Fsk Encoder Mpf1Extension user guide

October 2025

Contents

Introduction

The Fsk Encoder application is a usefull tool for SW development in retro computing enviornment.
It's capable of converting binary code or data files into FSK encoded sound samples which can be played on the computers sound card. Together with an appropriate interconnection cable the sound output of the host computer can be connected to the sound input of a retro computer system to upload the data.

The extension (*Z80TrainerExtension*) enables the FskEncoder application to upload data to the **Multitech Microprofessor-I** (Mpf-1) board. It does the conversion of the source data to sound samples according to the *protocoll* designed by Multitech.

1 Description

A so called *protocoll* defines the frequencies and the encoding rules to be used for binary zeros and ones. Please refer to Appendix A for detailed information on this topic.

2 GUI

The Z80TrainerExtension's specific GUI panel is shown in the *Target specific information area* of the FskEncoder main GUI and offers a way to enter the so called *Program Number* of the upload candidate.

The *Program Number* is a 4-digit hexadecimal number, used to identify a program on tape in the earlyer days. In FskUploader there is only one file as upload candidate and the *Program Number* can easily be left at the default setting.

Note:
Please consult the Z80-Trainer user manual for more information how to start an upload on the boards monitor system.

The layout of the panel is shown in the picture on the right..

3 Installation

The Z80TrainerExtension comes as *Z80TrainerExtension-m.s.b.zip* file containing all the necessary folders and files, which must be unpacked in the installation folder of the FskEncoder application.

Note: The .zip file contains a version code consisting of

m the main line of the build,

s the stream of the build and

b the build number.

The main line of the extension **must** match the one of the application to function propperly.

The extension .zip file provides confiruration snippets which must be merged into the FskEncoder configuration to make the extension accessible (see next section).

4 Configuration

After unpacking the .zip file, three new files apear in the installation directories:

1. `./bin/Z80TrainerExtension.bat` containing the Java `CLASS_PATH` extension and must be merged into the `./bin/FskEncoder.bat` file,
2. `./cfg/Z80TrainerExtension.properties` containing a template for the configuration of the `./cfg/Plugin.properties` file.
3. And at least the extension file *Z80TrainerExtension.jar* in the `./extension` directory.

Both, the .bat and .properties files contain instructions on how to merge the extensions into the FskEncoder configuration.

A Z80-Trainer encoding protocol

```
SEL Z80-Trainer tape format


Bit format
-------------------------------------------------------------------------------

'0'   1 cycle   568,18Hz (1760µs)
'1'   1 cycle  1136,36Hz ( 880µs)


Envelope (Byte format)
-------------------------------------------------------------------------------

  1 start bit '0'
  8 data bits, lsb first (b0 to b7)
  3 stop bit '1'


File format
-------------------------------------------------------------------------------

 1.    12'288   bit '1'   Lead sync                        ->leadIn

 2.         1   bit '0'   Measurement                     ->syncPatern
 3.        16   bit '1'     for period length            +>

 4.         1   envlp     Programm number - high byte     ->programNumber
 5.         1   envlp     Programm number - low byte      +>

 6.         1   envlp     Start address - high byte       ->startAddress
 7.         1   envlp     Start address - low byte        +>
 8.         1   envlp     Start address checksum          +>

 6.         1   envlp     Data block length - high byte   ->dataBlockLength
 7.         1   envlp     Data block length - low byte    +>
 8.         1   envlp     Data block length checksum      +>

 9.        16   bit '1'   Idle time for chksum calculation  ->idleTime

10.         n   envlp     Data block                      ->dataBlock

11.         1   envlp     Data block checksum             +>
```