VTAP

Integration Guide - Serial Integration Guide

Firmware from v2.2.5.0

VTAP50 and VTAP100

Revised July 2024 v1.3

DOT ORIGIN

If you need help to set up or use your VTAP reader, beyond what is contained in this Integration Guide, then please contact our support team.

Email: **vtap-support@dotorigin.com**

Download the latest documentation and firmware from **https://vtapnfc.com**

Telephone UK and Europe: +44 (0) 1428 685861

Telephone North America and Latin America: +1 (562) 262-9642

If you have any feedback on setting up or using your VTAP reader or this documentation, then please contact our support team. The product is constantly being reviewed and improved and we value feedback about your experience.

Dot Origin Ltd
Unit 7, Coopers Place Business Park, Combe Lane, Wormley
Godalming GU8 5SZ United Kingdom
+44 (0) 1428 685861

Contents

1 Integrating VTAP with other systems

The VTAP reader is designed to be flexible, with many different ways that it can interface to other systems, to accommodate a wide range of requirements. It is important to choose the most suitable interface(s) when integrating a VTAP reader with other systems.

Interfaces available on the VTAP reader

- USB keyboard emulation and mass storage
- USB COM port
- Serial RS-232 interface
  (VTAP50-OEM and VTAP100-OEM option)
- Serial RS-485 interface (VTAP100-PAC-485-CC only)
- Serial TTL/3.3V logic interface (VTAP50-OEM option)
- Wiegand interface (VTAP100-PAC-W only).

Note: The Wiegand interface is specific to Access Control applications and discussed in other VTAP Application Notes on that topic.

The tables in Section **2 Choice of interface** will help you understand the pros and cons of choosing one or more of these interface options.

The following sections explain the difference between a basic use of the VTAP reader USB interface, assumed in the VTAP Configuration Guide, and the more advanced use of a VTAP reader involving one or more serial interfaces.

1.1 Basic - keyboard emulation and mass storage

A simple point-of-sale or kiosk application, with a small number of VTAP readers on one site, is likely to use USB keyboard emulation and mass storage options. The VTAP Configuration Guide deals with the essential functions of a VTAP reader for this situation.

- Detecting a reader is present;
- Reading data from the VTAP reader;
- Changing configuration file settings for reading passes, cards or tags, and providing the configured user feedback;

- Loading any necessary private ECC or application key files;
- Upgrading VTAP reader firmware.

The standard mass storage drive of the VTAP allows most of these functions to be performed by third-party software integration or development, using standard file write functions, under most combinations of programming language and operating system.

1.2 Advanced - USB COM, serial RS-232 or RS-485 communications

If more advanced functionality is required, the USB keyboard emulation and mass storage drive are not always suitable, or may not be the best choice. You should **consider which of the VTAP interfaces best suits each function**.

It may be that your integration requires more advanced functionality, such as:

- 'Background' pass reading - the pass payload is received by an application running on an operating system in the background, rather than the application in focus on that PC (with the cursor correctly positioned to receive the keyboard input);
- Dynamically changing configuration, either to suit different transaction types or switching to card emulation mode and back (for detail refer to VTAP Application Notes on VTAP NFC Tag Emulation).

All of the functions, both basic and advanced, can be reduced to three broad types:

- **Receiving data from the VTAP reader** - pass payload, card or tag data;
- **Sending commands to the VTAP reader**- for dynamic configuration, to control LED/buzzer, or request data or current settings;
- **Transfer files** - either firmware or configuration files.

After considering your choice(s) of interface, this Integration Guide will look at the best approach for each of these types of action one by one.

2 Choice of interface

The tables in sections 2.1 to 2.3 present all of the interface options and their pros and cons.

The first option in each table is the most basic configuration scenario, where pass payload or tag data is sent from a single VTAP reader to the attached PC over a USB keyboard emulation interface. Copying a firmware file to the VTAP mass storage drive is an easy way to update firmware in this situation, if required.

Sending commands or remote transfer of files to the VTAP reader will require at least one of the USB ComPort, RS-232 serial, RS-485 serial or TTL serial interfaces. These are called the VTAP 'Command Interface(s)'.

Three common implementations are:

1. One of the serial ports is used for sending commands and file transfer, while pass payload is sent over USB keyboard emulation;
2. The USB ComPort is used for sending commands and file transfer, with pass payload sent over a serial RS-232 interface;
3. Pass payload, sending commands and file transfer all use a single type of serial port, in passive mode. (If you are using command-response actions and receiving routine data on the same interface, choose passive mode to avoid the risk of pass reads interrupting other actions.)

Active or passive interface

The USB or serial communication interfaces, can be described as being in either **active** or **passive** mode.

Active mode means that pass payload or card data read by the VTAP reader will be **sent immediately over the interface in use**, whenever it is read. Active mode allows simultaneous bi-directional communication, such as continuous reception of any incoming data from the reader and sending any commands. (Active mode is not therefore suitable for RS-485 where communications are half-duplex.)

Passive mode means that the VTAP reader will only **send data in response to a command** (query), listed in the VTAP Commands Reference Guide. Passive mode is useful if your system or application polls for data to process, rather than reacting to data which could arrive at any time. It ensures that data received can always be treated as a response to the last command sent. Passive mode allows uni-directional communication only, and so is recommended for the RS-485 interface as that is half-duplex. It can also be used, on any of the full-duplex interfaces on your VTAP reader (USB COM, serial RS-232, TTL serial), if required for your application.

Your preference is likely to be driven by your existing systems, and in-house technical expertise.

If you need pinouts and hardware connection information please refer to the Installation Guide for your VTAP reader (available on **https://vtapnfc.com**).

2.1 Interfaces to receive data from the VTAP reader

	USB keyboard emulation	**USB ComPort and Serial (RS-232, RS-485, TTL/3.3V logic)**
Read mode	Active	Active/Passive
Command-response	No	Yes
Pros	•Very simple set up •Easily understood by non-technical staff •No third-party software integration or development required	•Can read pass/card data without the receiving application being in focus on an operating system •Commands can be sent to request data, read or change settings •Dynamic configuration changes possible •Can be integrated in existing custom software/script
Cons	•Requires the receiving application to be open and in focus for every pass read (cannot be in the background). The cursor must be in the correct position to receive keyboard input. •As keyboard responses differ between UK/US and Apple keyboards some characters may change with operating system keyboard language settings, requiring a special keyboard map (see KBmap setting).	•Equipment reboot needed to initiate USB ComPort

2.2 Interfaces to send commands or transfer files (firmware, config or keys) to the VTAP reader

	USB mass storage emulation	**USB ComPort and Serial (RS-232, RS-485, TTL/3.3V logic)**
Pros	•Very simple set up •Easily understood by non-technical staff •USB connection on most PCs •Updating firmware requires simple copy/paste operation •Config.txt file (and LED.ini file on VTAP50) can be opened and edited simply in a text editor such as Notepad	•Serial connections widely used in industry •Remote file transfer can be done easily using Zmodem, without the operating system file manager •More suitable for scenarios with multiple VTAP readers connected •RS-485 ideal for long cable run (up to 1200m)
Cons	•File transfer can be done locally over USB connection •Detecting addition or removal of mass storage not always reliable on Windows - difficulties for automating file transfer •May be file caching issues where a deleted file is only detected by the operating system on reboot/remount	•RS-232 is only suited to short cable runs (up to 15m) •RS-485 is half-duplex, which means communication is in only one direction at a time.

2.3 Available interfaces on the VTAP models

The following table illustrates available interfaces on all the VTAP models

VTAP model	Keyboard wedge/ barcode emulation	USB ComPort	TTL Serial	Serial RS-232	Serial RS-485	Bluetooth keyboard
VTAP50-OEM	Available	Available	Available	Available	-	-
VTAP100-OEM	Available	Available	-	Available	-	-
VTAP100-USB-CC	Available	Available	-	-	-	-
VTAP100-PAC-485-CC	Available	Available	-	Available	Available	-
VTAP100-PAC-W	Available	Available	-	Available	-	-
VTAP100-PRO-BW	Available	Available	-	Available	-	Available

3 Receiving pass payload from a VTAP reader

Receiving pass payload (and card data) and sending it over one (or more) of its interfaces is core functionality for a VTAP reader. You should decide which interface(s) will receive this data, and whether that data should be sent automatically whenever a pass or card is read (active mode) or whether another system is responsible for polling the reader on a regular basis to request data (passive mode).

Although there is no restriction on the type of system that can be connected over the interface you choose, there may be electrical safety measures to be observed in the hardware, such as using rated power settings for the interface and the VTAP reader, grounding on both sides, and cable shielding. You must ensure the commands used by the connected system respect warnings or cautions that appear in this Integration Guide or the VTAP Commands Reference Guide. (The next section discusses the commands to use if you want to poll for data.)

The key advantage of choosing any option other than USB keyboard emulation, is that the data can be read in the background and does not rely on a system cursor being present in an entry box. It can also avoid the need for complex keyboard input handling code, or setting the keyboard input language.

The configuration settings to send pass payload over USB keyboard emulation or send pass payload over a virtual COM port are included in the VTAP Configuration Guide.

The settings to use other interfaces all follow the same basic pattern as those for keyboard emulation, where `<interface>Mode` enables or disables the interface.

Most of the same settings are then used, with the appropriate `KB`, `ComPort`, `Serial`, or `Serial2` prefix, to determine which data to read, from which passes, cards or tags. There are then a few extra settings specific to each interface. Refer to the VTAP Commands Reference Guide for the specific settings to use with different interfaces.

This Integration Guide discusses the integration and operational aspects of using the serial interfaces, including RS-232 and RS-485. It includes using the VTAP interfaces for sending commands, which may be required for your integration and custom software development.

3.1 Send pass payload over a serial RS-232 or TTL interface

The VTAP reader will, by default, send pass payload or a command response to the connected system (such as a PC) over any serial RS-232 or TTL interface.

Physical RS-232 or TTL connector

The RS-232 interface is available on the captive USB/RS-232 connector on VTAP100-OEM and VTAP50-OEM. The TTL serial interface is only available on VTAP50-OEM on the 12-pin expansion connector (if fitted). Refer to the Installation Guide for your VTAP reader for wiring serial RS-232 and TTL interfaces.

Note: RS-232 and TTL serial present as different physical interfaces but use the same UART, so only one of these interfaces should be used at a time, if both are available.

The VTAP reader can communicate in either **Active or passive interface?** mode over a serial RS-232 interface. The settings in this section assume active mode, as this is the default. See **Use the virtual COM port in passive mode** for additional settings and considerations if you need to use passive mode.

The following list describes optional settings. You may need to use some of them to suit your VTAP reader application:

- The RS-232 interface is enabled by default. (It can be disabled by setting `SerialMode=0` if required.) The default serial port settings are 9600 baud rate, no parity, 8 data bits and 1 stop bit, but this can be changed to match your system, in `config.txt`, using `SerialSettings` and defining the parameters in the sequence `= <baud rate>, <parity>, <data bits>, <stop bits>`, such as `SerialSettings=115200,n,8,1`.

- Use the setting `SerialSource` in `config.txt` to restrict the types of data to be passed through the serial RS-232 interface, if needed. The default value is `=A5`, which allows every mobile pass or card/tag read, scanner input, all commands and interface messages.

SerialSource values

You restrict the types of data passed through the serial RS-232 interface by setting a hex value for `SerialSource` in `config.txt`. The most common values are:

- `=A5` (default) allows every mobile pass, card/tag read, scanner input;
- `=83` for all mobile pass, card/tag reads, all commands and interface messages;
- `=80` for mobile pass reads only.

The values come from a bitwise combination of the following hexadecimal values:

FIRST HEX DIGIT				SECOND HEX DIGIT			
BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
0x80	0x40	0x20	0x10	0x08	0x04	0x02	0x01
Mobile wallet pass	-	Write to card emulation mode	-	-	Scanners	Command interface message	Card/Tag UID

So, for example setting SerialSource=80 (hex) will send only NFC mobile pass data to the serial interface because 80 corresponds to choosing 1000 0000 in the table above.

- You could add extra prefix or postfix characters (up to 16 characters) to the data being sent, using `SerialPrefix` or `SerialPostfix` settings. (Only when the serial interface is used in active mode).

SerialPrefix and SerialPostfix syntax

- URL encoding of ASCII characters is supported.
- Use `$t` command to return the pass type (A, G, 0, 2, 4, or 6, E, Q , X or W) together with the payload.

  0 = MIFARE Classic
  2 = NFC Forum type 2 / MIFARE Ultralight / NTAG
  4 = NFC Forum type 4 / DESFire
  6 = NFC Forum type 5 / I-CODE
  A = Apple VAS pass
  G = Google Smart Tap pass
  E = Card emulation (when in write back mode)
  Q = QR / Barcode scanner
  X = Apple Wallet Access iPhone
  W = Apple Wallet Access Watch
- `$n` will add VAS merchant ID / Smart Tap collector ID index (1 to 6) and the keyslot number used (1 to 6). For cards/tags, QR/barcode or card emulation, `$n` will return no value `--`.

Example: Using prefix and postfix on Serial RS-232 interface for pass and card/tag payload

```
SerialPrefix=%0a%0d

SerialPostfix=%20Type:$t,Slot:$n
```

This will result in Google pass output:

```
(New line)
3|F6I53ZaPv2Ys2PAb|Employee Name|Dot Type:G,Slot:16
```

or NFC Type-2 card output:

```
(New line)
0473816A831E80 Type:2,Slot:--
```

- `SerialPass...` settings allow you to extract only a part of the full pass payload, if preferred. Refer to the VTAP Commands Reference Guide for more about these settings.
- A start-up message is set up on the VTAP reader, which defaults to "VTAP100/<firmware version>\r". Change this using the setting `SerialStartup` in `config.txt`.

After changing these `Serial...` settings in `config.txt` and saving it (if something other than default behaviour is required), you can start reading pass payloads over the serial RS-232 interface, in either active or passive mode.

You can continue to modify or add any of the `Serial...` settings by sending them as a command over the serial RS-232 interface. This is useful if you need to change a setting value on a VTAP reader which is not accessible over USB. Section **5** details how to **Use the virtual COM port in passive mode**, and important factors to bear in mind when using this approach.

3.2 Send pass payload over a serial RS-485 interface

RS-485 is a half-duplex interface, so the VTAP reader should be set to passive mode to avoid contention when bi-directional communication with a host system is required.

In passive mode the VTAP reader connected over RS-485 becomes subordinate to a system, such as a PC, which is the master. The PC (master) sends a command (query) to the VTAP reader and waits for a response. The PC can send a serial command to the VTAP reader such as `?r` to retrieve the last pass payload, or `?t` to retrieve the type of the last pass read. The VTAP reader only responds to incoming commands.

Physical RS-485 connector

The RS-485 interface is only available on the VTAP100-PAC-485-CC. Refer to the Installation Guide for your VTAP reader for advice on wiring a serial RS-485 interface.

The following list describes optional settings. You may need to use some of them to suit your VTAP reader application:

- The Serial2 RS-485 interface is enabled by default. (It can be disabled by setting `Serial2Mode=0` and/or `Serial2RS485=0` to disable the Serial2 interface and RS-485 transmission driver respectively, if required.) The default serial2 port settings are 9600 baud rate, no parity, 8 data bits and 1 stop bit, but this can be changed to match your system in `config.txt`, using `Serial2Settings` and defining the parameters in the sequence `= <baud rate>, <parity>, <data bits>, <stop bits>`, such as `Serial2Settings=115200,n,8,1`.
- Use the setting `Serial2Source` in `config.txt` to restrict the types of data to be passed through the serial2 RS-485 interface, if needed. The default value is `=A5`, which allows every mobile pass, card or tag read, scanner input, all commands and interface messages. This setting works in the same way as `SerialSource` for RS-232. Refer to **SerialSource values** in the previous section for more, if it needs to be changed to suit your application.

- You could add extra prefix or postfix characters (up to 16 characters) to the data being sent, using `Serial2Prefix` or `Serial2Postfix` settings. (Only when the serial interface is used in active mode). This setting works in the same way as `SerialPrefix` and `SerialPostfix` for RS-232. Refer to **SerialPrefix and SerialPostfix syntax** in the previous section for more, if this is needed for your application.

Example: Using prefix and postfix on Serial2 RS-485 interface for pass and card/tag payload

```
Serial2Prefix=UID:

Serial2Postfix=%20Type:$t,Slot:$n
```

This will result in Google pass output:

```
UID:3|F6I53ZaPv2Ys2PAb|Employee Name|Dot Type:G,Slot:16
```

or MIFARE Classic card output:

```
UID:0473816A831E80 Type:4,Slot:--
```

- `Serial2Pass...` settings allow you to extract only a part of the full pass payload, if preferred. Refer to the VTAP Commands Reference Guide for more about these settings.

After changing the `Serial2...` settings in `config.txt` and saving it (if something other than default behaviour is required), you can start sending commands (queries) over the serial2 RS-485 interface, to receive pass payloads in response.

To set up the VTAP for passive mode and start sending different commands and receive responses, follow the instructions in the section **Use an interface in passive mode**.

You can then modify or add any of the `Serial2...` settings by sending them as a command over the serial RS-485 interface. This is useful if you need to change a setting value on a VTAP reader which is not accessible over USB.

4 Sending commands to a VTAP reader

You can send VTAP commands to perform various operations including:

- Data requests, such as a command to retrieve the last pass or card/tag payload from the VTAP reader.
- Remote management commands, to set any of the VTAP settings in the main configuration (`config.txt`), or retrieve any of the files stored on the VTAP reader, such as `config.txt` or `boot.txt`.
- Dynamic configuration commands, to perform operations not saved in `config.txt` like enabling/disabling the NFC field to save power, or perform integration tasks, such as consuming newly loaded keys.

Any serial port (USB ComPort, RS-232, or RS-485 serial port) can be a Command Interface, if your VTAP reader hardware permits.

You can send the commands to the VTAP through any terminal emulator such as TeraTerm or PuTTY, or your own custom software.

Note: End any command sent with a <CR> or <LF>, but not both.

By default, with `CommandInterfaces=7`, sending commands is enabled on all available serial interfaces of the VTAP reader. You can change this value to restrict which interfaces can be used to send commands: `CommandInterfaces=1` for USB ComPort only, `=2` for Serial RS-232, and `=4` for Serial2 RS-485. You add these values together to enable commands on multiple interfaces.

You might use the Command Interface on one communications port to control and configure the VTAP reader in passive mode, while receiving pass and tag data on a different port in active mode.

If you use the Command Interface on any serial port that is also set to receive pass or tag data, we recommend you enable **passive mode**. This avoids potential for confusion between command responses and pass or tag data. When a serial interface is set to use passive mode, your application will need to poll the VTAP reader (by sending an `?r` command) to retrieve pass or tag data. This makes half-duplex operation possible (for example when using RS-485).

When developing your own software, use the data request commands to pull data from the VTAP reader, dynamic commands to make changes in VTAP reader behaviour to suit your system and connected hardware, and use remote management commands to modify `config.txt` if a setting is to be permanently changed in the VTAP reader.

The following sections discuss the use of different types of commands, during integration or operation, over the ComPort, RS-232 or RS-485 interfaces. The full list of possible commands is provided in the VTAP Commands Reference Guide.

4.1 Data request commands

These can be used to retrieve data (a Get operation) from the VTAP reader. Some of the most common data request commands are listed below. The full list of possible commands is provided in the VTAP Commands Reference Guide.

- Use `?r` when in **passive mode** to request the NFC pass or card/tag payload last read. The tap data will be sent as long as it was read within the last `InvalidDataCacheMS` period and the request comes from an interface with a `...Source` setting that permits this type of data to be sent over that interface.
  Up to three serial interfaces can request and receive this cached data for a single tap, at different times, up to the `InvalidDataCacheMS` period. The data will not be overwritten by a more recent tap until all potential interfaces have read the tap data or the `InvalidDataCacheMS period` is reached. [Note: On hardware VTAP100 v4 or earlier, there is only one cache, so there is a greater risk that the data could be overwritten by a more recent tap before it has been read on all interfaces].

- Use the `?type` command to request the type of the NFC pass last read, along with VAS merchant ID/ ST Collector ID and ECC key slot. For cards/tags `?t` is more suitable. This command is also used in **passive mode** only.

  Example: Send type? command
  within 3000ms after a successful Google pass read

  ```
  ?type
  ```

  This will result in a VTAP response, such as:

  ```
  G16
  ```

  This shows pass type is `G` for Google, Collector ID is `1` and ECC key slot used is `6`.

  Note: Always use `?type` to extract the pass type when in passive mode. In passive mode, you cannot set a prefix/postfix for your payload to output the payload using `?r` with `$t$n` set as prefix/postfix.

- Using the `%<setting>` command to request the current value of any setting that is included in `config.txt`. This command can be used in both **active and passive mode**.

Example: Send %PassLED to get the value for the LED response pattern set for a successful pass read

```
%PassLED
```

This will result in a VTAP response such as:

```
PassLED=00FF00,100,50,2
```

- Send a `?b` command to retrieve boot information from your VTAP reader, as contained in `boot.txt`. This includes model, serial number, hardware and firmware version, connected expansion (if any), and ECC key slots used.
- Use the `?temp` command to retrieve the internal temperature (in Celsius) of the VTAP reader .

Example: Send %temp to check the internal temperature of the VTAP reader

```
?temp
```

This will result in a VTAP response such as:

```
40.41C
```

4.2 Dynamic commands

These commands are used to make dynamic changes in VTAP reader behaviour. They are not saved in the `config.txt`, so they can be used to make frequent changes in the VTAP reader settings without degrading the flash memory. There are a wide range of dynamic commands, some of the most common are listed below. The full list of possible commands is provided in the VTAP Commands Reference Guide.

- Change the current mode of the VTAP reader to active or passive, using `?a` and `?p` respectively.
- Power cycle the VTAP reader using `?reboot`. (An alternative to saving `reboot` in a `command.txt` file).
- Lock the VTAP reader configuration with a password using `?l` (for example `?l APa55w0rd`) and unlock with `?u` (for example `?u APa55w0rd`).
- Consume any newly added keys into secure storage, from the file system on your VTAP reader, using `?keyload`
- Enable or disable card emulation mode using `?cardmode` (for example `?cardmode 1` to enable card emulation mode), and set the card emulation data using `?card <type><:lang>,<data>` (for example `?card text,Hello world`). For more details refer to VTAP Application Notes on NFC tag emulation.
- Send a message from one VTAP reader interface to another, such as from RS-232 (Serial) to RS-485 (Serial2), or from RS-485 (Serial2) to USB COM (ComPort). This can be done using the `>interface:type:message` command. This feature could be used when integrating the VTAP reader into your custom software or script development including actions such as:
    - Trigger an operation on another connected device (through the chosen interface) when your VTAP reader reads an appropriate pass.
    - Send a particular message based on the `<type>` of pass, card or tag read by the VTAP reader, where your application requires different operation according to the type of pass, card or tag.
    - Copy the pass payload to another interface when in passive mode, since using `?r` on any interface will remove the cached payload from the VTAP reader.
    - Send an emulated NFC pass or card UID, during testing, to see if the device connected on a particular interface reacts as required.

Sending a message from one interface to another is only allowed if the receiving interface has the appropriate bit set (0x02) in its `...Source` setting. The default value for all `...Source` settings is `=A5` which does not include 0x02. This ensures that the feature must be explicitly enabled when required. Refer to **SerialSource values** for help with this. To discriminate between a genuine pass data read from the VTAP reader and an interface message coming from the 'source' interface you could use a dummy pass type in the interface message.

Example: Send >interface:type:message command from serial RS-232 to serial2 RS-485

```
>t:a:Pass received!
```

This command sent on the serial interface, will send a message '`Pass received!`' as an Apple VAS pass (`a`) to the device connected to the VTAP reader over the serial2 RS-485 interface (`t`).

- Toggle the NFC field of the VTAP reader on or off using `?NFC 0` to turn off, or `?NFC 1` to turn on, in order to reduce power consumption.
- Dynamically change the active VAS or ST pass data group, using `?VASPasses <slot>` or `?STPasses <slot>` respectively.

Example: Choose a different active VAS or ST pass

```
?VASPasses 3 ; enable the Apple VAS pass VAS3
             ; (if present in config.txt)

?STPasses 4  ; enable the Google Smart Tap pass ST4
             ; (if present in config.txt)
```

- Drive LEDs on the VTAP reader without saving a pattern in `config.txt` using `?LEDR <value>,<on time>,<off time>,<repeat#>`. If you are using a VTAP50 v2, you can also trigger sequences using `?LEDR 0000:seq.<name>@<target filename>`. The target file must be a `.ini` file. Multiple files describing complex LED patterns and sequences can be saved in the VTAP reader file system, and any one of those triggered using `?LEDR`.

Example: Drive LEDs dynamically with ?LEDR command

```
?LEDR FF0000,10,50,3 ; flash red 3 times for 100ms on and 50ms off.

?LEDR 000000:seq.test@led.ini ; trigger the sequence seq.test saved
                              ; in led.ini in the VTAP reader file system.
```

Refer to the VTAP Configuration Guide for instructions on creating LED patterns and sequences. If you are using VTAP50 with external LEDs, there are specific VTAP Application Notes with detailed instructions.

- Drive the buzzer using the command `?BEEPR <on time>,<off time>,<repeat#> [,<frequency>][:<section_name>@<text_file> for beep sequence]`. For single a beep, use `?BEEPR <on time>`.

Example: Drive LEDs dynamically with ?LEDR command

```
?BEEPR 200,100,2 ; beep the buzzer twice,
                 ; each time 200ms on and 100ms off.

?BEEPR 100 will beep the buzzer once for 100ms.
```

4.3 Remote management commands

These commands can be used to remotely configure VTAP reader settings, without the need to connect the VTAP reader over USB and use its mass storage.

- Modify or add settings in `config.txt` using `$<setting>=<value>`.

Example: Send $PassBeep to set the buzzer response on successful pass read

```
$PassBeep=100,100,2
```

This will result in the VTAP response:

```
OK
```

Note: Any update to `config.txt` using a `$<setting>` command will cause the USB file system to be remounted. This is forcing the operating system to refresh its file system cache, and will appear as if the mass storage drive has been momentarily removed then reconnected.

CAUTION: Frequent changes in `config.txt` causes flash memory degradation. If your application requires frequent temporary configuration changes, use dynamic commands to avoid frequent updates to the `config.txt` file.

- Use `!<filename>` to retrieve files saved on the VTAP reader to any connected system, over Zmodem. When this command is sent to the VTAP reader, it will start polling for the 'file transfer initiation acknowledgement' from your connected system, by sending a header. When TeraTerm, PuTTY, or your own custom software, acknowledges the first header, a Zmodem file transfer will begin and the file will be transferred. Use this command when you need to modify or add many settings in `config.txt`, or change patterns/animations in `leds.ini`. You could also use this setting to make a backup copy of the current `config.txt` or `boot.txt`.

5 Use an interface in passive mode

Passive mode allows you to implement a master-slave architecture, which may be needed to support half-duplex physical connections or improve security, by preventing continuous transmission of pass payloads.

In passive mode the VTAP reader will only respond to queries from the master system. USB COM, serial RS-232 and TTL serial are full-duplex, so can optionally use passive mode. Serial RS-485 is half-duplex, so can only use passive mode.

If you want to receive the pass payload collected by the VTAP reader into a system connected by a serial interface, but only when requested, you will need to set that Command Interface to passive mode. This can be done by setting values of `CommandInterfaces` and `PassiveInterfaces` to suit your application. `CommandInterfaces=7` is the default, which means all interfaces are enabled.

By default, `PassiveInterfaces=0`, and all interfaces are active. You change this value to choose which interfaces are in passive mode: `PassiveInterfaces=1` for USB ComPort only, `=2` for Serial RS-232, and `=4` for Serial2 RS-485. You add these values together for multiple command interfaces in passive mode.

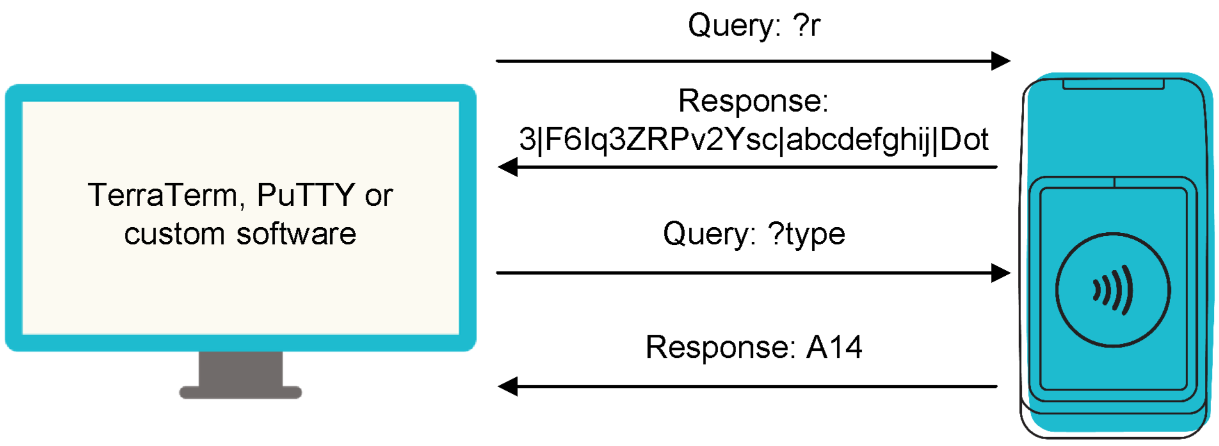
Example: Setting Serial RS-232 only as the command interface in passive mode

```
!VTAPconfig

CommandInterfaces=2 ; Enable Serial RS-232 only as command interface
PassiveInterfaces=2 ; Switch Serial RS-232 only in to passive mode
```

You can send any of the commands listed in **Sending commands to a VTAP reader**, or others in the VTAP Commands Reference Guide, over a serial interface to the VTAP reader. They can extract data, get or set VTAP configurations, or transfer files to and from the VTAP reader as required. You do this using a program such as TeraTerm or PuTTY, or your own custom software or script.

The diagram below is an example of query-response exchange between the VTAP reader (in passive mode) and a connected system (here a PC). This exchange retrieves the pass payload and pass type, when an Apple VAS pass is successfully presented to the VTAP reader:


Query: ?r
Response:
3|F6Iq3ZRPv2Ysc|abcdefghij|Dot
Query: ?type
Response: A14
TerraTerm, PuTTY or custom software

If you want to switch between active and passive modes over, for example, Serial RS-232 or ComPort, use `?a` for switching to active mode and `?p` for passive mode. If changes will not be frequent, you could use a `$PassiveInterfaces` command to disable or enable passive mode on a particular (or all) interface(s).

Note: All `Serial...` settings are valid in passive mode except `SerialPrefix` and `SerialPostfix` settings (and the equivalents for other interfaces). For this reason, if you have set `$t` to retrieve pass type with each pass read as a prefix/postfix in active mode, you must use `?type` or `?t` to separately retrieve this data in passive mode.

6 Transfer files over a command interface

If you need to manage a VTAP reader at a remote location or you want to update many devices, it becomes impractical to simply access the VTAP reader as a mass storage device on a PC, to upload new files. Also, on many Android builds, the USB mass storage FAT12 format is not supported by default, which means a custom module has to be added to the operating system before files can be transferred.

To avoid these difficulties you can use the Zmodem protocol to transfer files, such as a firmware update, key update or configuration file. By default the VTAP reader is always listening for Zmodem commands on any ports where the Command Interface is enabled. There are two ways that you might make use of the Zmodem file transfer facility:

Option 1: Use a free terminal emulation application such as TeraTerm or PuTTY to transfer a file

- **To send a file:** Simply choose to 'transfer a file', select the 'Zmodem' option and select the file to send. TeraTerm or PuTTY-nd are examples applications to use on Windows. Similar applications exist for other platforms. You simply choose to transfer a file, select the Zmodem option and select the file to send.

  Note: You need to use PuTTY-nd, Le Putty or extraPuTTY, because the original version of PuTTY does not support Zmodem

  Note: In TeraTerm, use File > Transfer, not File > Send. You may want to edit the ZModem sending parameters `ZmodemDataLen=<data sub packet length in bytes>` and `ZmodemWinSize=<window size for sending in bytes>`. These default to larger sizes for faster sending, but the VTAP has very low data processing capabilities, so these numbers should be set to 100. If you need to send a receipt set `ZmodemRcvCommand=rz`, and adjust timeouts `ZmodemTimeouts=10,0,10,3` (where the first number relates to the serial port, second the TCP/IP port, 3rd initial packet, 4th final packet).

- **To receive a file:** Use the command `!<filename>` over any serial port on which commands are allowed (by the `CommandInterfaces` setting). Then run 'Zmodem receive' in the client. The VTAP reader will continue sending the file until it is acknowledged.

Option 2: Implement your own code including actions that trigger a Zmodem transfer

It is expected that developers would be more likely to implement their own code, which includes actions that trigger a Zmodem transfer. We generally recommend the use of our

Zmodem Developer Toolkit and the serial command interface to customers who are building a deeper integration between their systems and our VTAP readers.

The following code examples are currently available on request, to help you take advantage of the Zmodem file transfer facility:

- VTAP Serial Interface and File Transfer API C/C++

  C language source and header files implementing a serial communications interface to the VTAP reader. This provides easy access to the full functionality of the VTAP command interface and the ability to send and receive files to/from the VTAP file system.

- VTAP Android Java Developer Toolkit

  Java classes that provide a lightweight interface to the VTAP command interface via the USB virtual COM port. An example Android app that demonstrates the use of these classes is included. This provides easy access to the full functionality of the VTAP command interface and the ability to send and files to the VTAP100 file system using the ZModem protocol.

- ZmTest - Zmodem Command Line Test Utility

  A Windows command line test tool for file transfer between Windows and the VTAP reader. It can be used to transfer files, such as firmware or `config.txt` between the VTAP reader and a connected Windows system. It can also be useful to test Zmodem functionality when implementing a serial communications interface to the VTAP reader using the other toolkit resources (Java Developer Toolkit and C language source code).

To obtain these developer toolkits, or if you need other examples in C/C++/Python/Java contact **vtap-support@dotorigin.com** and we will do our best to help.

We recommend that you set the interface you are using for a Zmodem transfer to passive mode, or receive routine pass (or card/tag) read data on a different interface. This is because user taps could interrupt other operations.

7 About this Integration Guide

This Integration Guide provides specific details about choosing between and using the wide variety of interfaces on different VTAP models.

The main documents available to support your use of the VTAP50 and VTAP100 are the Installation Guide for your VTAP reader model and the VTAP Configuration Guide. You will find the latest versions of these, and other useful information at **https://vtapnfc.com**.

If you need further help do contact us by email to **vtap-support@dotorigin.com**, or by phone +44 (0) 1428 685861 from Europe and Asia, or +1 (562) 262-9642 from Northern and Latin America.