The Beaglebone Pinmux Tool

Synopsis

The pinmux tool helps to set correct pin muxing on beaglebone. Instead of manually figuring out what is the mode number for required pin functionality or which bit to set to enable pull up, the user has to create a simple text file describing the setup and the tool will create a setup script.
The tool can also build and decompile binary content of cape's EEPROM.

How to use the tool

Operation of BeagleBone pinmux tool requires at least a name of the input file to operate properly The names of the input and output files, file formats and additional functionality are controlled by the command line options.

--input-file,-i	Name of the input file.
--input-format	Input file format. If omitted, the tool expects config file. The argument can be either *config* or *binary*.
--output-file,-o	Output file name. By default the produced output is send to STDOUT.
--output-format	Output file format. If omitted, binary EEPROM contents is generated. The argument can be *binary, config, shell* or *python*.
--function, -f	List pins which support given function. Takes a regexp as an argument.
--name, -n	List pins whose names match the parameter. Takes a regexp as an argument.
--modes, -m	List modes for a pin. Takes a regexp as an argument.
--run, -r	Set pinmuxing without generating files. Output still can be generated if *–output-file* or *–output-format* options are used.

Example: *(to be entered on one line)*

```
$ ruby ./bbone_pinmux_tool.rb ↩
      --input-file examples/in.txt ↩
      --input-format config ↩
      --output-format shell ↩
      --output-file setmuxing.sh
```

The *name* option allows to find pins with names matching the pattern.
Example1:

```
$ ruby ./bbone_pinmux_tool.rb --name i2c.*
i2c1_scl: p9-17(i2c1_scl)
i2c1_sda: p9-18(i2c1_sda)
i2c2_scl: p9-19(i2c2_scl)
i2c2_sda: p9-20(i2c2_sda)
```

Example 2:

```
$ ruby ./bbone_pinmux_tool.rb --name p8-1.*
```

```
p8-10: p8-10(timer6)
p8-11: p8-11(gpio1_13)
p8-12: p8-12(gpio1_12)
p8-13: p8-13(ehrpwm2b)
p8-14: p8-14(gpio0_26)
p8-15: p8-15(gpio1_15)
p8-16: p8-16(gpio1_14)
p8-17: p8-17(gpio0_27)
p8-18: p8-18(gpio2_1)
p8-19: p8-19(ehrpwm2a)
```

The *function* option allows to identify pins which support function matching the pattern. Since one function can be routed to more than one pin, multiple pins can be listed for one function.
Example:

```
$ ruby ./bbone_pinmux_tool.rb --function i2c.*
i2c1_scl: p9-17(i2c1_scl) p9-24(uart1_txd)
i2c1_sda: p9-18(i2c1_sda) p9-26(uart1_rxd)
i2c2_scl: p9-21(uart2_txd) p9-19(i2c2_scl)
i2c2_sda: p9-20(i2c2_sda)
```

Configuration file

The configuration file used as the input to the tool is divided into several sections. Each section starts with a header, containing section's name enclosed in square brackets. Some sections require an additional argument following the section's name. The lines following the header are considered configuration directives. Section ends on another section header or at the end of the file.
Text following the '#' character is considered a comment and will be ignored during parsing. The comment can start any point, so it is possible to insert short documentation messages after each of the configuration directives. Blank lines and lines containing only comments will be ignored.
The order of sections in the file and the order of configuration directives inside sections is not important. Section names and key words are not case sensitive. It is possible to have multiple instances of a section, with configuration directives from the later section overwriting contents of the configuration defined earlier.

The structure of the config reflects the structure of cape's EEPROM: there can be a header section, current information, multiple pin configuration sections and (in the future) a section describing contents of the user part of the EEPROM, following the cape's configuration.

Header

The header section contains information about cape's name, manufacturer, circuit revision, serial number and part number. The number of used pins is calculated automatically, based on the number of *pin* sections present in the file. This is an example of the header section:

```
[header]
      name Test board
      version 123
```

```
        manufacturer John Smith
        part FlyingBone
        serial 11224P160001
```

[header]	Marks the beginning of the header section.
name	Defines the name of the cape. Max. 32 characters.
version	Board's revision. Up to 4 ASCII characters.
manufacturer	Name of the board's manufacturer. Max. 16 characters.
part	Part number. Max. 14 characters.
serial	Board's serial number. Max. 16 characters.

Trailing and following white characters (tabs, spaces, etc) will be stripped. Strings shorter than the maximum number of characters will be left-padded with spaces, longer strings will be trimmed on the right (the beginning of the string will be preserved). The case of the characters will not be modified.

Current information

The section contains information about current consumption by the cape and, if the cape provides such capability, how much current can be supplied to the VDD_5V rail.

```
[current]
      3.3vdd 123mA
      5vdd 500mA
      5sys 2000mA
      supplied 1.234A
```

[current]	Section header
3.3vdd	Maximum current consumed from the VDD_3V3EXP rail.
5vdd	Maximum current consumed from the VDD_5V rail.
5sys	Maximum current consumed from the SYS_5V rail.
supplied	Maximum current that can be provided by the cape to the VDD_5V rail

All values are positive numbers. Currents can be specified in miliamps or amps. 0 means no current drawn or supplied.

Pin configuration

This is probably the part of the config file most often sued by the BeagleBone users. The *pin* sections describe pin functions and configurations. Each section starts with a header containing the *pin* keyword and a pin name. The pin name can be specified as a Px-y, where x is the connector number (either 8 or 9) and y is the pin number in the connector, or using pin's name. The pin names and numbers can be found on table 15 in the BeagleBone SRM.

```
[pin p9-22]
      bidir
      slow
      rx
      pull up
```

```
            pull disabled
            mux gpio0_2
```

[pin p9-22]	Section header. The name following the keyword *pin* tells which pin is being configured. The pin can be also specified using its name, so *P9-22* and *UART2_RXD* describe the same pin.
bidir/in/out	Direction of the pin. Valid keywords are *in*, *out* and *bidir*.
slow	Slew rate on the bin. Can be either *slow* or *fast*.
rx	Present if the pin is an input.
pull up/down	Direction of the pull on the pin.
pull enable/disable	Controls whether the pull up/down should be enabled or disabled on this pin.
mux gpio0_2	Specifies which of the pin's functions should be enabled. Can be function name (*gpio0_2*) or mode number (*7*).

Order of the configuration directives is not important. If there are multiple sections describing configuration of a pin, only latest occurrence of a configuration directive will be used. If a configuration section for previously configured pin is found, the tool will print a warning message.

Alternative function names can be found in tables 9, 10, 12 and 13 in the SRM.
NOTE: several functions can be assigned to different pins. In such case the SRM adds '*_muxX*' appendix to the name, but only the base name should be used as the function name. E.g: *gpmc_clk* can be configured on pins P8-18 and P8-21. The functions are described as *gpmc_clk* on pin 21 and *gpmc_clk_mux0* on pin 18. When configuring GPMC clock function on any of the pins, only the *gpmc_clk* name should be used.