2019/ver 1.0

RFID Reader

HOW TO SET READER BY APPBLATER

RFID reader is prommning through producer Elatec by their configuration software AppBlaster. In AppBlaster user must create or work with firmware, which contain specification control reader. Firmware is configuralbe and save inAppBlaster folder. Could be use for other readers in different machines.

RFID card/chip listing

SUPPORTED TRANSPONDERS (STANDARD)	ISO14443A: LEGIC Advant, MIFARE Classic EV1[1], MIFARE Classic, MIFARE Mini, MIFARE DESFire EV1, MIFARE DESFire EV2[1], MIFARE Plus S, X, MIFARE Pro X[2], MIFARE Smart MX[2], MIFARE Ultralight, MIFARE Ultralight C, MIFARE Ultralight EV1, NTAG2xx, PayPass[2], SLE44R35, SLE66Rxx (my-d move)[2] ISO14443B: Calypso[2], CEPAS[2], HID iCLASS[3], Moneo[2], Pico Pass[3] ISO18092 ECMA-340: NFC Peer-to-Peer, Sony FeliCa[4], NFC Active and passive communication mode, Passive peer-to-peer mode - initiator, NFC Tag 2, 3, 4 ISO15693: EM4x33[2], EM4x35[2], HID iCLASS[3], HID iCLASS SE/SR[3], ICODE SLI, LEGIC Advant, M24LR16/64, SRF55Vxx (my-d vicinity)[2], Tag-it, PicoPass[3] LEGIC Prime: LEGIC Prime 125 kHz, 134.2 kHz: AWID, Cardax, CASI-RUSCO, Deister[5], EM4100, 4102, 4200[6], EM4050, 4150, 4450, 4550, EM4305[7], FDX-B, EM4105, HITAG 1[8], HITAG 2[8], HITAG S[8], ICT[7], IDTECK, Isonas[7], Keri, Miro, Nedap[5], PAC, Pyramid, Q5, T5557, T5567, T5577, TIRIS/HDX, TITAN (EM4050), UNIQUE, ZODIAC
SUPPORTED TRANSPONDERS (VERSION P)	All Standard Transponders, Cotag, G-Prox[5], HID DuoProx II, HID ISO Prox II, HID Micro Prox, HID ProxKey III, HID Prox, HID Prox II, Indala, ioProx, Nexwatch

Short list of individual steps

1. Determining type of card

1.1. Upload of tracer (default determination of card) firmware to RFID

1.2. Load of card by Hyperterminal application

1.3. Load of card by Terminal application

2. Basic programming for concrete card type (preparation of customer firmware)

3. Uploaded customer firmware to RFID

4. Advance programming (manipulations with number of card, example)

4.1. Data source (UID)

4.2. Output format

4.3. Card number shortening

4.4. Prefix/Sufix

4.5. Bit/Byte Manipulation

5. Image creation and upload to RFID

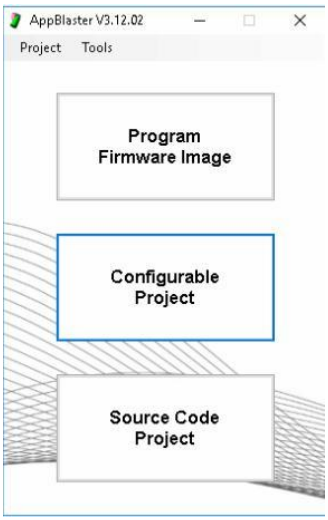
6. Name of configuration file (.bix)

RFID settings

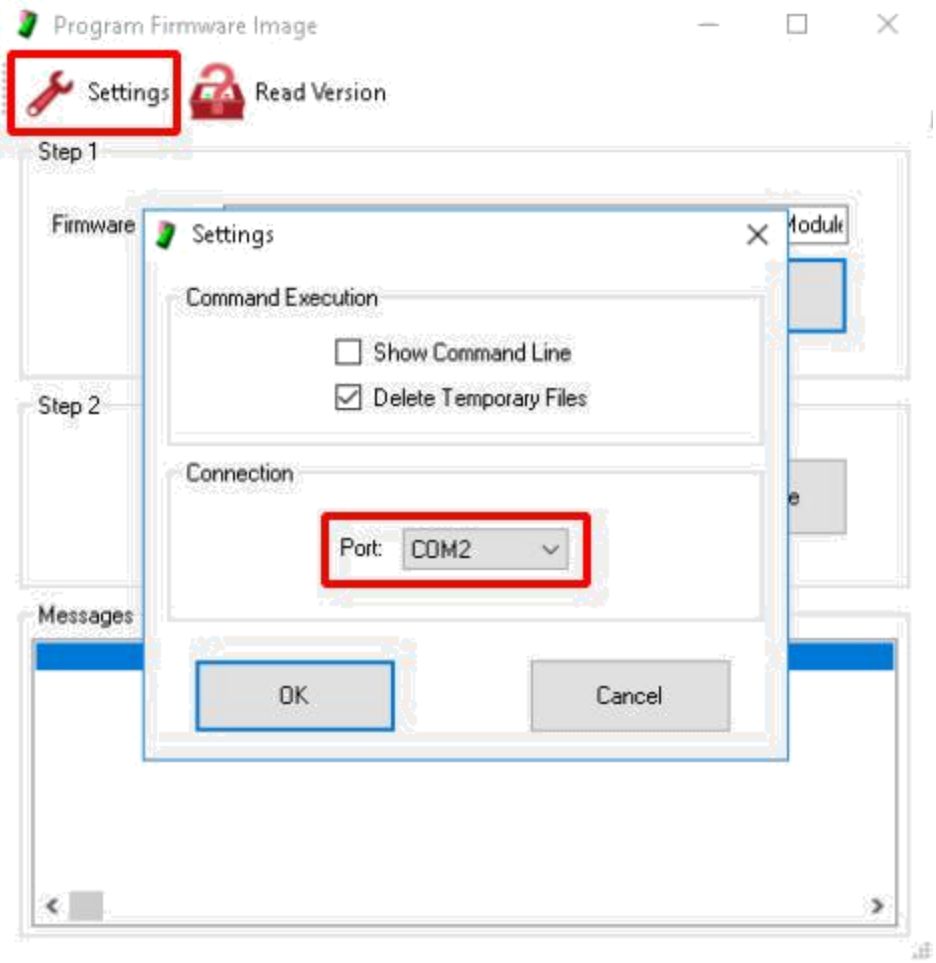
SW Guard must be off !!! Then continue and open AppBlaster.

Path to application:
`C:\Drivers\TWN4DevPack312c\AppBlaster.exe`

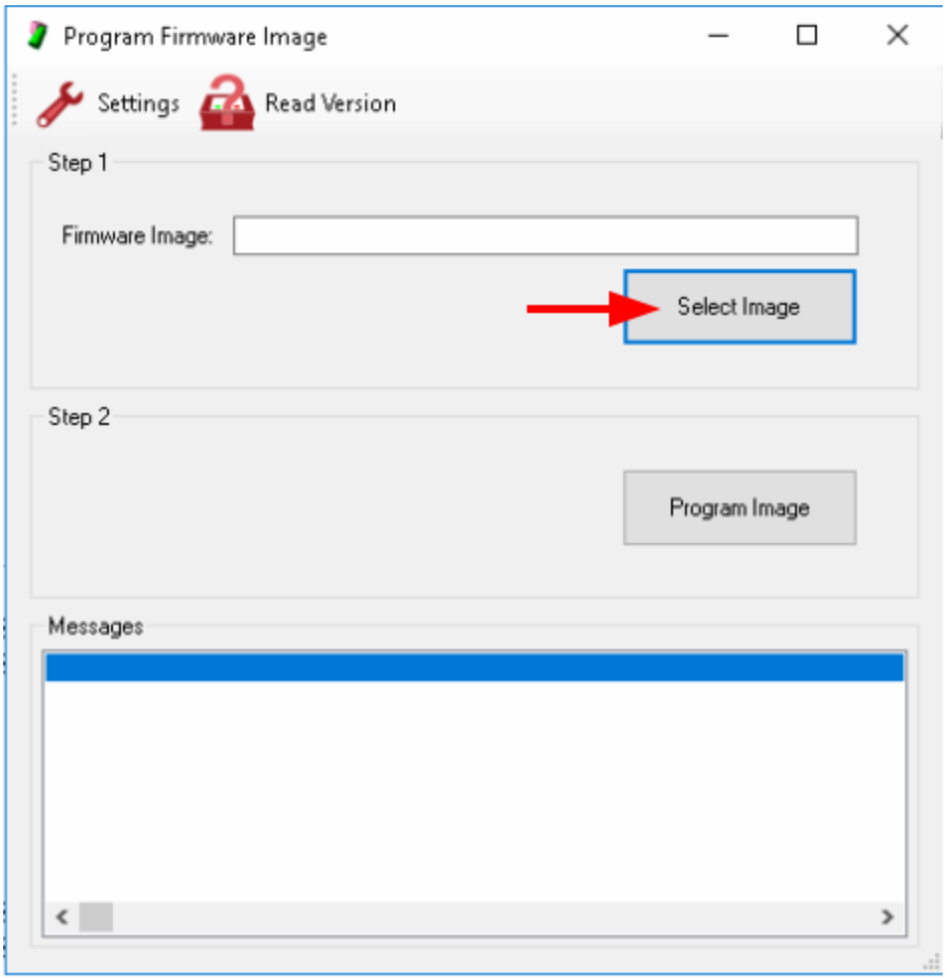
Determining type of card



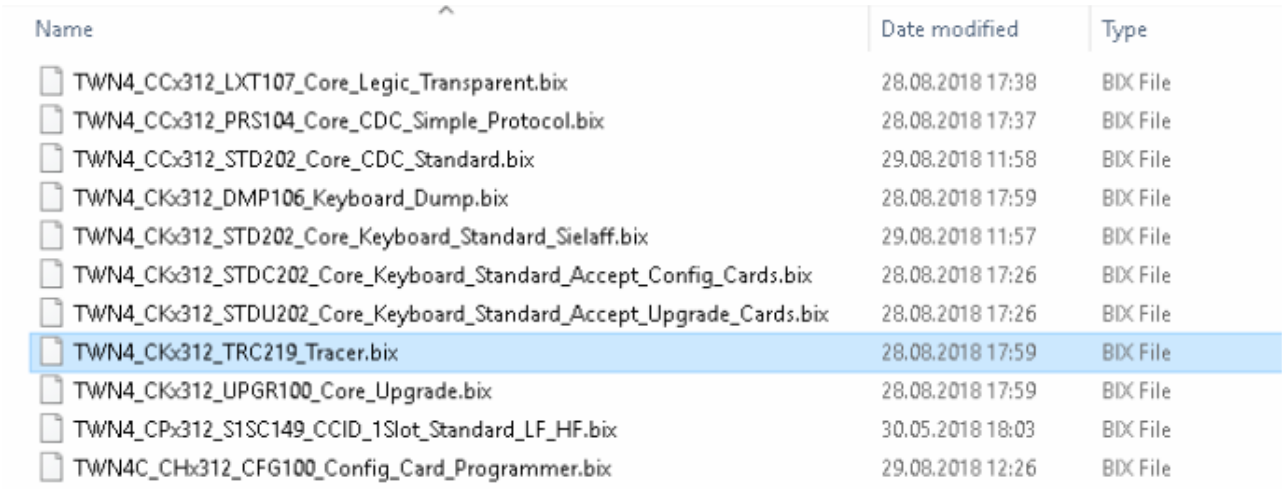
Choose Program Firmware Image



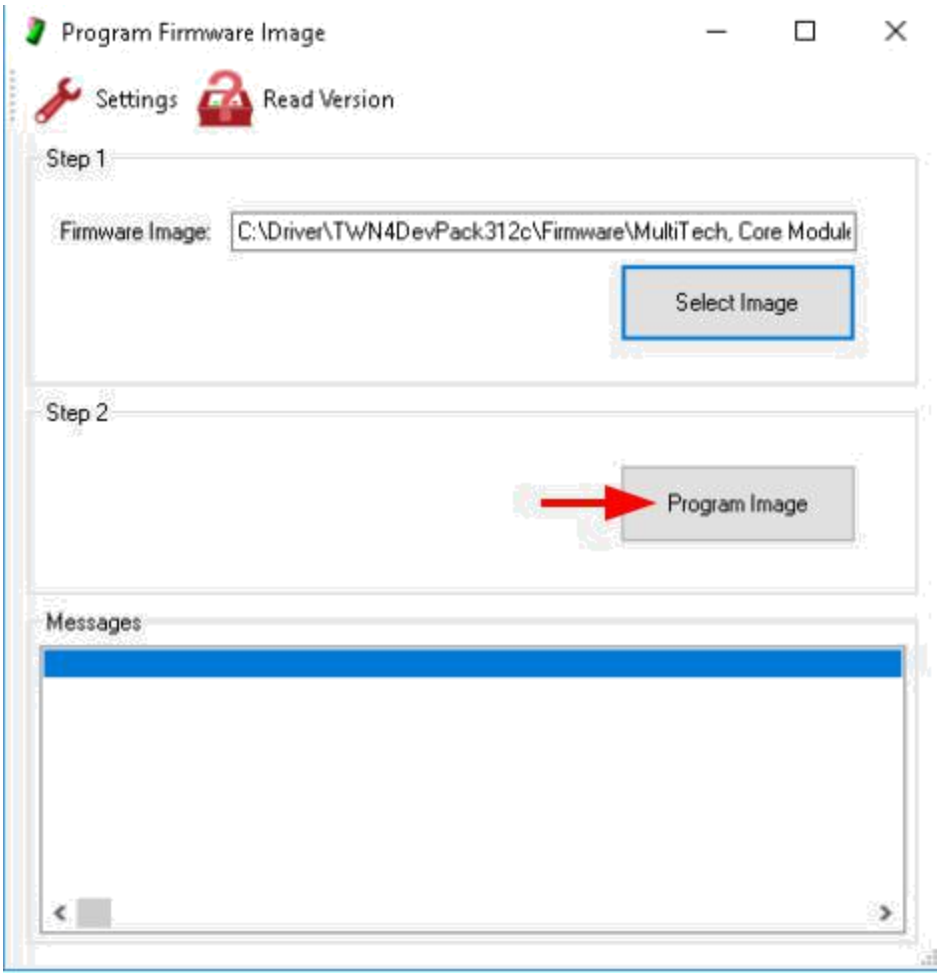
RFID reader is COM2



C:\Driver\TWN4DevPack312c\Firmware\MultiTech, Core Module, Desktop, OEM PCB



Choose Tracer



And Program Image

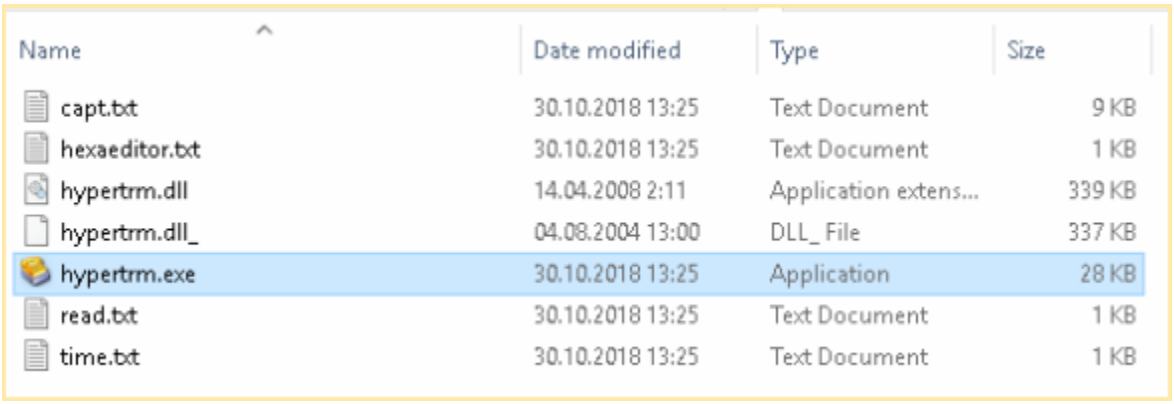
In messages table example extract - Successful programming of firmware

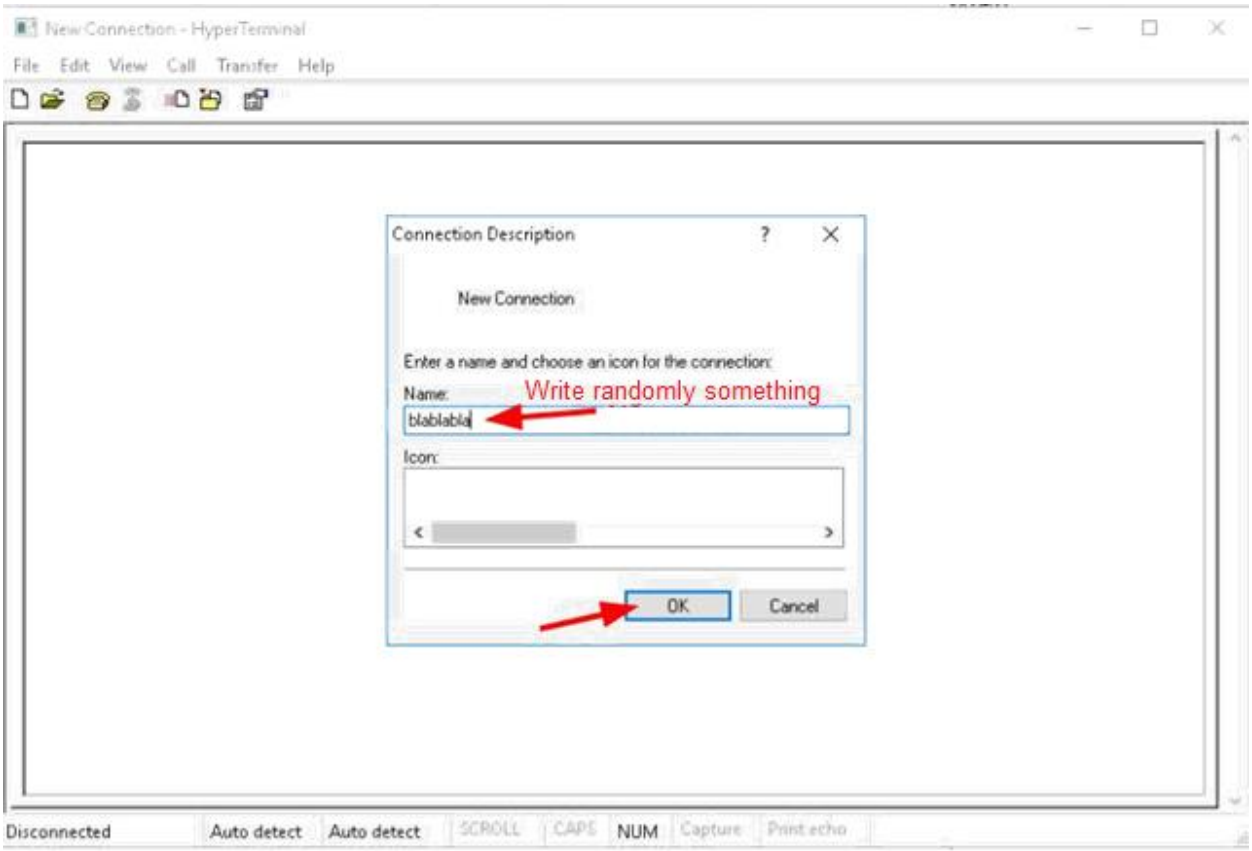
```
------ Program Image ------------------------------------------------
Programming image "C:\Driver\TWN4DevPack312c\Firmware\MultiTech, Core Module, Desktop, OEM PCB\TWN4_CKx312_TRC219_Tracer.bix"
Starting boot loader: OK
Connecting. OK
Bootloader: V1.06 (TWN4 Core)
Firmware before programming: TWN4 Core, V3.12, Keyboard
Check compatibility: OK
Programming Firmware: OK
Programming App: OK
Firmware after programming: TWN4 Core, V3.12, Keyboard
Restarting: OK
Done.
```

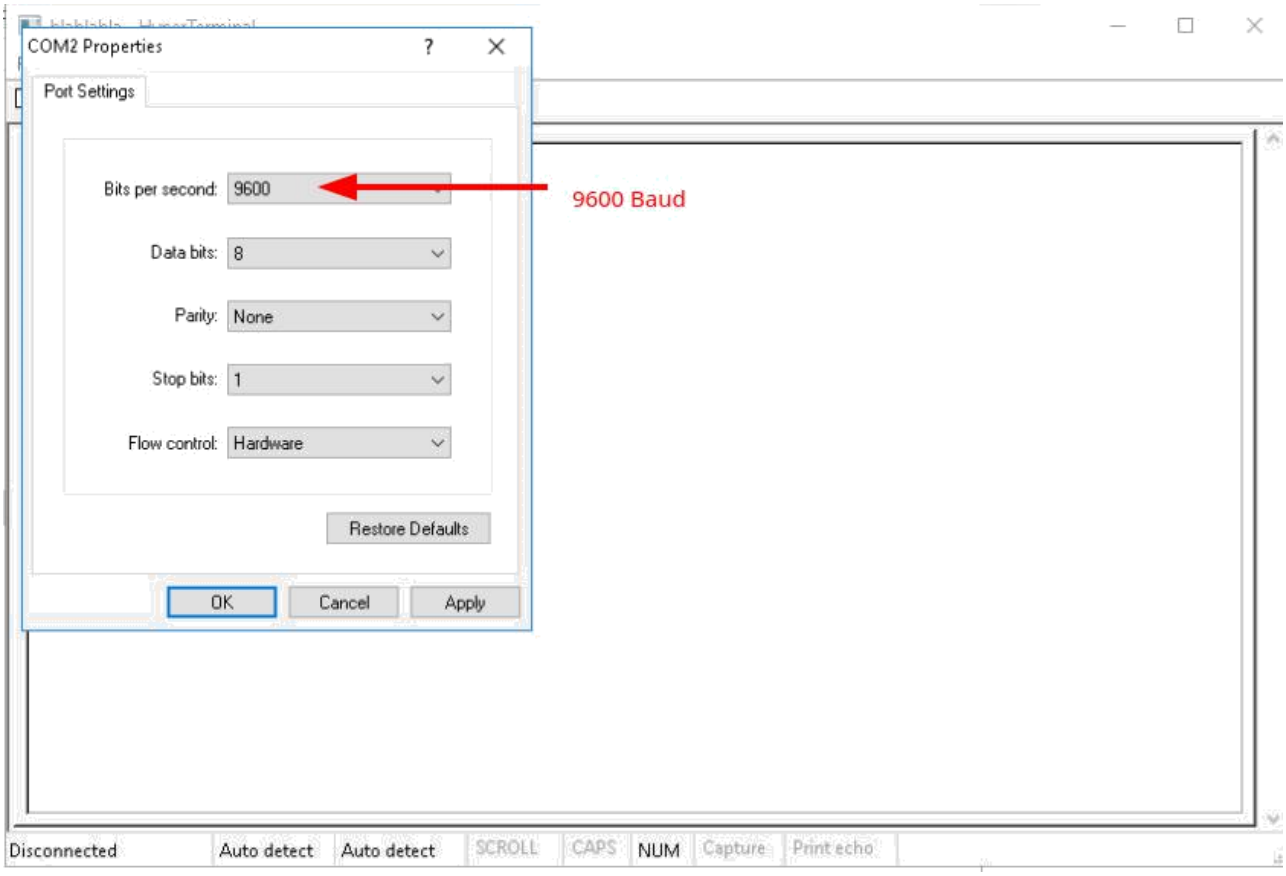
Attach test of card — hyperterminal (older)

Start up application at first. Choose beetwen two versions of Hyperteminal

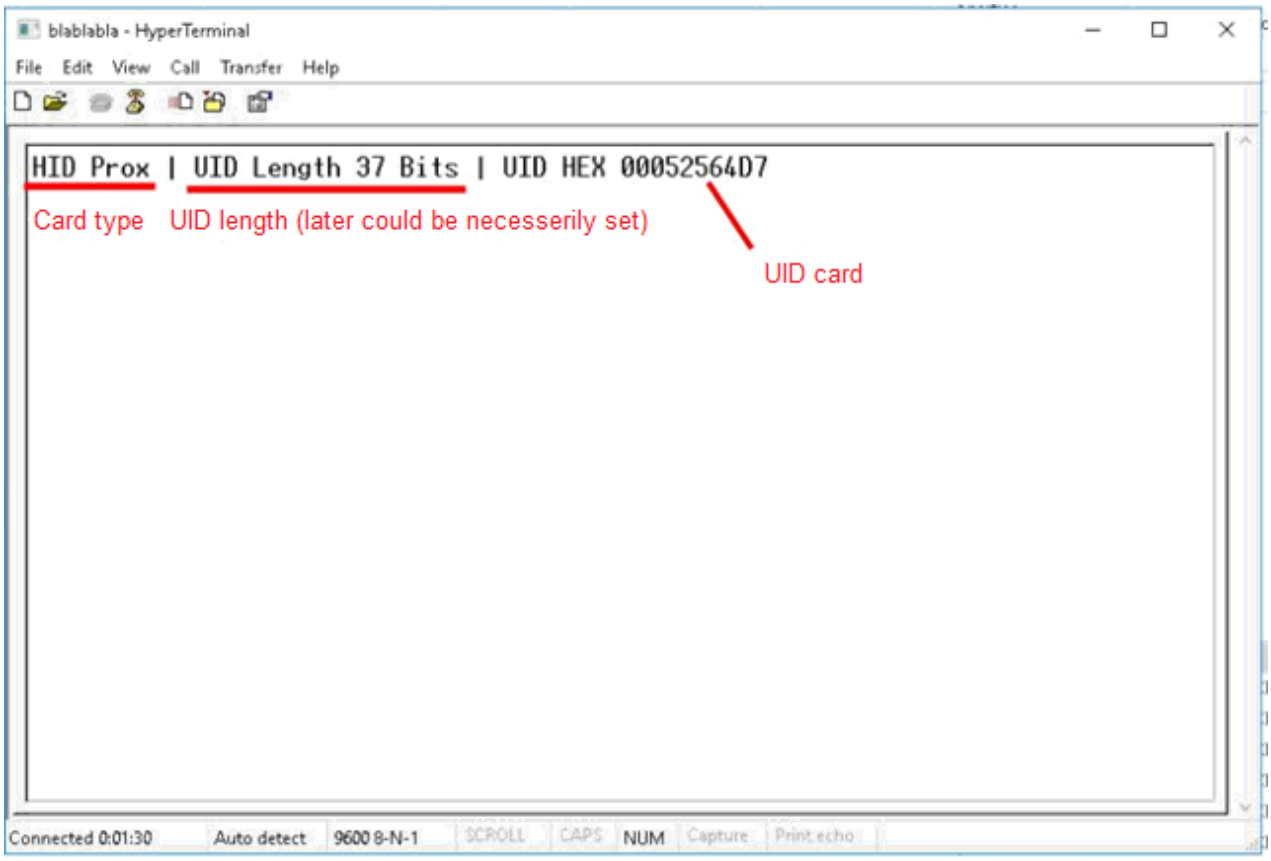
C:\Drivers\hyperterminal.exe







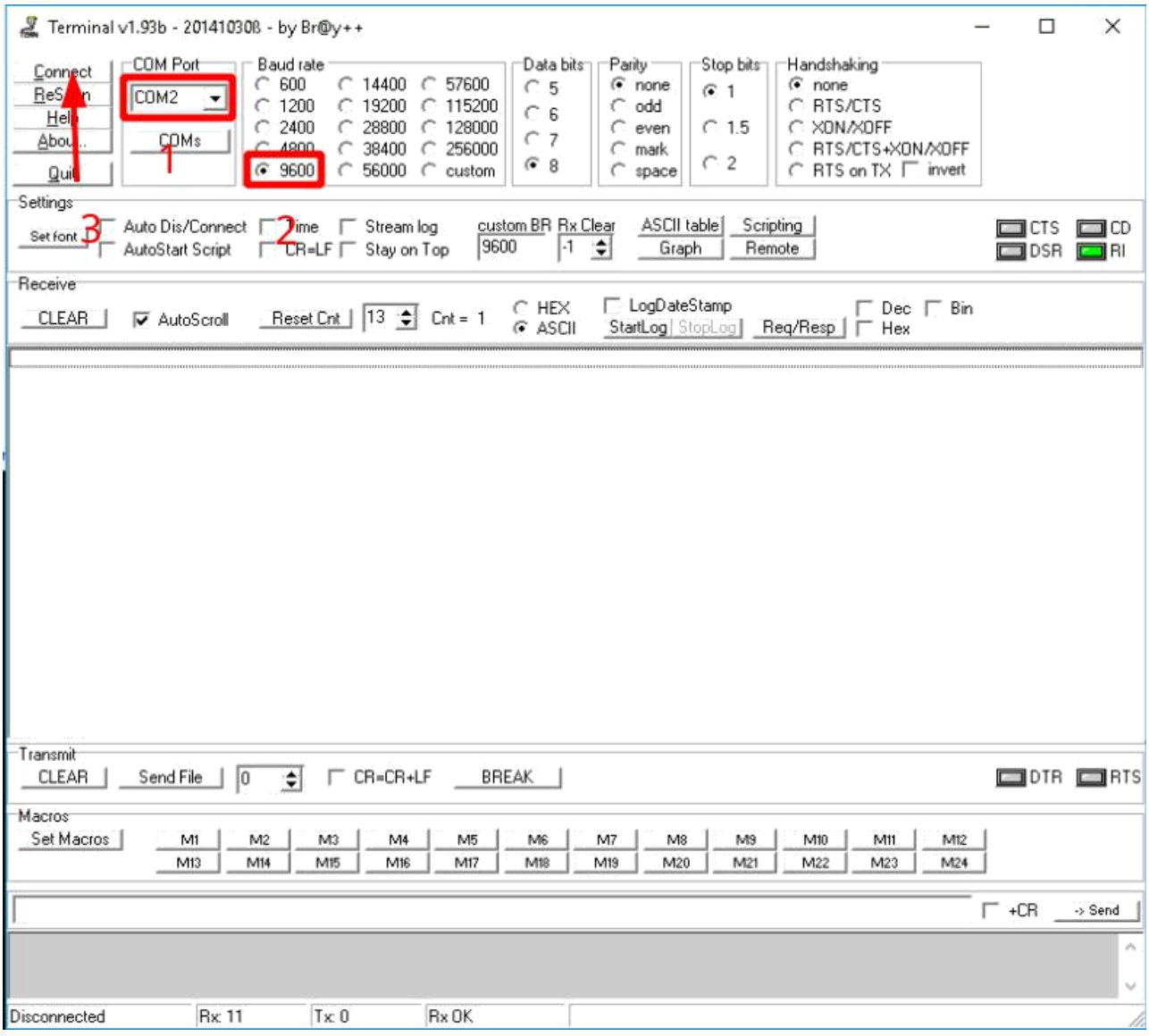
Then attach card, if you heard beep from RFID (HW working correctly) you should get result.



When you have information you need from hyperterminal, turn off application, then continue.

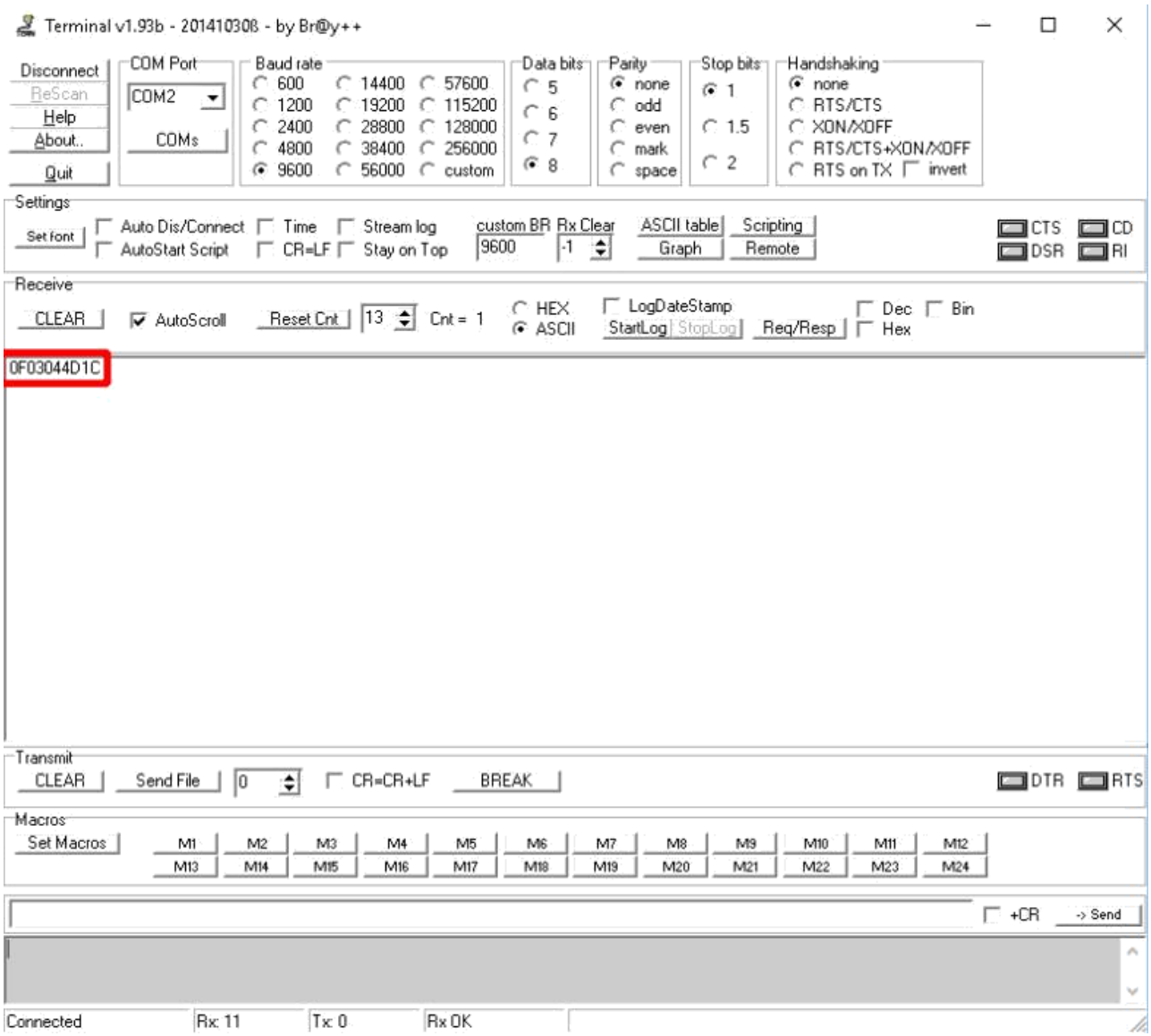
Attach test of card — Terminal (newer)

Newer Terminal is located in `C:\Tools\Terminal.exe`.



Again reader on COM2, rate 9600 Bd

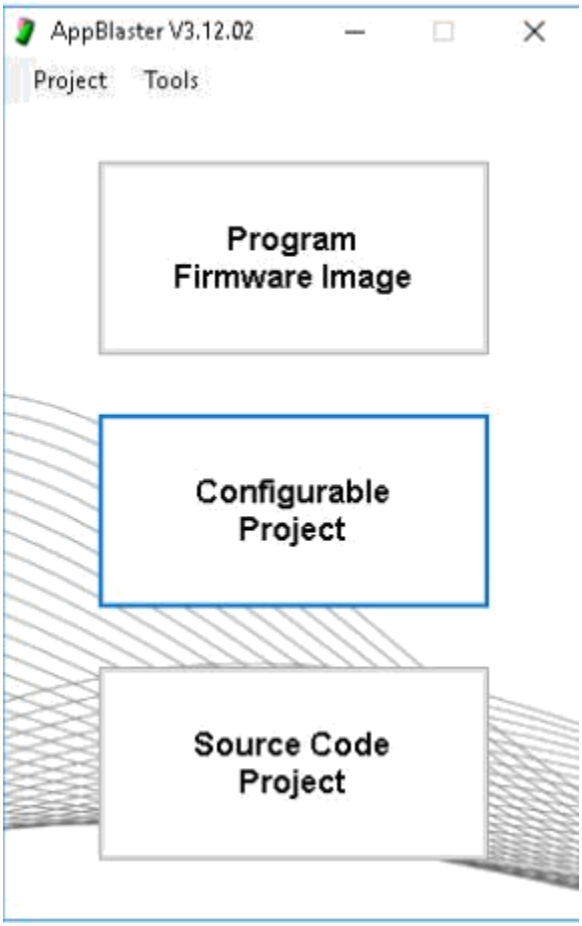
Then attach card, you should get result.



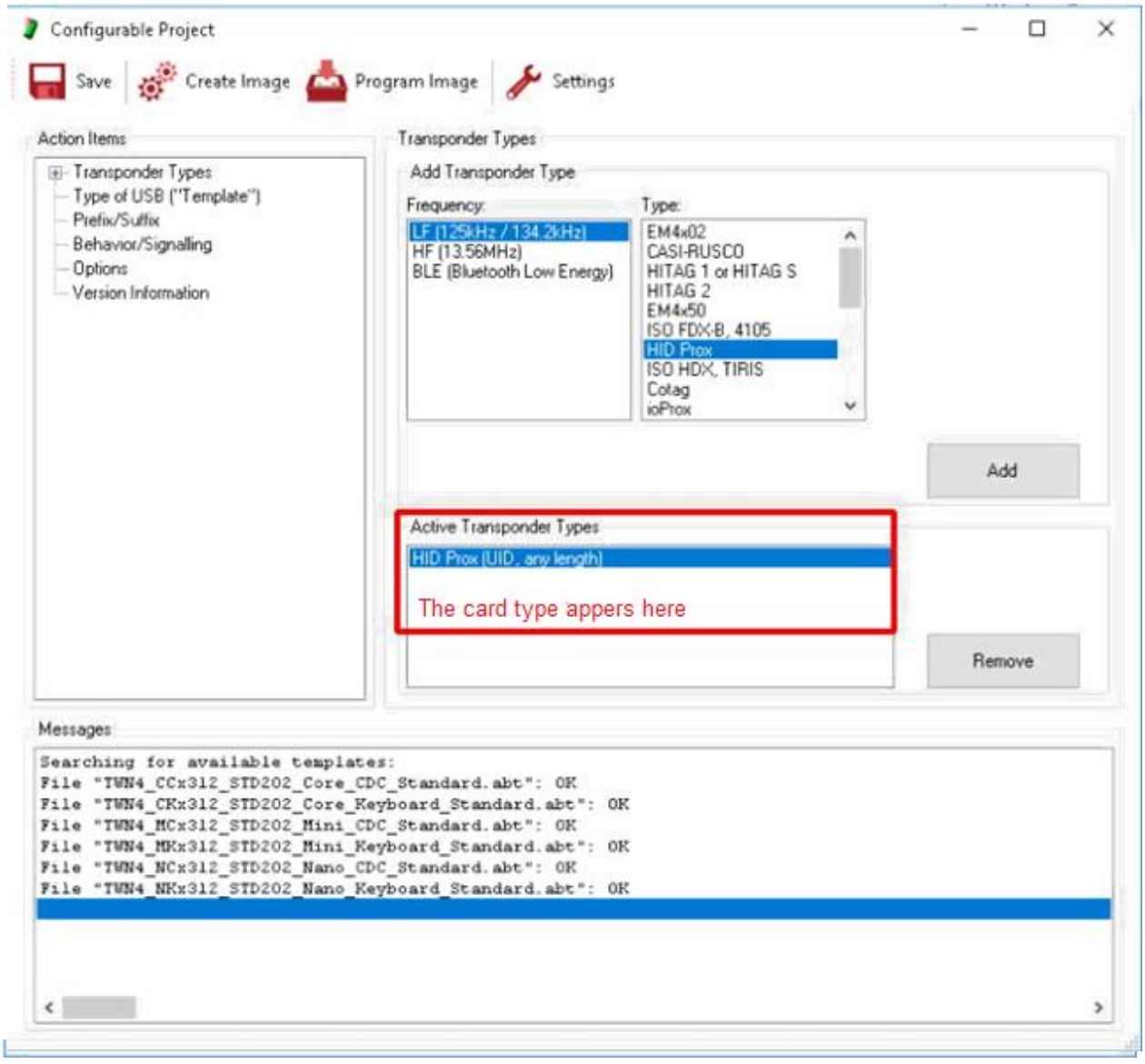
After card attach

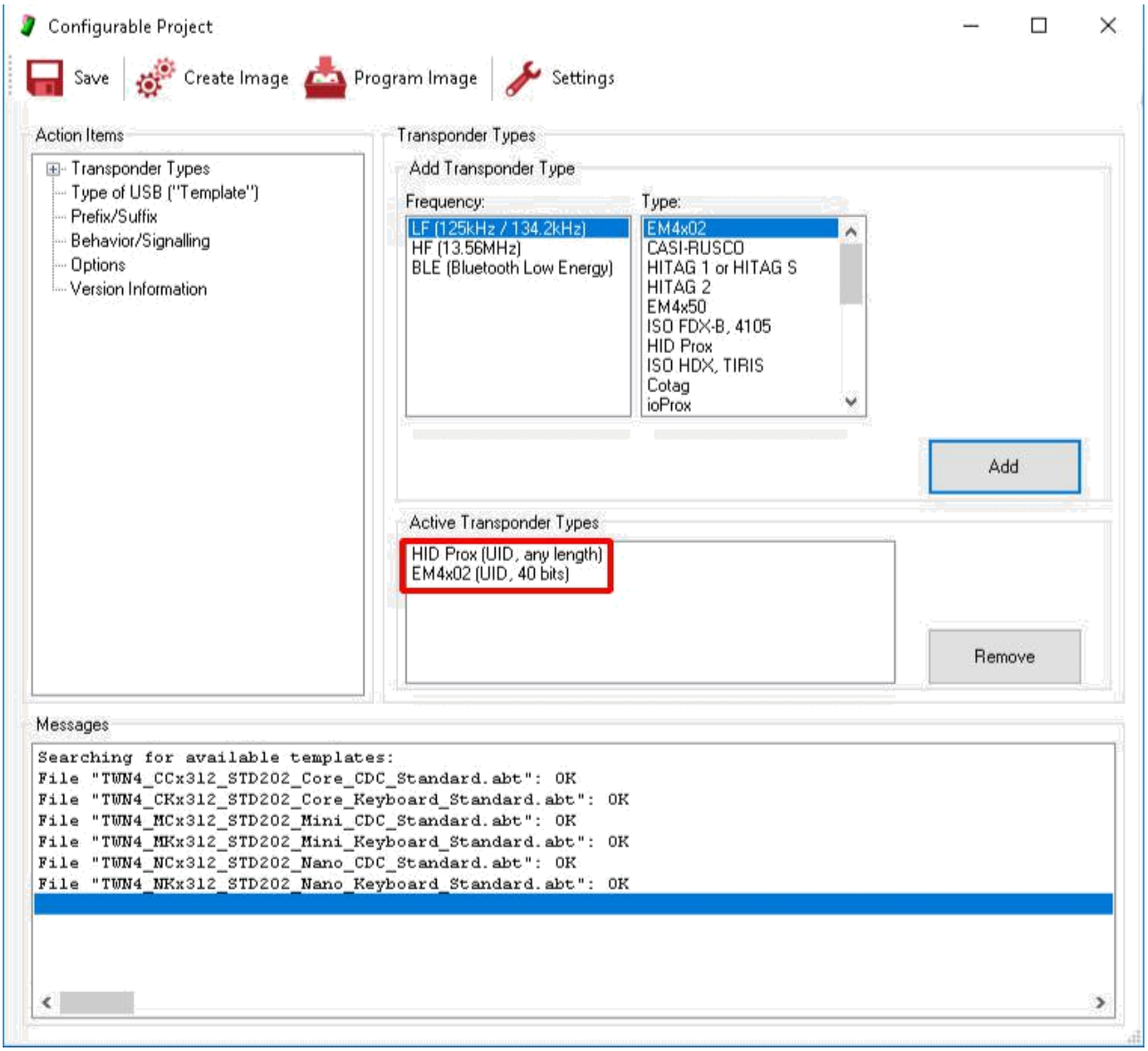
When you have information you need from hyperterminal, turn off application, then continue.

Basic programming

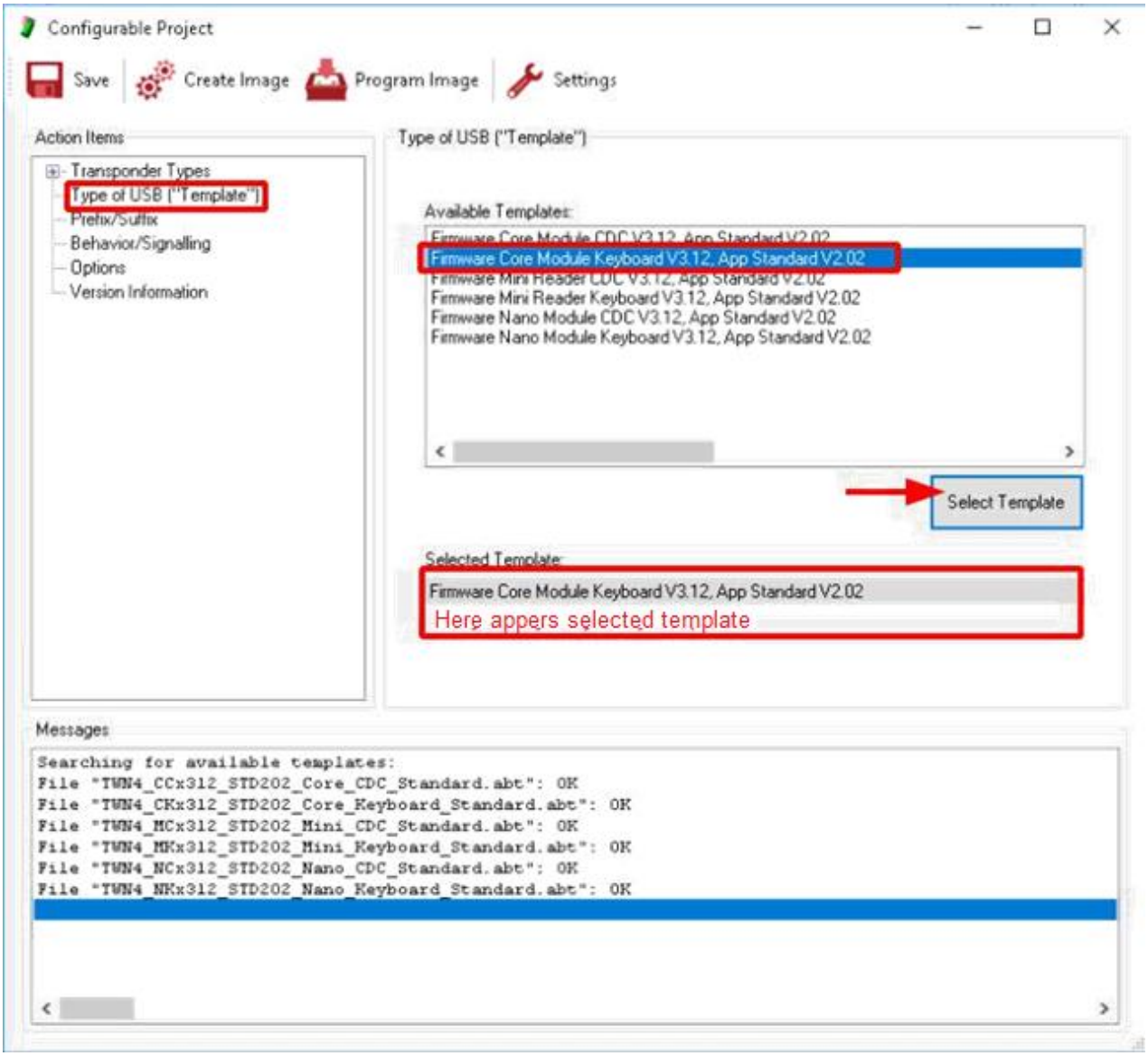


Choose Configurable Project

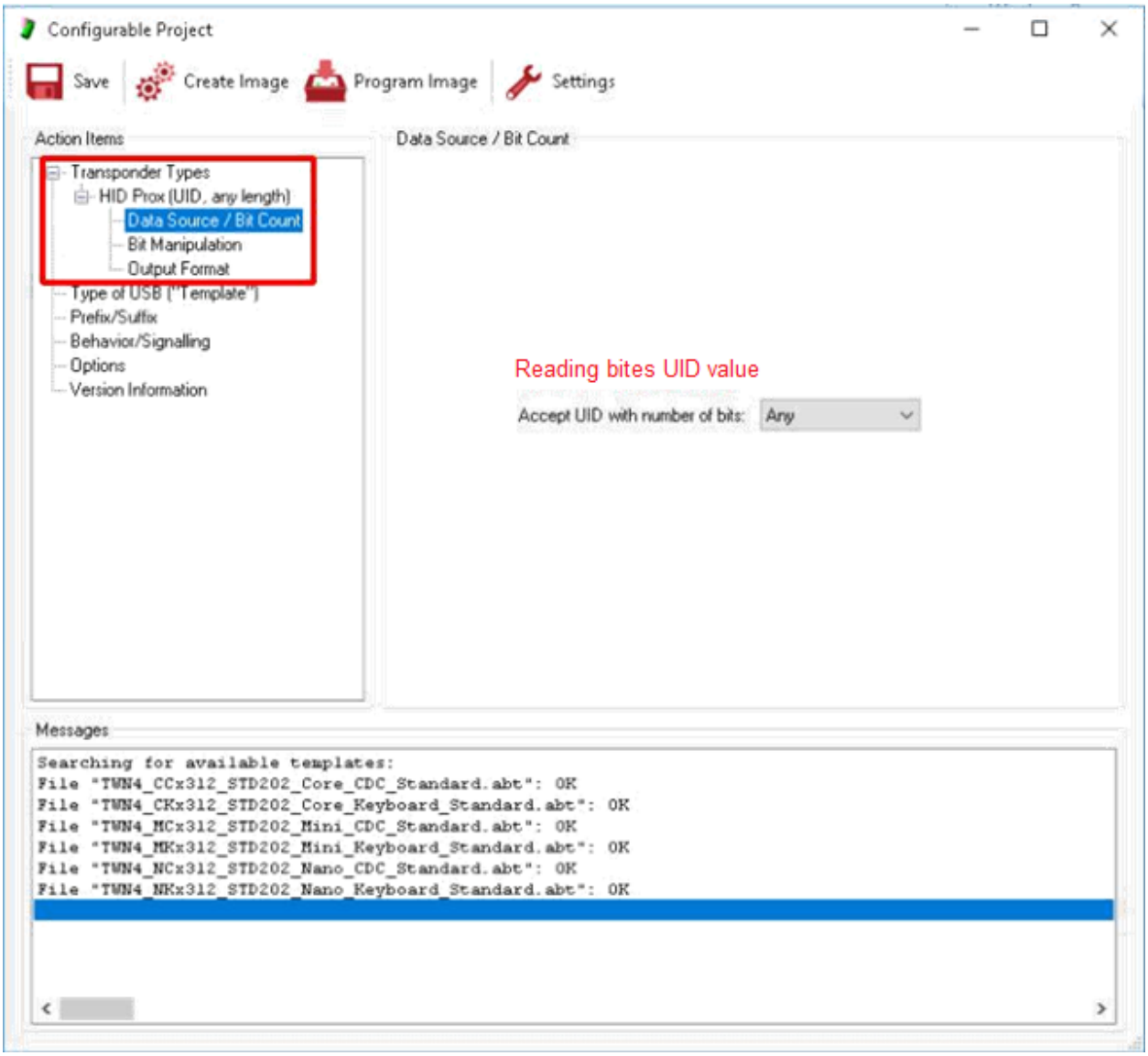

Configurable Project
Save
Create Image
Program Image
Settings
Action Items
Transponder Types
Type of USB ("Template")
Prefix/Suffix
Behavior/Signalling
Options
Version Information
Transponder Types
Add Transponder Type
Frequency:
LF (125kHz / 134.2kHz)
HF (13.56MHz)
BLE (Bluetooth Low Energy)
Type:
EM4x02
CASI-RUSCO
HITAG 1 or HITAG S
HITAG 2
EM4x50
ISO FDX-B, 4105
HID Prox
ISO HDX, TIRIS
Cotag
ioProx
Add
Active Transponder Types
HID Prox (UID, any length)
The card type appers here
Remove
Messages
Searching for available templates:
File "TWN4_CCx312_STD202_Core_CDC_Standard.abt": OK
File "TWN4_CKx312_STD202_Core_Keyboard_Standard.abt": OK
File "TWN4_MCx312_STD202_Mini_CDC_Standard.abt": OK
File "TWN4_MKx312_STD202_Mini_Keyboard_Standard.abt": OK
File "TWN4_NCx312_STD202_Nano_CDC_Standard.abt": OK
File "TWN4_NKx312_STD202_Nano_Keyboard_Standard.abt": OK

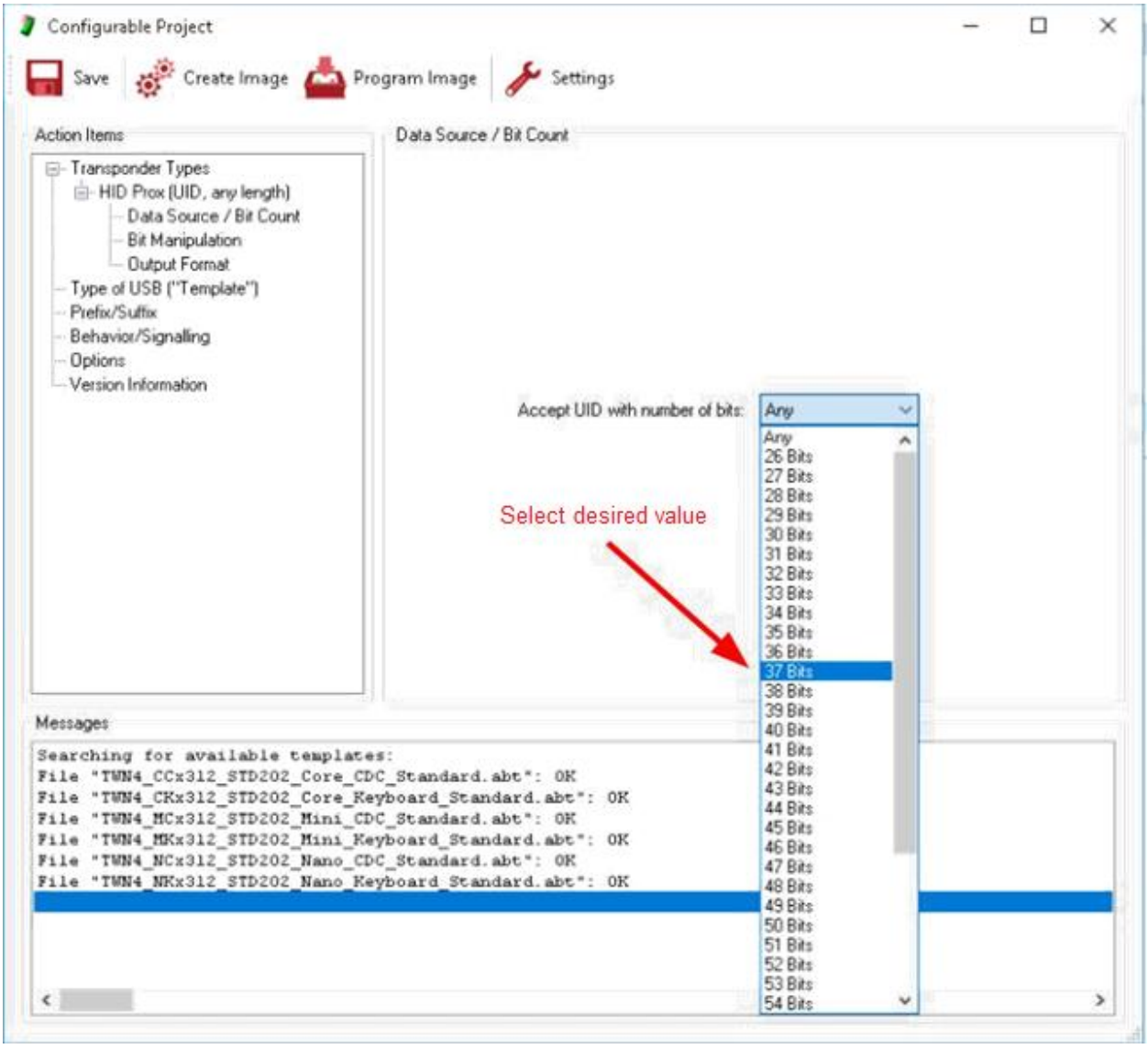


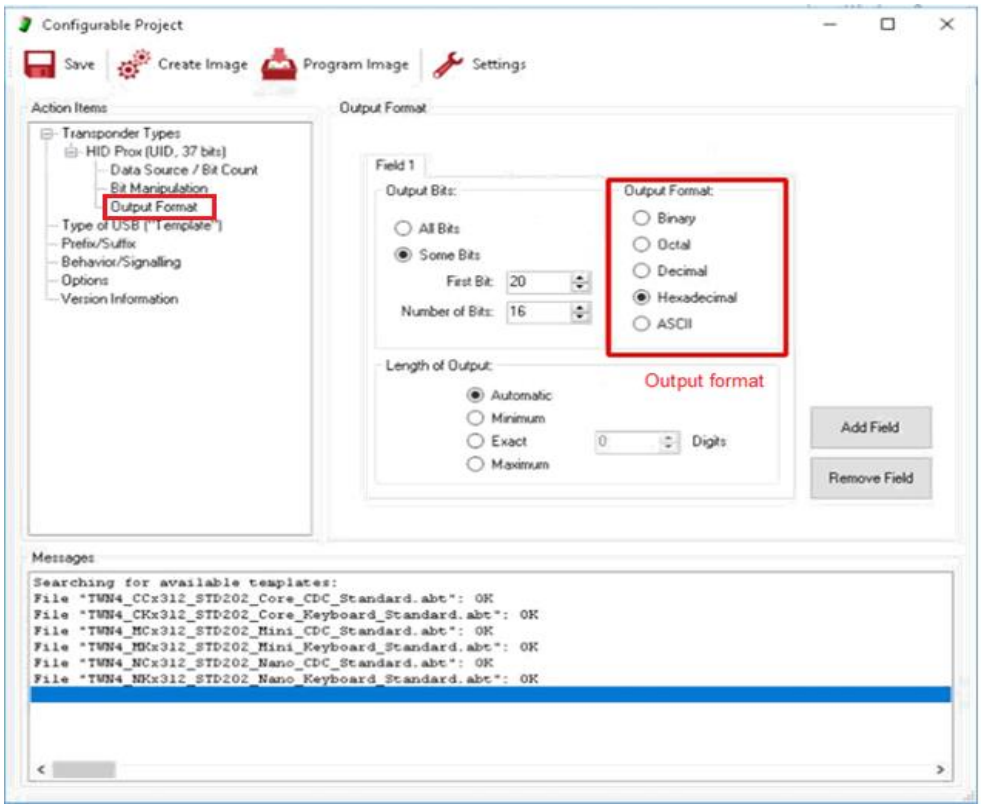
Is possible add more types

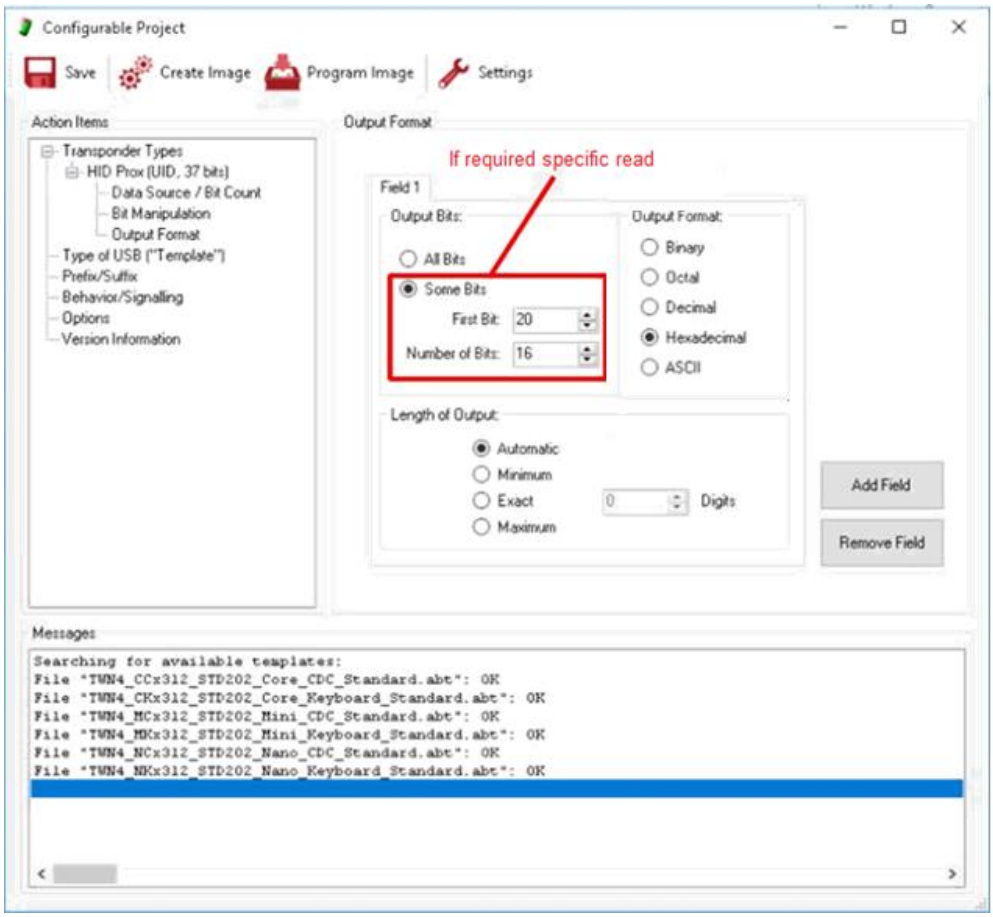

Configurable Project
Save
Create Image
Program Image
Settings
Action Items
Transponder Types
Type of USB ("Template")
Pretix/Suffix
Behavior/Signalling
Options
Version Information
Type of USB ("Template")
Available Templates:
Firmware Core Module Keyboard V3.12, App Standard V2.02
Firmware Mini Reader Keyboard V3.12, App Standard V2.02
Firmware Nano Module CDC V3.12, App Standard V2.02
Firmware Nano Module Keyboard V3.12, App Standard V2.02
Select Template
Selected Template:
Firmware Core Module Keyboard V3.12, App Standard V2.02
Here appers selected template
Messages
Searching for available templates:
File "TWN4_CCx312_STD202_Core_CDC_Standard.abt": OK
File "TWN4_CKx312_STD202_Core_Keyboard_Standard.abt": OK
File "TWN4_MCx312_STD202_Mini_CDC_Standard.abt": OK
File "TWN4_MKx312_STD202_Mini_Keyboard_Standard.abt": OK
File "TWN4_NCx312_STD202_Nano_CDC_Standard.abt": OK
File "TWN4_NKx312_STD202_Nano_Keyboard_Standard.abt": OK

Settings of bites reading

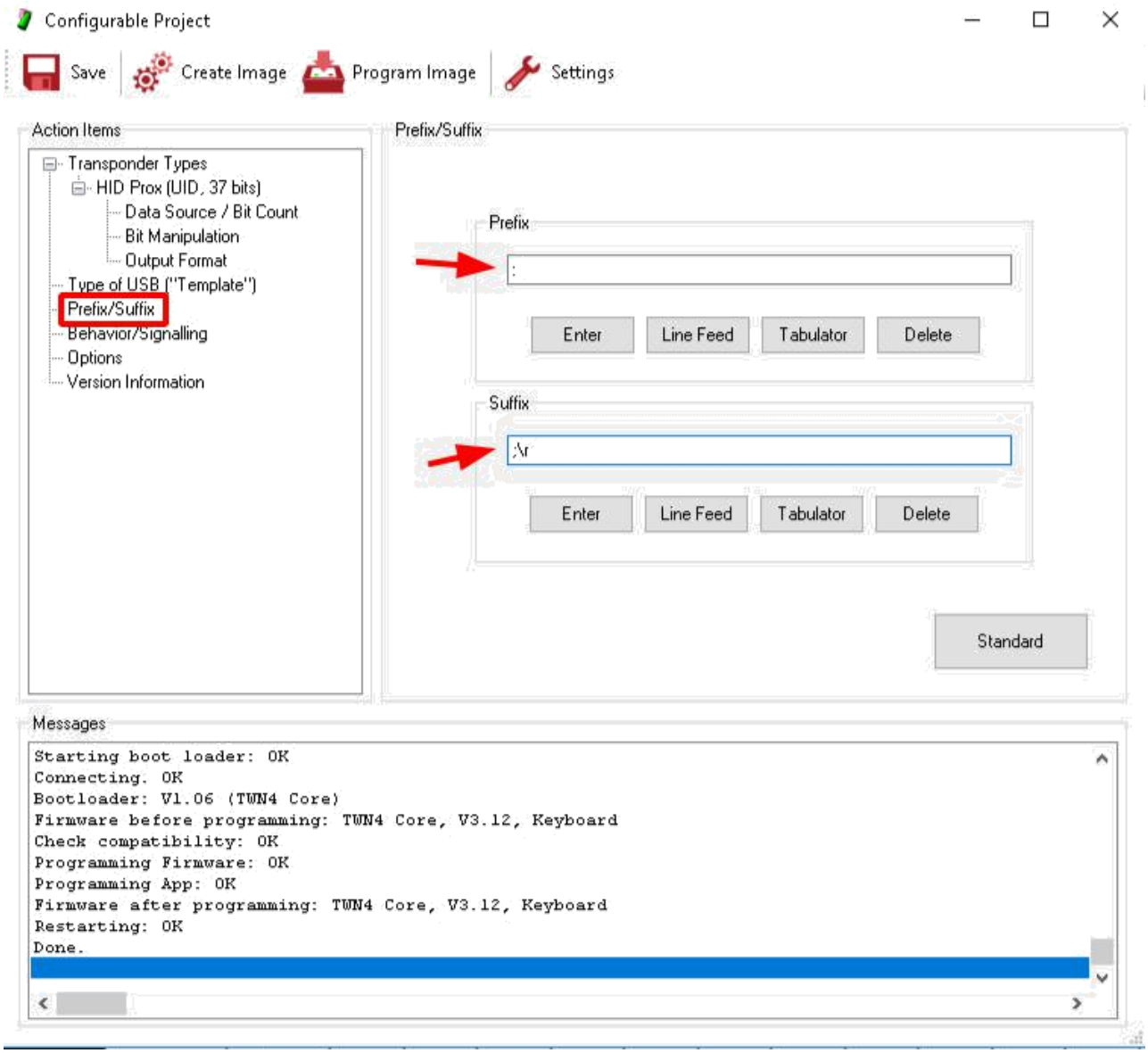



Configurable Project
Save
Create Image
Program Image
Settings
Action Items
Transponder Types
HID Prox (UID, any length)
Data Source / Bit Count
Bit Manipulation
Output Format
Type of USB ("Template")
Prefix/Suffix
Behavior/Signalling
Options
Version Information
Data Source / Bit Count
Accept UID with number of bits:
Any
Select desired value
Any
26 Bits
27 Bits
28 Bits
29 Bits
30 Bits
31 Bits
32 Bits
33 Bits
34 Bits
35 Bits
36 Bits
37 Bits
38 Bits
39 Bits
40 Bits
41 Bits
42 Bits
43 Bits
44 Bits
45 Bits
46 Bits
47 Bits
48 Bits
49 Bits
50 Bits
51 Bits
52 Bits
53 Bits
54 Bits
Messages
Searching for available templates:
File "TWN4_CCx312_STD202_Core_CDC_Standard.abt": OK
File "TWN4_CKx312_STD202_Core_Keyboard_Standard.abt": OK
File "TWN4_MCx312_STD202_Mini_CDC_Standard.abt": OK
File "TWN4_MKx312_STD202_Mini_Keyboard_Standard.abt": OK
File "TWN4_NCx312_STD202_Nano_CDC_Standard.abt": OK
File "TWN4_NKx312_STD202_Nano_Keyboard_Standard.abt": OK


Configurable Project
Save
Create Image
Program Image
Settings
Action Items
Transponder Types
HID Prox (UID, 37 bits)
Data Source / Bit Count
Bit Manipulation
Output Format
Type of USB ("Template")
Prefix/Suffix
Behavior/Signalling
Options
Version Information
Output Format
Field 1
Output Bits:
All Bits
Some Bits
First Bit: 20
Number of Bits: 16
Output Format:
Binary
Octal
Decimal
Hexadecimal
ASCII
Length of Output:
Automatic
Minimum
Exact
Maximum
Digits
Output format
Add Field
Remove Field
Messages
Searching for available templates:
File "TWN4_CCx312_STD202_Core_CDC_Standard.abt": OK
File "TWN4_CKx312_STD202_Core_Keyboard_Standard.abt": OK
File "TWN4_MCx312_STD202_Mini_CDC_Standard.abt": OK
File "TWN4_MKx312_STD202_Mini_Keyboard_Standard.abt": OK
File "TWN4_NCx312_STD202_Nano_CDC_Standard.abt": OK
File "TWN4_NKx312_STD202_Nano_Keyboard_Standard.abt": OK


Configurable Project
Save
Create Image
Program Image
Settings
Action Items
Transponder Types
HID Prox (UID, 37 bits)
Data Source / Bit Count
Bit Manipulation
Output Format
Type of USB ("Template")
Prefix/Suffix
Behavior/Signalling
Options
Version Information
Output Format
If required specific read
Field 1
Output Bits:
All Bits
Some Bits
First Bit: 20
Number of Bits: 16
Output Format:
Binary
Octal
Decimal
Hexadecimal
ASCII
Length of Output:
Automatic
Minimum
Exact
Maximum
Digits
Add Field
Remove Field
Messages
Searching for available templates:
File "TWN4_CCx312_STD202_Core_CDC_Standard.abt": OK
File "TWN4_CKx312_STD202_Core_Keyboard_Standard.abt": OK
File "TWN4_MCx312_STD202_Mini_CDC_Standard.abt": OK
File "TWN4_MKx312_STD202_Mini_Keyboard_Standard.abt": OK
File "TWN4_NCx312_STD202_Nano_CDC_Standard.abt": OK
File "TWN4_NKx312_STD202_Nano_Keyboard_Standard.abt": OK

Prefix/Suffix



After card attach you´ll get mark **:**

Result in RFID then **:86336727;**

Bits manipulations

From customer is needs to know, how to read card.

Reading just specified bits

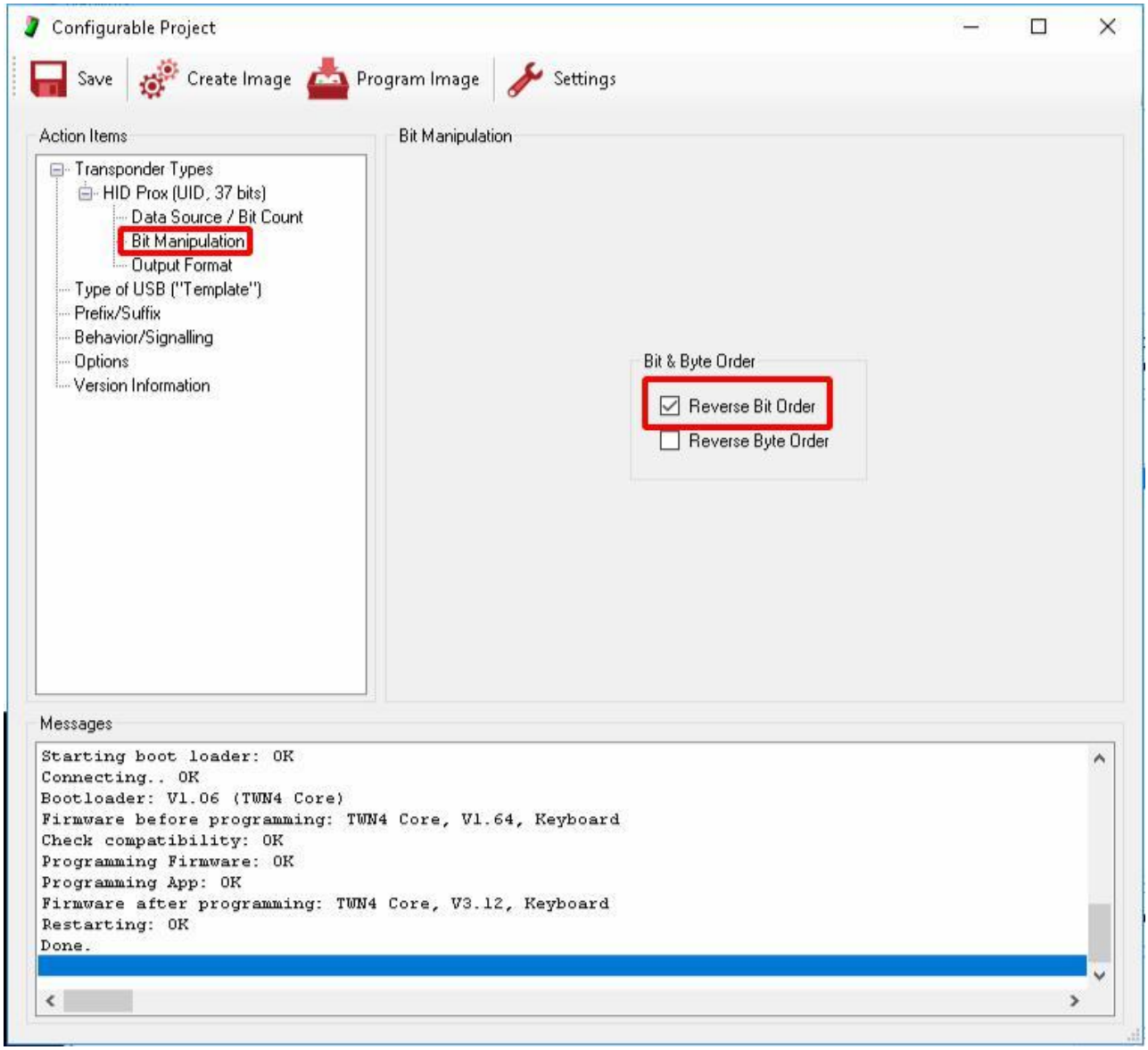
```
Load card:
HEX: 00052564D7
DEC: 86336727
BIN: 01010010010101100100110101111 (RFID reads 28 bitů)

Customer has in database number 45675:
DEC: 45675
BIN: 1011001001101011 (We want read 16 bits)

Compare both BIN results:
01010010010101100100110101111
             1011001001101011
```

We have 11 bits at beginning, we want read 16 bits, but on end of chain is still 1 bit.It means we have 11 + 16 + 1 = 28 bit (what RFID reads). UID size is 37 bits (see above). It means there are 9 bits (zeros) in front of read number.

To get just the right piece of number, we need to take 11 bits (which is at the begining) + 9 bits (those zeros before they are not visible) = 20 bit. Later in chapter ***Bits read settings*** fill as First bit and Number of bits, in this case number 16.

Bits/bytes reverse read

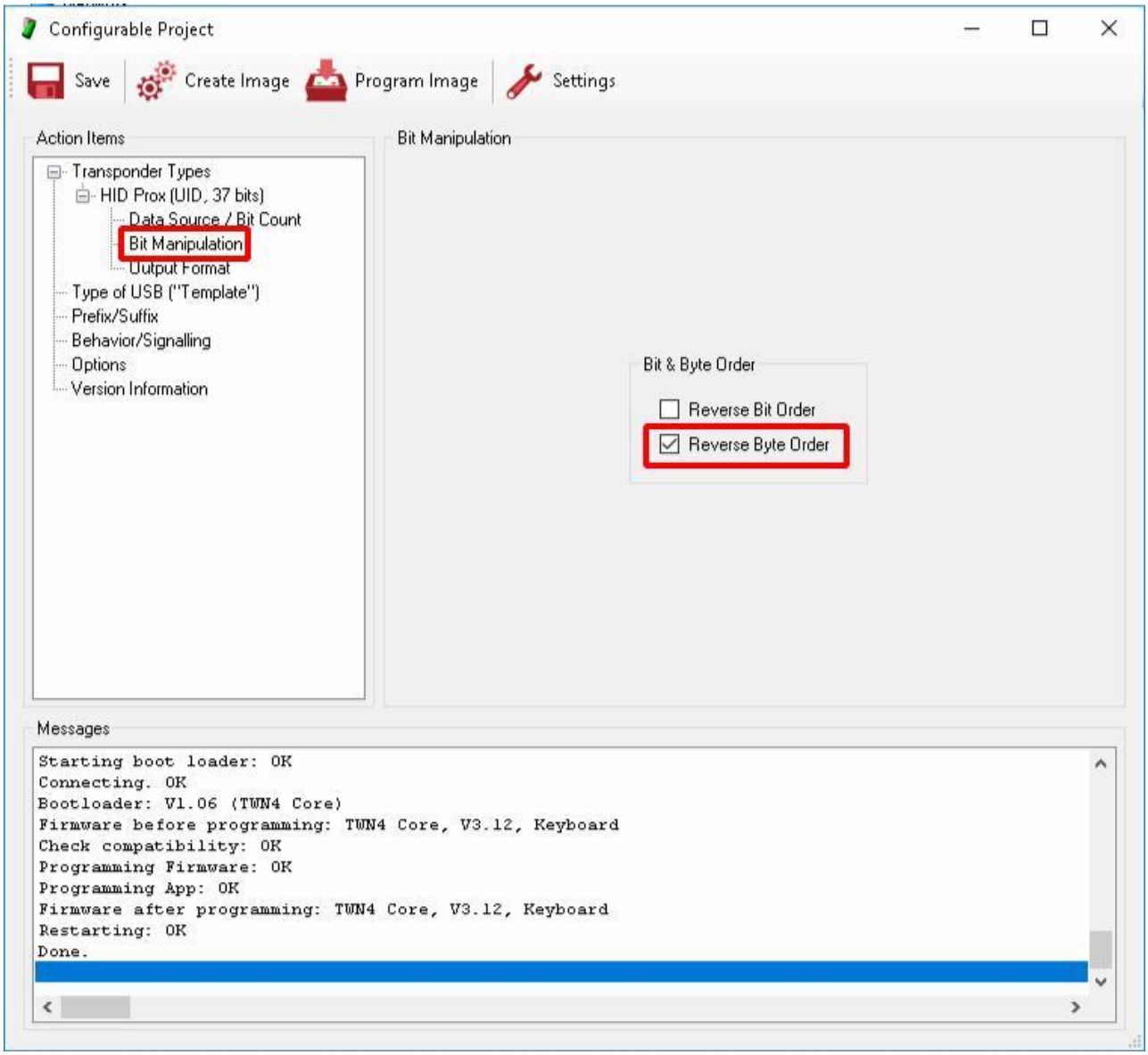
From same example in previous chapter set on Reverse bit order



Binary number after load will be readed conversely /reverse, bit by bity (don´t forget for 9 bits to 37b):

```
Origin:                  0000000000101001001010110010011010111
With reverse bit order: 1110101100100110101001001010000000000
After cropping on 16b:                                100101000000000
```

Reverse byte order is similar:



In result will be swap 8 bits pars/couples:

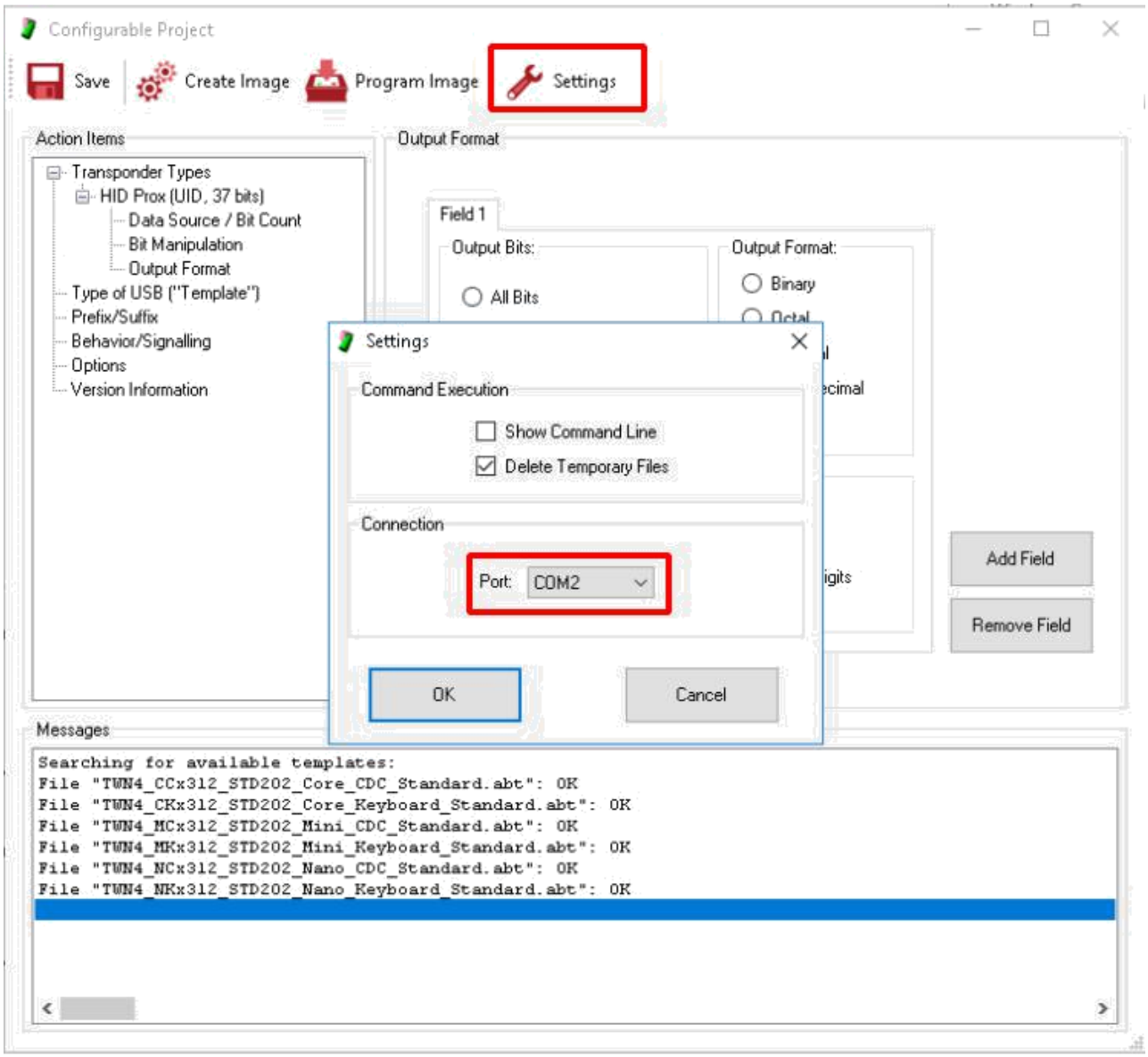
```
Origin:                   0000000000101001001010110010011010111
With reverse byte order: 1011100000100110001010110010100100000
```

```
After cropping on 16b:                                  1011001010010000
```

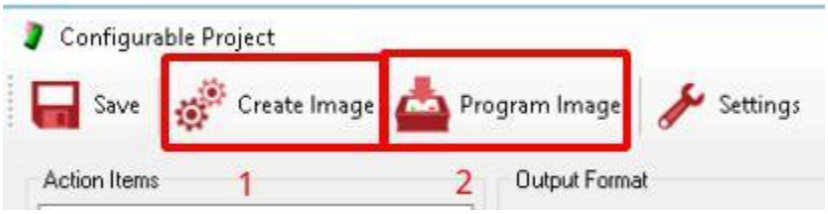
The way of swap is a little more complicated. Origin number is divided on bytes from left. I have 37b, so 32b would be whole and last byte will have 5b.

```
                  B0         B1         B2         B3         B4
Origin number: 00000000  00101001  00101011   00100110     10111


New number:       B4         B3         B2         B1         B0
                10111000  00100110  00101011  00101001   00000
```

Image creation and upload to RFID reader

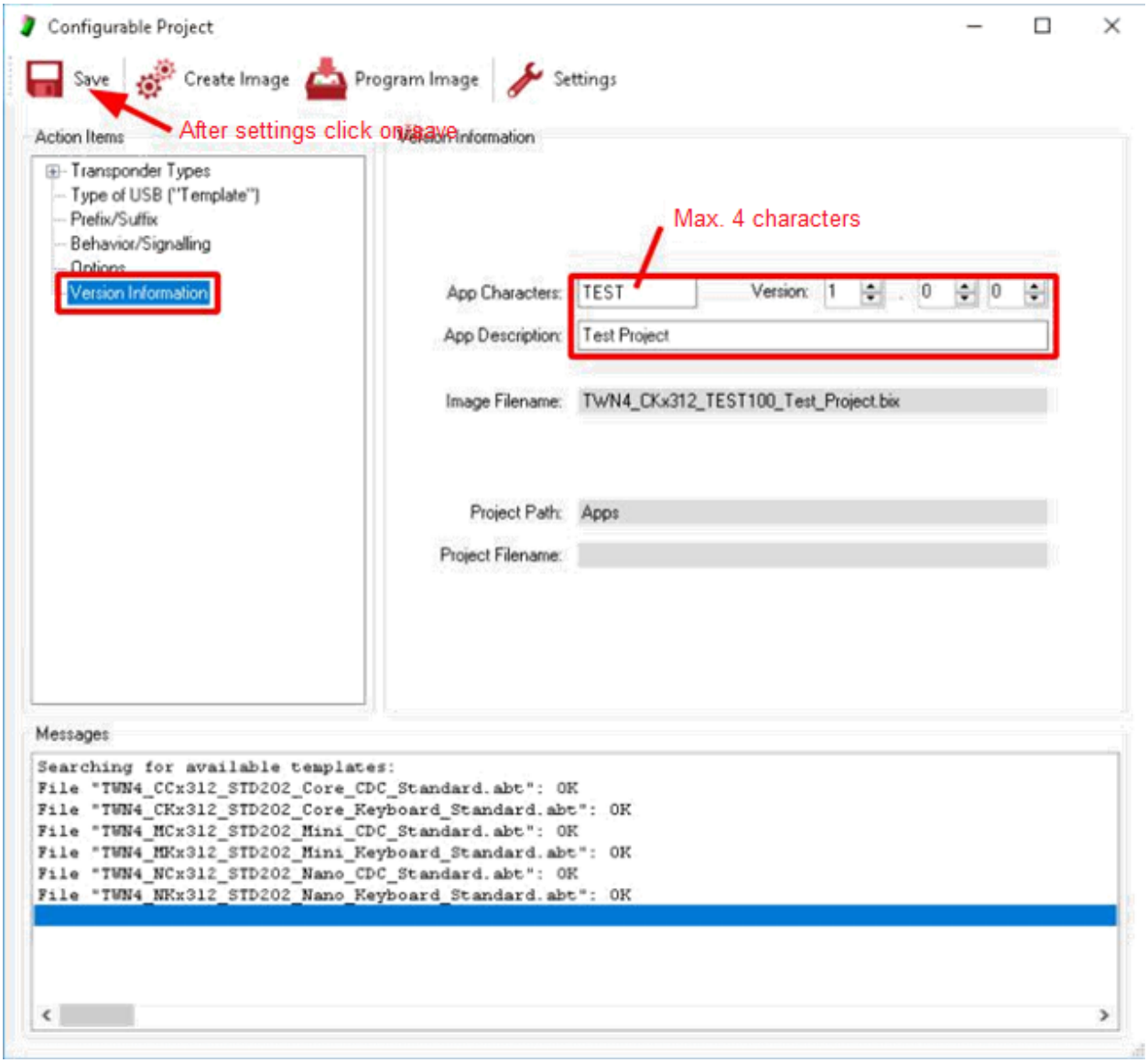


RRID is on COM2



Click on Create Image and then on Program Image

Name and save settings



IF YOU DID EVERYTHING RIGHT, YOU HAVE RFID PROGRAMMED.