TRUDU, Laurent

FXP20 DEMO POS

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Author | Version | Comments |
| 28th March 2025 | Laurent Trudu | 1.0 | Initial Version |
|  |  |  |  |

[1. Overview 5](#_Toc199335234)

[2. Software considerations 6](#_Toc199335235)

[1. Distribution 6](#_Toc199335236)

[2. Licence 6](#_Toc199335237)

[3. Pre-requisites 6](#_Toc199335238)

[4. Screen resolution 7](#_Toc199335239)

[5. Documentation and reading 7](#_Toc199335240)

[6. One drive folders 7](#_Toc199335241)

[7. Architecture 8](#_Toc199335242)

[3. Demo Workflow 9](#_Toc199335243)

[1. Welcome Screen 9](#_Toc199335244)

[2. Basket Screen: Waiting for products 10](#_Toc199335245)

[3. Basket screen: Displaying detected products 11](#_Toc199335246)

[4. Empty Basket 12](#_Toc199335247)

[5. Settings screen 13](#_Toc199335248)

[1. Adding a new product 14](#_Toc199335249)

[2. Modifying a product line in the database 15](#_Toc199335250)

[3. Deleting a product entry 15](#_Toc199335251)

[4. Setting up the demo 16](#_Toc199335252)

[1. Installation 16](#_Toc199335253)

[2. Mosquitto setup 16](#_Toc199335254)

[3. FXP20KeyInjector configuration 17](#_Toc199335255)

[1. COM Port 17](#_Toc199335256)

[2. Antenna Config 17](#_Toc199335257)

[3. Reader Param 18](#_Toc199335258)

[4. MQTT Protocol 19](#_Toc199335259)

[5. Saving the configuration 20](#_Toc199335260)

[6. Config.xml file 20](#_Toc199335261)

[5. Running the Demo 21](#_Toc199335262)

[6. Running the demo without physical objects 21](#_Toc199335263)

[7. Appendix 23](#_Toc199335264)

[1. Farraday cage metallic basket 23](#_Toc199335265)

[2. Pre-configured EPCs 24](#_Toc199335266)

# Overview

The FXP20 DEMO POS is a demonstration that allows to showcase a POS user experience using the RFID technology with the FXP20 reader.

This demonstration shows how a FXP20 can be used to detect the content of a basket and provide a seamless payment experience to the user.

The application uses on the FXP20KeyInjector to retrieve EPCs from the FXP20 reader.

The protocol MQTT is used between the demonstration and the reader.

It allows the application to control the FXP20KeyInjector as well as the reader.

# Software considerations

## Distribution

Please note that this software must not be shared with partners and must remains internal only for demonstration purposes.

Contact me if you have any questions about its distribution: [laurent.trudu@zebra.com](mailto:laurent.trudu@zebra.com)

## Licence

The software is available under the Zebra EULA License.

A copy of the license is available in the One Drive folder that contains the demo:

Or directly on the Zebra Devs repository:

[**https://github.com/ZebraDevs/About/blob/master/ZEBRA\_EULA\_LICENSE.md**](https://github.com/ZebraDevs/About/blob/master/ZEBRA_EULA_LICENSE.md)

## Pre-requisites

Hardware:

* A PC with usb >= 2.0 ports
* A FXP20 connected to one of the USB ports of your PC (the green light should not be blinking, and the blue light should be blinking)

Software:

* The application has been tested on Windows 10 and Windows 11 and should work on Windows server as well.
* .NET Framework runtime >= 4.8 ([Download Link for .NET Runtime 4.8](https://dotnet.microsoft.com/en-us/download/dotnet-framework/net48))
* MQTT Broker. The Mosquito MQTT Broker version 2.0.20 has been tested successfully ([Download Mosquitto MQTT](https://mosquitto.org/download/))
* The KLite Codec Pack to allow the display of the POS video inside the demonstration, the full release has been tested successfully ([Download KLite Codec Pack](https://www.codecguide.com/download_kl.htm))

All the necessary software can be found in the “SoftwareToInstall (Dependencies)” folder inside the FXP20DemoPos One Drive folder.

## Screen resolution

To work properly, the screen resolution of the host PC must be setup to Full HD 1920x1080.

The font scaling must be set to 100%.

The layout must be set to Landscape.

A screenshot of a computer

AI-generated content may be incorrect.

## Documentation and reading

Please first read the FXP20KeyInjector documentation for information on how to set it up and run it.

It contains information on how to setup the Mosquitto MQTT Broker as well.

This document supposes that you have read this documentation before.

## One drive folders

The FXP20KeyInjector is available in the folder: [FXP20KeyInjector](https://zebra-my.sharepoint.com/:f:/p/hpd438/Esi349jFlqZOvExvICAGQIoBNA9z0OzJzbX4c8sace3iRQ?e=jKYCVV)

The FXP20DemoPOS software is available in the folder: [FXP20PosDemo](https://zebra-my.sharepoint.com/:f:/p/hpd438/EvVS4OmGrNxFoZON8hhB5KgBDbpJ3UXWJ1rglQcJUrenIA?e=4Wa5bC)

## Architecture

The application needs to have access to a MQTT broker to get a connection with the FXP20KeyInjector and then the Reader.

The address and the login/password are set inside the configuration of the FXP20 Key Injector.

The FXP20 Key injector will register itself to a specified control topic and will send all the EPC read to specific data topic.

They must be defined as stated in this document.

The MQTT Broker will dispatch the EPC read to the registered application and will dispatch the commands received from the applications to the FXP20 Key Injector app.

The MQTT Broker can be launched on the host computer (localhost) as well as in a separate server.

FXP20 DEMO POS

MQTT Broker

RFID SDK

FXP20 Key Injector

FXP20

Control commands

EPC data channel

EPC data topic

Control topic

EPC read

# Demo Workflow

The FXP20DemoPos has a workflow that tries to simulate a POS experience with RFID.

## Welcome Screen

When launched, the application automatically shows the Welcome Screen.

A person holding a tablet

AI-generated content may be incorrect.

From this screen, you can:

* 1: Launch the Demo by touching or clicking on the button Start
* 2: Quit the Demo with password Zebr@123
* 3: Setup the Demo with password Zebr@123

Click on Start to go to start the demo to go to the basket screen.

## Basket Screen: Waiting for products

A screenshot of a video game

AI-generated content may be incorrect.

This screen appears to invite you to put some products inside the basket.

On the demo screen, you’ll have access to:

* 1: The number of items detected.
* 2: The total price of all the detected items.
* 3: A Checkout button that you can use at any time to exit the demo workflow.

## Basket screen: Displaying detected products

Once the basket screen is opened, the reader starts reading for EPC Tags.

If a registered tag is detected, the associated product will be inserted inside the basket list.

The number of products will be increased, and the total price will be updated.

A screenshot of a screen

AI-generated content may be incorrect.

Once all products have been detected, you can click on the Checkout button.

## Empty Basket

The demo does not showcase a checkout process.

When the user touch or click on checkout, it will display the empty basket screen.

A blue basket with a green arrow pointing up

AI-generated content may be incorrect.

To go back to the welcome screen, simply click or touch the green arrow or the basket.

## Settings screen

To open the settings screen, click on the FXP20 picture on the top left of the welcome screen.

Enter the password Zebr@123 and you’ll see the following interface:

A screenshot of a video game

AI-generated content may be incorrect.

This screen contains the following elements to manage the products database:

* 1: The database of products displayed as a list.
* 2: Add button to insert new product.
* 3: Delete button to erase the selected product line in the product database.
* 4: Modify button to change the content of the selected product in the product database.
* 5: Settings button to open the settings window.

These elements are for debugging purposes:

* 6: Connect the FXP20KeyInjector to the reader. It will check the MQTT connection and the ability to control the FXP20KeyInjector. If the MQTT message is received by the reader and the FXP20KeyInjector properly configured (especially the COM port), the “Connect” button of the FXP20KeyInjector app should appears greyed and the “Disconnect” button should appear as available.
* 7: Disconnect the FXP20KeyInjector to the reader. When disconnected, the “Disconnect” button of the FXP20KeyInjector application gets greyed and the “Connect” button gets available.
* 8: Start Reading. The FXP20KeyInjector must be connected before trying to start reading tags. When the MQTT message is properly received by FXP20KeyInjector, it should setup the reader for tag reading. The blue light should start to blink rapidly.
* 9: Stop Reading. Will stop the reader to read tags. The blue light should stop blinking rapidly and start blinking slowly.

Two more buttons are available:

* 10: POS Demo will let the demo return to the welcome screen.
* 11: Quit will quit the demo without the need to enter a password.

### Setting the application locale

To set the locale, click on the Settings button.

A blue line on a white surface

AI-generated content may be incorrect.

Change the localisation in the drop-down list.

It will change the language of the GUI and the currency used by the application.

The choices are:

* Europe : Texts in English, Currency €
* France : Texts in French, Currency €
* UK : Texts in English, Currency £
* USA : Texts in English, Currency $

Click on the Save button to save the configuration and update the GUI elements.

### Adding a new product

To Add a new product, click on the Add button.

A screenshot of a computer

AI-generated content may be incorrect.

The following data can be set:

* 1: The product name as it will appear in the basket products list
* 2: The price of the product that will be used to calculate the total price of the basket
* 3: The EPC of the RFID tag attached to the product
* 4: Double click on this zone to add/modify the product picture

Example of added product:

A screenshot of a computer screen

AI-generated content may be incorrect.

Once you have entered the data and set an image you can click on the Add button to insert a new line in the products database.

### Modifying a product line in the database

To change the values associated with a product, you can directly update the following values in the product list:

* Product Name
* Price
* EPC

As soon as you will change the focus of the modified input element, the database will be updated.

If you want to change the picture associated with the product, you must:

* Select the line of the product in the list.
* Click on the button Modify.
* A dialog will appear similar to the Add dialog:  
  A screenshot of a computer

  AI-generated content may be incorrect.  
  You can change all the values in their respective input field.  
  To change the image, double click on it and it will open a browser to select the new image.
* Click on the update button to change the values of the product entry in the database.

### Deleting a product entry

To delete a product entry, select the line of the product inside the database list.

Click on the Delete button and the product will be erased from the database.

There is no dialog to validate if you are sure or not, so be careful when using the Delete button.

# Setting up the demo

## Installation

Install the .NET Framework 4.8 to the target PC.

Unzip the FXP20 Key Injector archive in a folder on the PC that will run the application (read the FXP20KeyInjector documentation for information on how to setup and run the application).

Install the MQTT broker on the target PC or on a separate server depending on your needs (read FXP20KeyInjector documentation for more information).

Sample config file and launcher for Mosquitto MQTT Broker are provided in the documentation of the FXP20KeyInjector, and in the OneDrive folder containing the application in the file MosquittoMQTTBrokerLauncher.7z.

Install the KLite Codec Pack on the host computer.

Unzip the FXP20DemoPOS archive on the host PC hard drive.

A version of the Mosquitto MQTT Broker and KLite Codec Pack can be found in the One Drive folder of the FXP20DemoPos software.

## Mosquitto setup

Before running the demo, you need to install and run Mosquitto MQTT Server (or any other MQTT server as long as you respect the following user/password combination for login).

Please read the section “Mosquitto MQTT setup and sample files” of the FXP20KeyInjector documentation on how to setup Moquitto server.

The demo’s user and password data are hardcoded inside its code.

To work, the following user must be added to the Mosquitto server passwd file:

Login: user

Password: sko

Once the Mosquitto server has been setup, you can run it to test FXP20KeyInjector configuration.

Alternatively, if you put the Mosquitto bin folder in your path as explained in the FXP20KeyInjector documentation, you can :

* Download the MosquittoMQTTBrokerLauncher.7z file from the FXP20KeyInjector one drive folder
* Unzip the archive
* Execute the runMosquitto.bat batch file

This archive is pre-configured with the login and password required for this demonstration.

## FXP20KeyInjector configuration

Read the documentation of FXP20KeyInjector to understand how to setup the FXP20 Reader.

If you don’t have a Config.xml file inside the FXP20KeyInjector folder, run it to create a new file.

### COM Port

Ensure that the FXP20 reader is connected to a USB port, and that it is plugged in.

Start the FXP20KeyInjector.

It will discover the COM port on which the reader is connected.

Select this COM port in the drop-down list “Com Port”.

To check if the right COM port has been selected click on the Connect button.

If everything runs well, the blue light of the FXP20 should light up and blink slowly.

Click on the Start Reading button and the blue light should start blinking rapidly.

Now you can stop the reading, but keep the reader connected as you will need it to setup the antenna(s) and the reader parameters.

### Antenna Config

To change the antenna configuration, click on the “Antenna Config” button in the FXP20KeyInjector app.

The recommended antenna settings are:

A screenshot of a computer

AI-generated content may be incorrect.

It is advised that you only change the Transmit Power according to your needs.

The current transmit power of 2000 may be too high and detect your objects even when they are not on top of the FXP20.

To fix this problem, it is recommended to

* Test different values until you find the best one that will detect all your product.
* Use a basket made of metal to do a Faraday cage and protect the products from being read by the FXP20 (see appendix for a picture of the type of basket we recommend you use)

### Reader Param

To change the reader parameters, click on the “Reader Param” button in the FXP20KeyInjector app.

The recommended parameters are:

A screenshot of a computer

AI-generated content may be incorrect.

You can change these parameters, but to work properly, the reader should be in a mode that read the EPC continuously.

The demonstration maintains an internal list of already read EPC to ensure that only one entry is added to the basket.

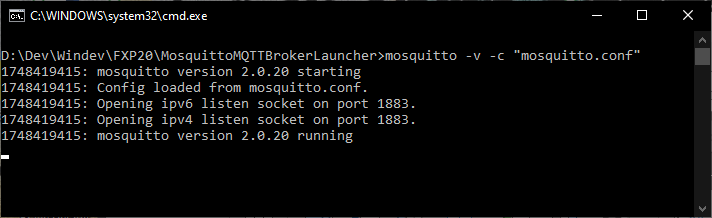
So, if the same EPC is read many times, it will appear only once inside basket.

Any new EPCs will be added to the internal list of already read EPC and added to the basket as well.

### MQTT Protocol

Before setting the MQTT protocol, launch the MQTT Server to get debug information, and to see if the FXP20 connects to it.

Here is the command window you should have if you launch the Mosquitto MQTT server in verbose mode:



Now you can change the protocol on the FXP20KeyInjector interface to MQTT in the “Send Protocol” drop down list.

The following information should appear in the command window:

A computer screen with white text

AI-generated content may be incorrect.

It will show you that a new client has connected to the MQTT server, and that it has subscribed to the FXP20/Control topic.

Wait a little bit, and you should see keep alive messages appearing inside the command window:

A screen shot of a computer

AI-generated content may be incorrect.

If you launch the MQTT Server after launching the FXP20KeyInjector, and if the latter has been configured for MQTT protocol, it will fail connecting as no server has been launched.

To test the connection, you can switch to either KeyInjection or ClipboardPaste, then switch back to MQTT to force the application to connect to the server.

### Saving the configuration

Once everything has been properly set, you can click on the “Save Config” button to update the Config.xml file.

### Config.xml file

The Config.xml file that is provided in the FXP20KeyInjector archive is pre-setup for this demo to run on a local MQTT Server.

There are some parameters that **MUST** be set to the following values for the demo to work properly:

* MQTTUser: user
* MQTTPassword : sko
* MQTTSendTopic : FXP20/EPC
* MQTTControlTopic : FXP20/Control
* Protocol: MQTT
* AutoConnect : false
* AutoStartReading : false
* BeepOnRead : false
* HookKeyboardToStartReading : false
* PerformReadingWithGPIO : false

By default, the application is setup to work with a local MQTT Server:

* MQTTServer: 127.0.0.1
* MQTTPort: 1883

You can change these parameters if your server runs on a separate host or on a different port.

# Running the Demo

Once everything has been setup properly, you **MUST** launch the software in that order to get the demo work properly:

* Launch the MQTT Server  
  You can use runMosquitto.bat from the MosquittoMQTTBrokerLauncher.7z archive if you have setup Mosquitto MQTT server to run on the PC that hosts the demo.
* Launch the FXP20KeyInjector.exe application.  
  The server verbose or logs should show you that a new client has appeared and has subscribed to the FXP20/Control topic.
* Launch FXP20DemoPOS.exe

Now you can update the database to your needs and run the demonstration!!!

# Running the demo without physical objects

If you don’t want to update the database with your own EPCs or if you don’t have any physical products to show for the demonstration, you can make it work with RFID tags only.

Check the appendix to get the list of the pre-configured EPCs of the demo.

You can use this list and write RFID tags with theses EPCs to demonstrate the POS workflow.

The visuals can be found in the OneDrive folder so you can print card boards with them, stick the RFID tag on the back of the cardboard and you are ready to go for the demo.

# Appendix

## Farraday cage metallic basket

To prevent the reader to read EPCs that are waiting to be put on the FXP20, we recommend you put your objects in a basket like this:

A wire basket with a handle

AI-generated content may be incorrect.

This will act like a faraday cage and prevent your objects to be detected too early in the demonstration workflow.

## Pre-configured EPCs

Here is a list of the preconfigured EPCs of the demo:

|  |  |  |  |
| --- | --- | --- | --- |
| Product Name | Price | EPC | Visual |
| Aloha Shirt | 25,00€ | 966801000000000000000001 |  |
| Black Dress Shirt | 45,00€ | 966804000000000000000005 |  |
| T-Shirt Black | 10,00€ | 966804000000000000000002 |  |
| T-Shirt Blue | 12,00€ | 966804000000000000000003 |  |
| T-Shirt Red | 8,00€ | 966804000000000000000004 |  |
| T-Shirt Long Sleeves | 16,00€ | 966804000000000000000001 |  |