QuickChip Keyboard - Implementation & Integration FAQ

Premise:

This is a FAQ for what QuickChip Keyboard is, how the implementation typically goes, and several certification related scenarios.

This is intended for more technical related folks.

For more information, please consult our support team at support@idtechproducts.com

What is QC KB?

QC KB = Quickchip keyboard emulation. This allows a traditional POS that uses keyboard input to be compatibale with some of IDTECH's newer device offerings that can output both magstripe and EMV compliant data as keyboard strokes.

This is typically used in conjunction with a VPOS - virtual POS. These types of POS usually exist in the form of a web page served from a web browser.

Why QC KB?

This reduces the amount of complexity required for integration, and also reduces the time needed for cert. The solution complexity and time complexity drop, and the user experience / setup is easier.

What products support QC KB today?

As of 23 Apr 2018 , the Augusta and VP 3300 devices support QC KB - quickchip keyboard emulation.

How do we deal with the scenario that an ICC card is swiped?

In the case that an EMV (ICC) card is swiped, we recommend the following way to address the scenario:
When the MSR data is received (but not yet displayed), the Virtual Terminal would have to look through the returned data for an ICC flag (which will be either 0 or 1). This is reflected in Bit 5 of Field 8 of the Enhanced Encrypted MSR data. (Check this document for details about that data format, and its various fields.) If the bit is set, indicating that the card has a chip, the Virtual Terminal application would have to stop at that point (before any more of the data is processed and/or sent), and then display a message to the user indicating that he/she should use the ICC slot instead.

Implementation-wise, the Virtual Terminal Application would first need be able to differentiate between MSR or EMV tag data (they are formatted very differently). Then should the incoming data be MSR data, the Terminal application would look for the status flag to see whether ICC is supported on the card or not, and then make the decision.

How is the user going to be able to make a choice / select the appropriate application to handle the card?

The user won't be able to make a choice, as this decision is handled by the device kernel. At a high level, the card and the terminal would each go down their priority lists, and the transaction would terminate early should there not be the appropriate application supported, for example.There is not an option for a customer to choose which of the supported CVMs they want to use. The card brands determine which CVMs to attempt, the order to attempt them, and whether additional (remaining CVMs) should be attempted if the current CVM is not supported or fails. This process only sets bits in the TVR to indicate pass/failure.

Notet that, in general, the ID TECH implementation of Quick Chip and M/Chip Fast for Augusta in keyboard mode is predicated on the card reader using what we call Terminal Configuration 5C. (See this article for info about terminal configs.) This configuration is for no-special-user-interaction scenarios. When user interaction is required (such as for configuration 2C, also supported in Augusta, but not in keyboard mode), you need to set the configuration manually (in the U-Demo app) or programmatically using the SDK, then implement the Quick Chip choreography yourself, in USB HID mode.

How will a terminal deal with combo cards (containing credit AID + debit pair)? How does the data coming from the reader trigger this type of user interaction (to select credit/debit)?

The terminal setting 5C that we use for the KB QC will defer to the card's highest priority. As the Augusta and VP 3300 doesn't support PIN entry, if the card's highest priority is Debit, the kernel will instead select the next possible priority, which would be Credit, and proceed onward with that. So to fully answer that question: When the Augusta is in Keyboard Quick Chip mode, there won't need to be a trigger for user interaction as the choice is made by the Augusta's kernel before the data is output to, for example, a Virtual Terminal Application. Bottom line, there is no easy, built-in way to support PIN debit with Augusta. We have other products that can handle PIN debit more conveniently.

What are the supported AIDs for the device? What is actually going on in the background, then?

There is no concept of “supported AIDs” as far as the device kernel is concerned. An AID file is just a configuration file, or a data file, with a file name representing an AID name (either partial or full match), and data that will override terminal data or add to terminal data for that transaction. Our kernel as of 20 Mar 2017 supports up to 16 of these files (future rev’s will support more slots).

The kernel doesn’t review or make decision on any AID file nor have any mechanism to know if an AID file is meant for debit, credit or some other function.

Once the device loads the highest priority AID, it will then have the CVM list for that application. After CVM is complete, the TVR (Terminal Verification Results) will have the outcome, if CMV passed/failed and which CVM this happened on. These results are compared to the Terminal Action Codes to determine if the terminal should decline/approve offline (assuming an offline capable terminal) or go online for host approval. In the overwhelming majority of cases, the card will provide "go online" advice.

What if the card requires cardholder confirmation?

Any card that requires cardholder confirmation will be terminated by the device kernel while the Augusta is in QC KB mode.

This means that Quick Chip KB cannot support cards under Durbin requirements, and the user will need to use another card that does not require cardholder confirmation.

However, Quick Chip (the specification) can allow for card holder confirmation. In that case, it’s not as quick, as you need to wait for the confirmation, but it is not prevented. It is similar to CVMs. You can have Quick Chip and support offline PIN. Many of these "alternative scenarios" will need to be implemented yourself, with Augusta in USB HID mode. Out of the box, Augusta in keyboard mode does not support any interruption in the Quick Chip EMV sequence. The entire sequence, from ATR to Gen AC, runs uninterrupted. There is no way to pause it, then start up again.

Our implementation of Quick Chip with Augusta in KB mode is a "No CVM" configuration.

To support Quick Chip WITH card holder confirmation, our customers can use Augusta in HID mode, with configuration 2C. This allows cardholder interaction (Language, what application to pick [Credit or Debit, Common AID or Card brand AID for Debit], …) when there is a need for it.

At a high level - if you require cardholder confirmation, you will need to instead do your own implementation of Quick Chip from the application layer, which all of ID Tech's devices with our L2 Common Kernel do support, in configuration 2C.

Contributors:

Randy Palermo, Eric Lecesne, Kas Thomas, Jason Chiu