**EasyLink solution outline design**

Version 1.00.01

Release Date: June 15, 2016

Copyright © 2000-2016 PAX Computer Technology (Shenzhen) Co., Ltd.

All rights reserved. No part of the contents of this document may be reproduced or transmitted in any form without the written permission of PAX Computer Technology (Shenzhen) Co., Ltd. The information contained in this document is subject to change without notice. Although PAX Computer Technology (Shenzhen) Co., Ltd. has attempted to ensure the accuracy of the contents of this document, this document may include errors or omissions. The examples and sample programs are for illustration only and may not be suited for your purpose. You should verify the applicability of any example or sample program before placing the software into productive use.

# Revision history

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | Comments |
| V1.00.00 | June 07, 2016 | Idina Zhang | Initial version |
| V1.00.01 | June 15, 2016 | Idina Zhang | Revise |
| V1.00.02 | July 1, 2016 | Idina Zhang | Modified interface for EMV/CLSS; |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table1 – revision history

Contects

[Revision history 3](#_Toc455157532)

[1. Introduction 6](#_Toc455157533)

[1.1 Purpose 6](#_Toc455157534)

[1.2 Abbreviation 6](#_Toc455157535)

[1.3 Reference 6](#_Toc455157536)

[2. EasyLink solution overview 7](#_Toc455157537)

[2.1 Background and goals to achieve 7](#_Toc455157538)

[2.2 Overview 7](#_Toc455157539)

[2.3 Supported model 7](#_Toc455157540)

[3. Overall design 8](#_Toc455157541)

[3.1 System design 8](#_Toc455157542)

[3.2 Message protocol layer 9](#_Toc455157543)

[3.2.1 Communication protocol 9](#_Toc455157544)

[3.2.2 Messaging protocol 9](#_Toc455157545)

[3.3 Business process layer 9](#_Toc455157546)

[3.3.1 UI module 9](#_Toc455157547)

[3.3.2 Card module 9](#_Toc455157548)

[3.3.3 Security module 9](#_Toc455157549)

[3.3.4 EMV module 10](#_Toc455157550)

[3.3.5 CLSS module 12](#_Toc455157551)

[3.4 Local management 13](#_Toc455157552)

[3.4.1 Parameter management module 13](#_Toc455157553)

[3.4.2 TMS Proxy module 13](#_Toc455157554)

[3.5 UI display 13](#_Toc455157555)

[3.6 Module design for Android/IOS/Windows SDK 13](#_Toc455157556)

[4. Interface design 14](#_Toc455157557)

[4.1 Interface design for POS side 14](#_Toc455157558)

[4.1.1 Interfaces for COMM module 14](#_Toc455157559)

[4.1.2 Interfaces for UI module 14](#_Toc455157566)

[4.1.3 Interfaces for Security module 14](#_Toc455157567)

[4.1.4 Interface for transaction flow 14](#_Toc455157568)

[4.1.5 Interfaces for parameter and data in terminal 15](#_Toc455157569)

[4.1.6 Interfaces for TMS Proxy 16](#_Toc455157570)

[4.2 Interface design for Android/IOS/Windows side 16](#_Toc455157571)

[5. Parameter configuration 17](#_Toc455157572)

[5.1 Parameter format statement 17](#_Toc455157574)

[5.2 Parameters in POS side 17](#_Toc455157575)

[5.2.1 EMV Parameter configuration 17](#_Toc455157576)

[5.2.2 CLSS parameter configuration 23](#_Toc455157577)

[5.2.3 Application parameters 30](#_Toc455157578)

[5.3 Parameters in Android/IOS/Windows side 30](#_Toc455157579)

# Introduction

## 1.1 Purpose

This document is outline design of EasyLink solution. It mainly describe the overall design which give reference in detailed design.

## 1.2 Abbreviation

|  |  |
| --- | --- |
| Name | Description |
| **POS** | Point of sale |
| **PDK** | PAX Platform Development Kit |
| **SDK** | Software development kit |
| **RKI** | Remote key injection |
| **KMS** | Key management system |
| **PAN** | Primary account number |
| **TM** | Terminal management |
| **TMS** | Terminal management systme |
| **AID** | Application identifier |
| **CAPK** | Certification authority public key |

Table2 - Abbreviation

## 1.3 Reference

PAX\_PDK\_API\_Programming\_Guide\_V1.00.00.doc

EUI详细设计说明(V1.1.01).docx

4.3book3\_Application.pdf

PAX EMV Kernel API Programming Guide.pdf

# EasyLink solution overview

## 2.1 Background and goals to achieve

The original MPOS solution provides a set of commands with slight granularity which make it difficult for application in upper device (like Android/IOS Windows device) to use. So the purpose of this specification for EasyLink solution is to design an optimization program structure with better extendibility and provide a set of commands with larger granularity to improve the interaction efficiency and allow quick and easy integration with customized applications in upper device.

## 2.2 Overview

PAX POS device works in conjunction with smart device (like Android/IOS/Window device).

The development in PAX POS side of this solution is based on PAX PDK API level which support cross-platform to compatible with PAX Monitor and Prolin OS.

Figure1 - EasyLink close view



## 2.3 Supported model

Device with PAX Monitor lite platform: D180S (may support D150 in the future)

Device with Monitor platform: S200（for future）

Device with PAX Prolin platform: D200, D220 (may support S300, PX5, and PX7 in the future)

# Overall design

## 3.1 System design

For EasyLink solution, PAX POS device works in conjunction with Android/IOS/Windows device, among which PAX POS device provides a set of commands for Android/IOS/Windows app. Android/IOS/Windows device send request packet to PAX POS terminal, then PAX POS terminal return the corresponding response back to the Android/IOS/Windows device.

Below figure demonstrate the data processing flow of EasyLink solution.

Figure 2 – data processing flow



Below is a close view in PAX POS terminal:

Figure 3 – close view in PAX POS terminal



## Message protocol layer

### Communication protocol

The communication protocol is also call COMM module. It is responsible transmit data between terminal and Android/IOS/Windows device. It is based on PDK. The PDK shield up the difference of each communication mode and provides a set of APIs to make it easy for application to use.

The PAX POS EasyLink application only need to communicate with the Android/IOS/Windows device, and the Android/IOS/Windows application is going to connect to the host. The EasyLink application provides interfaces for Android/IOS/Windows application to connect, drop connect, send and receive data.

Supported communication mode: UART, USB, BT, WIFI, LAN, etc.

Note: D180S may use lite version.

### Messaging protocol

The messaging protocol is responsible for command parsing and delivery to the business layer for further processing or display.

## Business process layer

### UI module

EasyLink UI module adopts existing EUI solution. For EUI, it uses XML to describe user interface layout, and draw text and pictures on the screen by getting the layout settings from XML file via UI functions.

PAX defines and provides several type of UI XML files, then user needs to download the XML files into terminal before application initialization.

The PAX POS EasyLink application provides a UI type command for Android/IOS/Windows APP to draw the texts and pictures they want on the screen.

Note: D180S may use lite version (including UI XML of lite version).

### Card module

PAX POS EasyLink application provides a card type command for Android/IOS/Windows APP to get card data, then the operations of opening, detecting, reading and closing for card module shall be automatically done in POS side.

### Security module

* Key injection

EasyLink solution shall support RKI and key injected by Master POS. RKI is integrated into TM, while key injected by Master POS is not, so terminal needs to provide interface for master POS to inject key.

* Data security

Terminal shall support TDES, AES and RSA encryption for sensitive data including PAN, track1 data, track2 data, Tag 5A data, and Tag 57 data.

Furthermore, EasyLink solution shall support DUKPT and TDES for PIN encryption.

* Transfer security

### EMV module

All the EMV procedures shall be in compliance with EMV specification.

The application in terminal of EasyLink shall provide several interfaces for Android/IOS/Windows application to fulfill the EMV transaction. The following diagram shows the processing flow of EMV contact transaction:

Figure 4 – EMV contact transaction flow of EasyLink



The EMV contact parameter of EasyLink which should not be modified by the merchant will be configured into EMV contact parameter file. The EMV contact parameter file and UIs should be downloaded into terminal through TMS. The EasyLink application shall check if all the necessary parameter files and UIs have been downloaded.

D180S: may use xml of lite version

### CLSS module

All the CLSS interfaces in EasyLink shall conform to EMV contactless specifications.

The application in terminal of EasyLink solution shall provide several interfaces for Android/IOS/Windows application to fulfill the contactless EMV transaction. The following diagram shows the processing procedure of EMV contactless transaction:

Figure 5 - EMV contactless transaction flow of EasyLink



The EMV contactless parameter of EasyLink which should not be modified by the merchant will be configured into EMV contactless parameter file. The EMV contactless parameter file and UIs should be downloaded into terminal through TMS. The EasyLink application shall check if all the necessary parameter files and UIs have been downloaded.

## Local management

### Parameter management module

Parameters which should not be modified by the merchant shall be configured in corresponding parameter files and downloaded into terminal via TMS in TM. This type of parameters shall be EMV, CLSS, and UI files.

Parameters which relate to merchant or application level like data encryption type shall be also configured into corresponding parameter files and downloaded into terminal via TMS, however, terminal shall provide interface to let merchant to check and edit.

Note: D180S may use lite version.

### TMS Proxy module

Device model with remote communication capability can download parameters, font, and application files via TMS in terminal.

For the device model without remote communication capability, like D180S can only communicate via BT and USB, there should be a TMS proxy in Android/IOS/Windows device to download parameters, font, and application files from TMS server, then push the files into terminal. So there should be an interface for TMS proxy in terminal. For TMS proxy in Android/IOS/Windows device, PAX has implemented the TMS proxy in Android and Windows platform at present, so there still needs to implement the TMS proxy in IOS platform according to present TMS protocol.

## UI display

The UI display provides screen for UI module to show some message or pictures.

## Module design for Android/IOS/Windows SDK

Since the terminal works in conjunction with the Android/IOS/Windows device for EasyLink solution, so the Android/IOS/Windows SDK shall provide interfaces which are in consistent with the modules in POS side.

# Interface design

## Interface design for POS side

### Interfaces for COMM module

* int Connect();

This API is used to create connection between the POS and Android/IOS/Windows device.

* int Disconnect();

This API is used to drop connection between the POS and Android/IOS/Windows device.

### Interfaces for UI module

* int ShowMsgBox();

The terminal provides this API to Android/IOS/Windows device to fulfill all the UI related processing with the UI XML files, like show some message, pictures, and etc.

### 

### Interface for transaction flow

* Int AuthorizeCard();

This API is used to detect card, then do the corresponding processing according to the card type, if MSR, then read MSR data, if EMV chip, then do the EMV processing (all the offline processing) according to the EMV specification, if EMV contactless card, then do the corresponding CLSS processing (all the offline processing).

In this API, terminal will do the following procedures according to the EMV contact specification and EMV contactless specification, and also read data from MSR:

1. DetectCard, including MSR, EMV chip, EMV contactless card;
2. If EMV chip card inserted;
3. Go EMV contact transaction flow;
4. App Selection
5. App Initialization
6. Get Processing Options
7. Check if need fallback, if yes, then return error, if no, then return EMV data
8. Process Restrictions
9. Process CVM
10. Offline Data Authentication
11. Terminal Risk Management
12. First GAC
13. If transaction approved or declined, remove card
14. If EMV contatctless card tapped;
15. Go EMV contactless transaction flow;
16. App Selection
17. Get Final Select Data
18. Paywave/Paypass/AE(or other CLSS kernel) processing
19. Restriction Processing
20. Offline Data Authentication
21. CVM Processing
22. Remove card
23. Outcome Check, if need tap card again, go back to step 1, if not, return the result back to Android/IOS/Windows device.
24. If MSR swiped;
25. Read MSR data;

* int CompleteOnlineTxn();

If online authentication is needed for EMV or CLSS transaction, then Android/IOS/Windows device shall call this API to complete EMV/CLSS traction after online processing, below are the procedures in this API:

1. if EMV chip card
2. External authentications if exists
3. Issuer script processing if exists
4. Second GAC if online authenticated
5. Remove card
6. If CLSS card
7. External Authentication
8. Issuer Script Process

### Interfaces for parameter and data in terminal

* int GetData();

The Android/IOS/Windows device use this API to get specific data from terminal, like value of EMV/CLSS TAGs for current transaction, transaction parameters, hardware information of terminal, software information of terminal, track1/track2/track3 of MSR, and so on;

**Note:** This API shall return the encrypted data for some sensitive data like PAN, track2 data.

* int SetData();

The Android/IOS/Windows device use this API to set value for EMV/CLSS TAGs, transaction parameters, terminal parameters, and so on;

### Interfaces for TMS Proxy

* int DownloadFile();

The Android/IOS/Windows device download all the files from TMS server, then push all the files into terminal. So terminal uses this API to download parameter files, including EMV parameter file, EMV contactless parameter file, UI XML files, font, application, application parameter files, etc.

## Interface design for Android/IOS/Windows side

Since the terminal works in conjunction with the Android/IOS/Windows device for EasyLink solution, so the Android/IOS/Windows SDK shall provide interfaces which are in consistent with the modules in POS side.

# Parameter configuration



## Parameter format statement

**Hex**: HEX characters (0~9, A~E).

**N**: Numeric characters (0~9).

**B**: 0 or 1.

**Char**: Any available ASCII characters.

**Hex 10**: Data length fixed with 10 characters. For Hex format data, It means 5 bytes of Hex data (refer to the example below).

**Hex…496**: Data length maximumly being 496 characters (258 bytes of Hex data).

Example:

RID (Hex 10):<RID>A000000005<RID> -- The RID is “\xA0\x00\x00\x00\x05”

ExpDate(N 6):<ExpDate>140710<ExpDate> -- The expiration date is “140710”

AppName (Char…16):<AppName>MCHIP<AppName> -- The maximum length of application name is 16 ASCII characters

## Parameters in POS side

### EMV Parameter configuration

All the EMV contact parameters shall be configured into XML file.

#### 5.2.1.1 CAPK configuration

| **No.** | **Field Name** | **Required** | **Attribute** | **Description** |
| --- | --- | --- | --- | --- |
| 1 | RID | M | Hex 10 | Application Service Provider ID |
| 2 | KeyID | M | Hex 2 | Key Index |
| 3 | HashInd | M | N 1 | HASH Flag |
| 4 | ArithInd | M | N 1 | RSA Flag |
| 5 | ModulLen | M | N…4 | Module Length |
| 6 | Module | M | Hex…496 | Module |
| 7 | ExponentLen | M | N1 | Exponent Length |
| 8 | Exponent | M | N…6 | Exponent |
| 9 | ExpDate | M | N 6 | Expiry Date (YYMMDD) |
| 10 | CheckSum | M | Hex 40 | Key Check Sum |

Note: the CAPK parameter shall be configured into XML file.

#### 5.2.1.2 Revoked CAPK

| **No.** | **Field Name** | **Required** | **Attribute** | **Description** |
| --- | --- | --- | --- | --- |
| 1 | RID | M | Hex 10 | Application Service Provider ID |
| 2 | KeyID | M | Hex 2 | Key Index |
| 3 | CertificateSN | M | N…6 | Issuer Certificate Serial No. |

Note: the revoked CAPK parameter shall be configured into XML file.

#### 5.2.1.3 AID configuration

| **No.** | **Field Name** | **Required** | **Attribute** | **Description** |
| --- | --- | --- | --- | --- |
| 1 | AppName | M | Char…16 | Local Application Name |
| 2 | ApplicationID | M | Hex…34 | Application Identifier |
| 3 | AIDLen | M | N…2 | Application Identifier Length |
| 4 | SelectFlag | M | N 1 | Select Flag  The valid value shall be:  0: Partial Match  1: Full Match |
| 5 | TargetPer | M | N…2 | Target Percentage |
| 6 | MaxTargetPer | M | N…2 | Maximum Target Percentage |
| 7 | FloorLimit | M | N…12 | Terminal Floor Limit |
| 8 | Threshold | M | N…12 | Threshold |
| 9 | TACDenial | M | Hex 10 | TAC Denial |
| 10 | TACOnline | M | Hex 10 | TAC Online |
| 11 | TACDefault | M | Hex 10 | TAC Default |
| 12 | TermDDOLLen | M | N…3 | Terminal Default DDOL Length |
| 13 | TermDDOL | M | Hex…256 | Terminal Default DDOL |
| 14 | TDOL | M | Hex…256 |  |
| 15 | AppVersion | M | Hex 4 | Application Version |
| 16 | TermRisk | M | Hex 2 | b1: Floor Limit Check Flag  b2: Random Selection  b3: Velocity Check  b4: Exception File  b5: Support TAC  b6~b8: Reserved |

Note: the AID parameter shall be configured into XML file.

#### 5.2.1.4 Terminal capability

| **No.** | **Field Name** | **Required** | **Attribute** | **Description** |
| --- | --- | --- | --- | --- |
| 1 | TermType | M | Hex 2 | Terminal Type. Below is the supported terminal type according to PX EMV L2 certificates:  22: attended, online with offline capability terminal  25: unattended, online with offline capability terminal |
| 2 | CardDataInputCapability | M | Hex 2 | Card Data Input Capability (2 characters, i.e. 1 byte HEX data). Each bit represents a flag of a specific capability describing the capability of the terminal to capture card data: bit set to 1 means that the capability is supported while 0 means not supported. Below is the representation of each bit :  (bit 8 indicates the most significant bit - i.e. the leftmost bit)  b1~b5: Reserved  b6: IC with contacts  b7: Magnetic stripe  b8: Manual key entry |
| 3 | CVMCapability | M | Hex 2 | CVM Capability (2 characters, i.e. 1 byte HEX data). Each bit represents a flag of a specific capability describing the capability of the terminal to perform CVM (Cardholder Verification Method): bit set to 1 means that the capability is supported while 0 means not supported. Below is the representation of each bit:  (bit 8 indicates the most significant bit - i.e. the leftmost bit)  b1~b3: Reserved  b4: No CVM  b5: Enciphered PIN for offline ICC verification  b6: Signature (paper)  b7: Enciphered PIN for online verification  b8: Plaintext PIN for offline ICC verification |
| 4 | SecurityCapability | M | Hex 2 | Security Capability (2 characters, i.e. 1 byte HEX data). Each bit represents a flag of a specific capability describing the security capability of the terminal: bit set to 1 means that the capability is supported while 0 means not supported. Below is the representation of each bit:  (bit 8 indicates the most significant bit - i.e. the leftmost bit)  b1~b3: Reserved  b4: CDA (Combined DDA/Application Cryptogram Generation)  b5: Reserved  b6: Card Capture (Always be 0 for POS terminal)  b7: DDA (Dynamic Data Authentication)  b8: SDA (Static Data Authentication) |
| 5 | AdditionalTermCapabilities | M | Hex 10 | Additional Terminal Capabilities (10 characters, i.e. 5 byte HEX data). Each bit (totally 40 bits) represents a flag of a specific additional terminal capability: bit set to 1 means that the capability is supported while 0 means not supported. Below is the representation of each bit:  (byte 1 indicates the leftmost byte)  (bit 8 indicates the most significant bit - i.e. the leftmost bit)  Byte 1 - Transaction Type Capability  b1: Administrative  b2: Payment  b3: Transfer  b4: Inquiry  b5: Cashback  b6: Services  b7: Goods  b8: Cash  Byte 2 - Transaction Type Capability  b1 ~ bit7: Reserved  bit 8: Cash Deposit  Byte 3 - Terminal Data Input Capability  b1 ~ b4: Reserved  b5: Function keys  b6: Command keys  b7:Alphabetic and special characters keys  b8: Numeric keys  Byte 4 - Terminal Data Output Capability  b1: Code table 9  b2: Code table 10  b3~b4: Reserved  b5: Display, cardholder  b6: Display, attendant  b7: Print, cardholder  b8: Print, attendant  Byte 5 - Terminal Data Output Capability  b1: Code table 1  b2: Code table 2  b3: Code table 3  b4: Code table 4  b5: Code table 5  b6: Code table 6  b7: Code table 7  b8: Code table 8  (Note: The code table number refers to the corresponding part of ISO/IEC 8859.) |
| 5 | Options | M | Hex 2 | b1: PSE Selection Support  b2: Card Holder Confirmation  b3: Bypass PIN Allowed  b4: Subsequent Bypass PIN Entry  b5: Get Data for PIN Try Counter  b6: Equipped PIN Pad (Always be 1)  b7: Amount Entered At Same Pad (Aways be 0)  (b6 & b7 are mutually exclusive)  b8: Reserved |
| 6 | ComProc | M | Hex 2 | b1: Forced Online  b2: Support Advices  b3: Support Issuer Referrals  b4: Support Card Referral  b5: Batch Data Capture  b6: Online Data Capture  b7: Default TDOL  b8: reserved  (b5& b6 are mutually exclusive) |

Note: the EMV chip card parameter shall be configured into XML file.

#### 5.2.1.5 Other common parameter configuration

| **No.** | **Field Name** | **Required** | **Attribute** | **Description** |
| --- | --- | --- | --- | --- |
| 1 | CountryCode | M | Hex 4 | Terminal Country Code |
| 2 | CurrencyCode | M | Hex 4 | Terminal Currency Code |
| 3 | RefCurcyCode | M | Hex 4 | Reference Currency Code |
| 4 | CurrencyExp | M | Hex 2 | Terminal Currency Exponent |
| 5 | RefCurrencyExp | M | Hex 2 | Reference Currency Exponent |
| 6 | MerchantCategoryCode | M | Hex 4 | Classifies the type of business being done by the merchant, represented in accordance with [ISO 8583:1993] for Card Acceptor Business Code. |
| 7 | MerchantId | M | ASC 15 | When concatenated with the Acquirer Identifier, uniquely identifies a given merchant. |
| 8 | TerminalID | M | N 8 | Designates the unique location of the Terminal. |
| 9 | MerchantName | M | ASC 128 | Indicates the name of the merchant. |
| 10 | MerchantLocalAddress | M | ASC 128 | Indicates the location of the merchant. |
| 11 | ConversionRatio | M | N.. 6 | the conversion quotients between transaction currency and reference currency (default : 1000)  (the exchange rate of transaction currency to reference currency \*1000) |

### CLSS parameter configuration

All the EMV contactless parameters shall be configured into XML file.

#### Paywave parameter configuration

##### Paywave AID configuration

| **No.** | **Field Name** | **Required** | **Attribute** | **Description** |
| --- | --- | --- | --- | --- |
| 1 | SelectFlag | M | N 1 | Select Flag  The value is as below:  0: Partial Match  1: Full Match |
| 2 | ApplicationID | M | Hex…34 | Application Identifier |
| 3 | AppName | M | Char…16 | Local Application Name |
| 4 | AppNameLen | M | N…2 | Local Application Name Length |
| 5 | AppVersion | M | Hex 4 | Application Version |
| 6 | Crypto17Flg | M | N…1 | MSD CVN17 support flag,   1. not support 2. support |
| 7 | ZeroAmountNoAllowed | M | N…1 | Amount, Authorized of Zero Check] flag,  0- [Amount, Authorized of Zero Check] activated, online required  1- [Amount, Authorized of Zero Check] activated, amount zero not allowed  2- [Amount, Authorized of Zero Check] deactivated |
| 8 | StatusCheckFlg | M | N…1 | status check support flag,  0-not support ，1- support |
| 9 | ReaderTTQ | M | Hex…8 | Indicates reader capabilities, requirements, and preferences to the card.  Byte 1  bit 8: 1 = MSD supported  bit 7: RFU (0)  bit 6: 1 = qVSDC supported  bit 5: 1 = EMV contact chip supported  bit 4: 1 = Offline-only reader  bit 3: 1 = Online PIN supported  bit 2: 1 = Signature supported  bit 1: 1 = Offline Data Authentication (ODA) for Online Authorizations supported.  Note: Readers compliant to this specification set TTQ byte 1 bit 1 to 0b  Byte 2  bit 8: 1 = Online cryptogram required  bit 7: 1 = CVM required  bit 6: 1 = (Contact Chip) Offline PIN supported  bits 5-1: RFU (00000)  Byte 3  bit 8: 1 = Issuer Update Processing supported  bit 7: 1 = Mobile functionality supported (Consumer Device CVM)  bits 6-1: RFU (000000)  Byte 4  RFU ('00') |
| 10 | Inter\_WareFloorlimitByTransactionType | M | Node | Sub Node, refer to “2.1.2.2.1 Sub Node Inter\_WareFloorlimitByTransactionType description” |

##### Sub Node “Inter\_WareFlmtByTransType” description

| **No.** | **Field Name** | **Required** | **Attribute** | **Description** |
| --- | --- | --- | --- | --- |
| 1 | ContactlessCVMLimit | M | N…12 | Contactless floor limit |
| 2 | ContactlessTxnLimit | M | N…12 | Contactless transaction floor limit |
| 3 | ContactlessFloorLimit | M | N…12 | Contactless floor limit |
| 4 | TermFloorLimit | M | N…12 | Terminal floor limit |
| 5 | TermFloorLimitSupported | M | N…1 | Terminal offline limit check flag:  0 - Deactivated,  1 - Active and exist,  2 - Active but not exist |
| 6 | ContactlessTxnLimitSupported | M | N…1 | Card reader contactless transaction limit check flag,  0-Deactivated, 1-Active and exist, 2-Active but not exist |
| 7 | CVMLimitSupported | M | N…1 | Card reader CVM limit check flag,  0-Deactivated, 1-Active and exist, 2-Active but not exist |
| 8 | ContactlessFloorLimitSupported | M | N…1 | Card reader contactless Offline limit check flag,  0-Deactivated, 1-Active and exist, 2-Active but not exist |
| 9 | TxnType | M | N2 | Transaction type same as Tag“0x9C” |

##### Program ID configuration (only for chip card)

| **No.** | **Field Name** | **Required** | **Attribute** | **Description** |
| --- | --- | --- | --- | --- |
| 1 | ProgramID | M | N…17 | Program ID |
| 2 | ContactlessCVMLimit | M | N…12 | Contactless CVM floor limit |
| 3 | ContactlessTxnLimit | M | N…12 | Contactless transaction floor limit |
| 4 | ContactlessFloorLimit | M | N…12 | Contactless floor limit |
| 5 | TermFloorLimitSupported | M | N…1 | Terminal offline limit check flag:  0 - Deactivated,  1 - Active and exist,  2 - Active but not exist |
| 6 | TermFloorLimitSupported | M | N…1 | Terminal Offline limit check flag  0-Deactivated, 1-Active and exist, 2-Active but not exist |
| 7 | ContactlessTxnLimitSupported | M | N…1 | Card reader contactless transaction limit check flag,  0-Deactivated, 1-Active and exist, 2-Active but not exist |
| 8 | CVMLimitSupported | M | N…1 | Card reader CVM limit check flag,  0-Deactivated, 1-Active and exist, 2-Active but not exist |
| 9 | ContactlessFloorLimitSupported | M | N…1 | Card reader contactless Offline limit check flag,  0-Deactivated, 1-Active and exist, 2-Active but not exist |
| 10 | Crypto17Supported | M | N…1 | MSD CVN17 support flag,  0 - not support  1 - support |
| 11 | ZeroAmountNoAllowed | M | N…1 | Amount, Authorized of Zero Check] flag,  0 - [Amount, Authorized of Zero Check] activated, online required  1 - [Amount, Authorized of Zero Check] activated, amount zero not allowed  2 - [Amount, Authorized of Zero Check] deactivated |
| 12 | StatusCheckSupported | M | N…1 | status check support flag,  0-not support  1- support |
| 13 | ReaderTTQ | M | Hex…8 | Indicates reader capabilities, requirements, and preferences to the card.  Byte 1  bit 8: 1 = MSD supported  bit 7: RFU (0)  bit 6: 1 = qVSDC supported  bit 5: 1 = EMV contact chip supported  bit 4: 1 = Offline-only reader  bit 3: 1 = Online PIN supported  bit 2: 1 = Signature supported  bit 1: 1 = Offline Data Authentication (ODA) for Online Authorizations supported.  Note: Readers compliant to this specification set TTQ byte 1 bit 1 to 0b  Byte 2  bit 8: 1 = Online cryptogram required  bit 7: 1 = CVM required  bit 6: 1 = (Contact Chip) Offline PIN supported  bits 5-1: RFU (00000)  Byte 3  bit 8: 1 = Issuer Update Processing supported  bit 7: 1 = Mobile functionality supported (Consumer Device CVM)  bits 6-1: RFU (000000)  Byte 4  RFU ('00') |

#### 5.2.2.2 Paypass Parameter configuration

##### Paypass AID configuration

| **No.** | **Field Name** | **Required** | **Attribute** | **Description** |
| --- | --- | --- | --- | --- |
| 1 | SelectFlag | M | N 1 | Select Flag  The value is as below:  0: Partial Match  1: Full Match |
| 2 | ApplicationID | M | Hex…34 | Application Identifier |
| 3 | AppName | M | Char…16 | Local Application Name |
| 4 | AppNameLen | M | N…2 | Local Application Name Length |
| 5 | AppVersion | M | Hex 4 | Application Version |
| 6 | MagneticAppVersionNum | M | Hex 4 | 9F6D Mag-stripe Application Version Number (Reader) |
| 7 | TACDenial | M | Hex 10 | TAG DF8121 : TAC Denial  (only for chip card) |
| 8 | TACOnline | M | Hex 10 | TAG DF8122:TAC Online  (only for chip card) |
| 9 | TACDefault | M | Hex 10 | TAG DF8120: TAC Default  (only for chip card) |
| 10 | TermRisk | M | Hex 16 | (Tag 9F1D)  B7:Plaintext PIN for ICC verification (Contactless)  B6:Enciphered PIN for online verification (Contactless)  B5:Signature (paper) (Contactless)  B4:Enciphered PIN for offline verification (Contactless)  b3:No CVM required (Contactless)  b2:On device cardholder verification (Contactless)  (only for chip card) |
| 11 | ContactlessCVMLimit | M | N…12 | Tag DF8126 Contactless CVM Floor Limit |
| 12 | ContactlessTxnLimit\_NoOnDevice | M | N…12 | Tag DF8124 Indicates the transaction amount above which the transaction is not allowed, when on-device cardholder verification is not supported. |
| 13 | ContactlessFloorLimit | M | N…12 | Tag DF8123 Contactless Floor Limit |

##### Paypass configuraton

| **No.** | **Field Name** | **Required** | **Attribute** | **Description** |
| --- | --- | --- | --- | --- |
| 1 | KernelConfiguration | M | Hex 2 | DF811B Indicates the Kernel configuration options  b8:Only EMV mode transactions supported  b7: Only mag-stripe mode transactions supported  b6:On device cardholder verification supported  b5-1 RFU  decide by AID |
| 2 | TornLeftTime | M | N…4 | DF811C Maximum time, in seconds, that a record can remain in the Torn Transaction Log.  (only for chip card) |
| 3 | MaxTornNumber | M | Hex…2 | DF811D Max Number of Torn Transaction Log Records  (only for chip card) |
| 4 | MagnaticCVM | M | Hex…2 | DF811E Mag-stripe CVM Capability-CVM required(PayPass)  b8-5  0000: NO CVM  0001: OBTAIN SIGNATURE  0010: ONLINE PIN  1111: N/A  b4-1  NFU |
| 5 | MagneticNoCVM | M | Hex…2 | DF812C Mag-stripe CVM Capability-No CVM required  b8-5  0000: NO CVM  0001: OBTAIN SIGNATURE  0010: ONLINE PIN  1111: N/A  b4-1  NFU |
| 6 | MobileSupport | M | Hex..2 | Tag 9F7E Mobile Support Indicator  b8-3： RFU  b2 :Offline PIN Required  b1：Mobile supported |
| 7 | CardDataInput | M | Hex…2 | Tag DF8117 Card Data Input Capability |
| 8 | CVMCapabilit\_CVMRequired | M | Hex…2 | Tag DF8118 CVM Capability - CVM Required |
| 9 | CVMCapabilit\_NoCVMRequired | M | Hex 10 | Tag DF8119 CVM Capability - No CVM Required(0x08) |
| 10 | TermType | M | Hex…2 | Tag 9F35 Terminal Type |
| 11 | AccountType | M | Hex 2 | Tag 5F57 Account type |
| 12 | AdditionalTermCapability | M | Hex10 | Tag 9F40 Additional Terminal Capbilities |
| 13 | KernelID | M | Hex 1 | Tag DF810C Kernel ID |
| 14 | SecurityCapability | M | Hex 1 | Tag DF811F security capability  b1~b3: reserved  b4: CDA  b5: reserved  b6: Card Capture (Always be 0)  b7: DDA  b8: SDA |

#### 5.2.2.3 CAPK configuration

Same to the EMV contact CAPK configuration [5.2.1.1 CAPK configuration](#_5.2.1.1_CAPK_configuration);

#### 5.2.2.4 Revoked CAPK

Same to the EMV contact Revoked CAPK [5.2.1.2 Revoked CAPK](#_5.2.1.2_Revoked_CAPK);

### Application parameters

The below are the application parameters of the application in terminal. And the Android/IOS/Windows application can configured the value of these parameters by transmitting the value to the terminal.

| **No.** | **Parameter** | **Description** | **Default** |
| --- | --- | --- | --- |
| 1 | SleepModeTimeout |  |  |
| 2 | DataEncryptionType | Indicates which kind of encryption is going to adopt, valid value shall be as below:  0 – no encryption  1 – TDES  2 – RSA |  |
| 3 | DataEncryptionKeyIdx | Indicates which key is going to be used for data encryption |  |
| 4 | LanguageType |  |  |
| 5 | FallbackAllowFlag | 0 – Fallback not allowed  1 – Fallback allowed |  |
| 6 | PANMaskStartPos | Indicates the first few number of clear digits for the masked PAN, valid value: 0 to 6. |  |

## Parameters in Android/IOS/Windows side

The below transaction parameters showed below shall be transmitted to from Android/IOS/windows device to terminal via related commands.

| **No.** | **Field Name** | **Required** | **Attribute** | **Description** |
| --- | --- | --- | --- | --- |
| 1 | TxnAmount | M | N…12 | Transaction Amount |
| 2 | TransactionType | M | Hex 2 | Transaction type |
| 3 | TxnCurcyCode | M | Hex 4 | Transaction Currency Code |
| 4 | TxnCurcyExp | M | Hex 2 | Transaction Currency Exponent |
| 5 | TxnDate | M | N 6 | Transaction Date (YYMMDD) |
| 6 | TxnTime | M | N 6 | Transaction Time (hhmmss) |
| 7 | F55TagList | M | Hex…512 | The Tag List Which Should Be Involved In Field55(ICC Data) for Online Authentication |

Table 3 – Parameters controlled by Android/IOS/Windows device