

<https://www.halvorsen.blog>



RFID

Radio Frequency Identification System

Hans-Petter Halvorsen

Contents

- [Introduction](#)
- [RFID Overview](#)

Overview of different RFID Readers:

- [Parallax USB RFID Reader](#)
 - [Code Examples](#) (Python, LabVIEW, Visual Studio/C#)
- [Eccel OEM-MICODE-USB RFID Reader](#)
 - [Setup and Configuration](#)
 - [Code Examples](#) (Python, LabVIEW, Visual Studio/C#)



Introduction

Different Technologies

- Barcode
- QR Code
- RFID
- NFC
- Beacons

Barcode



- A barcode or bar code is a method of representing data in a visual, machine-readable form.
- Initially, barcodes represented data by varying the widths and spacings of parallel lines.
- These barcodes, now commonly referred to as linear or one-dimensional (1D)
- These can be scanned by special optical scanners, called barcode readers

QR Code

A QR code (abbreviated from Quick Response code) is a type of matrix barcode (or two-dimensional barcode)



RFID

- Radio-frequency identification (RFID)
- RFID is the method of uniquely identifying items using radio waves
- An RFID system comprises a tag, a reader, and an antenna
- Unlike a barcode, the tag does not need to be within the line of sight of the reader

NFC

- NFC - Near Field Communication
- NFC describes a technology which can be used for contactless exchange of data over short distances
- RFID is the process by which items are uniquely identified using radio waves
- NFC is a specialized subset within the family of RFID technology

Beacons

- The concept of beacons is as old as, for example, hand watches. Like lighthouses guide ships and show where the land is, beacon devices provide information and navigation to smartphone users. Beacon technology offers a new context for an old concept.
- Typically use Bluetooth

https://en.wikipedia.org/wiki/Bluetooth_low_energy_beacon

<https://www.intellectsoft.net/blog/what-are-beacons-and-how-do-they-work/>



RFID Overview

Radio Frequency Identification System

RFID Parts

- An **RFID tag** in its most simplistic form, is comprised of two parts – an antenna for transmitting and receiving signals, and an RFID chip (or integrated circuit, IC) which stores the tag's ID and other information. RFID tags are affixed to items in order to track them using an RFID reader and antenna.
- An **RFID reader** is the brain of the RFID system and is necessary for any system to function
- **RFID Antennas** are necessary elements in an RFID system because they convert the RFID reader's signal into RF waves that can be picked up by RFID tags
- Many RFID readers has an integrated antenna

RFID

- RFID System is an abbreviation of **Radio Frequency Identification System**.
- It is a system for identification of items
- It uses using wireless communication that transfer data between Tags and the RFID Reader/Antenna
- We get RFID systems with different Frequencies
 - LF (125KHz), HF (13.56MHz), UHF
- We have Active and Passive Tags.
 - Passive tags are powered by energy from the RFID reader
 - Active tags have a battery

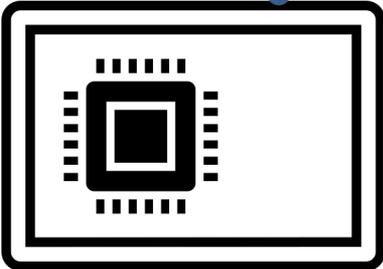
RFID System

PC (or a Microcontroller, e.g., Arduino, Raspberry Pi)

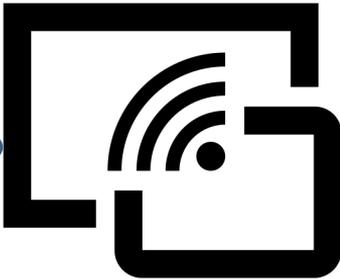


USB

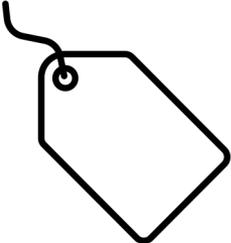
RFID Reader



RFID Antenna



RFID Tag



RFID Applications

How is RFID Used in the Real World? Some Examples:

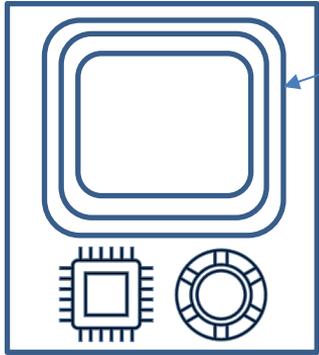
- Inventory Tracking
- Race Timing
- Attendee Tracking
- Materials Management
- Access Control
- Library System

..

https://www.atlasfidstore.com/rfid-insider/what-is-rfid-used-for-in-applications/?utm_source=RFID-Beginners-Guide&utm_medium=eBook&utm_campaign=Content&utm_content=real-world-applications

RFID System

RFID Reader



RFID Antenna



RFID Tags



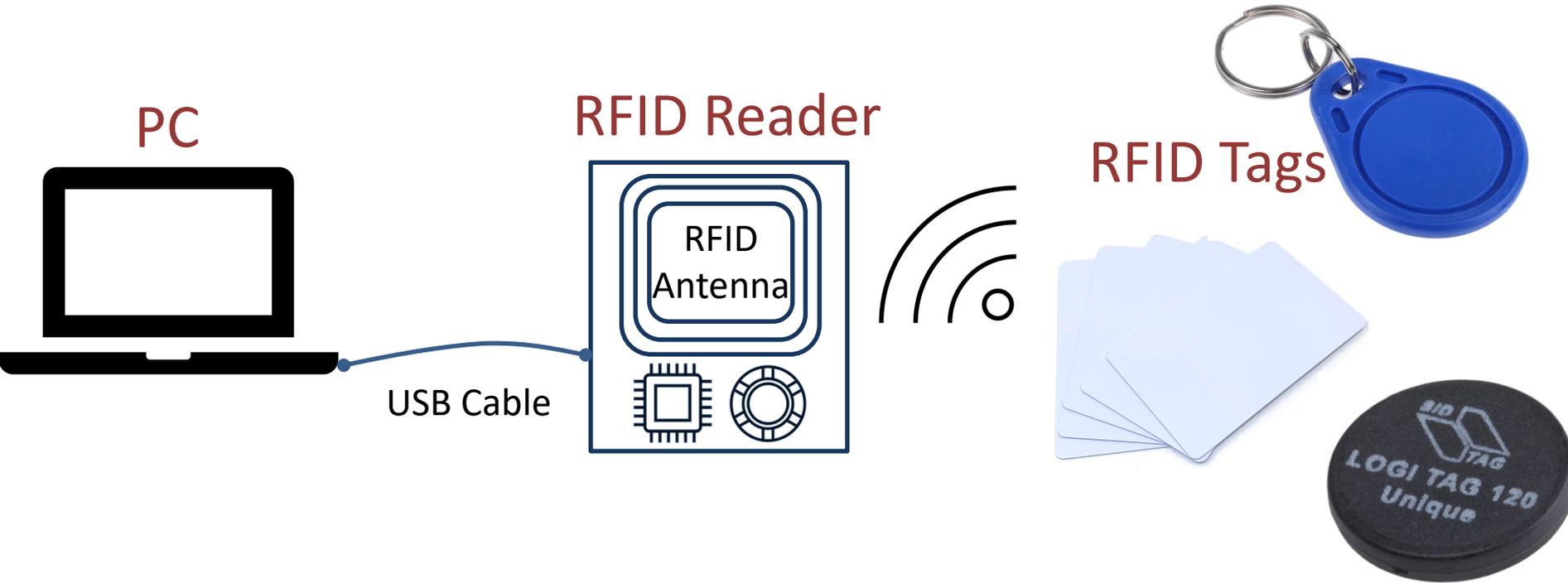
RFID Tags exist in many flavors and shapes

The RFID Reader is typically a Microcontroller

The Antenna is typically integrated within the RFID Reader, but you can also get external Antennas for better range



RFID System



RFID Frequencies

- Low Frequency (LF) - 125KHz
 - Range up to 20cm
 - Read Only
 - It is suitable for applications with a short distance, low transmission rate, and a small amount of data. For example, access control, attendance, electronic billing, electronic wallet, and so on.
- High Frequency (HF) - 13.56MHz
 - Range up to 1m
 - Read/Write
- Ultra High Frequency (UHF) - 433 MHz and higher
 - Range up to 100m
 - Read/Write

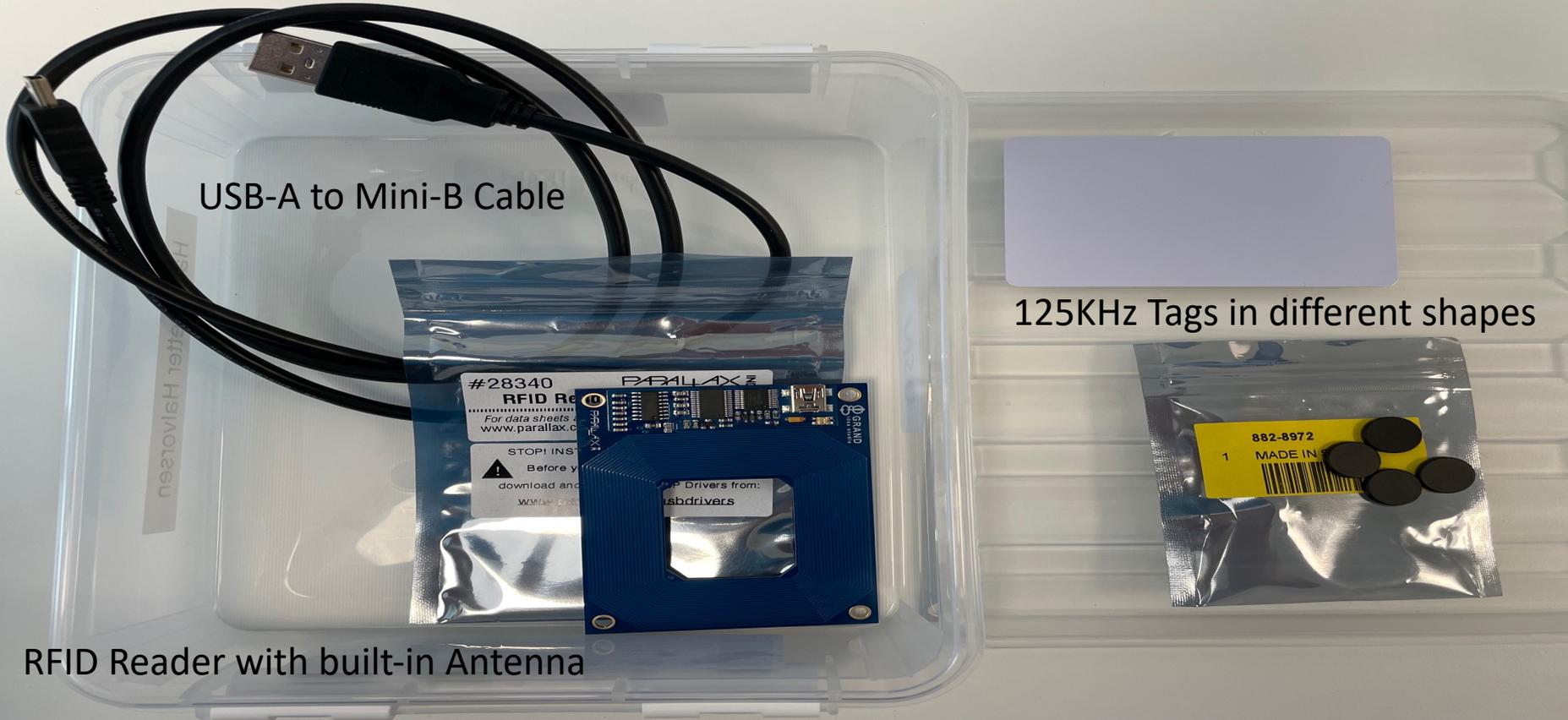
<https://learn.sparkfun.com/tutorials/rfid-basics/>

<https://www.asiarfid.com/different-types-of-rfid-tags.html>



Parallax USB RFID Reader

Parallax USB RFID Reader



USB-A to Mini-B Cable

125KHz Tags in different shapes

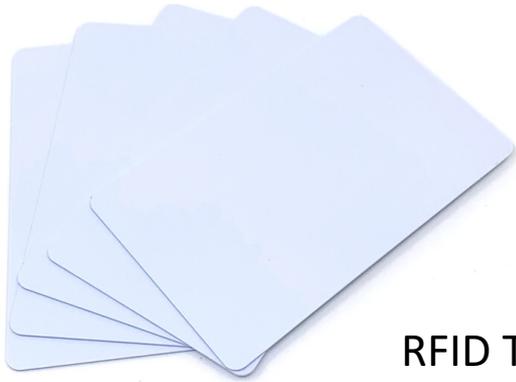
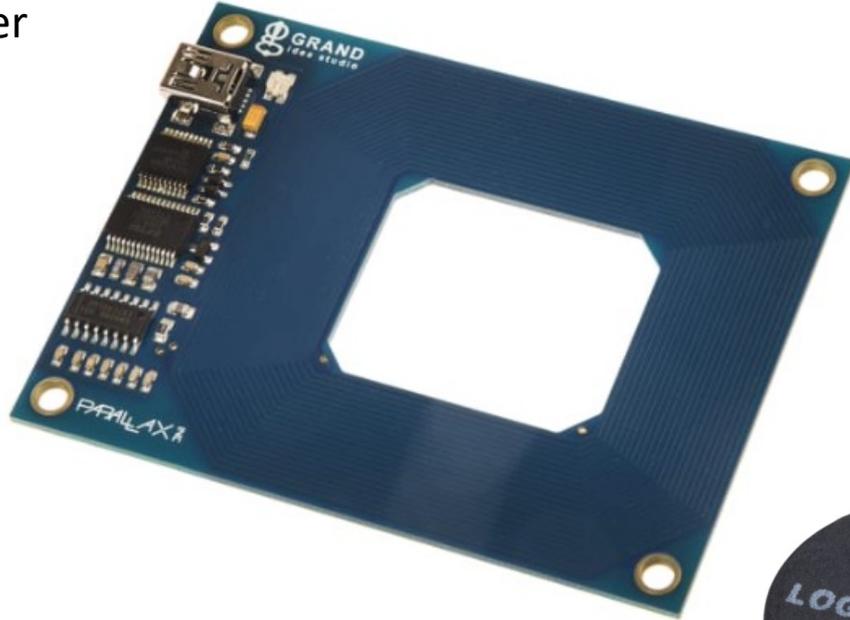
RFID Reader with built-in Antenna

RFID 125KHz

Reads 125kHz Tags with EM4100 protocol

Parallax Inc 28340 RFID Reader

USB RFID Reader



RFID Tags

RFID Tags



<https://www.parallax.com/product/rfid-card-reader-usb/>

<https://no.rs-online.com/web/p/rf-modules/7813061/>

Parallax USB RFID Reader

From Parallax USB RFID Reader Documentation

- It reads passive **125 kHz** RFID transponder tags
- The Parallax RFID Card Reader USB version can be connected directly to any PC, Macintosh, or Linux machine that has a USB port and the appropriate drivers installed. The module is powered from the host computer's USB port and uses an industry-standard **FTDI FT232R** device to provide the USB connectivity
- A visual indication of the state of the RFID Card Reader is given with the on-board LED. When the module is successfully powered-up and is in an idle state, the LED will be **GREEN**. When the module is in an active state searching for or communicating with a valid tag, the LED will be **RED**.
- The RFID Card Reader USB version is activated via the **DTR** line of the USB Virtual COM port. When the DTR line is set HIGH, the module will enter the active state. When the DTR line is set LOW, the module will enter the idle state.
- RFID Tag read distance of approximately 4 inches (**10cm**).

Parallax USB RFID Reader

Communication Protocol:

- The RFID Card Reader USB version transmits the data through the USB Virtual COM Port driver
- All communication is **8 data bits, no parity, 1 stop bit**, and least significant bit first (8N1) at **2400 bps**.
- When the RFID Card Reader is active and a valid RFID transponder tag is placed within range of the activated reader, the tag's unique ID will be transmitted as a **12-byte printable ASCII string** serially to the host in the following format:

Parallax USB RFID Reader

Communication Protocol:

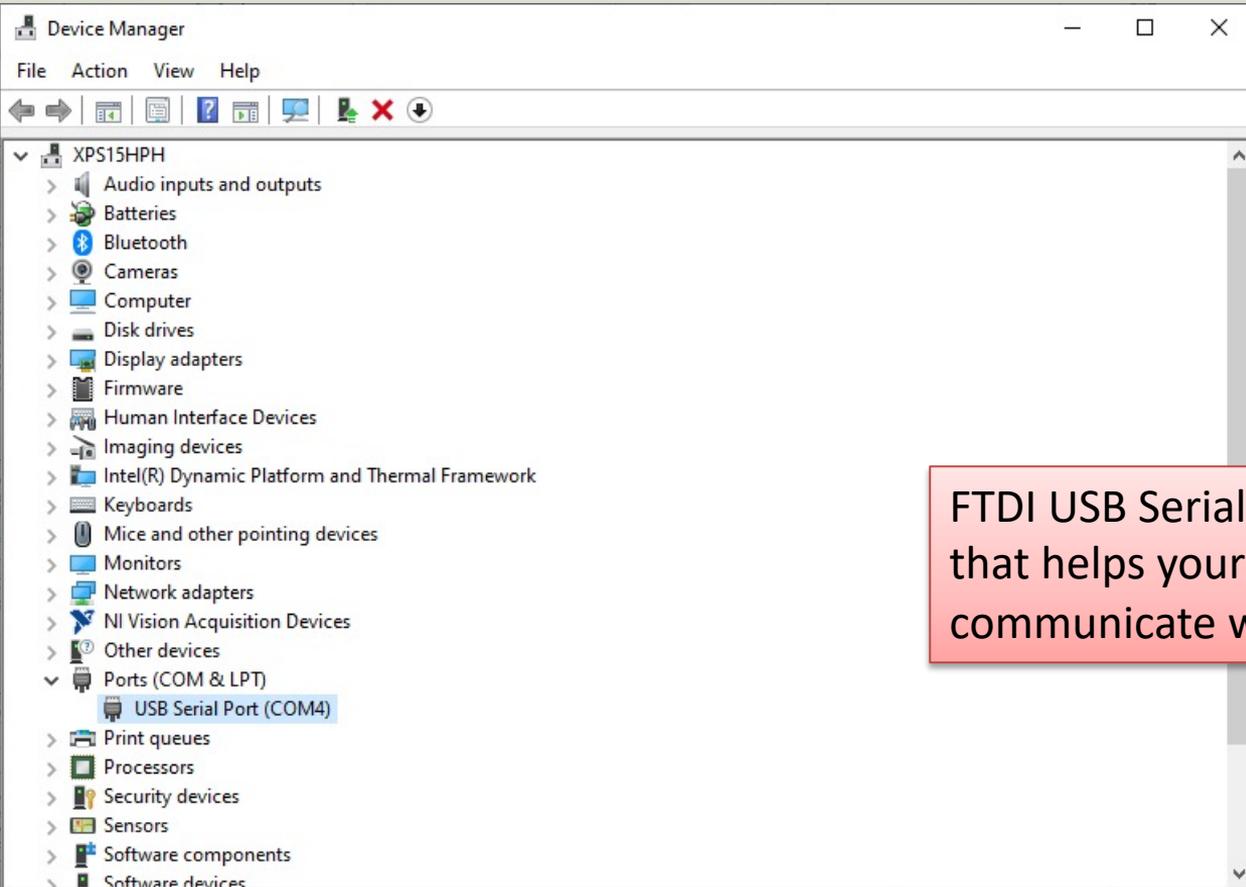
Start Byte (0x0A)	Unique ID Digit 1	Unique ID Digit 2	Unique ID Digit 3	Unique ID Digit 4	Unique ID Digit 5	Unique ID Digit 6	Unique ID Digit 7	Unique ID Digit 8	Unique ID Digit 9	Unique ID Digit 10	Stop Byte (0x0D)
----------------------	----------------------	----------------------	----------------------	----------------------	----------------------	----------------------	----------------------	----------------------	----------------------	-----------------------	---------------------

The start byte and stop byte are used to easily identify that a correct string has been received from the reader (they correspond to **line feed (\n)** and **carriage return (\r)** characters, respectively).

The middle ten bytes are the actual tag's unique ID.

For example, for a tag with a valid ID of 0F0184F07A, the following bytes would be sent: 0x0A, 0x30, 0x46, 0x30, 0x31, 0x38, 0x34, 0x46, 0x30, 0x37, 0x41, 0x0D.

Setup and Configuration



Device Manager

FTDI USB Serial Port driver is the software that helps your operating system to communicate with USB Serial Port devices



Code Examples

Parallax USB RFID Reader



Code Examples

Note!

- The examples provided can be considered as a “proof of concept”
- The sample code is very simplified for clarity and doesn't necessarily represent best practices.

Python Example

The screenshot shows the Thonny Python IDE interface. The main editor window displays the following Python code in `rfid_ex.py`:

```
1 import serial
2 import time
3
4 ser = serial.Serial('COM4', 2400, timeout=1)
5
6 response = ser.read(12)
7 if response != "":
8     print(response)
9
10 ser.close()
```

The Shell window shows the execution output for Python 3.7.9 (bundled):

```
>>> %Run rfid_ex.py
b'\n0800296663\r'
>>>
```

The Assistant window displays a warning:

Warnings
May be ignored if you are happy with your program.
[rfid_ex.py](#)
Line 2: Unused import time
[Was it helpful or confusing?](#)

Python 3.7.9

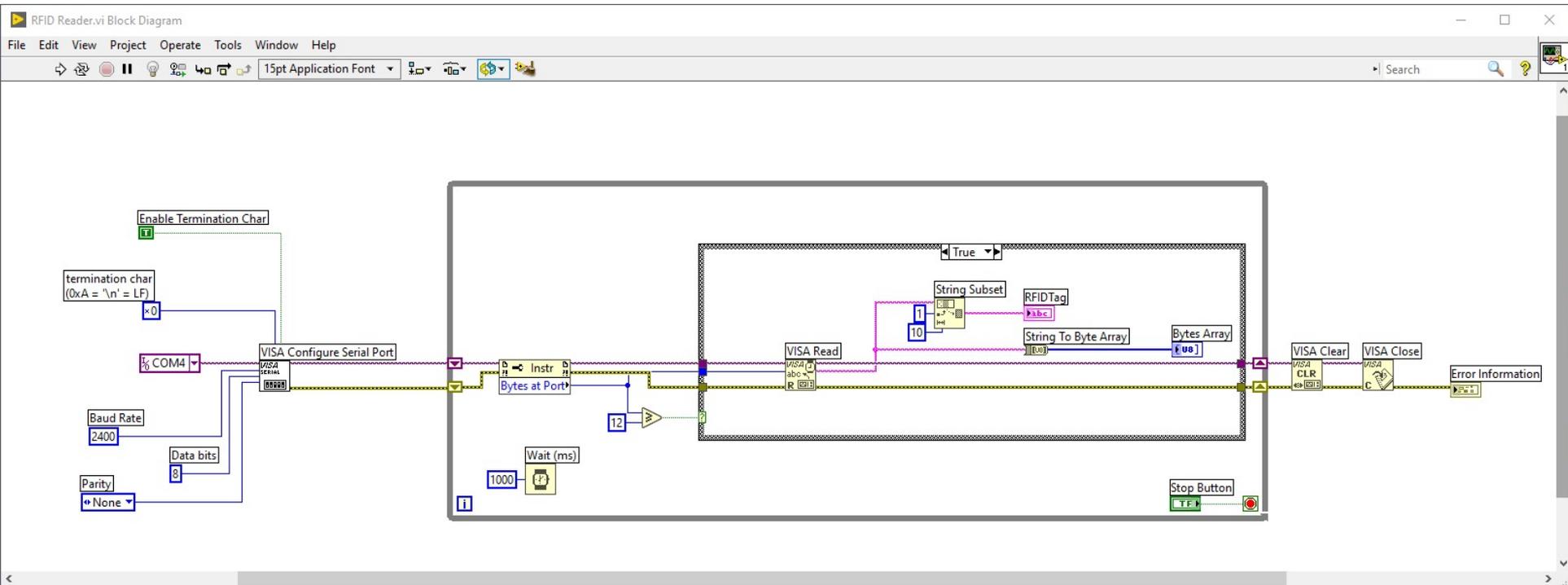
LabVIEW Example

The screenshot displays the LabVIEW front panel for 'RFID Reader.vi'. The interface includes a menu bar (File, Edit, View, Project, Operate, Tools, Window, Help), a toolbar with various icons, and a search field. The main panel is divided into three sections:

- RFIDTag**: A text box containing the hexadecimal string '0800297F02'.
- Bytes Array**: A horizontal array of 12 hexadecimal characters: 'A', '30', '38', '30', '30', '32', '39', '37', '46', '30', '32', 'D'.
- Error Information**: A panel with a green status indicator, a source dropdown menu showing 'VISA Read in RFID Reader.vi', and a code field containing '1073676'.

A red 'Stop' button is located in the bottom right corner of the front panel.

LabVIEW Example



Read RFID Tag with C#

```
using System.IO.Ports;
```

```
SerialPort port = new System.IO.Ports.SerialPort("COM4", 2400, System.IO.Ports.Parity.None,  
8, System.IO.Ports.StopBits.One);
```

```
port.Open();  
port.DtrEnable = true;
```

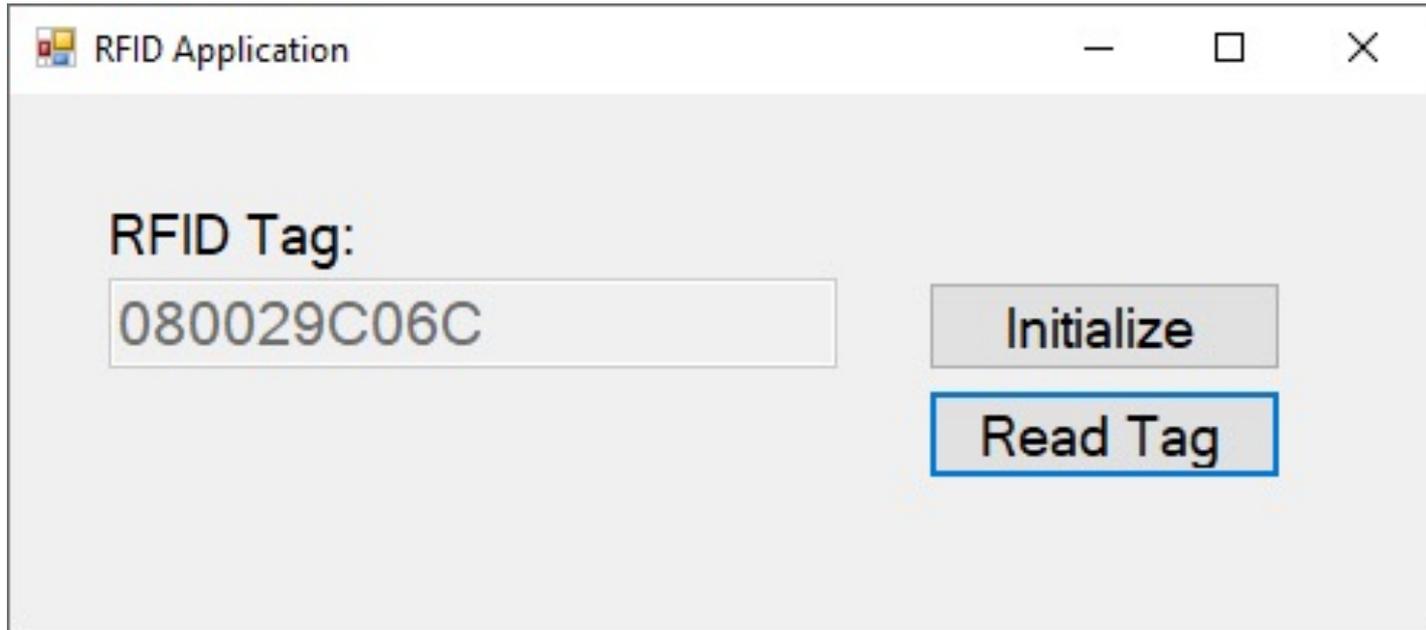
```
int numberBytesToRead = 12;  
byte[] data = new byte[numberBytesToRead];  
port.ReadTimeout = 1000;  
port.Read(data, 0, numberBytesToRead);
```

```
string rfidTag;  
rfidTag = System.Text.Encoding.UTF8.GetString(data, 0, data.Length);
```

```
rfidTag = rfidTag.Replace("\n", "");  
rfidTag = rfidTag.Replace("\r", "");
```

```
port.Close();
```

Visual Studio/C# Example



C# Example

```
using System;
using System.IO.Ports;
using System.Windows.Forms;

namespace ReadRfidApp
{
    public partial class Form1 : Form
    {
        string rfidTag;
        SerialPort port = new System.IO.Ports.SerialPort("COM4", 2400, System.IO.Ports.Parity.None, 8, System.IO.Ports.StopBits.One);

        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {}

        private void btnInitialize_Click(object sender, EventArgs e)
        {
            port.Open();
            port.DtrEnable = true;

            txtTagData.Text = "";
        }

        private void btnReadTag_Click(object sender, EventArgs e)
        {
            int numberBytesToRead = 12;
            byte[] data = new byte[numberBytesToRead];
            port.ReadTimeout = 1000;
            port.Read(data, 0, numberBytesToRead);

            rfidTag = System.Text.Encoding.UTF8.GetString(data, 0, data.Length);

            rfidTag = rfidTag.Replace("\n", "");
            rfidTag = rfidTag.Replace("\r", "");

            txtTagData.Text = rfidTag;

            port.Close();
        }
    }
}
```



Excel Technology Ltd

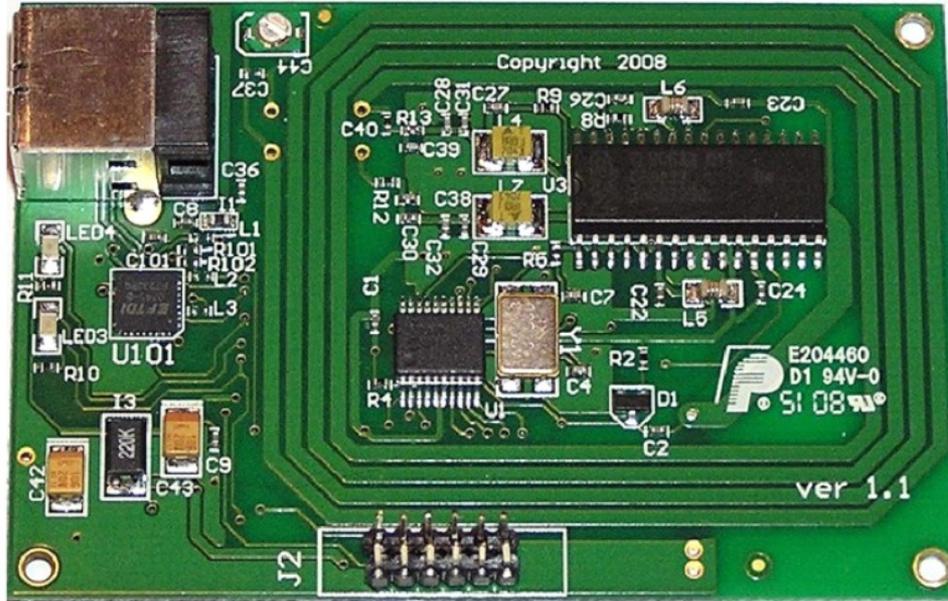
OEM-MICODE-USB RFID

Reader

RFID 13.56MHz

Eccel Technology Ltd OEM-MICODE-USB (000128) RFID Reader

Board Case



RFID Tags



RFID standard:
13.56MHz ISO14443-A
MIFARE Classic Card 1K 13.56mhz

USB A-B Male Cable 1m



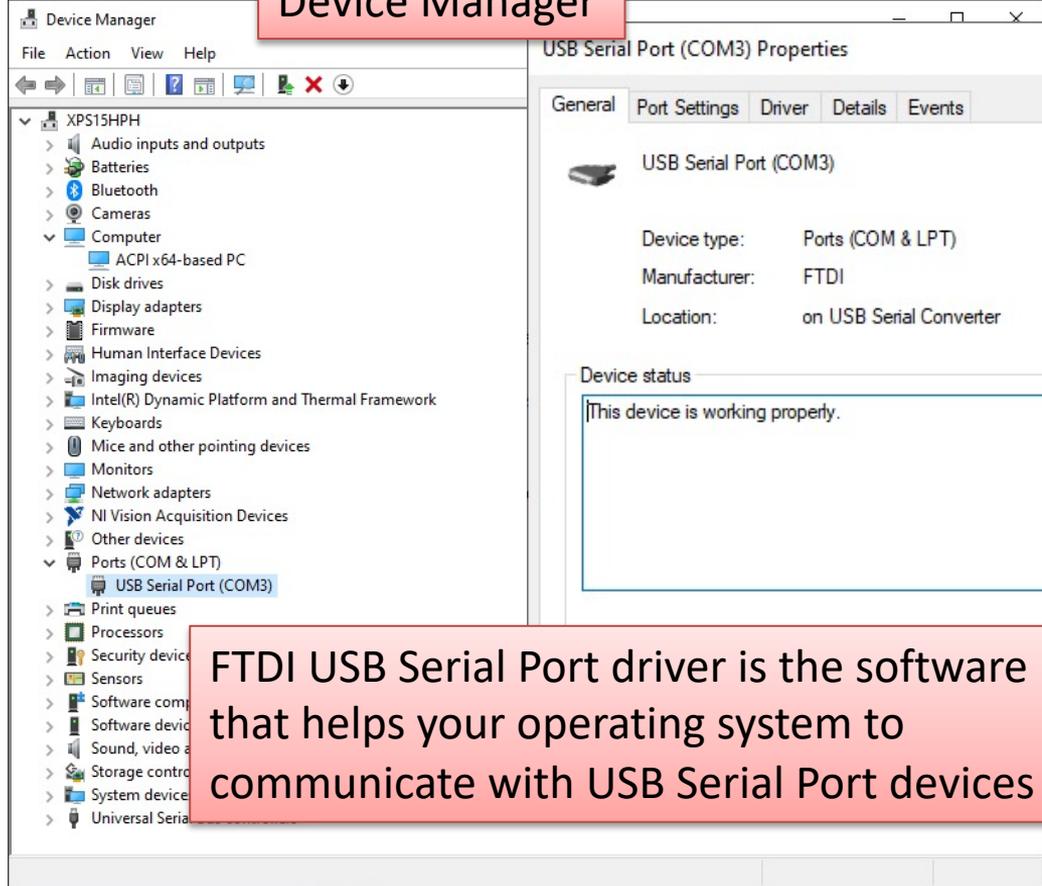
RS Online: <https://no.rs-online.com/web/p/rf-modules/1262181/>

Setup and Configuration

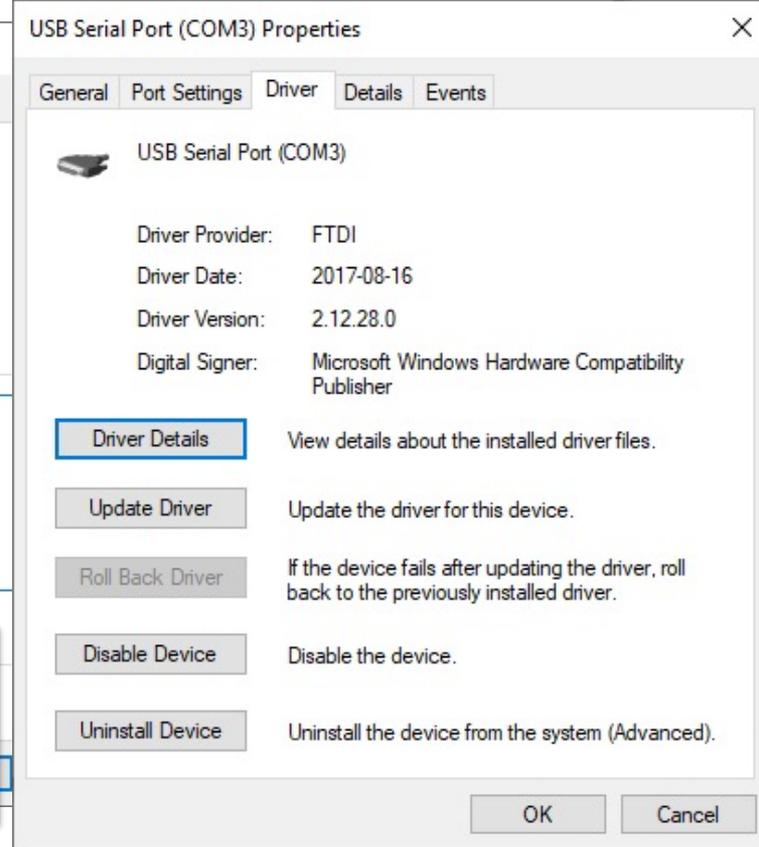
- Connect the Device to your PC using the USB Cable
- Open the **Device Manager** in Windows and find the allocated **COM Port** for the device
- Make sure the Device and the **FTDI USB Serial Port driver** is installed properly
- Install the Configuration and Test Software (Micro RWD MFIC) from <https://eccel.co.uk/product/oem-micode-usb/>
- Read the Datasheet
- Start developing a Test application that can read data from the RFID reader

Setup and Configuration

Device Manager



FTDI USB Serial Port driver is the software that helps your operating system to communicate with USB Serial Port devices

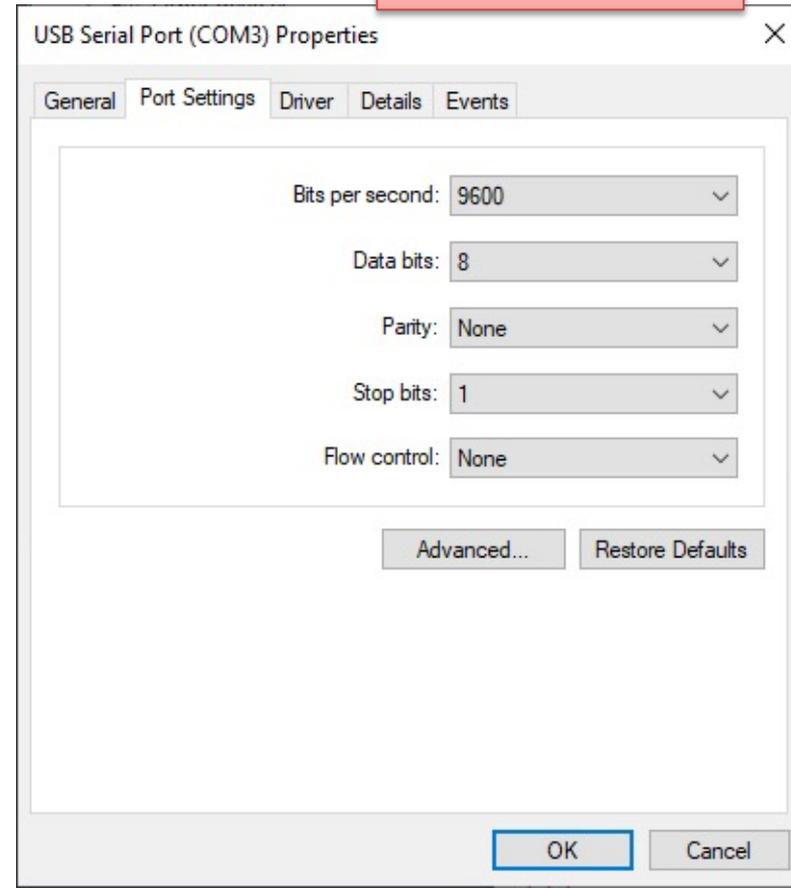


Excel RFID Reader

Device Manager

Communication (from the Datasheet):

- 9600 baud
- 8 bits
- 1 stop
- No parity



Configuration and Test Software

MicroRWD ICODE/Mifare Combination Reader

File Configure Window Help

MicroRWD MFIC RFID Reader Module

Supports Mifare (1K, 4K and Ultralight) and ICODE SLI transponder/tag types

Click the appropriate image to configure

Mifare Mode

ICODE Mode

MicroRWD Status: **CONNECTED**

Exit

MicroRWD ICODE/Mifare Combination Reader

File Configure Window Help

Card RWD Keys RWD Parameters

Mifare Memory

Block	Data (HEX)	ASCII	Description
00	----	Serial no. + Mfr. data
01	----	User Data
02	----	User Data
03	----	Keys A/B + Access Bits

Do not change Sector Trailer Blocks or RWD Keys unless Mifare operation is understood

Serial Number: 6B D9 82 47 00 00 00

Read Card Write to Card

RWD Status

Continuous Poll Poll Now

Status: Binary 1 0 0 0 0 1 1 0 Hex 86

EEPROM error
 Card OK
 Rx OK
 RS232 Error

Card Type: 1K 4K Ultralight

MFRC error

Exit

Configure Tx Output

- The default for the OEM-Micode and RWD products is to output the received UID number on the OP0 pin. (Connector J2, Pin 2 on the OEM products).
- If you want the automatic output to be redirected to the TX pin of the serial port instead, then you must program a control byte from its factory default to do this.
- See datasheet, page 12). Link to Datasheet:
https://eccel.co.uk/wp-content/uploads/2018/05/MF_ICBprot_030518.pdf
- If the UID automatic output is redirected to the TX pin, then there will be no acknowledge byte sent by the reader after you send any commands to it. This is to avoid data clashes with the automatic UID transmission.
- To change the direction of the UID output to the TX pin you have to program byte 9 of the EEPROM control registers to 0x01.
- So, send a command string as follows : 0x50, 0x09, 0x01.
- You will receive no acknowledge but after presenting a card/tag, you should receive the UID back on your terminal screen.



Configure Tx Output

MicroRWD ICODE/Mifare Combination Reader

File Configure Window Help

Card RWD Keys RWD Parameters

MicroRWD Memory

Page	Description
00	Polling Delay (SLEEP / Power down) period
01	Aux data output
02	*Reserved* (Checksum)
03	Mifare/ICODE option byte
04	Wiegand parity option
05	Aux block address on card
06	Key no./type for read of aux data
07	Beep delay parameter (x40mS)
08	Aux output source data selection
09	Aux out (serial data) redirection
0A	Aux output serial format (Hex or ASCII)
0B	Aux output byte order option
0C	Card 12 (MS byte)
0D	Card 12 (Byte 2)
0E	Card 12 (Byte 1)
0F	Card 12 (LS byte)
10	Card 16 (MS byte)
11	Card 16 (Byte 2)
12	Card 16 (Byte 1)

Write to MicroRWD

RWD Status

Continuous Poll Poll Now

Status: Binary Hex

1 0 0 0 0 0 0 80

EEPROM error
 Card OK
 Rx OK
 RS232 Error
 MFRC error

Card Type
 1K 4K
 Ultralight

Exit

To change the direction of the UID output to the TX pin you have to program byte 9 of the EEPROM control registers to 0x01

Write To MicroRWD Memory

WARNING:
Changing locations marked as *Reserved* may render the MicroRWD temporarily inoperable.

Address: 09 Description: Aux out (serial data) redirection Hex: 01 ASCII:

Close Write

From the Datasheet:

Byte 9: Auxiliary output switch (redirects serial o/p)

0x00 = Aux output from OP0 pin (default)

0x01 = Aux output from Tx pin

RealTerm

- RealTerm is a tool for capturing, entering and debugging Serial Communication
- RealTerm is a very old program
- RealTerm is available to download from SourceForge:
- Use RealTerm in combination with the Datasheet for the device to learn more about the communication protocol used for the device

<https://sourceforge.net/projects/realterm/files/Realterm/>

<https://learn.sparkfun.com/tutorials/terminal-basics/real-term-windows>

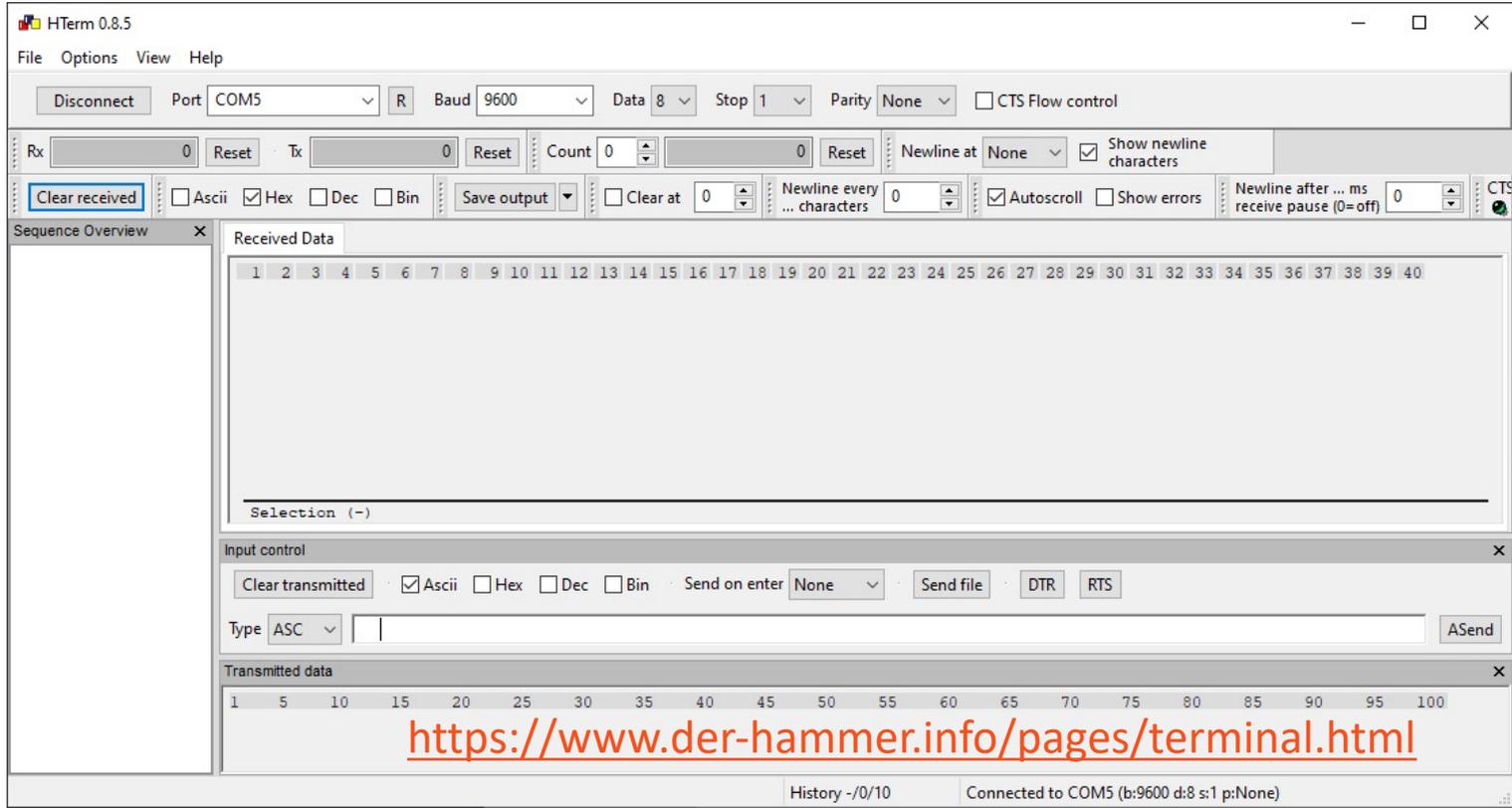
YAT

- Another program is YAT. It has a more modern graphical interface than RealTerm.
- YAT is a tool for capturing, entering and debugging Serial Communication, etc.
- YAT is available to download from SourceForge:
<https://sourceforge.net/projects/y-a-terminal/files/>
- Use YAT in combination with the Datasheet for the device to learn more about the communication protocol used for the device

<https://learn.sparkfun.com/tutorials/terminal-basics/yat---yet-another-terminal-windows>

HTerm

Another Terminal Program like RealTerm and YAT



Hterm – Check Device

The RFID reader then respond with information about him self

Enter "z"

The screenshot shows the Hterm 0.8.5 terminal window. The top menu bar includes File, Options, View, and Help. Below the menu is a toolbar with a Disconnect button, a Port dropdown set to COM3, a Baud rate dropdown set to 9600, Data bits set to 8, Stop bits set to 1, Parity set to None, and a checked CTS Flow control checkbox. Below the toolbar is a status bar showing Rx: 103, Tx: 2, Count: 0, and Newline at: None. The main terminal area displays the following text:

```
1 5 10 15 20 25 30 35 40 45 50 55 60 65 70 75 80 85 90 95 100 105  
m IDE RWD Mifare ICODE (SECMFI_CWAX_LP V3.04 09/08/11) Copyright: IB Technology (Eccel Technology Ltd) w
```

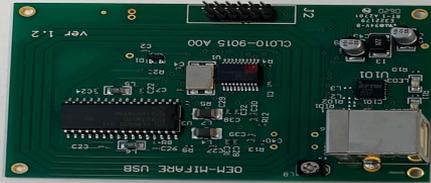
Below the terminal area is an input control section with a Clear transmitted button, a checked Ascii checkbox, and Send on enter set to None. The input field contains the text "type ASC" and a cursor. Below the input control is a Transmitted data section showing the text "zz". The status bar at the bottom indicates "History -/2/10" and "Connected to COM3 (b:9600 d:8 s:1 p:None cts)".



Code Examples

Excel OEM-MICROCODE-USB RFID Reader

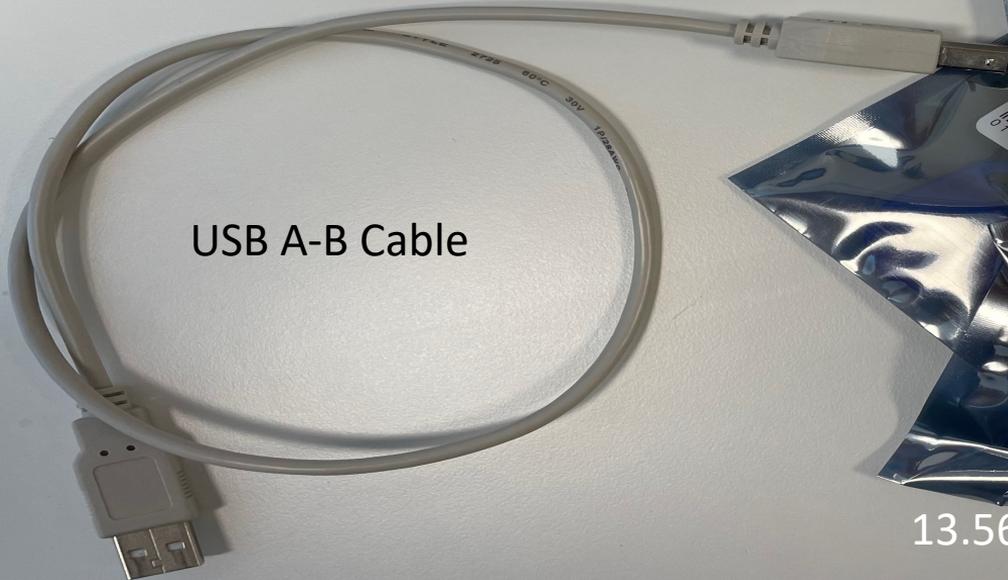
RFID Reader with built-in Antenna



Case for RFID Reader



USB A-B Cable



13.56MHz ISO 14443-A (Mifare Classic 1k) Tags

Code Examples

Note!

- The examples provided can be considered as a “proof of concept”
- The sample code is very simplified for clarity and doesn't necessarily represent best practices.

LabVIEW

Eccel RFID Reader.vi

File Edit View Project Operate Tools Window Help

Run Stop Pause

RFIDTag

D58A5047

Bytes Array

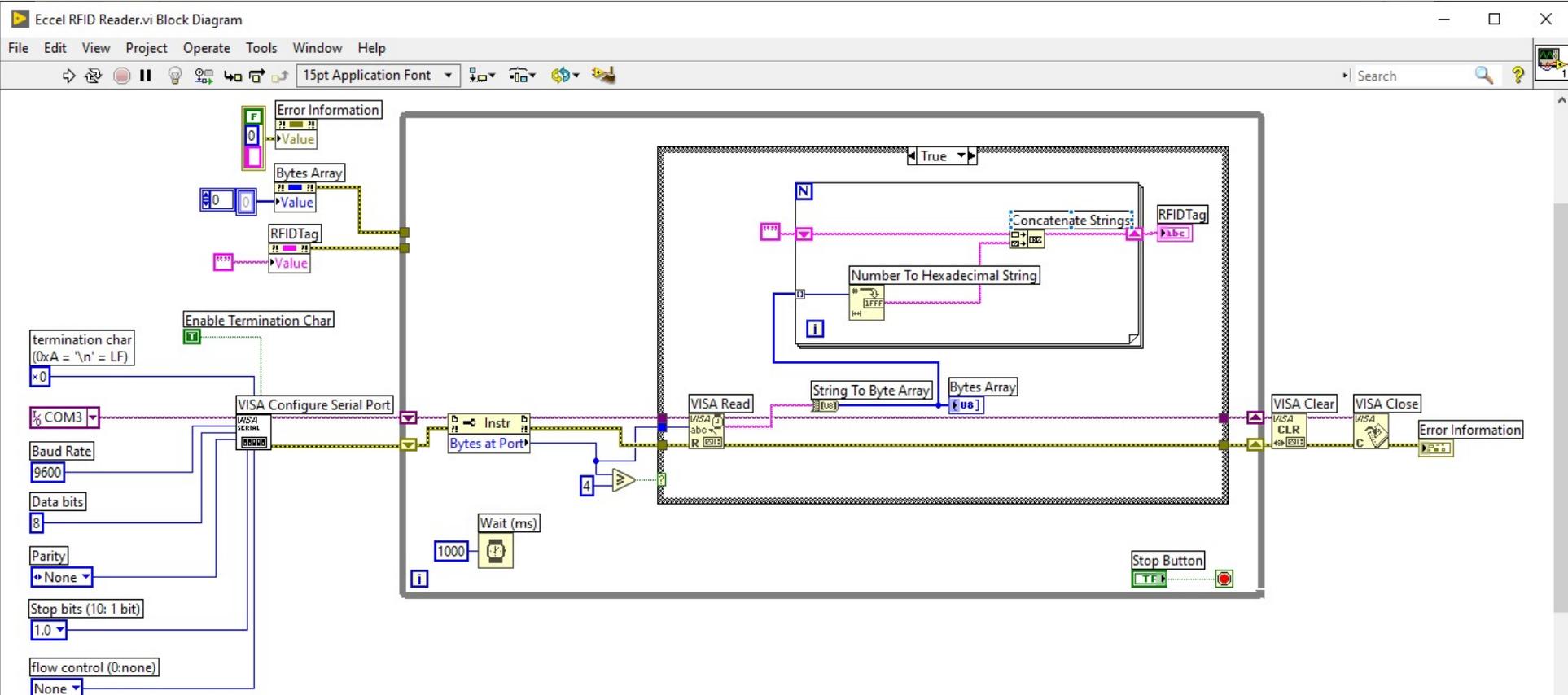
x D5	x 8A	x 50	x 47	x 0	x 0	x 0	x 0
------	------	------	------	-----	-----	-----	-----

Error Information

status	code
	0
source	

Stop

LabVIEW



Python

The image shows a screenshot of the Thonny Python IDE. The window title is "Thonny - C:\Users\hansha\OneDrive\Programming\Visual Studio Examples\RFID\Eccel RFID Reader\Python\rfid_lo...". The menu bar includes "File", "Edit", "View", "Run", "Tools", and "Help". The toolbar contains icons for file operations, running, and stopping. The editor shows two tabs: "rfid_loop_ex.py" and "test.py". The code in "rfid_loop_ex.py" is as follows:

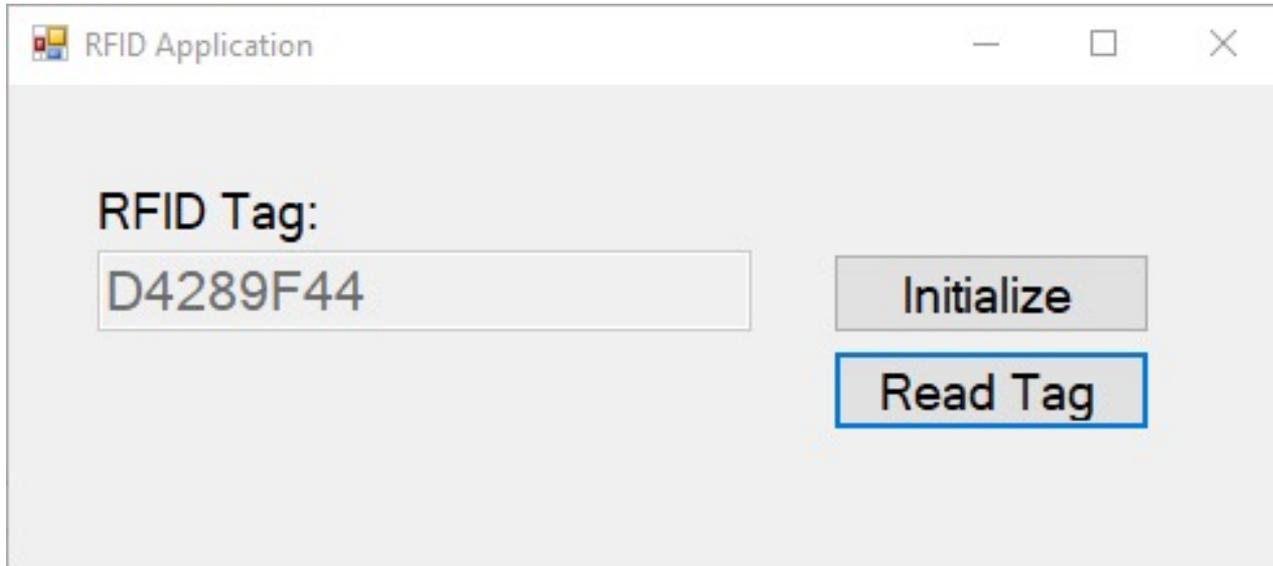
```
1 import serial
2 import time
3
4 ser = serial.Serial('COM3', 9600, timeout=1)
5
6 while True:
7     response = ser.read(4)
8
9     if response != "":
10        hexvalue = "".join(map(hex, response))
11        hexvalue = hexvalue.replace("0x", "", 4)
12        hexvalue = hexvalue.upper()
13        print(hexvalue)
14
15    time.sleep(1)
16
17 ser.close()
```

The Shell window shows the execution of the script:

```
Python 3.7.9 (bundled)
>>> %Run rfid_loop_ex.py
D58A5047
6BD98247
BB8FA847
```

Python 3.7.9

Visual Studio/C#



```
using System;
using System.IO.Ports;
using System.Windows.Forms;
```

```
namespace ReadRfidApp
```

```
{
    public partial class Form1 : Form
    {
        string rfidTag;
        SerialPort port = new System.IO.Ports.SerialPort("COM3", 9600, System.IO.Ports.Parity.None, 8, System.IO.Ports.StopBits.One);

        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
        }

        private void btnInitialize_Click(object sender, EventArgs e)
        {
            port.Open();
            port.DtrEnable = true;
            txtTagData.Text = "";
        }
    }
}
```

```
private void btnReadTag_Click(object sender, EventArgs e)
{
    int numberBytesToRead = 4;
    byte[] data = new byte[numberBytesToRead];
    port.ReadTimeout = 1000;
    port.Read(data, 0, numberBytesToRead);

    rfidTag = "";
    for (int i = 0; i < numberBytesToRead; i++)
    {
        rfidTag = rfidTag + data[i].ToString("X");
    }

    txtTagData.Text = rfidTag;

    port.Close();
}
```

Resources

- <https://en.wikipedia.org/wiki/Barcode>
- https://en.wikipedia.org/wiki/Radio-frequency_identification
- <https://www.atlasrfidstore.com/rfid-beginners-guide/>
- <https://no.rs-online.com/web/p/rf-modules/1262181/>
- <https://eccel.co.uk/product/oem-micode-usb/>

Hans-Petter Halvorsen

University of South-Eastern Norway

www.usn.no

E-mail: hans.p.halvorsen@usn.no

Web: <https://www.halvorsen.blog>

