IDWedgeBT™ Users Guide
HID/Keyboard Mode
Version 2.9 Software

Copyright

Copyright 2002 - 2013 TokenWorks, Inc. Printed in the USA

Information in this document is subject to change without notice. No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of TokenWorks, Inc.

www.TokenWorks.com (Company Web Site)

www.CardVisor.com, www.IDVisor.com, www.IDScanner.com (Product Web site)

TokenWorks®, CardVisor®, CardTool®, IDVisor® and IDWedge® are registered trademarks of TokenWorks, Inc. IDWedgeBT is a TokenWorks Inc trademark.

Version	Description of Change	Author	Date
1.0	Initial	PC	05182012
2.0	Updates to Config.xml	PC	07272012
2.1	Change wording for accessing SD card	PC	08062012
2.2	Clarify Pause command	PC	09122012
2.3	Integrate changes from firmware rev 1.5	PC	09282012
2.4	Integrate changes from firmware rev 1.6	PC	11262012
2.5	Remove Mode, Profile from Config.xml Improve Magnetic.txt text.	PC	01092013
2.6	Change Data Receive to hand Track1 only loyalty cards, modify Self test for new RN-42 revision	PC	01292013
2.7	Update DLLs and Divide/3 class to improve identification of Mag, CC, ID	PC	02132013
2.8	Add DocIsDate field as '&' in formula	PC	02282013
2.9	Increase ReadFile buffer for long formula	PC	03272013
2.9	BT Conn. LED stays on during SD access Function button displays 1st LED = HID	PC	04112013
3.0	Parse DOB, EXP into mm dd yyyy fields Also add raw data output for Credit cards	PC	04252013
3.1	Add Raw data for Magnetic cards	PC	04262013
3.2	Fix typo for BT Auth, add appendix 3	PC	05202013
3.3	Change power up initialization	PC	05312013
3.4	Fix IL parsing, add 2ndAddr,Country, Rank and 1st initial only, code 39 1D barcode, Change BIT	PC	08272013
3.5	Add ON Health, fix CaC date, FuncKey bool	PC	10072013

1 Introduction

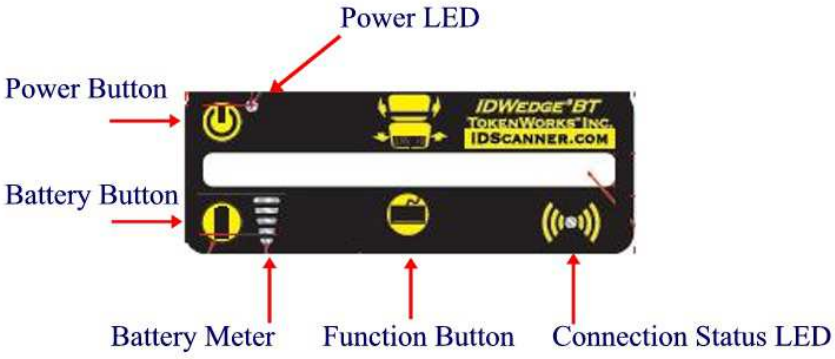
The IDWedgeBT is a barcode/magnetic card scanner that supports a Bluetooth connection to a host computer/tablet and functions as a HID/Keyboard device. Cards are scanned and parsed into fields, and these fields are sent as keyboard strokes to the host. The sequence of fields sent is controlled by formulas stored on the IDWedgeBT SD memory card.

1.1 Document overview

This document will describe the basic functionality of the IDWedgeBT and how to configure the application parameters and formulas used to control the output of data from the scanner.

1.2 Operational overview

The diagram below shows the top view of the IDWedgeBT scanner and the location of the LED indicators and button switches used during the operation of the device.



1.2.1 Power

The Power Button switches the battery power on/off to the unit. Power can be applied by connecting the 12VDC power supply, which charges the internal battery. The USB cable can also supply power, but does not charge the battery. If either of these two cables is connected the device powers on and ignores the state of the Power Button. When the device is powered on, the device will triple beep, and perform a Built in Test. After approximately 5 seconds the unit is ready for Scanning. See Appendix 1 for a description of the Built In Test (BIT)

1.2.2 Battery

The Battery button displays the state of the internal battery, press it once and view the state of the battery on the 5 segment LED. The Battery button serves a dual purpose of providing access to the SD card. To access the SD card, with the power off, press and hold the Battery button, and then power on the device. Continue to hold the Battery button for 3-4 seconds, then release it. When the USB cable is connected to a PC, the IDWedgeBT will now appear as an external USB memory device, and the Connection Status LED will stay lit.

NOTE: The USB cable supplied with the Scanner supplies operational power. The preferred method of accessing the SD card is to hold the battery button down, and plug in the USB cable to the PC; this will avoid having to press the power button.

1.2.3 Function Button

The Function button is used to toggle the built in keyboard on the host computer. If you press and hold this button during power up, you will put the device into programming mode and the device will not be operational, this mode is used to re-flash the firmware of the IDWedgeBT. During normal operation, when you press the Function button, the 1st segment of the 5 segment LED will display to identify the HID application. The toggle feature can be disabled by changing the variable in the config.xml, see 2.1.5.

1.2.4 Connection Status

This LED will flash when the IDWedgeBT has successfully paired with a compliant Bluetooth host. The default host of the IDWedgeBT is Ipad – iOs 5 or later. To pair with other hosts requires reconfiguring the application settings parameters. The default setting is for automatic pairing with iOs devices (iPad, iPhone, etc.). When the unit is set up to access the SD card via USB cable connection to PC, the LED will stay on solid until the next power cycle.

2 Configuration file

The IDWedgeBT maintains configuration file called *Config.xml*. This file stores the application parameters. If the file gets erased, it will be created automatically using default settings on power on. If the file contains editing errors or is corrupted and cannot be opened, then a file called *ConfigError.txt* will be written to the SD card. Once the errors have been corrected and the file can be read, the *ConfigError.txt* file will be automatically erased.

2.1 Config.xml

Below is an example of the contents of the default *Config.xml* file.

```
  <add key=" Version" value="2.9H" />
<!—version string, do not modify. Used to control updates to Config.xml  -->
 <add key="Name" value="IDWedgeBT" />
<!-- Bluetooth name the device shows when pairing -->
  <add key="PIN" value="1234" />
<!-- Pin is the code the device request when pairing -->
  <add key="Auth" value="2" />
<!-- Auth Possible values are 0,1,2,4 -->
  <add key="FuncKey" value="Eject" />
<!-- Keystroke sent when keyboard button is pressed in HID mode -->
  <add key="CreditCard" value="True" />
<!-- Controls how credit card data is parsed output - creditcard.txt  -->
  <add key="1Dbarcode" value="False" />
<!-- Controls how 1Dbarcodes output in HID mode - OneDbar.txt  ->
  <add key="MagTrackCard" value="False" />
<!-- Controls how mag stripes are parsed output in HID mode - Magnetic.txt  -->
  <add key="AddFKIdDL" value="True" />
<!—Add FuncKey after valid ID/DL scan  -->
  <add key="AddFKCC" value="False" />
<!-- Add FuncKey after valid Credit Card scan  -->
  <add key="AddFK1D" value="False" />
<!-- Add FuncKey after valid 1D barcode scan  -->
  <add key="AddFKMag" value="False" />
<!-- Add FuncKey after valid Magnetic Card scan  -->
<add key="MidInitial" value="False" />
<!--Only send first digit of Middle name field  -->
<add key="FirstInitial" value="False" />
<!--Only send first digit of First name field  -->
<add key="5DigitZip" value="True" />
<--Only  send first 5 digits of Zip Code field  -->
<add key="ScanMsg" value="False" />
<--Send out Scan again message if card does not read  -->
<add key="DobYYOnly" value="True" />
<--Only  send last2 digits of YYYY  field  -->
<add key="ExpYYOnly" value="True" />
<--Only  send last 2 digits of YYYY field  -->
<add key="CharDelay" value="5" />
<--inter-character delay X*10Ms (range is 1-20)  -->
<add key="CtrlDelay" value="5" />
<--Control-character delay X*20Ms (range is 1-20)  -->
```

2.1.1 Version

This is the firmware version; it has the letter 'H' appended to the end of the string, to differentiate it from the SPP version of firmware. E.g. 2.9H

2.1.2 Name

This is the Bluetooth name that gets broadcast to the host during pairing and is displayed in the list of Bluetooth devices on the Host.

2.1.3 Pin

This the 4-digit pin code used during the pairing process.

2.1.4 Auth

This is the Authentication setting. It is set to 0 for Serial Port Profile, it also support a value of 1,2,4. Use caution when changing this setting, you may have to reset the host after making changes from the default value of 2.

Auth Value	Description
0	Bluetooth 2.0 Encryption disabled (uses 4 digit pin)
1	Bluetooth 2.1 Keyboard I/O (verify 6 digit code)
2	Bluetooth 2.1 Simple Secure Pair (no pin code)
4	Bluetooth 2.0 Pin Code Authentication (uses 4 digit pin)

NOTE: These settings are included for backwards compatibility with legacy systems, iOS/Windows users should use the default setting of 2, Andriod users may need to use Auth=1 if they encounter problems with pairing while using the default value. See appendix 3 for details on Auth setting.

2.1.5 FuncKey

FuncKey defines the keystroke that will be sent when the button is pressed, the default value="Eject" will toggle the iOS built in keyboard. If the value does not equal "Eject" then the keyboard will not toggle when the button is pressed.

2.1.6 CreditCard

CreditCard Controls the output of Credit Card scans

2.1.7 1DBarcode.

1DBarcode controls the output of Code 39 and Code128 barcodes

2.1.8 MagTrackCard

MagTrackCard controls the output of Magstripe cards like Student Ids, AAA cards, etc.

2.1.9 AddFKIdDL

The AddFKIdDL determines if the FuncKey stroke will be sent at the end of an ID/DL card scan.

2.1.10 AddFKCC

The AddFKCC determines if the FuncKey stroke will be sent at the end of a Credit card scan.

2.1.11 AddFK1D

The AddFK1D determines if the FuncKey stroke will be sent at the end of a 1D barcode card scan.

2.1.12 AddFKMag

The AddFKMag determines if the FuncKey stroke will be sent at the end of a Magnetic card scan (Student ID, AAA, etc).

2.1.13 MidInitial

The MidInitial controls the output of the Middle name filed, if set to true then only the middle initial will be output, if set to false then the whole middle name (if available) will be output.

2.1.14 FirstInitial

The FirstInitial controls the output of the first name filed, if set to true then only the first initial will be output, if set to false then the whole first name will be output.

2.1.15 5DigitZip

The 5DigitZip controls the output of the Zip Code filed, if set to true only the first 5 digits of the zip code will be output. If set to false then the entire zip code field will be output, this can range from 5 to 10 digits.

2.1.16 ScanMsg

The ScanMsg controls the output of card error messages, when set to true, you will see messages like “scan again” or “decode error”. These messages can be helpful when developing formulas, but may cause extra operator effort when attempting to fill out forms.

2.1.17 DobYYOnly

The DobYYOnly variable only sends out the last two digits of the YYYY DOB field. If true then DOB_yy outputs YY, if false DOB_yy outputs YYYY

2.1.18 ExpYYOnly

The ExpYYOnly variable only sends out the last two digits of the YYYY EXP field
If true then EXP_yy outputs YY, if false EXP_yy outputs YYYY

2.1.19 CharDelay

The CharDelay controls the amount of time to wait between the transmissions of characters, the range is 1-20 and this number is multiplied by 10Ms. A setting of 0 defaults to 1Ms

2.1.20 CtrlDelay

The CtrlDelay controls the amount of time to wait between the transmissions of Control characters, the range is 1-20 and this number is multiplied by 20Ms. A setting of 0 defaults to 1Ms

2.2 Reading/Writing Config.xml

To Access the Config.xml file, you must boot the IDWedgeBT while holding the battery button as described in section 1.2.2. Once you have connected a USB cable and can explore the contents of the SD memory card, you will be able to read and write the *Config.xml* file. You can use any text editor to change items in the file and save them to the SD card. To ensure you have not made any errors, open up the Config.xml file using an Internet browser, if a browser cannot open the file, then there probably are editing mistakes that must be corrected before the IDWedgeBT can be operational.

3 Formula Files

The Formula used to send fields to the host is stored in a file on the SD card. To Access the Config.xml file, you must boot the IDWedgeBT while holding the battery button as described in section 1.2.2. Each of the 4 card type supported has it's own formula stored in it's own formula file. The 4 formula files are: *Driverslicenses.txt* for ID/DL cards, *CreditCard.txt* for Credit cards, *OneDbar.txt* for 1D Barcode cards, and *Magnetic.txt* for Magnetic stripe cards.

The concept behind the formula is that each parsed field on a card is represented by a single character. Control characters are enclosed with curly braces and appear in between each field in the formula. Below is a simple formula for filling out an address from a Drivers license:

F{TAB}L{TAB}A{TAB}C{TAB}S{TAB}Z{ENTER}

This formula would fill out a form that had text boxes for each item. The **F** represents the first name, the **L** represents the last name, followed by **A** for address, **C** for city, **S** for state and **Z** for zip code. The {TAB} in between each field moves the cursor to the next text box on a form, and the {ENTER} would act as if the user hit the Enter key to submit the data.

NOTE: There must be at least 1 valid field in each formula to create an output message and a field must be the first entry.

The formula file has a maximum length of 500 characters. If your formula is greater than 500 characters it may cause problems. For reference, the formula above contains 38 characters.

3.1 Driverslicense.txt

The Drivers license formula uses capital letter to represent the fields on a drivers license, ether barcode or magstripe cards;

F – First name
M – Middle name
L – Last name
T - Title
A - Address
C - City
S - State
Z – Zip code
B – DOB
E – Expiration
D – License number
H – Height
I – Eye color
O – Class
P – Hair
U – Weight
V- Endorsements
X – Sex
N – Scanner S/N
? - Restrictions
& - Document Issue Date (Available on PDF417 2D Barcodes only)
R – Rank (CaC only)
@ - Country (USA or CAN)
# - 2nd Address field
Q - DOB_mm
W - DOB_dd
Y - DOB_yy (yyyy format via config.xml variable)
J - EXP_mm
K - EXP_dd
% - EXP_yy (yyyy format via config.xml variable)

Example:
```
F{TAB}L{ENTER}C[ ]S[,]Z{ENTER}
```

3.2 CreditCard.txt

The Credit Card formula uses lower case letters to represent the fields on a credit card.

a = Track1 Raw data
b = Track2 Raw data
f = first name,
l= last name
p = PAN or credit card number
y = ExpYY,
m = ExpMM
n = Scanner S/N

Example:
```
f{TAB}l{TAB}p{ENTER}
```

NOTE: Raw data means all the information on a track including the start and end sentinels in ASCII format.

3.3 OneDbar.txt

There is only 1 field on a 1D barcode and that is represented with a lowercase **w**

Example:
```
w{ENTER}
```

3.4 Magnetic.txt

The Magnetic Formula used to send fields to the host and is stored in a file on the SD card. To Access the Config.xml file, you must boot the IDWedgeBT while holding the battery button as described in section 1.2.2

The magnetic parsing formula will allow for two tracks to be parsed and two fields of specific lengths and specific offsets to be parsed per track for a total of 4 fields. Each field is represent by a track number, and offset starting from zero and a length. There are three tracks and each track as 2 fields associated with it, for a total of 6 field designators

The total number of possible field is 6 and each field is mapped to a lower case letter (field designator):
q – Track1 Field1
r – Track1 Field2
s – Track2 Field1
t – Track2 Field2
u – Track3 Field1
v – Track3 Field2

The *Magnetic.txt* must have two lines of data; the first line defines the track number, offset, and length associated with each field. These values are stored as a comma separated string of 10 integer values. Any formula with less than 10 values will be rejected. If you do not need all of the 4 possible fields, then populate the unused track, field and offset with zeros.

Example of first line of Magnetic.txt: 1,2,4,6,9,2,1,5,6,5

(Track 1, offset1 = 2, length1 = 4, offset2 = 6, length9 = 5, track 2, offset1 = 1, length1 = 5, offset2 = 6,lenght2 = 5).

The second line of the Magnetic.txt uses the lowercase field designators, along with the control words in curly braces, to place the fields on the form

Example of second line of Magnetic.txt:
q{TAB}r{TAB}s{TAB}t{ENTER}

This formula will place the four fields identified in the first line on a form with TAB keys in between each field

Below is an example of the Magnetic.txt file for the formula above.
Example:
1,2,4,6,9,2,1,5,6,5
q{TAB}r{TAB}s{TAB}t{ENTER}

If no second offset is required, then these fields are populated with zero
and the unused lowercase field designators are not required

For example, 1,1,6,0,0,2,1,6,0,0
q{TAB}s{TAB}

In version 2.5 and later, Raw data can be extracted from each track by using the value 99 for the length.

If you want to extract the Raw Data from tracks 1 and 2:

1,0,99,0,0,2,0,99,0,0
q{TAB}s{TAB}

in Raw data mode for magnetic cards, if a track does not read, and ScanMsg is set to true then the output will be %NR? (track 1 no read) Or %ND? (track 1 no data)

If ScanMsg is set to false and a track does not read, the output will be just the sentinels, E.G
Track1 no read /no data = %?
Track2 no read /no data = ;?
Track3 no read /no data = %

The Magnetic formula and parsing has been designed to handle cards on any of the 3 possible tracks. The Magnetic formula applies to cards that have 25 or less characters of information encoded on any track (Except old New Mexico ID/DL). The following list of cards is used to identify Drivers License/ID cards, Credit Cards and AAA cards.

Track2 starts with	and a length of	Card Type
;000000	>28	old AZ state ID/DL
;06360	>28	old AK state ID/DL
;10	>36	Newfoundland ID/DL
;20	>36	Newfoundland ID/DL
;34	>28	AMEX Credit Card
;37	>28	AMEX Credit Card
;4	>31	VISA Credit Card
;4290	>36	AAA Membership Card
;4381	>36	AAA Membership Card
;4382	>36	AAA Membership Card
;51	>31	MasterCard Credit Card
;52	>31	MasterCard Credit Card
;53	>31	MasterCard Credit Card
;54	>31	MasterCard Credit Card
;55	>31	MasterCard Credit Card
;5490	>36	AAA Membership Card
;6006	>28	old CA state ID/DL
;6011	>31	Discover Credit Card
;610054	>28	Ontario Health ID Card
;6202	>36	AAA Membership Card
;6360	>28	United States ID/DL
;636005	>28	South Carolina ID/DL

NOTE: AAA membership cards fall into the Magnetic formula, Credit Cards fall into Credit Card parsing, and ID/DL cards are treated as Drivers Licenses.

Each of the above numbers is also tested for the presence of the '=' sign and a length, to help identify the card type. Cards that do not meet the test will be parsed using Magnetic formula. There is one exception and that is the South Carolina ID/DL, that card is not tested for the '=', only the length is tested.

The Location of the '=' sign for Credit cards use a value consistent with a 15 or 16 digit PAN.

Revision information: Rev 1.8 and newer allows magnetic card of less than 10 characters to be parsed, previous versions of IDWedgeBT required that all cards have 10 or more characters encoded.

3.5 Control keys

The control keys are the non-printing keyboard strokes used in between the parsed field in the formula. The examples so far have included the Tab key {TAB} and the Enter key {ENTER}. The format of the control keys is an all-capital key word surrounded by curly braces (except for Pause). Each curly brace must be paired with another (left, right) or the formula will not work. Below is a list of the supported control keys that can be used in the formula. If multiple keystrokes are required, then an integer can be included inside the curly braces with a space between the key word, for example {TAB 2} will output 2 TAB characters. The values for repeating keys are 2 through 9.

{ALT x}	Alt plus key *
{CONTROL x}	Ctrl plus key *
{SHIFT x}	Shift plus key *
{TAB}	Tab key
{ENTER}	Enter
{UP}	Up arrow
{DOWN}	Down arrow
{LEFT}	Left arrow
{RIGHT}	Right arrow
{PGUP}	Page up
{PGDN}	Page Down
{INS}	Insert
{DEL}	Delete
{BACKSPACE}	Backspace
{Pause x}	Pause in (x *100) Ms increments **

*Note: Alt, Ctrl, Shift are used in combination with a single case insensitive character in the range of 0-9 or a-z or A-Z. These commands do not support repeating, so a command like {SHIFT 8} will produce the * character, likewise {CONTROL 1} will be interpreted as the Ctrl key plus the 1 key (Ctrl +1).

** Note: Pause command is case sensitive and must be followed by a space and then a numerical value from 1 –9

3.6 Inserting text

To insert static text into the formula simply enclose the text in square brackets [ ] You can insert a single character or a whole word. If you examine the formula from the *Driverslicense.txt* you can see the brackets are used to insert a comma into the output.

The Length of text is limited to 10 characters. If you exceed 10 characters within the brackets, an error message will be output if ScanMsg is enabled. If ScanMsg is disabled, you may not get any output from the scan.

4 Additional Information and Examples

Tips
Do NOT end the formula's with a tab as this will move cursor off last field and if there are no more fields on the form, then the keyboard may not be able to display because the cursor was moved off the form.

Ensure nothing is trying to pair with the device during power up/Bit test or else the BIT test will fail, see appendix 1.

Example of the Pause command in a formula:

F{TAB}M{TAB}L{TAB}A{TAB}C{TAB}S{TAB}Z{Pause 5}

The above formula prints out a typical contact with name and address and after the last field, which is Zip code; there is a pause for 500 Milliseconds (1/2 second)

NOTE:
If you type Pause with all capitals it will be ignored, if you type in all lower case it will be ignored. If you do not include an integer value preceded by a space it will be ignored.

The pause command should be used sparingly and works best at the end of formulas, it was included to allow for time for the information to fill in the form, prior to launching the keyboard, however it could be used to control the cursor in between fields if the form requires a slower input.

CharDelay and CtrlDelay:
These two variables are stored in the Config.xml file on the SD card. They control the time to wait between sending out the next character (CharDelay) or next control character (CtrlDelay). If the form needs extra time due to field verification or slow host processor, increase the value of the CharDelay and or CtrlDelay. If you need to speed up the output of keystrokes, decrease the value of CharDelay and or CtrlDelay.

Appendix 1 – Power on BIT (Built in Test) Description

On power up, the device will perform a self test and display the results on the 5 bar LED

BIT Sequence

LED test, turn on/off each LED one at a time, then all 5 on/off

LED 1 - SD card Test, verify file create/read/erase

LED 2 - Test ESEEK Communication, verify send/receive hardware signals

LED 3 – Test BT Communication, verify send/receive hardware signals

LED 4 - Read Device.txt and verify BT revision string and ESEEK S/N string

LED 5 – If Bluetooth settings match Config.xml settings then turn on LED 5

Note: If the Scanner is paired or trying to pair during power up, it will fail to light LED 3,4 and 5. This is normal behavior.

If the Config.xml file is erased, the Scanner will triple beep on the next power cycle, this is confirmation that the scanner has received the initialization command and has been configured to read barcodes and magstripe card.

If the Device is being re-programmed to a different functionality (SPP to HID, etc) then the 5th LED will not light on the first power up, as the Bluetooth settings will change.

Appendix 2 – LED interpretation

When the Function button is pressed the bottom LED (LED1) will light up to identify the IDWedgeBT HID application. The IDWedgeBT can run other applications, the HID application will only light up the first LED, and the SPP application will light up LED1 and 2. The IDWedgeBT USB Keyboard application will light three LEDs 1,2,3. This is helpful for all users to identify what application is running on the device.

The Connection Status LED will only light when the device is in “SD Card Access” mode, see section 1.2.2. The LED will stay light until the device has been turn off/on.

Appendix 3 – Auth settings

Authentication can have the values 0, 1, 2, or 4, depending on the mode desired with some limitations on implementation.

0 // With this mode, the module uses Bluetooth version 2.0 NO encryption (open mode). This mode is useful for legacy devices that do not need security. For this mode to work, both devices must support open mode. If either device requests authentication, the PIN code will be required.

1 // In Bluetooth version 2.1, the default is keyboard I/O mode (which is considered as a secure mode). For Android devices, the user is prompted with a 6-digit code and is asked to verify that the code matches on the module. Because the module cannot display a code, simply press OK or Yes on the remote device to authenticate.

2 // This mode corresponds to Bluetooth version 2.1 Secure Simple Pairing (SSP), or just works mode. This mode works with iPhones and PCs, however it may not work appropriately with some Android devices.

4 // This mode is PIN code mode, which forces Bluetooth version 2.0 PIN code authentication. The functionality is for Serial Port Profile only and it does not work in HID mode for iOS devices.