Toprammer - TOPxxxx OpenSource suite

February 28, 2010

1 Dependencies

- Python 2.5 or Python 2.6 is required

  `http://python.org/`

- python-usb module is required:

  `http://sourceforge.net/apps/mediawiki/pyusb/index.php?title=Main_Page`

- python-pkg-resources module is required 'pkg_resources' is part of the Python 'setuptools' package.

  `http://pypi.python.org/pypi/setuptools`

2 Supported chips

Chip	State
Atmel Mega32 DIP40	supported
Atmel Mega8 DIP28	supported
Atmel Mega88 DIP28	supported
m2764a EPROM	supported

3 Installation

Just execute

```
python setup.py install
```

as root inside of the toprammer distribution package's root directory. Note that "setuptools" have to be installed on your system. See dependencies.

4 Usage

A .BIT file is always needed to do operations on the inserted chip. The .BIT file must match the chip. For example, do do operations on an Atmel Mega8 DIP microcontroller, the atmega8dip28.bit file is needed. Prebuilt .BIT files can be found in the "libtoprammer/bit/" subdirectory. Sourcecode for these .BIT files is located in "libtoprammer/bit/src/". However, the full path to the .BIT file doesn't need to be specified to the toprammer tool. If you just pass the chip-name, toprammer will search for a matching .BIT file in some standard search paths. The .BIT file basically is the firmware that runs on the FPGA in the TOP programmer. See toprammer –help for device actions to read/write stuff from/to the chip.

Example:

```
Checking which chips are supported:
  toprammer --list
Reading the flash of an Atmel Mega8 DIP to the file flash.img:
  toprammer --bitfile atmega8dip28 --read-prog flash.img
```

5 ZIF socket layout generator

The ZIF socket layout generator "toprammer-layout" will try to generate a chip-insert layout with the given parameters and print it as ASCII-art to the console.

For example, if you have a DIP28 packaged chip, that needs VCC on pin 7, VPP on pin 1 and GND on pin 8, you'd issue the following command:

```
toprammer-layout -d top2049 --package DIP28 --vccx 7 --vpp 1 --gnd 8
```

This will show you a layout of how to insert the chip into the programmer. It will also show which pins of the ZIF are powered. This will hopefully match your request. :) Alternatively, it will yield an error message, if it was unable to find a layout that fits the contraints. Note that the –vccx, –vpp and –gnd pins specified on the commandline are with respect to the package (_not_ the ZIF socket). The whole purpose of the tool is to map the package and its pin layout to the ZIF socket, while obeying the programmer constraints.

Alternatively you can specify one of the supported chip-IDs to toprammer-layout:

```
toprammer-layout -d top2049 --package atmega8dip28
```

This will show you the layout of the AtMega8 DIP28.