Smart card system requirements.

i) Purpose:

Enabling authentication and verification of card and card holder by a host.Enabling GUI at host machine to interact with the card holder/user for the required transactions, for example, financial transactions with a bank or credit card transactions.

ii) Inputs:

Received header and messages at IO port Port_IO from host through the antenna.

iii) Internal Signals, Events and Notifications Notifications:

On power up, radiation-powered charge-pump supply of the card activated and a signal to start the system boot program at resetTask.

Card start requestHeader message to task_ReadPort from resetTask.

Host authentication request requestStart message to task_ReadPort from resetTask to enable requests for Port_IO.

User PW verification message (notification) through Port _IO from host.

Card application close request request ApplClose message to Port _IO.

iv) Outputs:

Transmitted headers and messages at Port_IO through antenna Control panel:

No control panel is at the card. The control panel and GUIs activate at the host machine (for example, at ATM or credit card reader)

v) Functions of the system:

The card inserts at a host machine.

The radiations from the host activate a charge pump at the card.

The charge pump powers the SoC circuit consisting of card processor, memory, timer, interrupt handler and IO port, Port _IO.

On power up, system reset signals reset Task to start. The reset Task sends the messages request Header and request Start for waiting task task_ Read Port.

task_ Read Port sends requests for host identification and reads through the Port _IO the host-identification message and request for card identification.

task_ PW sends through Port_ IO the requested card identification after system receives the host identity through Port IO.

task_ Appl then runs required API. The requestApplClose message closes the application.

The card can now be withdrawn All transactions between cardholder/user now takes place through GUIs using at the host control panel (screen or touch screen or LCD display panel).

vi) Design metrics:

Power Source and Dissipation: Radiation powered contact less.

Code size: optimum. card system memory needs should not exceed 64 kB memory.

Limited use of data types; multidimensional arrays, long 64-bit integer and floating points and very limited use of the error handlers, exceptions, signals, serialization, debugging and profiling.

Microcontroller hardware: Generates distinct coded physical addresses for the program and data logical addresses. Protected once writable memory space.

Validity: System is embedded with expiry date, after which the card authorization through the hosts disables.

Extendibility: The system expiry date is extendable by transactions and authorization of master control unit (for example, bank server).

Performance: Less than 1s for transferring control from the card to host machine Process Deadlines: None.

User Interfaces: At host machine, graphic at LCD or touch screen display on LCD and commands for card holder (card user) transactions.

vii) Test and validation conditions:

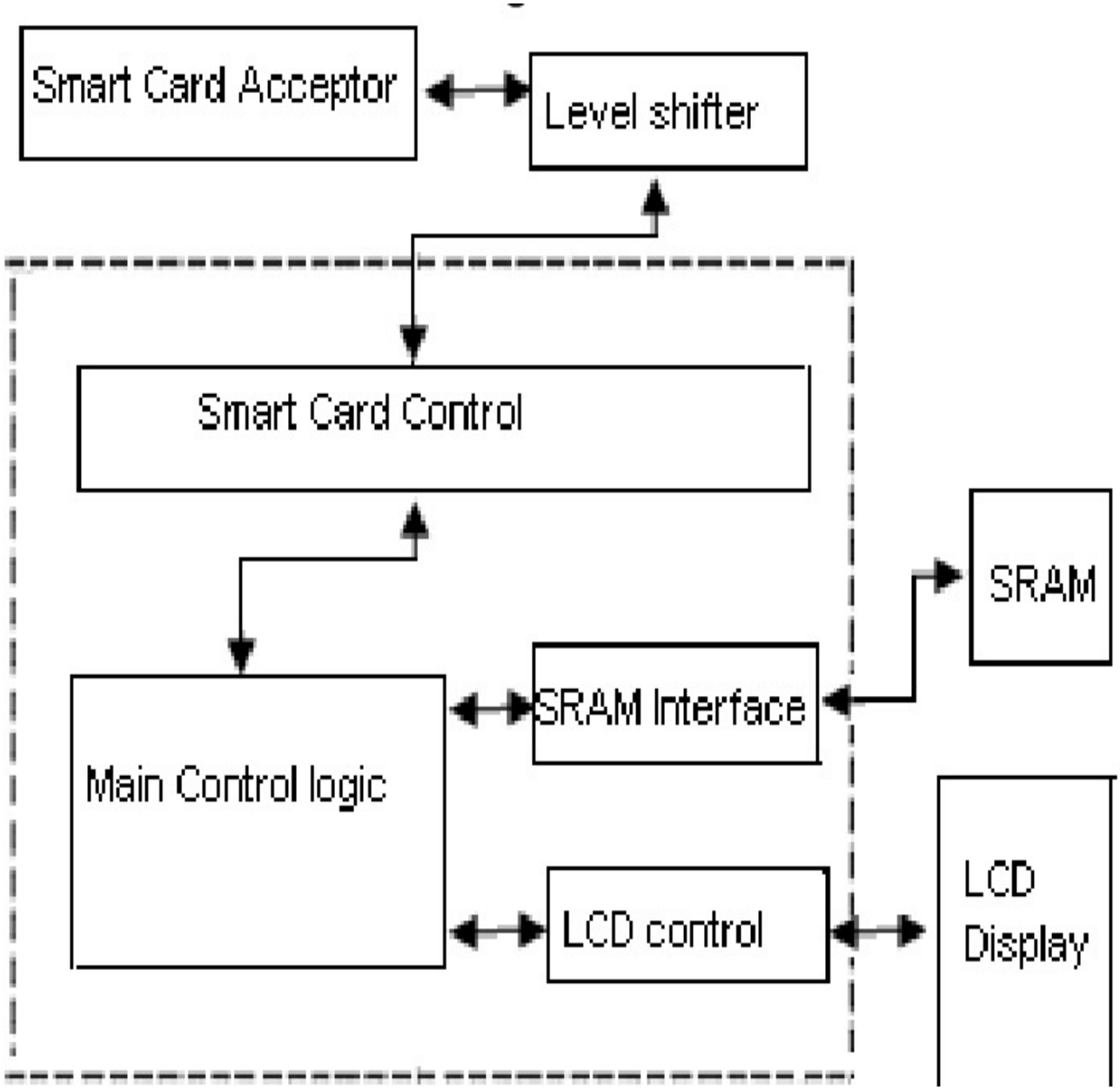
Tested on different host machine versions for fail proof card-host communication.

Classes:

Task_ Card Communication is an abstract class from which extended to class (es) derive to read port and authenticate.

The tasks (objects) are the instances of the classes Task_ Appl, Task_ Reset, Task_ Read Port and Task _PW.

ISR1_Port_IO, ISR2_Port_IO and ISR3_Port_IO are interfaces to the tasks



Smart card hardware:

A plastic card in ISO standard dimensions, 85.60 mm x 53.98 x 0.80 mm. It is an embedded SoC (System-OnChip). [ISO standards - ISO7816 (1 to 4) for host-machine contact based card and ISO14443 (Part A or B) for the contactless cards.]

Microcontroller MC68HC11D0 or PIC16C84 or a smart card processor Philips Smart XA or an ASIP Processor. Needs 8 kB+ internal RAM and 32 kB EPROM and 2/3 wire protected memory.

CPU special features, for example, a security lock.

CPU locks certain section of memory - protect 1 kB or more data from modification and access by any external source or instruction outside that memory.

Other way of protecting - CPU access through the physical addresses, which are different from logical address used in the program.

Standard ROM 8 kB for usual or 64 kB when using advanced cryptographic features.

Full or part of ROM bus activates take place after a security check only.

Chip-supply system using charge pump.

I/O system.

Software Architecture

Smart Card Software:

Needs cryptographic software, needs special features in its operating system over and above the MS DOS or UNIX system features.

Protected environment -OS stored in the protected part of ROM.

A restricted run-time environment.

OS, every method, class and run time library should be scalable.Optimum Code-size.

Limited use of data types; multidimensional arrays, long 64-bit integer and floating points and very limited use of the error handlers, exceptions, signals, serialization, debugging and profiling.

Three-layered file system for the data.

Master file to store all file headers (file status, access conditions and the file lock).

A header means file status, access conditions and the file lock.

Dedicated file – second file to hold a file grouping and headers of the immediate successor

Elementary file – third file to hold the file header and its file data.

Either a fixed length file management or a variable file length management with each file with a predefined offset.