IDTECH®

Value through Innovation

ViVOpay VP3350
Integration Manual

80178501-001 Rev. J
22 April 2024

ID TECH
10721 Walker Street, Cypress, CA 90630
Voice: (714) 761-6368 Fax: (714) 761-8880
idtechproducts.com

ID TECH
10721 Walker Street
Cypress, CA 90630 USA

Warranty Disclaimer
The services and hardware are provided "as is" and "as available" and the use of the services and hardware are at its own risk. ID TECH does not make, and hereby disclaims, any and all other express or implied warranties, including, but not limited to, warranties of merchantability, fitness for a particular purpose, title, and any warranties arising from a course of dealing, usage, or trade practice. ID TECH does not warrant that the services or hardware will be uninterrupted, error-free, or completely secure.

FCC warning statement
This device complies with Part 15 of the FCC Rules. Operation is subject to the following two conditions: (1) this device may not cause harmful interference, and (2) this device must accept any interference received, including interference that may cause undesired operation.

The user manual for an intentional or unintentional radiator shall caution the user that changes or modifications not expressly approved by the party responsible for compliance could void the user's authority to operate the equipment.

Note: The grantee is not responsible for any changes or modifications not expressly approved by the party responsible for compliance. Such modifications could void the user's authority to operate the equipment.

Note: This equipment has been tested and found to comply with the limits for a Class B digital device, pursuant to part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation. This equipment generates uses and can radiate radio frequency energy and, if not installed and used in accordance with the instructions, may cause harmful interference to radio communications. However, there is no guarantee that interference will not occur in a particular installation. If this equipment does cause harmful interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures:

- Increase the distance between the equipment and the receiver.
- Connect the equipment into an outlet on a circuit different from that to which the receiver is connected.
- Consult the dealer or an experienced radio/TV technician for help.

This device complies with FCC RF radiation exposure limits set forth for an uncontrolled environment.
The antenna(s) used for this transmitter must not be co-located or operating in conjunction with any other antenna or transmitter and must be installed to provide a separation distance of at least 20cm from all persons.

Innovation, Science and Economic Development Canada (ISED) statement
This device contains licence-exempt transmitter(s)/receiver(s) that comply with Innovation, Science and Economic Development Canada's licence-exempt RSS(s).
Operation is subject to the following two conditions:

1. This device may not cause interference, and
2. This device must accept any interference, including interference that may cause undesired operation of the device.

L'émetteur/récepteur exempt de licence contenu dans le présent appareil est conforme aux CNR d'Innovation, Sciences et Développement économique Canada applicables aux appareils radio exempts de licence. L'exploitation est autorisée aux deux conditions suivantes :

1. l'appareil ne doit pas produire de brouillage, et
2. l'utilisateur de l'appareil doit accepter tout brouillage radioelectrique subi, meme si le brouillage est susceptible d'en compromettre le fonctionnement.

Cautions and Warnings

	Caution: Danger of Explosion if battery is incorrectly charged. Use only standard USB 5V power source. Device contains a lithium battery. Approved temperature range for storage: -20°C to +60°C. Disposal: Contact your local recycling center.
	Warning: Avoid close proximity to radio transmitters, which may reduce the capabilities of the reader.

Caution: Exposure to Radio Frequency Radiation
To comply with the Canadian RF exposure compliance requirements, this device and its antenna must not be co-located or operating in conjunction with any other antenna or transmitter.

Attention: exposition au rayonnement radiofréquence
Pour se conformer aux exigences de conformité RF canadienne l'exposition, cet appareil et son antenne ne doivent pas être co-localisés ou fonctionnant en conjonction avec une autre antenne ou transmetteur.

Internal Rechargeable Battery Warning[1]
Danger: Do not attempt to replace the internal rechargeable lithium-ion battery. Replacing the original battery with an incompatible type may result in an increased risk of personal injury or property damage due to explosion, excessive heat, or other risks. Do not attempt to disassemble or modify the battery pack. Attempting to do so can cause a harmful explosion or battery fluid leakage.

When disposing of the battery, comply with all relevant local ordinances or regulations. Do not dispose of the battery pack in municipal waste. Dispose used batteries according to the instructions.
The battery pack contains a small amount of harmful substances.

To avoid injury:

- Keep the battery pack away from open flames or other heat sources.
- Do not expose the batter pack to water, rain, or other corrosive liquids.
- Do not leave the battery in an environment with extremely low air pressure. It may result in an explosion or the leakage of flammable liquid or gas from the battery.

To extend battery life, we recommend charging the battery to at least 30% to 50% capacity each time and recharging it every three months to prevent over discharge.

[1] Note that the SRED VP3350 is a PCI SRED certified device; any attempt to replace the internal Lithium-Ion battery will result in a device tamper, rending the unit inoperable.

Revision History

Date	Rev	Changes	By
12/08/2022	A	Initial release.	CB
02/06/2023	B	Updated LED and Sound State Indicators table and footnote.	CB
02/16/2023	C	Updated mounting guidelines and diagrams.	CB
03/22/2023	D	Tag 9F33: Updated byte 2 bit 5 (feature not supported).	CB
05/24/2023	E	Added specifications section; moved environmental/storage specs and power consumption there. Added cable-related power requirements to VP3350 Connectors and Interfaces section. Added note to iOS Connectivity section about iOS SDK integration for apps.	CB
03/07/2024	G	Updated Major VP3350 Features, Contactless NFC Features and Brand Certifications, and Other Agency Approvals and Compliances. Updated Specifications section. Updated VP3350 Connectors and Interfaces. Added power management commands. Added charging capability commands.	CB
04/19/2024	H	Added Configuring a VP3350 with the ID TECH Universal SDK section.	CB
04/22/2024	J	Added Appendix A: VP3350 Demo Setup for Mobile Devices	CB

Table of Contents

1. Introduction

The ViVOpay VP3350 is ID TECH's latest PCI 6.X SRED-certified and non-SRED mobile reader. The VP3350 supports Magstripe, EMV contact, and contactless transactions using either USB-C, Lighting, and/or Bluetooth connections.

The VP3350's compact form factor and Bluetooth interface make it ideal for mobile applications where smart card reading is required as well as incorporated into a case or stand associated with a countertop POS terminal. As a mobile reader, VP3350 works in conjunction with Android as well as iOS phones and tablets via BLE. The VP3350 is designed to be compatible with a wide range of third-party payment applications, and the ID TECH Universal SDK (described further below) is available for developing applications that communicate with VP3350.

Contact your acquirer, gateway, or POS partner for instructions on setting up and pairing the VP3350 to a compatible payment application and associated host device.

1.1. Major VP3350 Features

- Communicates via Bluetooth BLE (Bluetooth Low Energy)
- Bluetooth communication supports iOS and Android platforms
- Communicates to iPhone host devices via Lightning connector
- Communicates to iPad or Android host devices via USB-C male connector
- USB-C 2.0 female connector supports battery charging or communication with Android or Windows host systems
- Rechargeable battery, no external power necessary
- Supports pass-through charging
- ICC: EMV Level 1 Contact certified and integrated ID TECH's EMV L2 Gen 3 Common Contact kernel
- Front-facing contactless transaction support via Near Field Communication (NFC)
- Magstripe reading
- LED status indicator
- Audio feedback
- Field-upgradable firmware
- Low power consumption when reader is in standby mode
- Compact and ergonomic design to integrate with a variety of mobile devices
- Supports Triple DES, AES128, and TransArmor with DUKPT key management
- Supports 20 contact and 8 contactless AIDs, for a total of 28 AIDs

Refer to the table below for the differences among the VP3350 models.

1.2. Contactless NFC Features and Brand Certifications

- ISO 14443 Type A&B
- ISO 18092 (P2P)
- MasterCard® MChip (Formerly PayPass)
- Discover® DPAS
- American Express® ExpressPay
- Interac
- Interac Transit
- MIFARE: native support

- Apple Pay
- Apple VAS
- Samsung Pay NFC
- Android Pay
- Google Pay / Softcard Smart Tap 2.1
- JCB
- UPI and UPI Transit

1.3. Other Agency Approvals and Compliances

- CE (EN55032/EN55035, Class- B)
- FCC (Part 15, Class-B)
- RoHS (DIRECTIVE 2015/863/EU)
- REACH
- EMV Contact L1&L2
- EMV Contactless L1 and majority of Contactless 2
- TQM
- PCI PTS 6.X SRED

2. Specifications

The sections below describe VP3350 environmental and power specifications.

2.1. Operation and Storage: Environmental Limits

Item	Specification	Note
Operating Temperature	0 °C to 55 °C or 32°F to 131°F	Non-condensing. Product operation temperature is limited to this range due to constraints of the Li-Battery specification.
Storage Temperature	-20 °C to 60 °C or -4°F to 140°F	Non-condensing. Product storage temperature is limited to this range due to constraints of the Li-Battery specification.
Operating Humidity	Up to 95%	Non-condensing.
Storage Humidity	Up to 95%	Non-condensing.

2.2. Battery Life for Lightning and Bluetooth Models

The VP3350 Lighting and Bluetooth models come equipped with battery that can support transactions as below:

- Minimum 800 MSR transactions per battery charge (with 30 second interval)
- Minimum 500 contact transactions per battery charge (with 30 second interval)
- Minimum 400 contactless transactions per battery charge (with 30 second interval)
- Minimum 200 contactless + 200 contact + 100 MSR transactions per battery charge (with 30 second interval)

Maximum power draw is 500mA for VP3350 power consumption and internal battery charging.

2.3. 24-Hour Device Reboot

Per PCI Requirements, this device reboots every 24 hours. Please contact your device integrator if you need to check the reboot time for your unit.

3. VP3350 Connectors and Interfaces

The VP3350 is designed to work Windows, Android, and iOS via a physical USB-C, Lightning, and Bluetooth communication.

- All VP3350 models come equipped with USB-C female connector as the power source and communicate with Windows, mobile phones and tablets.
- VP3350 Model-B equipped with Lightning connector communicates with popular mobile iOS devices.
- VP3350 Model-C equipped with Bluetooth connection capability.
- VP3350 Model-F equipped with USB-C male connector allows for pass-through charging to the host device and communicate with Android or iOS mobile devices.

Part Number	IDMR-xLR93x	IDMR-xBT93x	IDMR-xUF93x	From IDMR-xUR93-63 From IDMR-xUR93D-66[2]
Picture				
Model (Device Tree)	B	C	D	F
USB-Female	○	○	○	○
USB-Male				○ (Rotatable)
Lighting-Male	○ (Rotatable)			
BLE 5.0		○		
Power Button		○		
Battery 400mAh	○	○		
Power Passthrough	○			○
MFi Certification	○			Q3 2024
Female Application	Android/Windows	Android/Windows	Android/Windows	Android/Windows
Male application	iOS			iOS/Android

When communicating over USB, the VP3350's default emulation mode is USB HID; the reader can also emulate a USB HID-KB interface[3].

3.1. Pass-Through Charging Power Requirements

VP3350 readers require the following for pass-through charging:

- Mobile phones: 5V/1A
- Tablets: 5V/1.5A

3.2. Cabling Power Requirements

The following power requirements apply to all cabling permutations (for example, USB-C to USB-C vs. USB-A to USB-C).

- When using a USB-C to USB-C cable with power input, the VP3350 provides a maximum of 1.5A through the VP3350 male connector to a phone or tablet.

[2] Note: Only model F readers with revisions 63 and 66 and up, respectively, have the features listed on this table.

[3] In development.

    - The Lightning version (IDMR-xLR93x) maximum is 1A.
    - The USB-C version (IDMR-xUR93x) maximum is 1.5A.
- When using a USB-A to USB-C cable with power input, the VP3350 provides a maximum of 0.5A through the VP3350 male connector to a phone or tablet. Note that the tablet will not show a charging symbol because it needs at least 1A for the charging symbol to appear.

4. Bluetooth Pairing Instructions

In addition to the following instructions, see the section on iOS Connectivity for more information on using an iPad or iPhone in conjunction with the VP3350.

1. Enable the Bluetooth device search function on the host device (smart phone or tablet).
2. Make sure the VP3350 is charged or connected to a power source via USB.
    a. When connected to a power source, the VP3350 automatically activates Bluetooth.
3. When not connected to a power source, press the VP3350's power button to manually activate Bluetooth.
4. Find a Bluetooth device named **IDTECH-VP3350-XXXXX** on the host smart phone or tablet and select **Pair**.
5. Enter the password for pairing. The default password is **123456**.
6. Follow the payment transaction instructions provided by a compatible payment application maker to complete a transaction.

The VP3350 will remain connected via Bluetooth to the host device indefinitely when powered via the USB cable. If the unit is operating on battery power, it will go to sleep after 20 seconds of idling to extend battery life. To perform a transaction again, press the power button to re-establish the Bluetooth connection.

When connecting to an iOS device, please install a compatible payment application and follow the instructions provided by that application's maker. See iOS Connectivity: BLE and VP3350 below for more information.

4.1. Battery Charging Instructions

VP3350 Lighting and Bluetooth readers are powered by a lithium-ion polymer battery and are delivered in a partially-charged state. Be sure to fully charge the VP3350 before using it for the first time. Allow two to three hours for the initial charge.

Use a standard USB to USB-C cable to charge the unit. An LED battery indicator displays the current battery charging status[4].

Warning: When using a "fast charger" with a VP3350 reader, only use a USB-C to USB-C cable.

4.2. Tamper and Failed Self-Check Indicators

The VP3350 displays the following indicators when it has been tampered or has any of the other following internal issues, such as an expired certificate, missing key, or similar fault discovered during a self-check.

Indicator	Tampered Status	Other Issue Status
Front Four LEDs	All LEDs blink red	All LEDs blink red
Buzzer	Alarm tone	Alarm tone

[4] Use the **Get Battery Level (F0-02)** command to retrieve the reader's battery level.

4.3. VP3350 LED and Sound State Indicators

The VP3350 uses the following LEDs and sounds to indicate various statuses, including power management, Bluetooth, transactions, and security.

Device State	LED1	LED2	LED3	LED4	Sound
Battery Charging Full[5]	● on	● on	● on	● on	None
Low Battery (Power under 25%)[3]	● on	○ off	○ off	○ off	None
Bluetooth Connected	● on	● on	● on	● on	Unit beeps once
Bluetooth Connection Failed	● on	● on	● on	● on	Unit beeps twice
Bluetooth Working	● blink	○ off	● blink	○ off	
Unit Ready	● blink	○ off	○ off	○ off	None
Transaction Started	● on	○ off	○ off	○ off	None
Transaction Successful	● blink once	● blink once	● blink once	● blink once	Unit beeps once
Transaction Failed	○ off	○ off	○ off	○ off	Unit beeps twice
Unit Tampered	● blink	● blink	● blink	● blink	Unit beeps
Device Deactivated	● blink	● blink	● blink	● blink	Unit beeps
Power Off/Low-Power Consumption Mode	○ off	○ off	○ off	○ off	None

[5] Use the **Get Battery Level (F0-02)** command to retrieve battery level. When charging, the reader displays three green LEDs to indicate 75% charge, two green LEDs to indicate 50% charge, and one red LED to indicate a charge under 25%.

4.4. iOS Connectivity: BLE and VP3350

Note: Applications for iOS must integrate ID TECH's iOS SDK. The .ZIP file for the SDK, available on the ID TECH Knowledge Base, includes the *Apple iOS SDK Guide* and information on connecting ID TECH devices via BLE.

The VP3350 uses Bluetooth 5.0, also known as Bluetooth BLE (Bluetooth Low Energy). Unlike previous versions of Bluetooth, BLE does not require users to first pair their devices through the Bluetooth Settings in Apple iOS. If a payment application provider has enabled BLE scanning in their application, Apple iOS scans and locates all BLE devices in range to automatically connect with the VP3350.

It is critical to note that if the VP3350 is paired via the iOS Settings page, it will display as a connected device but not function with a payment app.

Unlike other operating systems that can detect or specify a BLE device by its MAC address, Apple does NOT allow users to specify a BLE device by MAC address for security reasons. Instead, after a device is selected by its "friendly" name (see the next paragraph), the Apple iOS calculates a unique identifier to allow that device to make further connections directly.

The VP3350 has a default friendly name of **IDTECH-VP3350-XXXXX**[6]. This is the default name the ID TECH Universal SDK uses to connect to the first VP3350 it encounters if no other friendly name is set in the SDK, or when the iOS-generated device identifier is not provided. See links given near the end of this document for information about the Universal SDK.

Note: The Universal SDK is primarily of interest to developers. If an application provider or POS software partner has already provided software to use with the VP3350, you do not need to obtain the SDK.

[6] Note that the last five digits (here denoted as XXXXX) are the same as the last five digits of the reader's serial number.

5. ID TECH Universal SDK

By virtue of its EMV L2 kernel, VP3350 is designed to be compatible with a wide range of third-party payment applications. ID TECH offers a Universal SDK (available for iOS, Android, or Windows) to enable rapid application development using VP3350 as the target device. The languages supported include Objective C (on iOS), Java (on Android), and C# (on Windows). The Universal SDK includes rich, powerful libraries that make sending commands to the VP3350 comparatively easy while greatly facilitating debugging and event handling.

To obtain the Universal SDK free of charge, visit ID TECH's VP3350 product page and select the version of the SDK that applies to your desired host platform (Android, iOS, Linux, MacOS, or Windows).

Normally, development of applications that take advantage of VP3350's capabilities can be done in a high-level language like C# or Java (using convenience objects and data structures defined in the Universal SDK), obviating the need to send byte commands directly. Nevertheless, it is possible to communicate with the device at a low level if necessary. For a command reference for VP3350, contact your ID TECH representative to request the *NEO 2 Interface Developers Guide* (IDG), P/N 80139403-002. This document includes not only low level (firmware) commands but error codes, response codes, and information on various default settings.

5.1. Updating VP3350 Firmware

ID TECH strongly recommends updating firmware via the Universal SDK. Contact ID TECH support for your reader's firmware, then consult the Universal SDK Guide for your desired platform (included in the firmware update ZIP file) for integration details.

5.2. Using the Correct Firmware Files

Due to the multiple models of VP3350 readers, make sure to select the correct firmware update files. Contact your ID TECH support representative to receive a .ZIP archive with the necessary firmware for your reader.

5.3. Configuring a VP3350 with the ID TECH Universal SDK

The ID TECH Universal SDK exposes a variety of methods for obtaining low-level access to features of the kernel like AIDs and CAPKs, allowing users to override or update various features as needed.

Note: ID TECH's Universal SDK provides source code for two apps. One of the apps represents the project created using the Tutorial in the documentation and can be found here. The other demo app is more elaborate and used for general device testing and configuration. In the sections below, references to the "demo app" apply to the latter project, not the Tutorial project. The source code for the demo app can be found here, and the compiled executable of the demo app can be found here.

ID TECH readers can support at least 23 major kernel configurations (**1C** through **23C**). Refer to this document for more information about available configurations.

Many terminal capability settings are considered "minor" settings that can be tailored to fit a scenario. Some are considered "major" settings that can't be changed without producing a runtime error. For more information on Terminal Settings and kernel configurations (and the settings that can be changed without producing an error), consult Which Terminal Settings Am I Allowed to Change? on the ID TECH Knowledge Base.

5.3.1 Setting AIDs

The card reader needs to know which application identifiers (AIDs) you intend to support. Configure this using code similar to the following example:

```
RETURN_CODE rt;
      byte[] name = Common.getByteArray("a0000000031010");
      byte[] aid =
Common.getByteArray("9f01065649534130305f5701005f2a0208409f0902009
65f3601029f1b0400003a98df25039f3704df28039f0802dfee150101df1305000
0000000df14050000000000df15050000000000df180100df170400002710df190
100");

// Tell the reader the name and value of the AID you wish to
support: rt =
IDT_NEO2.SharedController.emv_setApplicationData(name, aid);

      if (rt == RETURN_CODE.RETURN_CODE_DO_SUCCESS)
            {
                  tbOutput.AppendText("Default AID Successful\r\n");
            }
      else
            {
                  tbOutput.AppenText("Default AID failed Error Code:
                  " +
                        "0x" + String.Format("{0:X}", (ushort)rt) +
                        ":" + IDTechSDK.errorCode.getErrorString(rt)
                        + "\r\n");
            }
```

As with Terminal Configuration, setting the default AIDs is something you will likely do only one time, not on a per-transaction basis.

5.3.2 Setting CAPKs

You will also want to set the reader's supported Certificate Authority Public Key values (each card brand has one or more CAPKs for doing offline data authentication). The example below shows sample code for setting a CAPK:

```
      byte[] capk =
Common.getByteArray("a000000003500101b769775668cacb5d22a647d1d9931
41edab7237b000100018000d11197590057b84196c2f4d11a8f3c05408f422a35d
702f90106ea5b019bb28ae607aa9cdebcd0d81a38d48c7ebb0062d287369ec0c42
124246ac30d80cd602ab7238d51084ded4698162c59d25eac1e66255b4db235252
6ef0982c3b8ad3d1cce85b01db5788e75e09f44be7361366def9d1e1317b05e5d0
ff5290f88a0db47");

      RETURN_CODE rt = IDT_NEO2.SharedController.emv_setCAPK(capk);
```

This again is something probably configured only one time, not on a per-transaction basis.

Note: If EMV transactions are failing because of inability to perform ODA/CDA (as indicated in the first byte of TVR data, Tag 95), make sure your CAPKs are up to date.

5.4. Performing Transactions

The section below walks through the steps required for performing transactions.

5.4.1 Start Transaction

The Start Transaction command, issued via the method **emv_startTransaction()**, does the following:

When a card is already seated:

- Initiates a **Power On** sequence with the ICC reader
- If an ICC cannot be detected (no **ATR:Answer To Reset**), a fallback condition occurs and the reader requests a swipe.
- If the reader receives an ATR, it establishes timeout for the receipt of an **Authenticate Transaction** command and transmits **Application Data** parameters to the card, and the EMV transaction starts.

When a card is not seated:

- The MSR Swipe and the Contactless antenna is enabled, along with monitoring for a card to be inserted.

When a Swipe is executed first:

- If a non-ICC card is swiped, the swipe data is returned and the transaction is over.
- If an ICC card is swiped, the swipe is rejected and the kernel prompts to use the ICC (insert the card).

When executing **emv_startTransaction()**, TLV data should, at a minimum, contain the following tags:

- **9F02:** Amount
- **9F03:** Other amount
- **9C:** Transaction type

In addition, you can optionally include a DFEE1A Tag to request which Tags to include in the final response callback. The DFEE1A Tag is a proprietary ID TECH Tag that wraps other Tags. For

example, to request that the output from the Authenticate Transaction stage includes Tags 9F36, 9F37, and 95, specify **DFEE1A059F369F3795** as part of the TLV byte array (the **05** after **DFEE1A** is the total length of the included Tags 9F36, 9F37, and 95). Note that this will override the default set of default Tags returned (see the discussion under Obtaining Extra Tags, further below).

Another optional Tag is DFEF1F, which causes the transaction to enter the the Authenticate Transaction phase automatically (without a call to **emv_authenticateTransaction()**) after finishing Start Transaction. To use this Tag, supply two bytes of data: a first byte (with **1** to signal auto-authenticate or **0** to signal no auto- authenticate) and a second byte that, if set to a value of **1**, will force the transaction to go online (the default is **0**). The complete TLV might look something like **DFEF1F020100**, where **02** is the length.

After issuing the Start Transaction command, the reader interacts with the card to determine the correct AID to use and carry out other low-level EMV operations. It calls the application back a minimum of two times (one time with **ACK** and one time with **TransactionData**), but there are typically additional status callbacks. Inspect the **DeviceState** object (the second argument to the callback) to determine if the reader is responding with device data, an EMV callback, or transaction data (the sample demo app that comes with the Universal SDK shows code for this).

Check the callback type when the reader returns an **EMVCallback**, and if it is **EMV_CALLBACK_TYPE_LCD**, have the terminal display the appropriate message on the device LCD or POS screen (the sample demo app contains code showing how to do this). If the transaction requires a customer action before proceeding, the payment app should prompt the customer as necessary (using the terminal UI) and collect any needed info. See the demo app's **processEMVCallback()** method for an example of how to handle **EMVCallback** messages.

When the reader responds with a **DeviceState** of **TransactionData** and an EMV result of **EMV_RESULT_CODE_AUTHENTICATE_TRANSACTION**, the transaction can go to the next step. At this stage, the reader responds only to three commands: **Authenticate EMV L2 Transaction**, **Cancel EMV L2 Transaction**, or **Retrieve EMV L2 Transaction Data**; these three commands correspond to the SDK methods **emv_authenticateTransaction()**, **emv_cancelTransaction()**,and **emv_retrieveTransactionResult()**. The reader will block, at this point, while your app decides which method to call, and eventually time out if no action is taken. Use this opportunity to inspect the available transaction data to determine whether to invoke any extra logic that might be appropriate to the transaction. For example, this might be when the app decides that the customer is eligible for a loyalty discount. If no special logic applies, simply proceed to the next step: Authenticate Transaction.

5.4.2 Authenticate Transaction

In this stage of the transaction, the card reader's EMV kernel attempts to apply cardholder verification methods, terminal risk management logic, and any other logic required under EMV processing rules, to determine whether the terminal needs to go online for authorization.

This stage allows the temporary interruption of program flow so an integration to (optionally) modify any existing Tags. For example, a common use of this stage is to determine (via business logic) if the cardholder is entitled to a discount, and if so, provide the discount by changing the previously submitted amount (Tag 9F02) to a lesser amount. Integrations can pass any Tags that need values changed passed as a TLV stream in this method's parameter list.

Often, this step of pausing the EMV transaction to evaluate collected card data is not necessary. An integrator can tell the reader to continue automatically (with no pausing) by passing Tag DFEF1F with a value of **0100** (or **0101** if force-online is requested) when executing the **emv_startTransaction()** method. This can be done on a per-transaction basis. But also note the SDK has an "**autoAuthenticate**" option **ON** by default. If **autoAuthenticate** is set to **ON** before the EMV transaction begins, the authenticate transaction step always occurs without pausing. The main class has a method to set this boolean: **emv_autoAuthenticate(bool authenticate)**.

The app should monitor the **DeviceState** via the second argument of the **MessageCallBack** method (the method that was registered with the device at the time the app was loaded). Direct inspection of the **DeviceState** shows if the callback is being called with device data, with a state of **EMVCallback** (requiring a change of UI message, or perhaps interaction with the user), or with transaction data.

The final callback has a **DeviceState** (second argument) of **TransactionData** and come with a standard EMV result code. Consult the most recent Universal SDK documentation for the latest list.

If the result code at the end of the Authenticate Transaction stage is **EMV_RESULT_CODE_GO_ONLINE**, the terminal app should go online to obtain final authorization. Note that going online is the responsibility of the payment app, not the reader. The EMV kernel has no knowledge of online endpoints, protocols, or APIs.

At the conclusion of Authenticate Transaction, the application should consider the transaction finished if the cryptogram received from the card was of type **TC** (approved) or **AAC** (declined). Otherwise, if the cryptogram was **ARQC**, the application should go online for authorization, then call **emv_completeTransaction()**.

Note: To determine what kind of cryptogram was received in Tag 9F26, inspect the top nibble of the CID in **9F27**. A hex value of **00** indicates **AAC**; **40** indicates **TC**; **80** indicates **ARQC**.

5.4.3 Complete Transaction

After going online for authorization, you must call **emv_completeTransaction()**. You must call this method even if you were unable to go online but still wish to proceed with the transaction (for

example, if the network was down or the gateway failed to respond).

You must provide the **emv_completeTransaction()** method with the parameters that tell it if you were able to successfully reach the approver. These parameters should be provided in **Tag 8A**. Typical values here are **0x3030** for approval, **0x3032** for referral, **0x3035** for decline, and **0x5A33** for "unable to go online." Issuer Authentication Data (Tag 91) and Issuer Scripts (Tags 71 or 72) are not always present, but if they were returned by the online approver, they should be passed to this method. Consult the SDK documentation for the method signature and calling conventions; also consult the sample demo app code for an actual example of how to use this method.

When the transaction is complete, the reader will invoke your **MessageCallBack** method with a **DeviceState** of **TransactionData**. The sample demo app code shows how to pull TLV (Tag) data out of the **IDTTransactionData** object provided in the fourth argument to the callback. Typically, this Tag data includes TVR and TSI data in Tags 95 and 9B, respectively.

Tags from the reader may contain masked and encrypted data. ID TECH encrypts some (but not all) Tag data, using criteria described in *ID TECH Encrypted Data Output*. The SDK has methods that parse a TLV stream as necessary when the card data is returned, so manual parsing of the raw TLV data should not be required.

5.4.4 TLV Data

EMV transactions produce TLV data. At any stage in a transaction, you can use the **emv_unencryptedTags** field of your **IDTTransactionData** object (which has an instance name of **cardData** in the SDK sample code) to obtain unencrypted Tags, or the **emv_encryptedTags** field to obtain encrypted Tags. These fields point to **byte[]** arrays that you then need to process further to obtain actual TLVs. For example, a routine like the following converts the byte arrays to a Dictionary from which you can obtain text versions of the TLVs:

```
private string tlvToValues(byte[] tlv) {

      string text = "";
      Dictionary<string, string> dict =
            Common.processTLVUnencrypted(tlv);
      foreach (KeyValuePair<string, string> kvp in dict)
            text += kvp.Key + ": " + kvp.Value + "\r\n";
      return text;
}
```

5.4.5 Default Tags

Each stage of the transaction produces specific TLV data. The default TLVs obtained at each stage vary somewhat by product, but the transaction typically returns the following:

Start Transaction:

- 4F
- 50
- 57
- 5A
- 5F20
- 5F24
- 5F25
- 5F2D
- 5F34
- 84
- 9F20
- DFEE12
- DFEE23

Authenticate Transaction:

- 95
- 9B
- 9F02
- 9F03
- 9F10
- 9F13
- 9F26
- 9F27
- 9F34
- 9F36
- 9F37
- 9F4D
- 9F4F

Complete Transaction

- 95
- 99
- 9B
- 9F02
- 9F03
- 9F10
- 9F13
- 9F26
- 9F27
- 9F34

- 9F36
- 9F37
- 9F4D
- 9F4F
- 9F5B

Note that almost all of these are EMVCo-defined standard Tags. ID TECH proprietary Tags (typically three bytes in length, starting with DFEE or DFEF) sometimes also appear. For example, DFEE12 contains the transaction's KSN (Key Serial Number).

For a complete listing of ID TECH proprietary Tags and their meanings, see the *ID TECH TLV Tag Reference Guide*.

5.4.6 Encrypted Tags

Some TLVs contain encrypted data if your device has been key-injected and has encryption turned on. Generally speaking, any Tags containing track data (56, 57, 9F6B), PAN data (5A), or any other data considered sensitive are encrypted according to the rules described in *ID TECH Encrypted Data Output*.

By default, the following Tags (if present in your data) will be encrypted, assuming you are using a device that has been key-injected, with encryption enabled:

- 5A
- 56
- 57
- 9F1F
- 9F20
- 9F6B
- FFEE13
- FFEE14
- DF812A
- DF812B
- DF31
- DF32

5.4.7 Obtaining Extra Tags

Including a **DFEE1A** Tag in your original call to **emv_startTransaction()** notifies the card reader to produce extra Tags (see Start Transaction above). To obtain extra Tags, first include a **DFEE1A** TLV in your call to **emv_startTransaction(), emv_authenticateTransaction()**, or **emv_completeTransaction()** (or any combination of the three). Then, after each phase has finished, call **emv_retrieveTransactionResult()** to retrieve the requested TLVs.

Note: You must call **retrieveTransactionResult()** within **15 seconds** after the transaction is over. For security reasons, this data does not persist in memory for an extended period of time after the transaction completes.

Important: When specifing additional Tags in **emv_startTransaction()**, the request only returns the specified Tags. The request for additonal Tags overrides the default set of EMV Tags.

Also note that Tags **57** and **5A** are available after **emv_startTransaction()** by default but are not included by default after **emv_authenticateTransaction()**, or **emv_completeTransaction()**. As a rule, collect the desired TLVs immediately after each transaction phase.

5.5. Contactless EMV Transactions

Most ID TECH readers, including the VP3350, have contactless EMV capability, allowing transactions to be initiated via RFID. The following section provides a very brief overview on performing contactless transactions with the Universal SDK.

5.5.1 Contactless Configuration

As with contact EMV, contactless EMV requires that you first configure the reader with terminal settings, CAPKs, and AIDs before doing any transactions. Contactless EMV configurations do not share settings with contact EMV configuration items; instead, integrators must set up contactless configurations separately. Use the Universal SDK's **ctls_setApplicationData()** and **ctls_setCAPK()** methods to load each AID and CAPK the payment app needs to support. Note that this setup is only required one time; the settings persist across device reboots.

Terminal settings are more complicated in contactless EMV than in contact EMV; contactless EMV requires multiple EMV kernels—one kernel per card brand with different terminal settings for different card brands. In contactless readers, groups of TLVs specify terminal settings Predefined Groups exist (with Group numbers, such as 00, 80, 90, A0, B0, and C0), containing configurable TLVs that allow a set of behaviors to be mapped to a particular card or set of cards.

ID TECH has predefined a number of Groups populated or set by default with the **Set Configuration Defaults (04-09)** command. In the Universal SDK, run the **Set Configuration Defaults (04-09)** command using the **device_sendVivoCommandP2()** method, with **04** as the command argument and **09** as the subcommand argument. After running this command, all Groups are initialized for all card brands with common defaults.

Integrators may need to override certain defaults. For example, integrators outside of the U.S. might need to set the currency code (Tag 5F2A) to something other than **0840**. In the Universal SDK, run the **Set Configuration Group (04-03)** command. To override terminal settings using an API command, provide the TLVs to override along with a specified Group number to the **ctls_setConfigurationGroup((byte[] tlv)** method. The first TLV in the byte array should be the Group number (specified using Tag DFEE2D in NEO 2 and NEO 3); then at least one TLV should follow, containing configuration information. For example, in a NEO 3 device, you could set Group 90 to have a Currency Code of **0978** (for Euros) by supplying the following TLV data (as a byte array):

```
DF EE 2D 01 90 9F 1A 02 08 40 5F 2A 02 09 78 5F 36 01 02 9F
35 01 15 9F 66 04 30 80 C0 00 DF EE 34 06 00 00 00 01 00 00
DF 81 26 06 00 00 00 00 60 00 9F 1B 04 00 00 1F 40 DF 5B 01
```

```
05 9F 33 03 E0 08 40 DF EC 2E 01 61 9F 40 05 80 00 00 00 00
9F 02 06 00 00 00 00 05 00
```

Note that the first Tag is **DFEE2D**, which has **90** as the data value (meaning all of the downstream TLVs should be applied to Group 90). Even though the intention is to only change Tag 5F2A to **0978** (highlighted in yellow), we supply all the TLVs for this group to make sure no values are dropped and no values get accidentally picked up from Group 0 (any TLVs that don't exist in your Group but that do exist in the default Group 00 will get picked up from Group 00 unless you explicitly override the default value explicitly).

Important: When editing Group 00 (the default group, which cannot be deleted), you can replace individual Tags without affecting other Tags. However, for all other Groups, the **ctls_setConfigurationGroup(byte[] tlv)** method overwrites all data and installs only the TLVs that were passed in as a byte array. The best practice to follow, even if you are only modifying one TLV, is to read all TLVs from the Group in question using **ctls_getConfigurationGroup(int group, ref byte[] tlv)** and then modify the desired Tags, then replace all TLVs. Also remember that any Tags you fail to supply to a custom group, if they already exist in Group 00 will inherit values from Group 00 by default.

5.5.2 Starting a Contactless Transaction

Contactless transactions start in any NEO 3 device with an **Activate Transaction Command (02-40)**, or the SDK method **device_startTransaction (double amount, double amtOther, int exponent, int type, int timeout, byte[] Tags, bool forceOnline)**. You can use the same callback routine that you defined for contact EMV (see Performing Transactions above). When your callback is called, you can inspect the **IDTechSDK.IDTTransactionData** instance (in the final argument of the callback) to determine what kind of transaction occurred, and many other types of info (see SDK documentation included with the ID TECH SDK for an exhaustive list of properties).

When issuing **device_startTransaction()**, you can tell the device to accept contactless presentations only, or you can tell it to accept all three types of presentations (MSR, contact, and contactless). To specify the types of desired presentations, use Tag **DFEF37** (as shown below); specify fallback support using **DFEF3C**.

DFEF37	01	Define the type of interface to activate with (02-40). Interface Select: **Bit 0:** MSR **Bit 1:** Contactless **Bit 2:** Contact	DF EF 37 01 07 07 = 0000 0111 This activates transactions for all 3 interfaces.

DFEF3C	03	Fallback support and Timeout value for waiting for the next command (mainly to support EMV workflow) **Byte 1:** Fallback support **Byte 2~3:** Timeout for next command (Unit: Sec)	DF EF 3C 03 01 00 60 Fallback supported and the timeout is set to 60 seconds before the transaction times out.

5.5.3 Example raw command (NEO 3 firmware instructions):

```
5669564F7465636832000240000221E9C01009F02060000000001009F0306000000
000000DFEF370107DFEF3C0301006018D1

5669564F746563683200026300004BB4
```

If the command parse is successful and the ICC transaction starts, the response contains the first command and the status code is 0x63.

To send this command via the SDK, use the **device_sendVivoCommandP2()** method with a command of **02**, a subcommand of **40**, and Tag data (as byte array) of **9C01009F02060000000001009F03060000000000000DFEF370107DFEF3C03010060**.

Note: Unlike contact EMV, a contactless transaction runs straight through to completion, without interruption, in one step. The response contains one cryptogram, in Tag 9F26, at the end, rather than two cryptograms (corresponding to the two Gen AC events of contact EMV). A contactless transaction occurs in one swift step lasting about 500 milliseconds. There is no **device_completeTransaction()** method because completion is automatic. If the transaction occurs without error, all the necessary TLVs (including clearing-record TLVs) are in the returned data.

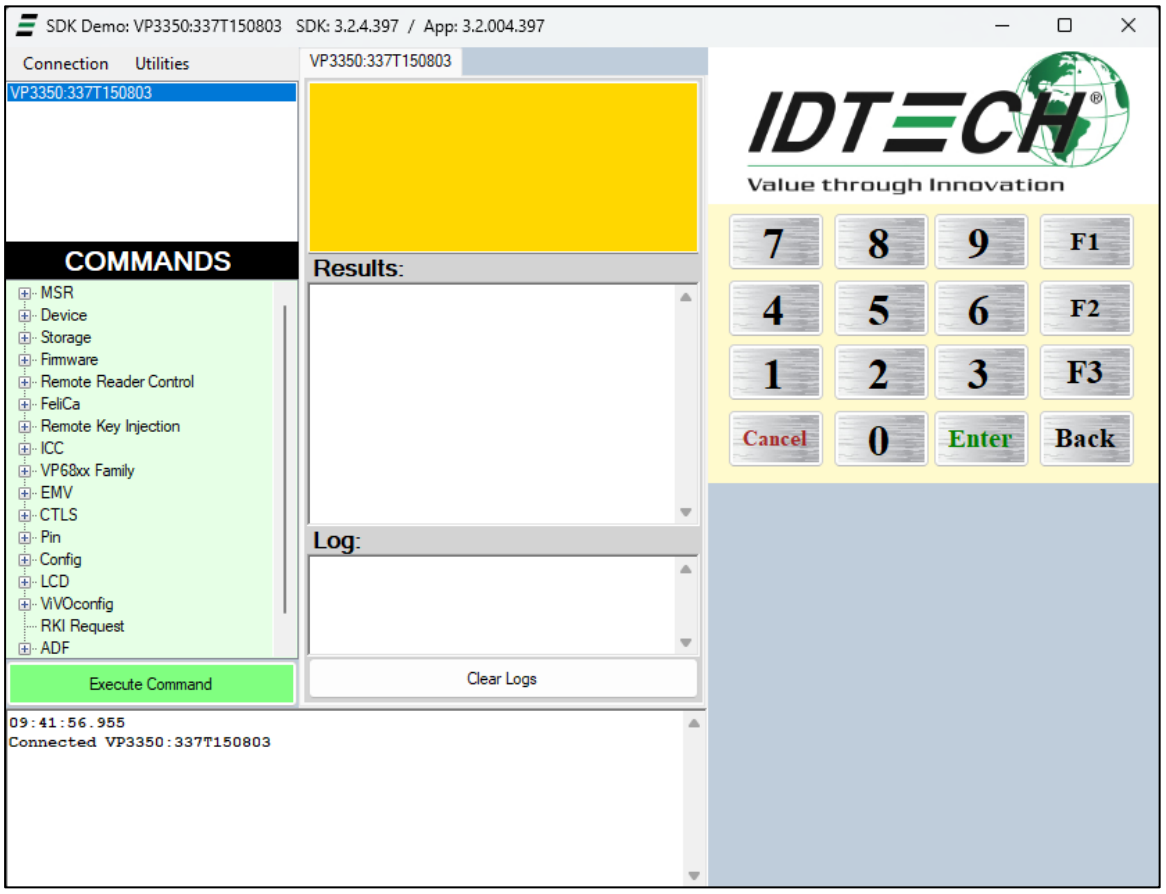
6. Universal SDK Demo App

The Universal SDK comes with a rich, fully featured demo app that allows users to run the VP3350 in USB mode. Visit the Universal Library for Visual Studio to download the Universal SDK Demo app as a standalone executable, separate from the SDK; the full SDK is not required to use the demo. Be sure to check out the Universal Demo QuickStart Guide for more detailed instructions about using the Universal SDK Demo app.

6.1. Using the Demo Application

Follow the steps below to run the Universal SDK Demo app on Windows:

1. Plug the VP3350 into the host device with a standard USB cable.
2. Open the **SDK Demo** application and allow a few seconds for the main window to appear (see illustration below).



The Universal SDK Demo app displays VP3350's available commands in a command tree, as shown above. Single-click on a command to populate the lower-right panel of the window with optional settings relevant to the command (for example, "Amount" and "Start EMV Additional Tags" above). In some cases, text fields appear, allowing users to enter custom values.

To execute a command, double-click it in the command tree or use the **Execute Command** button. The command executes in real time and a data trace appears automatically in the center panels. Use the **Clear Logs** button to clear both panels.

7. VP3350 Low-Level Commands

The following are commonly-used commands for the VP3350. For a full list of available commands, contact your ID TECH representative for the *NEO 2 Interface Developer's Guide* (available under NDA).

7.1. Activate Transaction Command (02-40)

The **Activate Transaction** command begins a contactless EMV or contactless MagStripe Card transaction.

Note: While an **Activate** command is in progress, readers only accept a **Cancel** or **Stop** command. Do not send other commands until **Activate Transaction** has completed, because the reader will interpret these as a **Cancel Transaction** command.

To control the behavior, use the DFEF37 and DFEF3C tags:

Tag	Length	Description	Example
DF EF 37	01	Define the type of interface to be activated with 02-40. Interface selection: • Bit 0: MSR • Bit 1: Contactless • Bit 2: Contact	DF EF 37 01 07 07 = 0000 0111 This activates transaction for all three interfaces.
DF EF 3C	03	Fallback support and Timeout value for waiting for the next command (mainly to support EMV workflow). Byte 1: Fallback support • 0x00: not support fallback • 0x01: support fallback Byte 2~3: Timeout for next command (Unit: Sec) (Hex format) • 00 0A = 10s • 01 00 = 256s	DF EF 3C 03 01 00 60 Fallback is supported, and the timeout is set to 96 seconds before the transaction times out.

Example:

```
5669564f7465636832000240002 21e9c01009f02060000000001009f030600000000
0000dfef370107dfef3c03010060 18d1
```

Command Frame

Byte 0-9	Byte 10	Byte 11	Byte 12	Byte 13	Byte 14 … Byte 14+n-1	Byte 14+n	Byte 15+n
Header Tag & Protocol Version	Command	Sub-Command	Data Length (MSB)	Data Length (LSB)	Data	CRC (LSB)	CRC (MSB)
ViVOtech2\0	02h	40h			See Data Format below		

Activate Transaction Command Frame Data Format

Data Item	Length (bytes)	Description
Timeout	1	Time in seconds that the reader waits for a card to be presented before timing out and returning an Error response. The reader will continue to poll for this amount of time if no card is found. Note that if a card is found, the transaction may not complete within the timeout period. This field must be present in the **Activate** command. Format: Binary
TLV Data	varies	See Activate Command TLVs in the *NEO 2 IDG*.

Response Frame

Byte 0-9	Byte 10	Byte 11	Byte 12	Byte 13	Byte 14 … Byte 14+n-1	Byte 14+n	Byte 15+n
Header Tag & Protocol Version	Command	Status Code	Data Length (MSB)	Data Length (LSB)	Data	CRC (MSB)	CRC (LSB)
ViVOtech2\0	02h	See Status Code Table			See Response Frame Data Format		

Note: The information above omits many command details, particularly TLV information. See the *NEO 2 Interface Developer's Guide* for the full **Activate Transaction (02-40)** description.

7.2. Set CA Public Key (D0-03)

The **Set CA Public Key** command adds a new key to the reader.

Command Frame

Byte 0-9	Byte 10	Byte 11	Byte 12	Byte 13	Byte 14 – 18	Byte 19	Bytes 19-n	Byte n+1	Byte n+2
Header Tag & Protocol Version	Cmd	Sub Cmd	Length (MSB)	Length (LSB)	RID (5 bytes)	Key Index (1 byte)	Key Data	CRC (LSB)	CRC (MSB)
ViVOtech2\0	D0h	03h	varies	varies	varies	varies	See below	Varies	Varies

Key Data is as follows: (all binary)

Byte	Name	Length (bytes)	Description
0	Hash Algorithm	1	The only algorithm supported is SHA-1. The value is set to 01h
1	Public Key Algorithm	1	The encryption algorithm in which this key is used. Currently support only one type: RSA. The value is set to 01h
3-22	Checksum/Hash	20	Checksum which is calculated using SHA-1 over the following fields: **RID** & **KeyIndex** & **Modulus** & **Exponent** where the exponent is either one byte or 3 bytes (although we store it in a 4-byte field)

Byte	Name	Length (bytes)	Description
23-26	Public Key Exponent	4	The real length of the exponent is either one byte or 3 bytes. It can have two values: 3, or 65537.
27-28	Modulus Length	2	Indicates the length of the next field, MSB followed by LSB.
29-n	Modulus	Variable	This is the modulus field of the public key. Its length is specified in the field above.

Response Frame

Byte 0-9	Byte 10	Byte 11	Byte 12	Byte 13	Byte 14	Byte 15
Header Tag & Protocol Version	Cmd	status	Length (MSB)	Length (LSB)	CRC (LSB)	CRC (MSB)
ViVOtech2\0	D0h	See Key Manager status codes	00h	00h	Calculated	Calculated

7.3. Get Processor Type (09-02)

The **Get Processor Type** command returns a processor type TLV.

Command Frame

Byte 0-9	Byte 10	Byte 11	Byte 12	Byte 13	Byte 14	Byte 15
Header Tag & Protocol	Command	Sub-Command	Data Length (MSB)	Data Length (LSB)	CRC (LSB)	CRC (MSB)
ViVOtech2\0	09h	02	00h	00h		

Response Frame

Byte 0-9	Byte 10	Byte 11	Byte12	Byte 13	Byte 14 … Byte 13+n	Byte 14+n	Byte 15+n
Header Tag & Protocol	Command	Status Code	Data Length (MSB)	Data Length (LSB)	Data	CRC (MSB)	CRC (LSB)
ViVOtech2\0	09h	See Status Code Table			See below		

The **Get Processor Type** sub-command returns a TLV string as follows:

- **Tag:** 0xDF61
- **Length:** 0x02
- **Value:** a field representing the processor type.

The following types of processors may be identified in the **Value** field:

Processor Type (hex values)	Description
45 00	ARM7/ LPC21xx
4D 00	ARM Cortex-M4/ K21 Family
4E 00	ARM Cortex-M4/ K81 Family

The following example shows the command and response.

Command: Get Processor Type: `56 69 56 4F 74 65 63 68 32 00 09 02 00 00 F0 F9`

Response: `56 69 56 4F 74 65 63 68 32 00 09 00 00 05 DF 61 02 4D 00 AC 4D`

7.4. Get Main Firmware Version (09-03)

The **Get Main Firmware Version** command returns main firmware version TLV.

Command Frame

Byte 0-9	Byte 10	Byte 11	Byte 12	Byte 13	Byte 14	Byte 15
Header Tag & Protocol	Command	Sub-Command	Data Length (MSB)	Data Length (LSB)	CRC (LSB)	CRC (MSB)
ViVOtech2\0	09h	03	00h	00h		

Response Frame

Byte 0-9	Byte 10	Byte 11	Byte 12	Byte 13	Byte 14	Byte 15	Byte 0-9
Header Tag & Protocol	Command	Status Code	Data Length (MSB)	Data Length (LSB)	Data	CRC (MSB)	CRC (LSB)
ViVOtech2\0	09h	See Status Code Table			See below		

The Get Main Firmware Version sub-command returns a TLV string as follows:

- **Tag:** 0xDF62
- **Length:** Varies
- **Value:** Varies field representing the main firmware version.

The following example shows the command and response.

Command: Get Main Firmware Version: `56 69 56 4F 74 65 63 68 32 00 09 03 00 00 C0 CE`

Response: `56 69 56 4F 74 65 63 68 32 00 09 00 00 14 DF 62 11 43 72 61 6E 65 56 65 6E 64 69 5F 31 2E 30 2E 30 00 E1 5D`

7.5. Get Hardware Information (09-14)

The **Get Hardware Information** command retrieves information about the reader's hardware.

Command Frame

Byte 0-9	Byte 10	Byte 11	Byte 12	Byte 13	Byte 14	Byte 15
Header Tag & Protocol	Command	Sub-Command	Data Length (MSB)	Data Length (LSB)	CRC (LSB)	CRC (MSB)
ViVOtech2\0	09h	14h	00h	00h		

Response Frame

Byte 0-9	Byte 10	Byte 11	Byte12	Byte 13	Byte 14 … Byte 13+n	Byte 14+n	Byte 15+n
Header Tag & Protocol	Command	Status Code	Data Length (MSB)	Data Length (LSB)	Data	CRC (MSB)	CRC (LSB)
ViVOtech2\0	09h	See Status Code Table			See below		

The format for hardware module version information returned is "human readable," consisting of fields that are separated by commas, and lines separated by carriage return and line feed characters:

```
<module type>,<module name><CRLF>
<chip version>
```

The following example shows the hardware version information subcommand and the information being returned (in ASCII format).

Command: Get Hardware Version Information: 56 69 56 4F 74 65 63 68 32 00 09 14 00 00 33 08

Response: For example, a VP5300 returns
`5669564f7465636832000900001548572c205650353330300d0a4b38314620526576347 7d5`
In ASCII: `HW, VP5300 <CR><LF>K81F.Rev4`

7.6. Get Module Version Information (09-20)

The **Get Module Version Information** command retrieves the reader's module information.

Command Frame

Byte 0-9	Byte 10	Byte 11	Byte 12	Byte 13	Byte 14	Byte 15
Header Tag & Protocol	Command	Sub-Command	Data Length (MSB)	Data Length (LSB)	CRC (LSB)	CRC (MSB)
ViVOtech2\0	09h	20h	00h	00h		

Response Frame

Byte 0-9	Byte 10	Byte 11	Byte12	Byte 13	Byte 14 … Byte 13+n	Byte 14+n	Byte 15+n
Header Tag & Protocol	Command	Status Code	Data Length (MSB)	Data Length (LSB)	Data	CRC (MSB)	CRC (LSB)
ViVOtech2\0	09h	See Status Code Table			See below		

If there is an error, the reader returns the appropriate status code with an empty data field (Data Length = 0000h).

The format for module version information returned is "human readable," consisting of fields separated by commas and lines separated by carriage return and line feed characters:

```
<module type>,<module name and spec. version>,[<implementation
version>],<CRLF>
```

The following example shows the module version information subcommand and the information being returned (in ASCII format).

Command: Get Module Version Information: 56 69 56 4F 74 65 63 68 32 00 09 20 00 00 56 11

Response:

```
56 69 56 4F 74 65 63 68 32 00 09 00 01 2A 46 57 2C 56 65 6E 64 69 20
56 31 2E 30 30 2C 2C 0D 0A 20 46 53 26 44 42 20 56 31 2E 30 30 2C 2C
0D 0A 20 43 4C 20 41 70 70 53 65 6C 2C 50 50 53 45 20 4D 6F 64 75 6C
65 2C 20 76 31 2E 30 30 2C 2C 0D 0A 20 43 4C 20 41 49 44 2C 4D 61 73
74 65 72 43 61 72 64 20 50 61 79 50 61 73 73 20 4D 2F 43 68 69 70 20
76 33 2E 30 2E 32 2C 20 56 65 6E 64 69 20 76 31 2E 30 2E 30 2C 2C 0D
0A 20 43 4C 20 41 49 44 2C 56 69 73 61 20 56 43 50 53 20 32 2E 31 2E
33 2C 20 76 30 2E 39 39 2C 2C 0D 0A 20 43 4C 20 41 49 44 2C 41 6D 65
78 20 45 78 70 72 65 73 73 50 61 79 20 33 2E 30 2C 20 76 31 2E 30 30
2C 2C 0D 0A 20 43 4C 20 41 49 44 2C 44 69 73 63 6F 76 65 72 20 44 50
41 53 20 31 2E 30 20 5A 69 70 20 33 2E 31 2E 32 2C 20 76 31 2E 30 30
2C 2C 0D 0A 20 43 4C 20 41 49 44 2C 49 6E 74 65 72 61 63 20 31 2E 35
2C 20 76 31 2E 30 30 2C 2C 0D 0A 20 43 4C 20 4C 31 2C 45 4D 56 20 34
2E 33 20 4C 31 2C 20 76 31 2E 30 30 00 8C 33
```

ASCII translation of the data field:

```
FW,Vendi V1.00,,<CR><LF> FS&DB V1.00,,<CR><LF>
CL AppSel,PPSE Module, v1.00,,<CR><LF>
CL AID,MasterCard PayPass M/Chip v3.0.2, Vendi v1.0.0,,<CR><LF> CL
AID,Visa VCPS 2.1.3, v0.99,,<CR><LF>
CL AID,Amex ExpressPay 3.0, v1.00,,<CR><LF>
CL AID,Discover DPAS 1.0 Zip 3.1.2, v1.00,,<CR><LF> CL AID,Interac
1.5, v1.00,,<CR><LF>
CL L1,EMV 4.3 L1, v1.00<NUL>
```

7.7. Get Serial Number (12-01)

The **Get Serial Number** command instructs the ViVOpay reader to return the 15-digit serial number stored in its non-volatile memory. If a serial number has not been previously set in the reader, this command fails with a **Command Not Allowed** error status. If the command frame is not valid, the reader returns an error response frame.

Command Frame

Byte 0-9	Byte 10	Byte 11	Byte 12	Byte 13	Byte 14	Byte 15
Header Tag & Protocol Version	Command	Sub-Command	Data Length (MSB)	Data Length (LSB)	CRC (LSB)	CRC (MSB)
ViVOtech2\0	12h	01h	00h	0Fh		

The serial number will be returned tail-padded with **0x00** to a length of 15.

Example:

TX: 56 69 56 4F 74 65 63 68 32 00 12 01 00 00 18 A5

RX: 56 69 56 4F 74 65 63 68 32 00 12 00 00 0F **36 33 30 5A 30 30 30 30 30 31 00 00 00 00 00** 94 BC

Response Frame

Byte 0-9	Byte 10	Byte 11	Byte 12	Byte 13	Byte 14 … Byte 14+n-1	Byte 14+n	Byte 15+n
Header Tag & Protocol Version	Command	Status Code	Data Length (MSB)	Data Length (LSB)	Data	CRC (MSB)	CRC (LSB)
ViVOtech2\0	12h	See Status Code Table	00h	0Fh	15-digit Serial Number		

7.8. Contact Set ICS Identification (60-16)

The **Contact Set ICS Identification** command sets the reader's ICS terminal configuration number (Contact Command EMV L2 includes 4 approved configurations of certification: 1C, 2C, 3C, or 4C).

This command affects the **Contact Set Terminal Data (60-06)** command. Generally, readers call 60-16 first, then 60-06.

Examples:

Set 3C configuration: `5669564f7465636832006016000103 92ed`

Reader responds with: `5669564f746563683200600000003d35` (Success)

Now set terminal data (TLVs) with config values appropriate to 3C:

```
5669564f746563683200600600c818005f3601029f1a0208409f3501259f330360d8
c89f40056000f050019f1e085465726d696e616c9f150212349f160f303030303030
303030303030303030309f1c0838373635343332319f4e22313037323120576 16c6b65
722053742e20437970726573732c204341202c5553412edf260101df1008656e6672
65737a68df110101df270100dfee150101dfee160100dfee170107dfee180180dfee
1e08f0dc24f0c20e1400dfee1f0180dfee1b0830303031353130 30dfee20013cdfee
21010adfee2203323c3caa88
```

Reader responds with: `5669564f746563683200600000003d35` (Success)

Command Frame

Byte 0-9	Byte 10	Byte 11	Byte 12	Byte 13	Byte 14 … Byte 14+n-1	Byte 14+n	Byte 15+n
Header Tag & Protocol Version	Command	Sub-Command	Data Length (MSB)	Data Length (LSB)	Data	CRC (LSB)	CRC (MSB)
ViVOtech2\0	60h	16h	00h	01h	ICS Identification		

ICS Identification:

0x01 → Identification 1C
0x02 → Identification 2C (Default)
0x03 → Identification 3C
0x04 → Identification 4C

Response Frame

Byte 0-9	Byte 10	Byte 11	Byte 12	Byte 13	Byte 14 … Byte 14+n-1	Byte 14+n	Byte 15+n
Header Tag & Protocol Version	Command	Status Code	Data Length (MSB)	Data Length (LSB)	Data	CRC (MSB)	CRC (LSB)
ViVOtech2\0	60h	See Status Code Table	00	00			

7.9. Contact Set Terminal Data (60-06)

The **Contact Set Terminal Data** command creates new terminal data according to the TLVs passed in. Terminal data is mandatory and seldom changes. It represents configuration data that usually gets set one time in pre-production and never changes after.

Command Frame

Byte 0-9	Byte 10	Byte 11	Byte 12	Byte 13	Byte 14 … Byte 14+n-1	Byte 14+n	Byte 15+n
Header Tag & Protocol Version	Command	Sub-Command	Data Length (MSB)	Data Length (LSB)	Data	CRC (LSB)	CRC (MSB)
ViVOtech2\0	60h	06h			Data Objects		

Data Objects: <TagCounterL><TagCounterH><TLV1><TLV2>…<TLVn>.

Where: <TagCounterL><TagCounterH>: the Number of <TLV> tags.

Response Frame

Byte 0-9	Byte 10	Byte 11	Byte 12	Byte 13	Byte 14 … Byte 14+n-1	Byte 14+n	Byte 15+n
Header Tag & Protocol Version	Command	Status Code	Data Length (MSB)	Data Length (LSB)	Data	CRC (MSB)	CRC (LSB)
ViVOtech2\0	60h	See Status Code Table			See Response Frame Data Format		

Note:
If a <TLV> format was bad, the response status code is **0x05**. If there is a flash error, the response status code is **0x62**.

Terminal Data List Example

Data ID	Tag	Value name	Length (Byte)	Data
1	5F36	Transaction currency exponent	1	02
2	9F1A	Terminal county code	2	08 40
3	9F35	Terminal type	1	21
4	9F33	Terminal capability	3	60 28 C8
5	9F40	Additional terminal capability	5	F0 00 F0 A0 01
6	9F1E	IFD serial number	8	54 65 72 6D 69 6E 61 6C
7	9F15	Merchant category code	2	12 34
8	9F16	Merchant identifier	15	30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
9	9F1C	Terminal identification	8	38 37 36 35 34 33 32 31
10	9F4E	Merchant name and location	<=64	31 30 37 32 31 20 57 61 6C 6B 65 72 20 53 74 2E 20 43 79 70 72 65 73 73 2C 20 43 41 20 2C 55 53 41 2E
11	DF26	Terminal supports CRL	1	01
12	DF10	Language	Var. up to 128	65 6E 66 72 65 73 7A 68
13	DF11	Support transaction log	1	00
14	DF27	support exception file	1	00
15	DFEE15	Terminal support ASI	1	01
16	DFEE16	Terminal encrypt mode	1	00
17	DFEE17	Terminal entry mode for ICC	1	07
18	DFEE18	Terminal encrypt mode for MSR	1	80
19	DFEE1E	Contact terminal configuration	8	D0 DC 20 D0 C4 1E 16 00
20	DFEE1F	Issuer script limit	1	80
21	DFEE1B	ARC define	8	30 30 30 31 35 31 30 30
22	DFEE20	ICC power on detect waiting time	1	3C
23	DFEE21	ICC L1 waiting time	1	0A
24	DFEE22	Driver waiting time. byte 1 -> Menu. byte 2 -> Get Pln. byte 3 -> MSR	3	32 3C 3C

The Contact Common EMV L2 comes with four approved configurations as shown below (1C, 2C, 3C, 4C); these correspond to the scenarios defined by EMVCo in tag 9F35. Parameters marked as Major Parameters usually cannot be changed without causing a checksum error, although certain flag bits (see tables below) can be changed. The Major Parameters are considered read-only because these are the settings the device was certified with for EMV L2 certification.

Terminal configuration					
Identification	**Tag**	**1C**	**2C**	**3C**	**4C**
Major parameters	9F33	60 F8 C8	60 28 C8	60 D8 C8	60 08 C8
	9F35	22	21	25	25
	9F40	F0 00 F0 A0 01	F0 00 F0 A0 01	60 00 F0 50 01	60 00 F0 50 01
	DF11	01	00	01	01
	DF26	01	01	01	01
	DF27	00	00	00	00
	DFEE1E	F0 DC 3C F0 C2 9E 96 00	D0 DC 20 D0 C4 1E 16 00	F0 DC 24 F0 C2 0E 16 00	D0 9C 20 F0 C2 0E 16 00

Tag 9F33 — **Description** — **Length 3**

Terminal Capabilities

Byte 1

b8	b7	b6	b5	b4	b3	b2	b1	Meaning	Change
1	x	x	x	x	x	x	x	Manual key entry	
x	1	x	x	x	x	x	x	Magnetic stripe	
x	x	1	x	x	x	x	x	IC with contacts	Major
x	x	x	0	x	x	x	x	RFU	
x	x	x	x	0	x	x	x	RFU	
x	x	x	x	x	0	x	x	RFU	
x	x	x	x	x	x	0	x	RFU	
x	x	x	x	x	x	x	0	RFU	

Byte 2

b8	b7	b6	b5	b4	b3	b2	b1	Meaning	Change
1	x	x	x	x	x	x	x	Plaintext PIN for IC verification	Major
x	1	x	x	x	x	x	x	Enciphered PIN for online verification	Major
x	x	1	x	x	x	x	x	Signature(paper)	Major
x	x	x	0	x	x	x	x	Not supported	Major
x	x	x	x	1	x	x	x	No CVM Required	Major
x	x	x	x	x	0	x	x	RFU	
x	x	x	x	x	x	0	x	RFU	
x	x	x	x	x	x	x	0	RFU	

Byte 3

b8	b7	b6	b5	b4	b3	b2	b1	Meaning	Change
1	x	x	x	x	x	x	x	SDA	Major
X	1	x	x	x	x	x	x	DDA	Major
X	x	1	x	x	x	x	x	Card capture	
x	x	x	0	x	x	x	x	RFU	
x	x	x	x	1	x	x	X	CDA	Major
x	x	x	x	x	0	x	x	RFU	
x	x	x	x	x	x	0	x	RFU	
x	x	x	x	X	X	x	0	RFU	

9F35 — Length 1

Terminal Type

Environment	Financial Institution	Merchant	Cardholder	Change
Attended				Major
Online only	11	21	--	
Offline with online capability	12	22	--	

Offline only	13	23	--		
Unattended				Major	
Online only	14	24	34		
Offline with online capability	15	25	35		
Offline only	16	26	36		

9F40 — Additional Terminal Capabilities — 5

Byte 1

b1	b2	b3	b4	b5	b6	b7	b8	Meaning	Change
1	x	x	x	x	x	x	x	Cash	Major
x	1	x	x	x	x	x	x	Goods	Major
x	x	1	x	x	x	x	x	Services	Major
x	x	x	1	x	x	x	x	Cashback	Major
x	x	x	x	1	x	x	x	Inquiry	
x	x	x	x	x	1	x	x	Transfer	
x	x	x	x	x	x	1	x	Payment	
x	x	x	x	x	x	x	1	Administrative	

Byte 2

b8	b7	b6	b5	b4	b3	b2	b1	Meaning	Change
1	x	x	x	x	x	x	x	Cash Deposit	
x	0	x	x	x	x	x	x	RFU	
x	x	0	x	x	x	x	x	RFU	
x	x	x	0	x	x	x	x	RFU	
x	x	x	x	0	x	x	x	RFU	
x	x	x	x	x	0	x	x	RFU	
x	x	x	x	x	x	0	x	RFU	
x	x	x	x	x	x	x	0	RFU	

Byte 3

b8	b7	b6	b5	b4	b3	b2	b1	Meaning	Change
1	x	x	x	x	x	x	x	Numeric keys	
x	1	x	x	x	x	x	x	Alphabetic and special characters keys	
x	x	1	x	x	x	x	x	Command keys	
x	x	x	1	x	x	x	x	Function Keys	
x	x	x	x	0	x	x	x	RFU	
x	x	x	x	x	0	x	x	RFU	
x	x	x	x	x	x	0	x	RFU	
x	x	x	x	x	x	x	0	RFU	

Byte 4

b8	b7	b6	b5	b4	b3	b2	b1	Meaning	Change
1	x	x	x	x	x	x	x	Print, attendant	
x	1	x	x	x	x	x	x	Print, cardholder	
x	x	1	x	x	x	x	x	Display, attendant	
x	x	x	1	x	x	x	x	Display, cardholder	
x	x	x	x	0	x	x	x	RFU	
x	x	x	x	x	0	x	x	RFU	
x	x	x	x	x	x	1	x	Code table 10	Major

	x	x	x	x	x	x	x	1	Code table 9	Major

Byte 5

b8	b7	b6	b5	b4	b3	b2	b1	Meaning	Change
1	x	x	x	x	x	x	x	Code table 8	Major
x	1	x	x	x	x	x	x	Code table 7	Major
x	x	1	x	x	x	x	x	Code table 6	Major
x	x	x	1	x	x	x	x	Code table 5	Major
x	x	x	x	1	x	x	x	Code table 4	Major
x	x	x	x	x	1	x	x	Code table 3	Major
x	x	x	x	x	x	1	x	Code table 2	Major
x	x	x	x	x	x	x	1	Code table 1	Major

DF11	Transaction Log Support (Default: Enable) (Major) 0 → Disable 1 → Enable	1
DF26	Revocation List Support (Default: Enable) (Major) 0 → Disable 1 → Enable	1
DF27	Exception File Support (Default: Disable) (Major) 0 → Disable 1 → Enable	1
DFEE1E	Contact Terminal Configuration (Default: F0 DC 3C F0 C2 9E 94 00)	

Byte 1

b8	b7	b6	b5	b4	b3	b2	b1	Meaning	Change
1	x	x	x	x	x	x	x	Keypad support	
x	1	x	x	x	x	x	x	LCD support	
x	x	1	x	x	x	x	x	PIN Pad support	
x	x	x	1	x	x	x	x	Print Support	
x	x	x	x	0	x	x	x	RFU	
x	x	x	x	x	0	x	x	RFU	
x	x	x	x	x	x	0	x	RFU	
x	x	x	x	x	x	X	0	RFU	

Byte 2

b8	b7	b6	b5	b4	b3	b2	b1	Meaning	Change
1	x	x	x	x	x	x	x	PSE support	Major
x	1	x	x	x	x	x	x	Cardholder confirmation	Major
x	x	1	x	x	x	x	x	Preferred display order	
x	x	x	1	x	x	x	x	Multi language	
x	x	x	x	1	x	x	x	EMV language selection method	
x	x	x	x	x	1	x	x	Default DDOL	Major
x	x	x	x	x	x	0	x	RFU	
x	x	x	x	x	x	x	0	RFU	

Byte 3

b8	b7	b6	b5	b4	b3	b2	b1	Meaning	Change
0	x	x	x	x	x	x	x	RFU (Revocation of Issuer Public Key Certificate (DF26))	Major
x	1	x	x	x	x	x	x	Manual action when CA PK loading fails	Major

x	x	1	x	x	x	x	x	CA PK verified with check sum	Major
x	x	x	1	x	x	x	x	Bypass PIN Entry	Major
x	x	x	x	1	x	x	x	Subsequent bypass PIN Entry	Major
x	x	x	x	x	1	x	x	Get data for pin try counter	Major
x	x	x	x	x	x	0	x	RFU	
x	x	x	x	x	x	x	0	RFU	

Byte 4

b8	b7	b6	b5	b4	b3	b2	b1	Meaning	Change
1	x	x	x	x	x	x	x	Amount before CVM processing	Major
x	1	x	x	x	x	x	x	Floor limit checking	Major
x	x	1	x	x	x	x	x	Random transaction selection	Major
x	x	x	1	x	x	x	x	Velocity checking	Major
x	x	x	x	0	x	x	x	RFU (Transaction Log (DF11))	Major
x	x	x	x	x	0	x	x	RFU (Exception File (DF27))	Major
x	x	x	x	x	x	0	x	RFU	
x	x	x	x	x	x	x	0	RFU	

Byte 5

b8	b7	b6	b5	b4	b3	b2	b1	Meaning	Change
1	x	x	x	x	x	x	X	Terminal action code support	Major
x	1	x	x	x	x	x	x	Terminal action code can be change	Major
x	x	1	x	x	x	x	x	Terminal action code can be deleted or disable	Major
x	x	x	1	x	x	x	x	Default Action code processing before 1st GAC	Major
x	x	x	x	1	x	x	x	Default Action code processing after 1st GAC	Major
x	x	x	x	x	1	x	x	TAC/IAC default process when unable to go online (Skipped)	Major
x	x	x	x	x	x	1	x	TAC/IAC default process when unable to go online (Normal)	Major
x	x	x	x	x	x	x	0	RFU	

Byte 6

b8	b7	b6	b5	b4	b3	b2	b1	Meaning	Change
1	x	x	x	x	x	x	x	Forced Online support	Major
x	1	x	x	x	x	x	x	Forced acceptance support	Major
x	x	1	x	x	x	x	x	Advice support	Major
x	x	x	1	x	x	x	x	Issuer referrals support	Major
X	x	x	x	1	x	x	x	Batch data capture	Major
x	x	x	x	x	1	x	x	Online data capture	Major
X	x	x	x	x	x	1	x	Default TDOL	Major
X	x	x	x	x	x	x	0	RFU	

Byte 7

b8	b7	b6	b5	b4	b3	b2	b1	Meaning	Change
1	x	x	x	x	x	x	x	amount and pin entered on the same keypad	
x	1	x	x	x	x	x	x	ICC/Magstripe reader combined	
x	x	1	x	x	x	x	x	Magstripe read first	

	x	x	x	1	x	x	x	x	Support account type selection	
	x	x	x	x	1	x	x	x	On fly script processing	
	x	x	x	x	x	1	x	x	Internal date management	
	x	x	x	x	x	x	1	x	Reversal Mode (1)Unable go online (2) ARC Error 0: (3) Online Approved but reader not approved. 1: (3) Online Approved but card response AAC.	
	x	x	x	x	x	x	x	0	RFU	

Byte 8

b8	b7	b6	b5	b4	b3	b2	b1	Meaning	Change
x	x	x	x	x	x	x	x	RFU	

7.10. Contact Set Application Data (60-03)

The **Contact Set Application Data** command creates a new AID configuration, up to a maximum of 16 sets.

Command Frame

Byte 0-9	Byte 10	Byte 11	Byte 12	Byte 13	Byte 14 … Byte 14+n-1	Byte 14+n	Byte 15+n
Header Tag & Protocol Version	Command	Sub-Command	Data Length (MSB)	Data Length (LSB)	Data	CRC (LSB)	CRC (MSB)
ViVOtech2\0	60h	03h			Data Objects		

Data Objects: <AID_LenL><AID_LenH><5~16 bytes AID><TagCounterL><TagCounterH><TLV1><TLV2>…<TLVn>.

Where: <TagCounterL><TagCounterH>is the Number of <TLV>.

Response Frame

Byte 0-9	Byte 10	Byte 11	Byte 12	Byte 13	Byte 14 … Byte 14+n-1	Byte 14+n	Byte 15+n
Header Tag & Protocol Version	Command	Status Code	Data Length (MSB)	Data Length (LSB)	Data	CRC (MSB)	CRC (LSB)
ViVOtech2\0	60h	See Status Code Table			See Response Frame Data Format		

Note:

- If there was a <TLV> format error, the response status code is 0x05.
- If the AID List is full (MAX is 16), the response status code is 0x61.

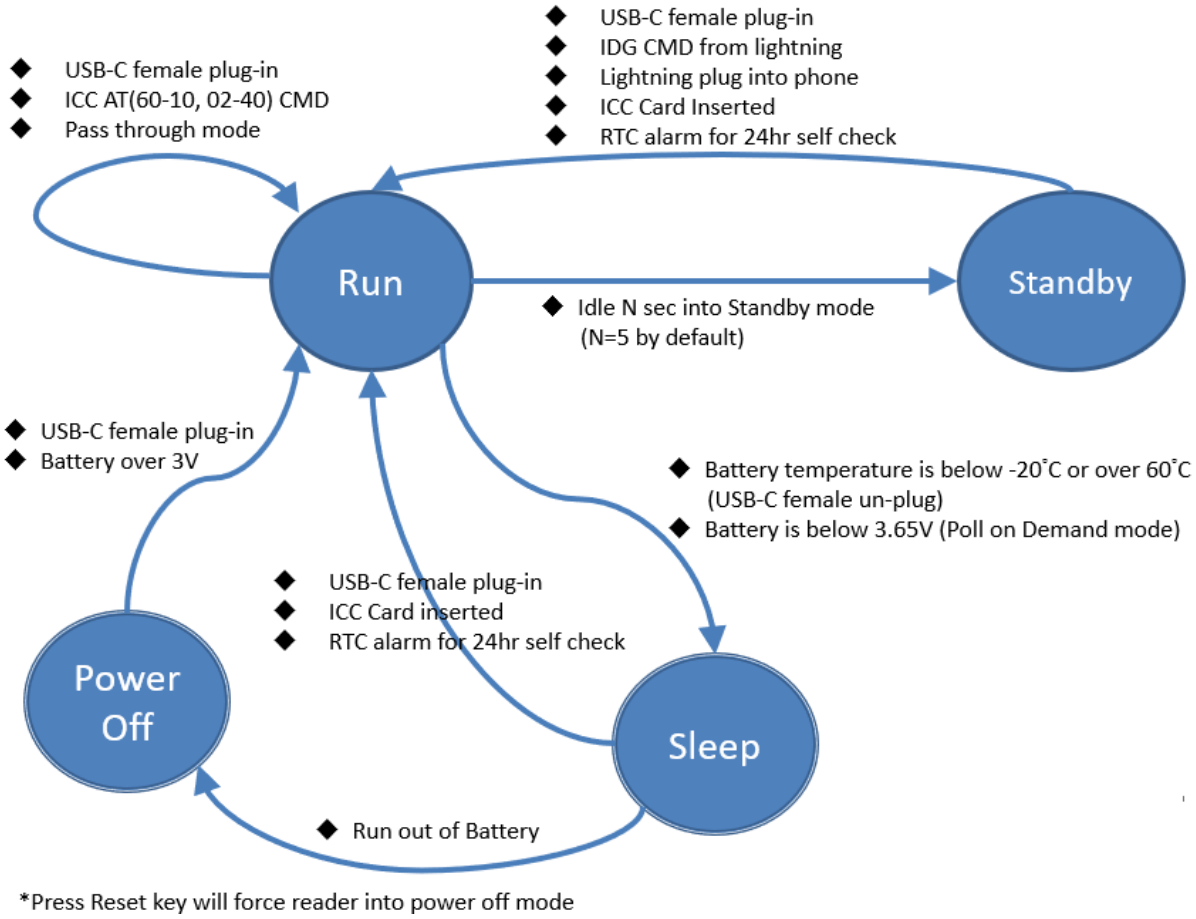
Application Data List Example

Data ID	Tag	Value name	Length (Byte)	Data
1	9F01	Acquirer identifier	6	56 49 53 41 30 30
2	5F57	Account type	1	00
3	5F2A	Transaction Currency	2	08 40
4	9F09	Terminal application version number	2	00 96
5	5F36	Transaction currency exponent	1	02
6	9F1B	Terminal floor limit	4	00 00 3A 98
7	DF25	Default DDOL	Var	9F 37 04
8	DF28	Default TDOL	Var	9F 08 02
9	DFEE15	ASI	1	01
10	DF13	TAC-Default	5	00 00 00 00 00
11	DF14	TAC-Denial	5	00 00 00 00 00
12	DF15	TAC-Online	5	00 00 00 00 00
13	DF18	Target percentage for random transaction selection	1	00
14	DF17	Threshold value for biased random selection	4	00 00 27 10
15	DF19	Maximum target percentage for random transaction selection	1	00

7.11. Power Management

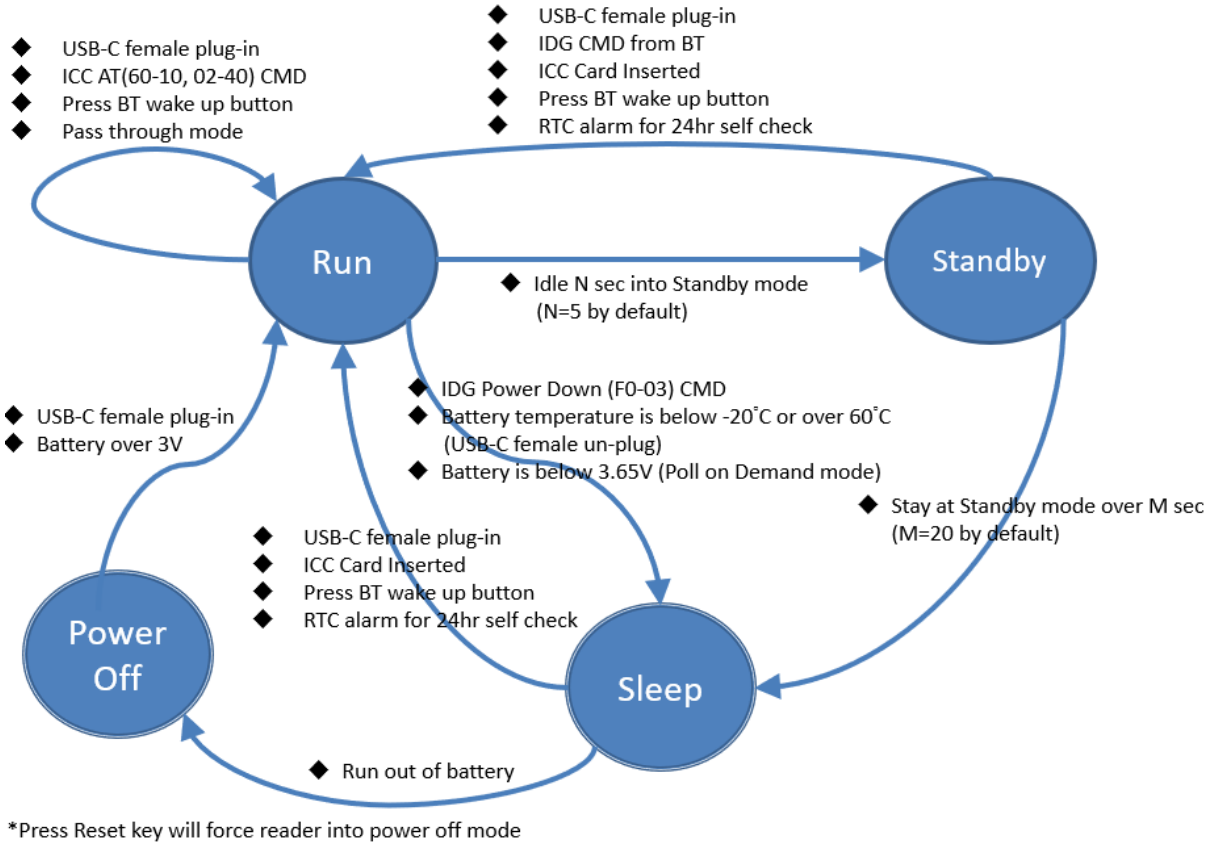
The diagrams and commands below describe power management for Lightning and Bluetooth VP3350 models.

7.11.1 VP3350 Lightning Connector PMC State Diagram



*Press Reset key will force reader into power off mode

7.11.2 VP3350 Bluetooth PMC State Diagram



7.11.3 Enter Low Power Mode (F0-03)

The **Enter Low Power Mode** command places the reader in low power mode.

Command Frame

Byte 0-9	Byte 10	Byte 11	Byte 12	Byte 13	Byte 14	Byte 15	Byte 16
Header Tag & Protocol Version	Command	Sub-Command	Data Length (MSB)	Data Length (LSB)	Data	CRC (LSB)	CRC (MSB)
ViVOtech2\0	F0h	03h	0x00 0x02		See Command Data Table		

Command Data

Data Item	Length (bytes)	Description/Example
Low Power Mode	1	0x00: Sleep Mode (resume from last instruction) 0x01: Stop Mode (power on reset POR required)
MSR / NFC Option	1	0x00: OFF 0x01: ON (Swipe/Tap will wake from low power)

Use `00 01` to put the reader in **Sleep Mode**. It will awaken upon card presentation.

With `00 00`, the reader will wake up with card *insertion*, but not with contactless (NFC) presentation (the card-seated switch is always enabled; it cannot be disabled).

Wakeup from Stop Mode involves a reboot (which takes several seconds). **Wakeup from Sleep Mode** does not reboot the reader.

Response Frame

Byte 0-9	Byte 10	Byte 11	Byte 12	Byte 13	Byte 14	Byte 15
Header Tag & Protocol Version	Command	Status Code	Data Length (MSB)	Data Length (LSB)	CRC (MSB)	CRC (LSB)
ViVOtech2\0	F0h	See status code table	00h	00h		

The reader will wake up automatically upon card presentation, or when a command is issued. On waking up (after being triggered by card presentation), the reader will output `5669564F746563683200F10000008D1E`.

Note:

- VP3350 Model B supports the use of `F0 03 00 00` to enter low power mode sleep.
- VP3350 Model C supports the use of `F0 03 00 00` and `F0 03 01 00` to enter low power mode sleep and stop.

Model	Low Power Mode		MSR / NFC Option	
	0x00: Sleep	0x01: Stop	0x00: Off	0x01: On
VP3350 Model B	Support	Not Support	Support	Not support
VP3350 Model C	Support	Support	Support	Not support

7.11.4 Set Low Power Consumption Configuration (F0-04)

The **Set Low Power Consumption Configuration** command enables or disables some level Low Power Consumption functions and the waiting timer of the previous state.

Note: Low Speed run and Standby0 is not supported in NEO3. Any attempt to set either of the two timers results in an error response.

Command Frame

Byte 0-9	Byte 10	Byte 11	Byte 12	Byte 13	Byte 14 ... Byte 14+n-1	Byte 14+n	Byte 15+n
Header Tag & Protocol Version	Command	Sub-Command	Data Length (MSB)	Data Length (LSB)	Data	CRC (LSB)	CRC (MSB)
ViVOtech2\0	F0h	04h			See Tables below		

Data Field	Length (bytes)	Description
The low power level ID	1	The range of low power ID is 1~4. The detailed supported level is defined by the PDS of the product.
low power timer	2	Time (**default 30s**) • 0000h = Does NOT support the level of low power consumption function (should be skipped). • 0001h ~ 0E10h (1~3600 Seconds) = Supports the level of low power consumption function. The value is defined by the previous state's timer "The low power level ID" level.

Note:

Default Setting for a VP3350 reader:

ID	Current Name	Timeout
1	Low speed run	30s
2	Standby0	30s
3	Standby1	30s
4	Sleep	30s

IDLE -(30s)->LowSpeedRun -(30s)->Standby0 -(30s)->Standby1 -(30s)->Sleep

After sending the following commands:

- `01 00 1F`: Enable Low Speed Run state and previous state (Idle) waiting timer is 31 seconds.
- `02 00 00`: Disable Standby 0.
- `03 00 21`: Enable Standby 1 state and previous state (Low Speed Run) waiting timer is 33 seconds.
- `04 00 22`: Enable Sleep state and previous state (Standby 1) waiting timer is 34 seconds.

The reader changes to these values:

ID	Current Name	Timeout
1	Low speed run	31s
2	Standby0	0
3	Standby1	33s
4	Sleep	34s

IDLE -(31s)-> LowSpeedRun -(33s)-> Standby1 -(34s)-> Sleep

Response Frame

Byte 0-9	Byte 10	Byte 11	Byte 12	Byte 13	Byte 14 … Byte 14+n-1	Byte 14+n	Byte 15+n
Header Tag & Protocol Version	Command	Status Code	Data Length (MSB)	Data Length (LSB)	Data	CRC (MSB)	CRC (LSB)
ViVOtech2\0	F0h	See status code table	00h	00h			

Default Settings for VP3350 readers:

ID	1	2	3	4
	Low speed run	Wait	Standby	Sleep
VP3350 Model B	Not support	Not support	5s	Not support
VP3350 Model C	Not support	Not support	5s	15s

7.11.5 Get Low Power Consumption Configuration (F0-05)

The **Get Low Power Consumption** command retrieves reader's the low power consumption function configuration.

Command Frame

Byte 0-9	Byte 10	Byte 11	Byte 12	Byte 13	Byte 14 … Byte 14+n-1	Byte 14+n	Byte 15+n
Header Tag & Protocol Version	Command	Sub-Command	Data Length (MSB)	Data Length (LSB)	Data	CRC (LSB)	CRC (MSB)
ViVOtech2\0	F0h	05h	00h	00h			

Response Frame

Byte 0-9	Byte 10	Byte 11	Byte 12	Byte 13	Byte 14 … Byte 14+n-1	Byte 14+n	Byte 15+n
Header Tag & Protocol Version	Command	Status Code	Data Length (MSB)	Data Length (LSB)	Data	CRC (MSB)	CRC (LSB)
ViVOtech2\0	F0h	See status code table	00h	09h	See Tables below		

Data Field	Length (bytes)	Description/Example
Modes number	1	Total number of supported low power modes. It will be 4.
1st lower power timer	2	
…		
4th lower power timer	2	

Note:

If the reader's configuration is:

ID	Current Name	Timeout
1	Low speed run	31s
2	Standby0	0
3	Standby1	33s
4	Sleep	34s

The data is: `04 00 1F 00 00 00 21 00 22`

7.11.6 Get Battery Level (F0-02)

The **Get Battery Level** command retrieves the reader's battery level.

Command Frame

Byte 0-9	Byte 10	Byte 11	Byte 12	Byte 13	Byte 14	Byte 14	Byte 15
Header Tag & Protocol Version	Command	Sub-Command	Data Length (MSB)	Data Length (LSB)	See Below	CRC (LSB)	CRC (MSB)
ViVOtech2\0	F0h	02h					

Data Format

Item	Length (bytes)	Description
Get Li Battery Voltage Percentage	0	
Get Battery Voltage Value	1	0x00: Coin Battery Voltage Value 0x01: Li Battery Voltage Value

Response Frame

Byte 0-9	Byte 10	Byte 11	Byte 12	Byte 13	Byte 14	Byte 15	Byte 16
Header Tag & Protocol Version	Command	Status Code	Data Length (MSB)	Data Length (LSB)	Data	CRC (MSB)	CRC (LSB)
ViVOtech2\0	F0h	See status code table			See Below		

Response Data Format

Item	Length (bytes)	Description
Li Battery Percentage	1	0x05 ~ 0x64 (5% ~ 100%)
Coin Battery Voltage Value	2	Low byte is first, unit is mV
Li Battery Voltage Value	2	Low byte is first, unit is mV

7.11.7 Set USB Sleep Configuration (F0-06)

The Set USB Sleep Configuration command enables or disables the reader's USB Sleep Configuration.

Command Frame

Byte 0-9	Byte 10	Byte 11	Byte 12	Byte 13	Byte 14	Byte 15	Byte 16
Header Tag & Protocol Version	Command	Sub-Command	Data Length (MSB)	Data Length (LSB)	Data	CRC (LSB)	CRC (MSB)
ViVOtech2\0	F0h	06h	00h	01h	00h: OFF 01h: ON		

Response Frame

Byte 0-9	Byte 10	Byte 11	Byte 12	Byte 13	Byte 14	Byte 15
Header Tag & Protocol Version	Command	Status Code	Data Length (MSB)	Data Length (LSB)	CRC (MSB)	CRC (LSB)
ViVOtech2\0	F0h	See status code table	00h	00h		

7.11.8 Get USB Sleep Configuration (F0-07)

The Get USB Sleep Configuration retrieves the reader's USB sleep configuration.

Note: When retrieving the **USB sleep configuration** value, the priority is as follows:

1. **F0-06** takes precedence over Device Tree settings.
2. Device Tree settings take precedence over any hardcoded value.
3. The USB sleep configuration hardcoded default value is OFF.

Ex: The **F0-06** operation is not active and there is NO Device Tree setting or a defective Device Tree setting: the USB sleep configuration value is OFF.
Ex: The **F0-06** operation is not active and valid there is a valid Device Tree setting: the USB sleep configuration value follows Device Tree settings.
Ex: The **F0-06** operation is active: the USB sleep configuration value follows the setting in **F0-06**.

Command Frame

Byte 0-9	Byte 10	Byte 11	Byte 12	Byte 13	Byte 14	Byte 15
Header Tag & Protocol Version	Command	Sub-Command	Data Length (MSB)	Data Length (LSB)	CRC (LSB)	CRC (MSB)
ViVOtech2\0	F0h	07h	00h	00h		

Response Frame

Byte 0-9	Byte 10	Byte 11	Byte 12	Byte 13	Byte 14	Byte 15	Byte 16
Header Tag & Protocol Version	Command	Status Code	Data Length (MSB)	Data Length (LSB)	Data	CRC (MSB)	CRC (LSB)
ViVOtech2\0	F0h	See status code table	00h	01h	USB Sleep Configuration		

Data:
00h: OFF
01h: ON

7.12. Charging Capability Configuration Commands

The section below describes commands for configuring VP3350 charging capabilities.

7.12.1 Set Charging Capability Configuration (F0-10)

IDTECH products facilitate passthrough charging via Lightning or USB-C male connectors. The **Set Charging Capability Configuration** command establishes the charging capability through the Lightning or USB-C male connector for device connection. This configured setting is permanently stored in the device's flash memory and becomes operational following either a re-plug of the female connector or a power cycle of the reader. Make sure the desired charging capability aligns with the power source of your connected device.

Command Frame

Byte 0-9	Byte 10	Byte 11	Byte 12	Byte 13	Byte 14 ... 13+n	Byte 14+n	Byte 15+n
Header Tag & Protocol Version	Command	Sub-Command	Data Length (MSB)	Data Length (LSB)	Data	CRC (LSB)	CRC (MSB)
ViVOtech2\0	F0h	10h	00	01	See below	Varies	Varies

Data Format

Data	Description
00h	Dynamic charging capability.
01h	Minimum charging capability.
02h	Maximum charging capability.

VP3350 Charging Capability		
Dynamic	Minimum	Maximum
Lightning: 0.5A ~ 1.5A @ 5V	**Lightning**: 0.5A @ 5V	**Lightning**: 1.5A @ 5V
USB-C: 0.5A ~ 1.5A @ 5V	**USB-C**: 0.5A @ 5V	**USB-C**: 1.5A @ 5V

Response Frame

Byte 0-9	Byte 10	Byte 11	Byte 12	Byte 13	Byte 14	Byte 15
Header Tag & Protocol Version	Command	Status Code	Data Length (MSB)	Data Length (LSB)	CRC (MSB)	CRC (LSB)
ViVOtech2\0	F0h	See status code table	00h	00h	Varies	Varies

7.12.2 Get Charging Capability Configuration (F0-11)

The **Get Charging Capability Configuration** command retrieves the current charging capability configurations from flash memory.

Command Frame

Byte 0-9	Byte 10	Byte 11	Byte 12	Byte 13	Byte 14	Byte 15
Header Tag & Protocol Version	Command	Sub-Command	Data Length (MSB)	Data Length (LSB)	CRC (LSB)	CRC (MSB)
ViVOtech2\0	F0h	11h	00h	00h	Varies	Varies

Response Frame

Byte 0-9	Byte 10	Byte 11	Byte 12	Byte 13	Byte14… 13+n	Byte 14+n	Byte 15+n
Header Tag & Protocol Version	Command	Status Code	Data Length (MSB)	Data Length (LSB)	Data	CRC (MSB)	CRC (LSB)
ViVOtech2\0	F0h	See status code table	00h	01h	See F0 10 command data format	Varies	Varies

8. Basic Card Reading Data Flow

The following examples describe the basic steps for setting a reader's terminal settings and activating a transaction.

8.1. Example: Reading a Card via Firmware Commands

Perform the following steps and commands to read a card with the VP3350 via NEO 2 protocol firmware commands.

1. Set the device's terminal settings:
    a. Run **Set Kernel Configuration (60-16)** to set the kernel configuration number, which acts as a filter to validate that the tags in the next step have the correct major parameters.
    b. Run **Set Terminal Configuration (60-06)** to set the TLV tags, which must have the correct major parameters set in step 1a.
2. Run **Contact Set Application Data (60-03)** to set the device's AID file. Note that, prior to device certification, devices require dummy AIDs to function.
3. Run **Activate Transaction (02-40)** and read the card.

8.2. Example: Reading a Card via Universal SDK Methods

Perform the following steps and commands to read a card with the VP3350 via USDK methods.

1. Set the device's terminal settings:
    a. Run **Set Kernel Configuration**:
    ```
    public RETURN_CODE
    emv_setTerminalMajorConfiguration(int configuration)
    ```
    b. Run **Set Terminal Configuration**:
    ```
    public RETURN_CODE emv_setTerminalData(byte[]
    tlv, string ident = "")
    ```
2. Run **Set Application Data**:
    ```
    public RETURN_CODE emv_setApplicationData(byte[] name, byte[]
    tlv, string ident = "")
    ```
3. Run **Activate Transaction**:
    ```
    public RETURN_CODE device_startTransaction(double amount, double
    amtOther, int exponent, int type, int timeout, byte[] tags, bool
    isFastEMV = false, string ident = "")
    ```
4. Read the card.

8.3. Example: Reading a Card via the USDK Demo App

1. Download and install the latest USDK Demo app from the ID TECH Knowledge Base (if you cannot access the link, please contact support).
2. Connect the VP3350 to your PC via USB or serial port.
3. Open the USDK Demo app from the Windows Start menu.
4. Select **EMV** > **Terminal Config** > **Set Kernel Config**, then input the Config Number (example 2).
5. Select **EMV** > **Terminal Config** > **Save Terminal Data** > **Pick Tag List** (example 2C) > **Execute.**
6. Select **EMV** > **AID** > **Load Default AID**.
7. Select **EMV** > **Activate Transaction**.

9. Application Development Considerations

When developing applications for the VP3350, make sure to consult the *ID TECH Universal SDK Guide* for your respective platform for best practices to follow. Download the *Universal SDK Guide* from the VP3350 product page as part of the ZIP file for your development platform.

ID TECH strongly recommends that integrators include a way for users to update their passwords.

9.1. Performing Key Injection on a VP3350 via Tablet

The VP3350 accepts key injection in two ways when integrated into a tablet:

- Via RKI with the tablet running an application that supports ID TECH RKI (for example, an ID TECH application or ISV application).
- Via the USB-C female port.

Note: When performing key injection via the USB-C female port, the tablet must be turned off. Using a Futurex machine requires the appropriate IDT-Futurex conversion box (ID TECH P/N: ID-80000001-012).

10. VP3350 Mounting and EMV Contactless Logo Requirements

Note that if the VP3350 is mounted behind any kind of casing or cover, that assembly MUST follow EMV requirements regarding contactless logo size and position. See EMVco Contactless Symbol Reproduction Requirements for details.

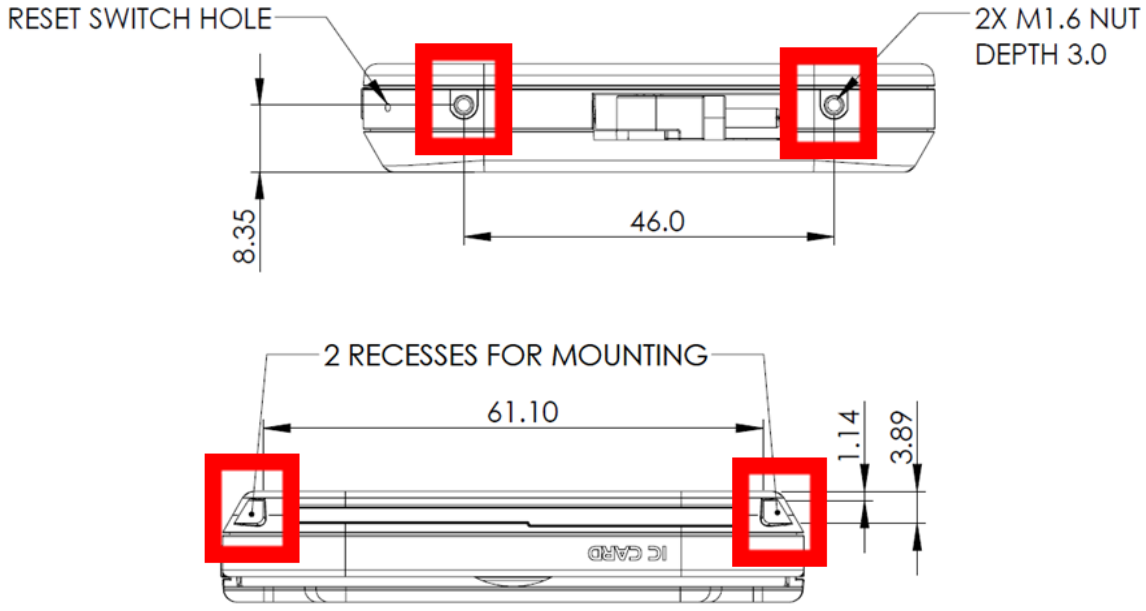
10.1. Contactless Mounting

To optimize performance, install VP3350 readers away from or in front of any metal surfaces or materials that have metallic content, which can interfere with the RF field. VP3350 readers perform optimally when mounted away from metal surfaces.

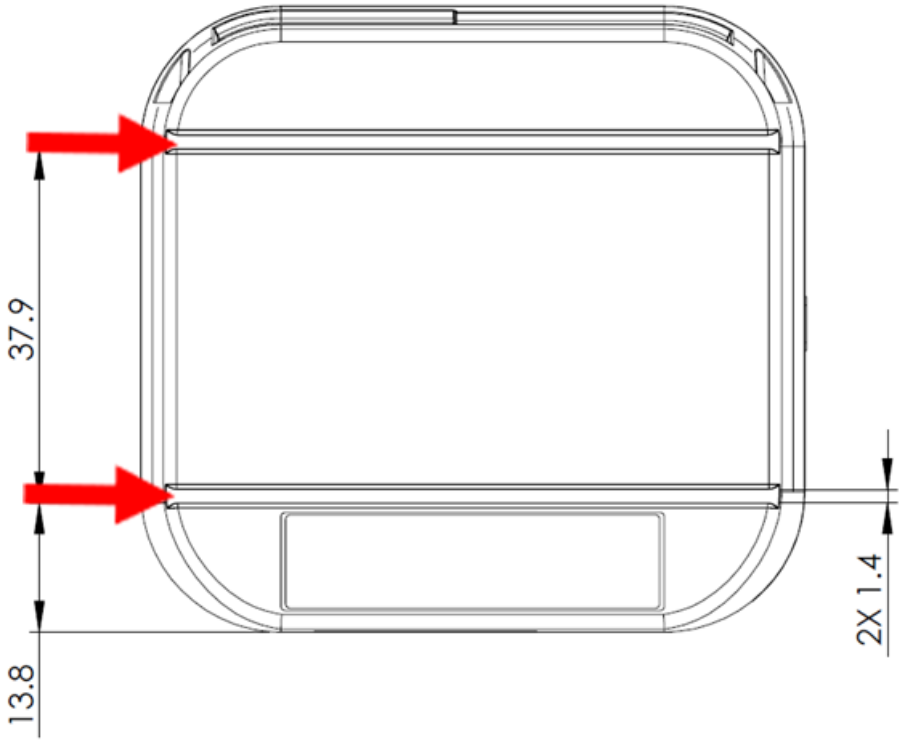
10.2. VP3350 Power Button Requirements and Mounting

When mounting the VP3350 to a surface or with another device (such as a tablet), use the following guidelines:

- For the Bluetooth version of the VP3350, make sure to keep the power button accessible.
- The USB-C male and USB-C female versions of the VP3350 do not require an exposed power button as those models do not have internal batteries.
- If desired, use the two screw holes on the bottom side of unit and two recesses on the top side of unit highlighted below for mounting:



- Likewise, the VP3350 has slots on the back for alignment, shown below:



11. Periodic Inspection Instructions

The VP3350 is an attended device; contact an ID TECH representative with any questions for the device's daily use.

Users are also required to complete the following checks daily:

- Check the device overlay to make sure it is intact.
- Power on the device to check the beeper and the display message:
    - Make sure there is no beeping that indicates the tamper was triggered.
    - Read the firmware version to make sure the version number is correct.
- Check the device appearance to make sure there are no holes on the device or any suspicious objects around the ICC card slot.

12. Decommissioning PCI-Certified Devices

All PCI-certified devices require proper decommissioning prior to device disposal to ensure the protection of all sensitive financial card data. For instructions on decommissioning your device, see Decommissioning of PCI-Certified Devices on the ID TECH Knowledge Base.

13. Troubleshooting

The VP3350 is designed to be reliable and easy to troubleshoot. The components that may require troubleshooting include the power module (if applicable), the reader, and the serial cable.

Symptom	Probable Cause	Remedy
General Issues		
4 RED LEDs blink and the device beeps	Unit has been tampered	Contact IDTECH Support
4 Yellow blink and the device beeps	Unit was not activated	Contact IDTECH Support
4 Yellow LEDs are on	LCL-KEK is not loaded or was erased	Contact IDTECH Support
4 Green LEDs blink	DEK is not loaded or was erased	Contact IDTECH Support
No communication and all LEDs off	• Power off • Battery out of charge • Host device not connected	• Charge the battery • Press the device Reset button • Make sure to enter the correct paring password
Bluetooth paring failed	• Host device running Android 6.x or earlier • Mobile does not work in BLE security mode	• Make sure the host device is running Android 6.x or later
Reader does not appear to be powered on after pressing the power button (no LEDs are lit)	• Reader not powered on • Battery out of charge	• Connect the device to PC via a USB cable • Replace the device with another unit known to work to verify that the installed USB cable wiring works correctly
Some cards or fobs read, but not all	• Possible bad card or fob. • Unsupported card used. • Wrong firmware	• Check to see if the card or fob is damaged • Verify that the correct firmware is loaded on reader; contact your ID TECH representative
LEDs do not light and the beeper is not audible when presenting a card or fob	• Card, fob, or phone not properly presented • RF interference • Unsupported card used • Wrong firmware	• Present card, fob, or phone closer to the antenna and ensure it is parallel to the face of the reader • Verify that the card, fob, or phone is valid and current • Verify that metal is not interfering with the antenna • Test with *ViVOcard Contactless Test Card* part number 241-0015-03, Rev A • Try a different card, fob, or phone • Check to see if the card, fob, or phone is damaged • Verify that correct firmware is loaded on reader; contact your ID TECH representative • Power cable plug is fully inserted • Make sure device is not set to Passthrough mode • Replace the unit
Communication Issues		
No data received or data is garbled	Faulty or incorrect cable connections.	Check that the cable connection is secure and in the correct port on the device
Fail to start transaction, 0x0B returned	Device in Passthrough mode or transaction mode	• Need to exit Passthrough mode • Need to cancel transaction
Fail to start transaction, 0x60 returned	No terminal data or application data	Need to load terminal data or application data for Contact EMV transactions
Fail to start transaction, 0x04 returned	Missing Key	Please re-start VP3350 and monitor the LEDs to confirm whether to Load LCL key or Data Key

Symptom	Probable Cause	Remedy
Firmware loading software indicates "Open device failed"	Device is not fully connected to PC	• Check the cable connection • Check the device
Firmware loading software indicates "Load firmware failed"	Device is not fully connected to PC	Check the cable connections
Firmware loading software indicates "Send Command failed"	Bootloader firmware in device was destroyed	Contact your ID TECH representative to reload manufacture's firmware

If you are unable to resolve the problem, please contact support@idtechproducts.com (sending an e-mail to this address will automatically open a support ticket).

13.1. Tamper Detection Codes

If a tamper event occurs, the VP3350 stores a tamper code in its security log. Check the security log with the **Get DRS Info (C7-3A)** command; see the *NEO 2 Interface Developer's Guide* for details.

Tamper Event Type	Code
EVENT_TYPE_TAMPER_ACTIVE	0
EVENT_TYPE_TAMPER_DEACTIVE	1
EVENT_TYPE_TAMPER_GENERIC	2
EVENT_TYPE_TAMPER_ACK	3
EVENT_TYPE_TAMPER_TIMEOVRF	4
EVENT_TYPE_TAMPER_MONOTONICOVRF	5
EVENT_TYPE_TAMPER_VOLT	6
EVENT_TYPE_TAMPER_CLK	7
EVENT_TYPE_TAMPER_TEMP	8
EVENT_TYPE_TAMPER_FLASH	9
EVENT_TYPE_TAMPER_TST	10
EVENT_TYPE_TAMPER_PIN	11
EVENT_TYPE_TAMPER_BAT	12
EVENT_TYPE_TAMPER_ALL	255

14. For More Information

- To learn more about VP3350 and other ID TECH products, visit the ID TECH Knowledge Base.
- Visit us online at http://idtechproducts.com.
- Find more Tech Support resources at the ID TECH Tech Support home page.

15. Appendix A: VP3350 Demo Setup for Mobile Devices

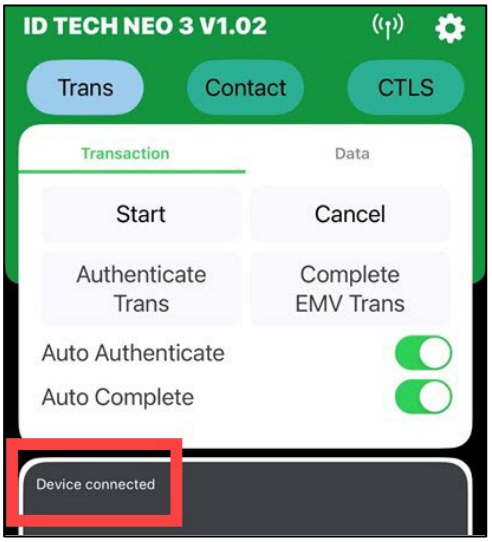
This document provides instructions for setting up a VP3350 for demo purposes.

Before You Begin:

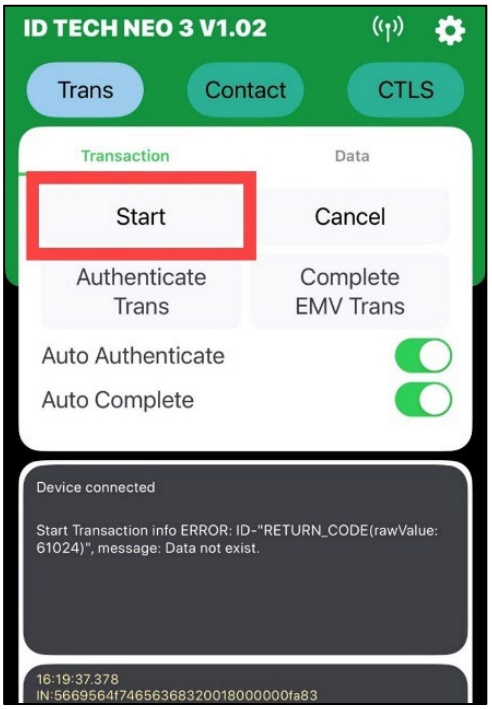
- Make sure there are no other ID TECH readers connected to your mobile device.
  - **iPhone:** Check for ID TECH devices under **Settings -> Bluetooth -> My Devices**. If any are present, select the blue information icon (ⓘ) and select **Forget This Device** at the bottom of the screen.
  - **Android:** Check for ID TECH devices under **Settings -> Connections -> Bluetooth -> Paired Devices**. If any are present, select the settings icon (⚙) for that device, then select **Unpair** on the next screen.
- Go to the Apple App Store or Google Play Store and install the ID TECH NEO 3 app.

Disclaimer: This procedure is for setting up VP3350 for demo purposes only. If a reader is configured for demo purposes, you must re-configure it for use in a production environment.

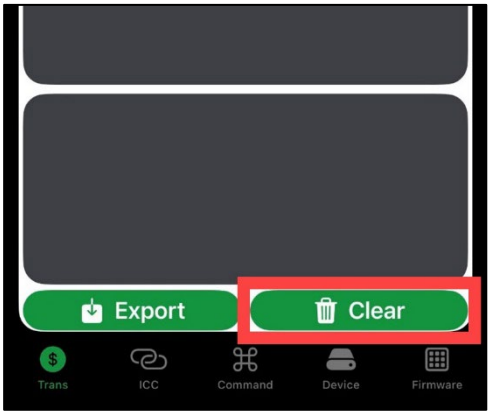
1. Plug the VP3350 into your mobile device (note that if your VP3350 is a Bluetooth model, make sure the reader is paired to your device).
2. Locate and open the ID TECH NEO 3 app. The message pane indicates there is a device connected.



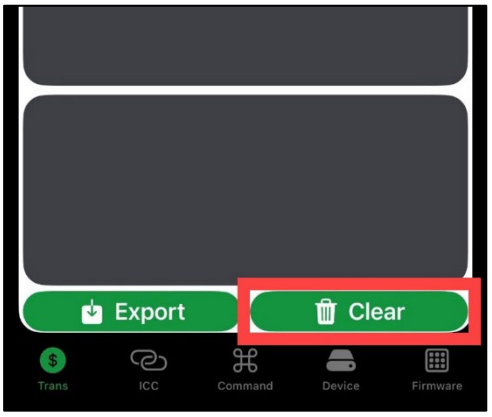
3. Select the **Start** button. The message pane displays **Start Transaction info ERROR: ID-"RETURN_CODE(rawValue:61024)", message: Data not exist**. This error indicates the device has not yet been configured.



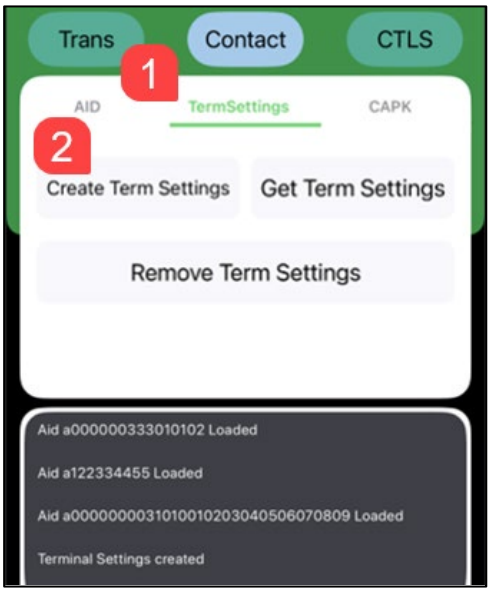
4. Select the **Clear** button to delete all messages in the bottom two panes.



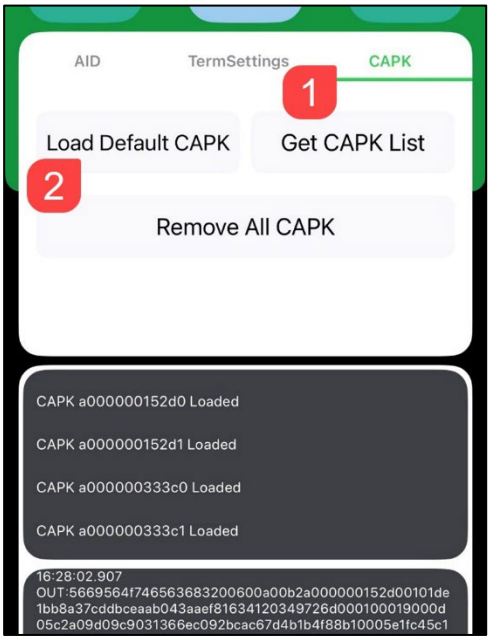
5. Select the **Contact** button at the top of the screen. The default tab is **AID**. Select the **Load Default AIDs** to load the demo AIDs. The message pane below will display that the app has loaded many AIDs onto the reader.



6. Select the **TermSettings** tab, then select the **Create Term Settings** button. The message pane below will display **Terminal Settings created**.



7. Select the **CAPK** tab, then select **Load Default CAPK**. The message pane below will display that the app has loaded many CAPKs onto the reader. VP3350 is now configured for demo purposes.



8. Select the **Trans** button and test the demo by selecting the **Start** button. The message pane below displays that a transaction command was accepted.