present Macro
SC_TC1Present Macro
SC_TC2Present Macro
SC_TD1Present Macro
SC_TD2Present Macro
SC_WAIT_TIME_EXT_RESPONSE Macro
SC_WI Macro
SC_WRITE_RECORD Macro

Files
SClib.h

Variables
scATR_HistoryBuffer Variable
scATR_HistoryLength Variable
scATRLength Variable
scCardATR Variable
scLastError Variable
scPPSresponse Variable
scPPSresponseLength Variable
scTA1 Variable
scTA2 Variable
scTA3 Variable
scTB1 Variable
scTB2 Variable
scTB3 Variable
scTC1 Variable
scTC2 Variable
Integrating with an Existing Application

**Revision History**

v1.02.8
v1.02.6
v1.02.4
v1.02.2
v1.02
v1.01
__SC_MCP_LIB__ macro

Configuration 1: PIC18 Explorer Board
Configuration 2: Explorer 16 Board
Configuration 3: Low Pin Count USB Development Kit
Configuration 4: PICDEM FS USB Board
Configuration using Explorer 16 Board
Configuration using PIC18 Explorer Board
Configuration using PIC18F14K50 + LPC Board
Configuration using PICDEM FS USB Board
Configuring the Hardware:
Configuring the Library

F
Files
Firmware
Functions

G
Getting Started - Smart Card

SC_Shutdown function
SC_START_SESSION macro
SC_STATE_CARD_ACTIVE macro
SC_STATE_CARD_INACTIVE macro
SC_STATE_CARD_NOT_PRESENT macro
SC_SUBMIT_CODE macro
SC_T0ProtocolType macro
SC_T1_PROLOGUE_FIELD structure
SC_T1ProtocolType macro
SC_TA1Present macro
SC_TA2Present macro
SC_TB1Present macro
SC_TB2Present macro
SC_TC1Present macro
SC_TC2Present macro
SC_TD1Present macro
SC_TD2Present macro
SC_TransactT0 function
SC_TransactT1 function
SC_WAIT_TIME_EXT_RESPONSE macro
SC_WI macro
SC_WRITE_RECORD macro
scATR_HistoryBuffer variable
scATR_HistoryLength variable
scATRLength variable
scCardATR variable
Integrating with an Existing Application

Library API

Library Architecture

Macros

Release Notes

Required Hardware

Resource Usage - dsPIC33F

Resource Usage - PIC18

Resource Usage - PIC24F

Resource Usage - PIC24H

Resource Usage - PIC32

Revision History

Running the Demo

SC_ABORT_RESPONSE macro

SC_APDU_COMMAND structure

SC_APDU_RESPONSE

scLastError variable

SClib.h

scPPSresponse variable

scPPSresponseLength variable

scTA1 variable

scTA2 variable

scTA3 variable

scTB1 variable

scTB2 variable

scTB3 variable

scTC1 variable

scTC2 variable

scTC3 variable

scTD1 variable

scTD2 variable

scTD3 variable

Smartcard Library Overview

SW License Agreement

T1BLOCK_TYPE enumeration

Types

v1.01

v1.02

v1.02.2

v1.02.4

v1.02.6

v1.02.8

Variables
structure
SC_AUTHENTICATE macro
SC_BWI macro
SC_CardPresent function
SC_CHANGE_PIN macro
SC_CLEAR_CARD macro
SC_CRC_TYPE_EDC macro
SC_CREDIT macro
SC_CWI macro
SC_DEBIT macro
SC_DoPPS function
SC_ERROR enumeration
SC_GET_RESPONSE macro
SC_GetCardState function
SC_IFS_RESPONSE macro
SC_Initialize function
SC_INQUIRE_ACCT macro
SC_LRC_TYPE_EDC macro
SC_PowerOnATR function
SC_READ_RECORD macro
SC_RESYNC_REQ macro
SC_REVOKE macro
SC_SELECT_FILE macro
## Library API Files

### Files

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SClib.h</td>
<td>FileName: SClib.h Dependencies: See INCLUDES section Processor: PIC18, PIC24 &amp; PIC32 Microcontrollers Hardware: This demo is natively intended to be used on Exp 16, LPC &amp; HPC Exp board. This demo can be modified for use on other hardware platforms. Compiler: Microchip C18 (for PIC18), C30 (for PIC24) &amp; C30 (for PIC32) Company: Microchip Technology, Inc. Software License Agreement: The software supplied herewith by Microchip Technology Incorporated (the “Company”) for its PIC® Microcontroller is intended and supplied to you, the Company’s customer, for use solely and exclusively on Microchip PIC Microcontroller products. The software is owned by the Company and/or... more</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>SC_CardPresent</td>
<td>This macro checks if card is inserted in the socket</td>
</tr>
<tr>
<td>SC_DoPPS</td>
<td>This function does the PPS exchange with the smart card &amp; configures the baud rate of the PIC UART module as per the PPS response from the smart card.</td>
</tr>
<tr>
<td>SC_GetCardState</td>
<td>This function returns the current state of SmartCard</td>
</tr>
<tr>
<td>SC_Initialize</td>
<td>This function initializes the smart card library</td>
</tr>
<tr>
<td>SC_PowerOnATR</td>
<td>This function performs the power on sequence of the SmartCard and interprets the Answer-to-Reset data received from the card.</td>
</tr>
<tr>
<td>SC_Shutdown</td>
<td>This function Performs the Power Down sequence of the SmartCard</td>
</tr>
<tr>
<td>SC_TransactT0</td>
<td>This function Sends/recieves the ISO 7816-4 compaliant APDU commands to the card.</td>
</tr>
<tr>
<td>SC_TransactT1</td>
<td>This function Sends/recieves the ISO 7816-4 compaliant T = 1 commands to the card.</td>
</tr>
</tbody>
</table>
## Library API Macros

### Macros

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SC_MCP_LIB</strong></td>
<td>Smart Card Library</td>
</tr>
<tr>
<td>SC_ABORT_RESPONSE</td>
<td>PCB byte for Abort Response of T1 Protocol</td>
</tr>
<tr>
<td>SC_AUTHENTICATE</td>
<td>Authenticate Command code to the Smart Card</td>
</tr>
<tr>
<td>SC_BWI</td>
<td>DEFAULT Value of BWI Indicator used in calculation of BWT for T=1 protocol</td>
</tr>
<tr>
<td>SC_CHANGE_PIN</td>
<td>Change Pin Command code to the Smart Card</td>
</tr>
<tr>
<td>SC_CLEAR_CARD</td>
<td>Clear Card Command code to the Smart Card</td>
</tr>
<tr>
<td>SC_CRC_TYPE_EDC</td>
<td>Cyclic Redundancy Check(CRC) type is used for EDC in Epilogue Field</td>
</tr>
<tr>
<td>SC_CREDIT</td>
<td>Credit Command code to the Smart Card</td>
</tr>
<tr>
<td>SC_CWI</td>
<td>DEFAULT Value of CWI Indicator used in calculation of CWT for</td>
</tr>
<tr>
<td>Command Code</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>T=1 protocol</td>
<td></td>
</tr>
<tr>
<td>SC_DEBIT</td>
<td>Debit Command code to the Smart Card</td>
</tr>
<tr>
<td>SC_GET_RESPONSE</td>
<td>Get Response Command code to the Smart Card</td>
</tr>
<tr>
<td>SC_IFS_RESPONSE</td>
<td>PCB byte for IFS Response of T1 Protocol</td>
</tr>
<tr>
<td>SC_INQUIRE_ACCT</td>
<td>Inquire Account Command code to the Smart Card</td>
</tr>
<tr>
<td>SC_LRC_TYPE_EDC</td>
<td>Longitudinal Redundancy Check (LRC) type is used for EDC in Epilogue Field</td>
</tr>
<tr>
<td>SC_READ_RECORD</td>
<td>Read Record Command code to the Smart Card</td>
</tr>
<tr>
<td>SC_RESYNC_REQ</td>
<td>PCB byte for Resync Request of T1 Protocol</td>
</tr>
<tr>
<td>SC_REVOKE</td>
<td>Revoke Command code to the Smart Card</td>
</tr>
<tr>
<td>SC_SELECT_FILE</td>
<td>Select File Command code to the Smart Card</td>
</tr>
<tr>
<td>SC_START_SESSION</td>
<td>Start Session Command code to the Smart Card</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-------------------------------------------------------</td>
</tr>
<tr>
<td>SC_STATE_CARD_ACTIVE</td>
<td>Card is powered and ATR received</td>
</tr>
<tr>
<td>SC_STATE_CARD_INACTIVE</td>
<td>Card present but not powered</td>
</tr>
<tr>
<td>SC_STATE_CARD_NOT_PRESENT</td>
<td>No Card Detected</td>
</tr>
<tr>
<td>SC_SUBMIT_CODE</td>
<td>Submit Code Command to the Smart Card</td>
</tr>
<tr>
<td>SC_T0ProtocolType</td>
<td>Returns '1' if T=0 protocol is supported &amp; Returns 0 otherwise</td>
</tr>
<tr>
<td>SC_T1ProtocolType</td>
<td>Returns '1' if T=1 protocol is supported &amp; Returns 0 otherwise</td>
</tr>
<tr>
<td>SC_TA1Present</td>
<td>Returns '1' if TA1 present &amp; Returns 0 otherwise</td>
</tr>
<tr>
<td>SC_TA2Present</td>
<td>Returns '1' if TA2 present &amp; Returns 0 otherwise</td>
</tr>
<tr>
<td>SC_TB1Present</td>
<td>Returns '1' if TB1 present &amp; Returns 0 otherwise</td>
</tr>
<tr>
<td>SC_TB2Present</td>
<td>Returns '1' if TB2 present &amp; Returns 0 otherwise</td>
</tr>
<tr>
<td>SC_TC1Present</td>
<td>Returns '1' if TC1 present &amp; Returns 0 otherwise</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>SC_TC2Present</td>
<td>Returns '1' if TC2 present &amp; Returns 0 otherwise</td>
</tr>
<tr>
<td>SC_TD1Present</td>
<td>Returns '1' if TD1 present &amp; Returns 0 otherwise</td>
</tr>
<tr>
<td>SC_TD2Present</td>
<td>Returns '1' if TD2 present &amp; Returns 0 otherwise</td>
</tr>
<tr>
<td>SC_WAIT_TIME_EXT_RESPONSE</td>
<td>PCB byte for Wait Time Extension Response of T1 Protocol</td>
</tr>
<tr>
<td>SC_WI</td>
<td>DEFAULT Value of WI Indicator used in calculation of WWT for T=0 protocol</td>
</tr>
<tr>
<td>SC_WRITE_RECORD</td>
<td>Write Record Command code to the Smart Card</td>
</tr>
</tbody>
</table>

Library API

Microchip Smart Card Library 1.02.8 - [July 18, 2012]
Copyright © 2012 Microchip Technology, Inc. All rights reserved.

Contents | Index | Reference | Home
## Library API Types

### Types

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SC_APDU_COMMAND</td>
<td>SmartCard APDU Command 7816-4</td>
</tr>
<tr>
<td>SC_APDU_RESPONSE</td>
<td>SmartCard APDU Response structure 7816-4</td>
</tr>
<tr>
<td>SC_ERROR</td>
<td>Smart Card error types</td>
</tr>
<tr>
<td>SC_T1_PROLOGUE_FIELD</td>
<td>Prologue Field for T=1 Protocol</td>
</tr>
<tr>
<td>T1BLOCK_TYPE</td>
<td>Block types in T=1 protocol</td>
</tr>
</tbody>
</table>

---

**Library API**

Microchip Smart Card Library 1.02.8 - [July 18, 2012]
Copyright © 2012 Microchip Technology, Inc. All rights reserved.
## Library API Variables

### Variables

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scATR_HistoryBuffer</td>
<td>Historical bytes sent by Smart Card</td>
</tr>
<tr>
<td>scATR_HistoryLength</td>
<td>Number of Historical bytes present</td>
</tr>
<tr>
<td>scATRLength</td>
<td>length of ATR data sent by smart card</td>
</tr>
<tr>
<td>scCardATR</td>
<td>ATR data sent by smartcard.</td>
</tr>
<tr>
<td>scLastError</td>
<td>Smart Card Error type is stored in this variable</td>
</tr>
<tr>
<td>scPPSresponse</td>
<td>PPS Response Bytes</td>
</tr>
<tr>
<td>scPPSresponseLength</td>
<td>Length of PPS Response</td>
</tr>
<tr>
<td>scTA1</td>
<td>TA1 determines the clock-rate conversion factor F &amp; bit-rate-adjustment factor D</td>
</tr>
<tr>
<td>scTA2</td>
<td>TA2 determines whether the smart card will operate in specific mode or negotiable mode following the ATR</td>
</tr>
<tr>
<td>scTA3</td>
<td>TA3 conveys the Information Field Size Integer (IFSI) for the smart card.</td>
</tr>
<tr>
<td>scTB1</td>
<td>TB1 conveys information on the smart card's programming voltage requirements.</td>
</tr>
<tr>
<td></td>
<td>Description</td>
</tr>
<tr>
<td>---</td>
<td>-------------</td>
</tr>
<tr>
<td>scTB2</td>
<td>TB2 conveys PI2, which determines the value of programming voltage required by the smart card. The value of PI1 in TB1 is superceded when TB2 is present.</td>
</tr>
<tr>
<td>scTB3</td>
<td>TB3 indicates the value of the Character Waiting Time Integer (CWI) and Block Waiting Time Integer (BWI) used to compute the Character Waiting Time (CWT) and Block Waiting Time (BWT).</td>
</tr>
<tr>
<td>scTC1</td>
<td>TC1 determines the extra guard time to be added between consecutive characters sent to the smart card from the terminal.</td>
</tr>
<tr>
<td>scTC2</td>
<td>TC2 is specific to protocol type T=0. TC2 conveys work waiting-time integer (WI) that determines the maximum interval between the leading edge of the start bit of any character sent by the smart card and the leading edge of the start bit of the previous character sent either by the card or the reader.</td>
</tr>
<tr>
<td>scTC3</td>
<td>When TC3 is present, it indicates the type of block-error detection to be used. When TC3 is not present, the default longitudinal redundancy check (LRC) is used.</td>
</tr>
<tr>
<td>scTD1</td>
<td>TD1 indicates if any further interface bytes are to be transmitted, and if so, which protocol will be used.</td>
</tr>
<tr>
<td></td>
<td>scTD2</td>
</tr>
<tr>
<td>---</td>
<td>----------</td>
</tr>
<tr>
<td></td>
<td>scTD3</td>
</tr>
</tbody>
</table>