tbot doc

version 2016.12

Heiko Schocher

November 25, 2016

Contents

Welcome to tbot's documentation!

tbot is a tool for automating commandline tasks. Especially tasks like compiling source code, installing the resulting images on the target and execute testcases on the new bootet images.

Sources are hosted at https://github.com/hsdenx/tbot

tbot overview

Short description

- execute testcases on real hw
- testcases written in python
- call testcases from another testcase
- Based on ideas from: http://www.denx.de/wiki/DUTS/DUTSDocs

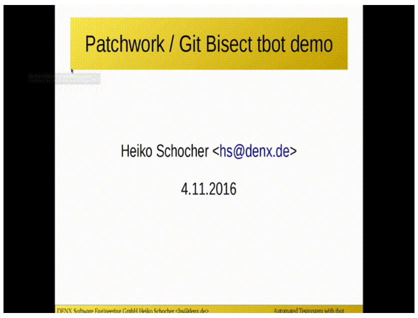
Usage

```
$ tbot.py --help
Usage: tbot.py [options]

Options:
  -h, --help            show this help message and exit
  -c CFGFILE, --cfgfile=CFGFILE
                        the tbot board configfilename
  -s LABFILE, --slabfile=LABFILE
                        the tbot lab configfilename
  -l LOGFILE, --logfile=LOGFILE
                        the tbot logfilename, if default, tbot creates a
                        defaultnamelogfile
  -t TC, --testcase=TC  the testcase which should be run
  -v, --verbose         be verbose, print all read/write to stdout
  -w WORKDIR, --workdir=WORKDIR
                        set workdir, default os.getcwd()
```

Demo

click on the gif to see the full video on youtube



Patchwork / Git Bisect tbot demo

Heiko Schocher <hs@denx.de>

4.11.2016

https://youtu.be/zfjpj3DLsx4

demo video for a CAN bus testcase:

https://youtu.be/hl7gI4b9CG8

demo for a buildbot integration:

http://xeidos.ddns.net/buildbot/tgrid

Installation

install tbot on your PC (linux only tested):

- get the source code:

```
$ git clone https://github.com/hsdenx/tbot.git
[...]
$
```

cd into the tbot directory.

- you need the for running tbot the python paramiko module, see: http://www.paramiko.org/installing.html

  paramiko is used for handling ssh sessions, and open filedescriptors on a ssh connection. Tbot open a ssh connection to a "lab PC" and opens on that connection 2 filehandles, one for control functions and one for the connection to the boards console. May it is worth to think about to open more filehandles and use them in tbot, but thats a point in the Todo list ...

  See [1] for more infos about tbot principles.

- prepare a directory for storing the logfiles and pass it with the commandline option "-l" to tbot. Default is the directory "log" in the tbot root (don;t forget to create it, if you want to use it)
- If your VL is not yet in tbot source, integrate it (This task has only to be done once for your VL):

  A VL has basic 3 tasks:

  a. power on/off the board
  b. get power state of the board
  c. connect to the boards console

  As tbot sends only shell commands (also to the Lab PC) this tasks must be executable through shell commands on your Lab PC:

- prepare a lab config file for your lab:

  - create a new folder in src/tc/lab/XXX replace XXX to a proper value

  Each VL needs a configuration file, passed with the option '-s' to tbot, example:

  https://github.com/hsdenx/tbot/blob/master/config/lab_hs_home.py simple copy this and rename it to https://github.com/hsdenx/tbot/blob/master/config/lab_XXX.py and adapt the settings to your specific needs.

- Then you have to setup Testcases for the 3 VL tasks:

  - Task a) power on/off board:

    default TC for this task is:

    https://github.com/hsdenx/tbot/blob/master/src/tc/lab/denx/tc_lab_denx_power.py

    - now copy this file to for example cp src/tc/lab/denx/tc_lab_denx_power.py src/tc/lab/XXX/tc_lab_XXX_power_onoff.py and adapt the "remote_power" command from the denx lab to your needs.

      As this TC powers on the board for all your boards in your VL, you can differ between the boards through the tbot class variable "tb.config.boardlabpowername" (which is in the default case the same as "tb.config.boardname"), but you may need to name the power target with an other name than boardname, so you can configure this case. The power state "tb.power_state" which the TC has to set is "on" for power on, or "off" for power off.

      If switching on the power is successful, call "tb.end_tc(True)" else "tb.end_tc(False)"

    - set in your lab config file: tc_lab_denx_power_tc = 'tc_lab_XXX_power_onoff.py'

  - Task b) power on/off board:

    default TC for this task is:

    https://github.com/hsdenx/tbot/blob/master/src/tc/lab/denx/tc_lab_denx_get_power_state.py

    - now copy this file to for example (replace XXX to a proper value) cp src/tc/lab/denx/tc_lab_denx_get_power_state.py src/tc/lab/XXX/tc_lab_XXX_get_power_state.py and adapt the commands to your needs.

      If the power of the board is on, call "tb.end_tc(True)" else "tb.end_tc(False)"

    - set in your lab config file: tc_lab_denx_get_power_state_tc = 'tc_lab_XXX_get_power_state.py'
  - Task c) connect to the boards console:

    default TC for this task is:

    https://github.com/hsdenx/tbot/blob/master/src/tc/lab/denx/tc_lab_denx_connect_to_board.py

    - now copy this file to for example cp src/tc/lab/denx/tc_lab_denx_connect_to_board.py src/tc/lab/XXX/tc_lab_XXX_connect_to_board.py and adapt the commands to your needs.

      If connect fails end this TC with "tb.end_tc(False)" else call "tb.end_tc(True)"

      If you want to use kermit for connecting to the boards console, you can use:

      https://github.com/hsdenx/tbot/blob/master/src/tc/linux/tc_workfd_connect_with_kermit.py

      Example for such a board in the VL from denx: tc_lab_denx_connect_to_board_tc = 'tc_workfd_connect_with_kermit.py'

      https://github.com/hsdenx/tbot/blob/master/config/tbot_dxr2.cfg#L20

    - set in your lab config file: tc_lab_denx_connect_to_board_tc = 'tc_lab_XXX_connect_to_board.py'
- prepare password.py file: This file contains all passwords tbot needs (for example for linux login on the boards) tbot searches this file in the tbot root directory. It is a simple python file, for example:

```
# passwords for the lab
if (board == 'lab'):
    if (user == 'hs'):
        password = 'passwordforuserhs'
    if (user == 'root'):
        password = 'passwordforrootuser'
# passwords for the boards
elif (board == 'mcx'):
    if (user == 'root'):
        password = 'passwordformcxrootfs'
else:
    if (user == 'root'):
        password = ''
```

tbot searches in the root folder for this file.

- prepare board config file

  Each board which is found in the VL needs a tbot configuration file pass the config file name with the option '-c' to tbot, tbot searches in the "config" folder for them.

  board Example (dxr2 board): https://github.com/hsdenx/tbot/blob/master/config/dxr2.py

  Now comes a list of variables TC needs, this vary from what you you want to test...

Thats it ... you now can call tbot and hopefully, it works ;-)

If you have problems in settings up tbot, please contact me (and may give me ssh access to your Lab PC ;-)

Heiko Schocher <hs@denx.de> v2 2016.11.02

Principles

Test case (TC):

A piece of python code, which uses the tbot class. Tbot provides functions for sending shell commands and parsing the shell commands output. Tbot waits endless for a shell commands end (detected through reading the consoles prompt). A TC can also call other TC-es.

remark: Tbot not really waits endless, for a shell commands end, instead tbot starts a watchdog in the background, and if it triggers, tbot ends the TC as failed. In the tbot beginning there was a lot of timeouts / retry cases, but it turned out, that waiting endless is robust and easy ...

Host PC

where tbot runs, currently only linux host tested must not be a powerful machine. For example, I run it on a raspberry Pi.

Lab PC:

Host PC connects through ssh to the Lab PC, so it is possible to test boards, which are not at the same place as the Host PC.

(Lab PC and Host PC can be the same of course)

curently only Lab PC with an installed linux supported/tested.

Boards(s):

the boards on which shell commands are executed.

Board state:

equals to the software, the board is currently running.

Currently tbot supports 2 board states:

- "u-boot", if the board is running U-Boot
- "linux", if the board is running a linux kernel

A board state is detected through analysing the boards shell prompt. In linux tbot sets a special tbot prompt, in U-Boot the prompt is static, and configurable through a board config file.

A TC can say in which board state it want to send shell commands. Tbot tries to detect the current board state, if board is not in the requested board state, tbot tries to switch into the correct state. If this fails, the TC fails. It is possible to switch in a single TC between board states.

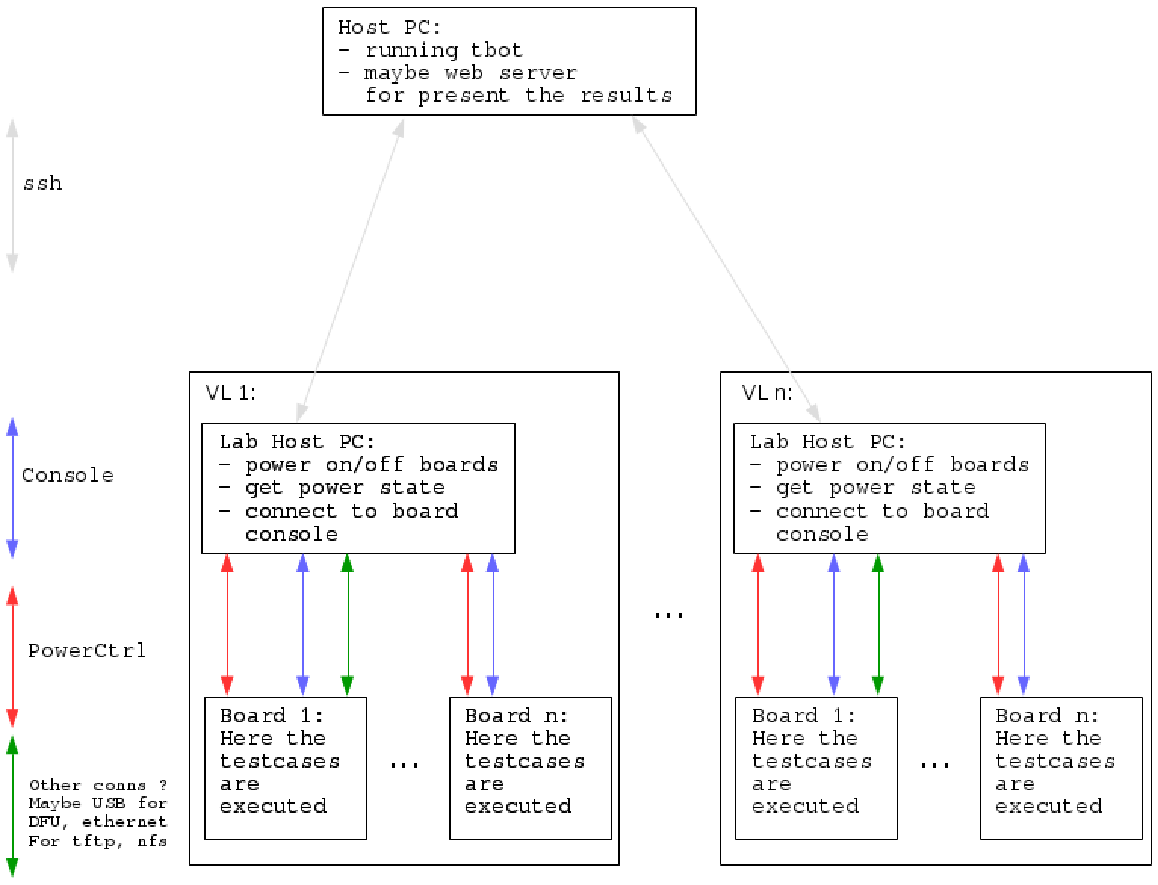
Virtual laboratory (VL)

VL is the basic environment that groups:

- [a number of] boards - target devices on which tbot executes testcases.
- one Lab PC
- Basic tasks: - power on/off boards - read current power state of a board - connect to boards console

read more in doc/README.install how to integrate your own VL

Overview:



Events

tbot creates while executing testcases so called events. After tbot ended with the testcase it can call event_backends, which convert the events to different formats.

There are standard Events which tbot create automatically, for example the Event "Start" is created when tbot starts a new Testcase.

It is also possible to create Testcases specific Events. Therefore a Testcase only has to call the function **create_event()**

Eventlist

current created standard Events

Event-ID	content
log	log content
Boardname	Name of board
BoardnameEnd	End of tests for Boardname
Start	Start of TC
End	End of TC

Testcases specific Events

tc_lab_set_toolchain.py

Event-ID	content
Toolchain	used Toolchain

tc_workfd_apply_patchwork_patches.py

Event-ID	content
PW_NR	current patchwork patchnumber
PW_CLEAN	current patchworknumber patch is clean or not
PW_AA	current patchworknumber patch is already applied
PW_APPLY	current patchworknumber patch is applies clean or not

tc_lab_compile_uboot.py

Event-ID	content
UBOOT_DEFCONFIG	used U-Boot configuration
UBOOT_SRC_PATH	path, where U-boot source is located

tc_ub_test_py.py

Event-ID	content
UBOOT_TEST_PY	path to test py result

tc_ub_get_version.py

Event-ID	content
UBOOT_VERSION	U-Boot/SPL version

tc_lx_get_version.py

Event-ID	content
LINUX_VERSION	Linux version

tc_workfd_compile_linux.py

Event-ID	content
LINUX_DEFCONFIG	used Linux configuration
LINUX_SRC_PATH	path, where Linux source is located

demos

dashboard

dashboard_source

pick some Events and put the content into a MYSQL database. Now the DB content can be readen with a simple php script to create a webpage, see for a minimal example:

http://xeidos.ddns.net/tests/test_db_auslesen.php

statistic

statistic_source

use gnuplot for a ploting some TC statistic results.

http://xeidos.ddns.net/tbot/id_189/statistic.jpg

dot

dot_source

Use the Eventinformation for creating nice DOT graphics from the test. see a raw example:

Demo Output of a git bisect Demotestcase

http://xeidos.ddns.net/tbot/id_171/graph.png

planned Event backends:

DUTS:

make from the logs tbot collected, DUTS specific textfiles, so the logs can integrated into the DULG

xunit:

create xunit files for presenting the results in jenkins

kernel CI:

adapt to a format, so the testresults can be presented at kernel CI (just an idea...)

Roadmap

Project's road-map

Here you will find "ideas" what can be done in tbot

tbot internal

tbotlib suggestions

Check if a TC ends with a prompt read

[simplify] create a "setup" script, which simplify setup after tbot is installed. (Adapt all config variables for a new lab ... Currently all defaultvalues are proper for the DENX lab)

[simplify] create testcases, specific for a specific SoC, so board testcases can use them. For Example, create a tc_soc_imx6_xxx.py testcases, which contains testcases we want to run on an imx based board.

[simplify] make it possible to move virtual lab specific config options into lab specific config files, and include them in board config files

[simplify] make a documentation of "tbot conventions" currently this are my personal conventions, if there are more user try to identify my conventions and discuss and document them

currently they are hidden in default settings of tbot variables ...

[simplify] Get Vars for TC from U-Boot Code started with, see TC:

- src/tc/tc_workfd_get_uboot_config_hex.py
- src/tc/tc_workfd_get_uboot_config_string.py

simplify tbot usage ... all ToDo points marked with [simplify] goal: show tbot installation/usage on 2 pages ... (maybe 1 is possible ?)

- one for tbot installation and how to setup a virtual lab
- one for how to setup a board testcase

call testcases with arguments

move testcases into functions and use python decorators ... I played with this approach, but I am not happy with it.

rework tbot completly into a more pythonic style

Event Backends

Dokumentation Backend

extract for each testcase the logs in files Filename: testcasename_connectionname_index_incnumber.txt

- testcasename: Name of TC
- connectionname: Name of the tbot connection
- index: counts, how often the TC was called, starts with 1
- incnumber: each switch to another connection increments this number starts with 1

Now you have all logs in a seperate file, you can integrate into a documentation

Saying the format of your documentation ist a rst file

You can for example define a keyword "tbotref:filename"

Now writting a script which searches in your document file for the above keword, and replaces it with the content of filename in for example http://docutils.sourceforge.net/docs/ref/rst/restructuredtext.html#literal-blocks format ...

and you can create a documentation with current logs.

jenkins backend

simple convert the collected events into jenkins format, so it is easy to integrate tbot into jenkins builds

kernel-ci backend

simple convert the collected events into a format, so we can integrate tbot into kernel-ci

tbot documentation

Documentation of tbotlib functions

tbotlib

`tbotlib.`**`raw`** (*`text`*)
Returns a raw string representation of text

class `tbotlib.`**`tbot`** (*`workdir`*, *`labfile`*, *`cfgfile`*, *`logfilen`*, *`verbose`*)
Bases: **`object`**
The tbot class
all begins here ...

- **parameters**, **types**, **return** and **return types**:

  Parameters:
  - **arg1** -- workdir for tbot
  - **arg2** -- labfile
  - **arg3** -- board config file
  - **arg4** -- name of logfile
  - **arg5** -- be verbose

`call_tc` **(*`name`*, *`**kwargs`*)**
Call another testcase.

Search for the TC name through all subdirs in 'src/tc'.

- **parameters**, **types**, **return** and **return types**:

Parameters:
- **arg1** -- name of testcase
- **arg2** -- optional testcase argumentlist

Returns: False: If testcase was not foundor testcase raised an execption ! called testcase sets the ret variable, whichthis function returns. If called testcase not set the ret variable default is false!

`check_args` **(*`args`*)**
Check if the args are in current argumentstack

- **parameters**, **types**, **return** and **return types**:

Parameters: **arg1** -- args

Returns: If args no found, end testcase with False

else return args argument

`check_debugger` **()**
checks if a debugger is attached
If so, run the target. For this tc "tc_lab_bdi_run.py" is called.

- **parameters**, **types**, **return** and **return types**:

Returns: True

`check_open_fd` **(*`c`*)**
check, if stream is open.

- **parameters**, **types**, **return** and **return types**:

Parameters: **arg1** -- connection

Returns: True: If open False: If stream open failed

`cleanup` **()**

`con_log` **(*`*args`*)**
logs a console string

- **parameters**, **types**, **return** and **return types**:

Parameters: **arg1** -- console string

Returns:

`connect_to_board` **(*`boardname`*)**
connect to the board

- **parameters**, **types**, **return** and **return types**:

Parameters: **arg1** -- boardname

`debugprint` **(*`*args`*)**
print a debug string on stdout.

This output can be enabled through self.config.debug

- **parameters**, **types**, **return** and **return types**:

Parameters: **arg1** -- argument list

`disconnect_from_board` **(*`boardname`*)**
disconnect from the board

- **parameters**, **types**, **return** and **return types**:

Parameters: **arg1** -- boardname

`end_tc` **(*`ret`*)**
end testcase.
simple end a testcase.
ret contains True if testcase ended successfully, False if not.

- **parameters**, **types**, **return** and **return types**:

Parameters: **arg1** -- return value True/False

Returns: calls sys.exit(0 if ret == True 1 else)

`eof_call_tc` **(*`name`*, *`**kwargs`*)**
call tc name, end testcase on failure

- **parameters**, **types**, **return** and **return types**:

Parameters:
- **arg1** -- name of Testcase
- **arg2** -- optional argument list

Returns: True if called testcase ends True, als call end_tc(False)

`eof_expect_string` **(*`c`*, *`string`*)**
expect a string, if prompt read end tc False

- **parameters**, **types**, **return** and **return types**:

Parameters:
- **arg1** -- connection
- **arg2** -- string expected

`eof_write` **(*`c`*, *`string`*)**
write a string to connection c

- **parameters**, **types**, **return** and **return types**:

Parameters:
- **arg1** -- connection
- **arg2** -- string

Returns: If write_stream returns not True, end tc

with failure

`eof_write_cmd` **(*`c`*, *`command`*)**
write a command to fd, wait for prompt

- **parameters**, **types**, **return** and **return types**:

Parameters:
- **arg1** -- connection
- **arg2** -- commandstring

Returns: True if prompt read, else end testcase with False

`eof_write_cmd_check` **(*`c`*, *`cmd`*, *`string`*)**
send a cmd and check if a string is read.

- **parameters**, **types**, **return** and **return types**:

Parameters:
- **arg1** -- connection
- **arg2** -- commandstring
- **arg3** -- string which must be read

Returns: True if prompt and string is read

else end Testcase with False

`eof_write_cmd_list` **(*`c`*, *`cmdlist`*)**

send a list of cmd to fd and wait for end

- **parameters**, **types**, **return** and **return types**:

  Parameters:
  - **arg1** -- connection
  - **arg2** -- list of commandstrings

  Returns: True if prompt found else endtestcase with False

`eof_write_con` **(*`string`*)**
write a string to console.

- **parameters**, **types**, **return** and **return types**:

  Parameters: **arg1** -- commandstring

  Returns: True if write_stream returns True, else end testcase with False

`eof_write_con_lx_cmd` **(*`command`*)**
write a linux command to console.

- **parameters**, **types**, **return** and **return types**:

  Parameters: **arg1** -- commandstring

  Returns: True if linux command was successful

else end testcase with False

`eof_write_con_passwd` **(*`user`*, *`board`*)**
write a passwd to console. Do not log it.

- **parameters**, **types**, **return** and **return types**:

  Parameters:
  - **arg1** -- username
  - **arg2** -- board

  Returns: If write_stream returns not True, end tc with failure

`eof_write_ctrl` **(*`string`*)**
write a string to control connection.

- **parameters**, **types**, **return** and **return types**:

  Parameters: **arg1** -- commandstring

  Returns: If write_stream returns not True, end tc with failure

`eof_write_ctrl_passwd` **(*`user`*, *`board`*)**
write a password to control. Do not log it.

- **parameters**, **types**, **return** and **return types**:

  Parameters:
  - **arg1** -- username
  - **arg2** -- board

  Returns: If write_stream returns not True, end tc with failure

`eof_write_workfd_passwd` **(*`user`*, *`board`*)**
write a password to workfd. Do not log it.

- **parameters**, **types**, **return** and **return types**:

  Parameters:
  - **arg1** -- username
  - **arg2** -- board

  Returns: If write_stream returns not True, end tc with failure

`failure` **()**

`flush` **(*c*)**
read out all bytes from connection

- **parameters**, **types**, **return** and **return types**:

Parameters: **arg1** -- connection

`get_board_state` **(*name*)**

`get_power_state` **(*boardname*)**
Get powerstate of the board in the lab

- **parameters**, **types**, **return** and **return types**:

Parameters: **arg1** -- boardname
Returns: True if power state is on, else False

`overwrite_config` **(*filename*)**

`read_line` **(*c*)**
read a line. line end detected through ' '

- **parameters**, **types**, **return** and **return types**:

param arg1: connection
return: **True: if a line is read**
self.buf contains the line

False :if prompt read

`send_console_end` **(*c*)**
write Ctrl-C to the opened stream

If stream is not open, try to open it

- **parameters**, **types**, **return** and **return types**:

Parameters: **arg1** -- connection
Returns: True: if write was successful None: not able to open the stream

`send_ctrl_c` **(*c*)**
write Ctrl-C to the opened stream

If stream is not open, try to open it

- **parameters**, **types**, **return** and **return types**:

Parameters: **arg1** -- connection
Returns:
Returns: True: if write was successful None: not able to open the stream

`send_ctrl_c_con` **()**
write Ctrl-C to the opened stream

If stream is not open, try to open it

- **parameters**, **types**, **return** and **return types**:

Parameters: **arg1** -- connection
Returns: True: if write was successful None: not able to open the stream

`send_ctrl_m` **(*c*)**
write Ctrl-M to the opened stream

If stream is not open, try to open it

- **parameters**, **types**, **return** and **return types**:

Parameters: **arg1** -- connection

Returns: True: if write was successful None: not able to open the stream

`set_board_state` **(*`state`*)**
set the board to a state
currrent states supported: 'lab' 'u-boot' 'linux'

- **parameters**, **types**, **return** and **return types**:

Parameters: **arg1** -- state string

Returns: True if switching to state had success

else testcase fails.

`set_power_state` **(*`boardname`*, *`state`*)**
set powerstate for the board in the lab

- **parameters**, **types**, **return** and **return types**:

Parameters:

- **arg1** (*string*) -- boardname
- **arg1** -- state on/off

Returns: True if setting state was successful, else False

`set_prompt` **(*`c`*, *`prompt`*, *`ptype`*)**
set the prompt for the connection c.
If ptype = 'linux' add some special settings to the prompt.

- **parameters**, **types**, **return** and **return types**:

Parameters:

- **arg1** -- connection
- **arg2** -- new promt string
- **arg3** (*string*) -- prompt type 'linux'

Returns: True: If setting the prompt was successful

False: If settting the prompt failed

`set_term_length` **(*`c`*)**
set terminal line length
ToDo How could this be set longer and do this correct - **parameters**, **types**, **return** and **return types**:: :param arg1: connection :return: no return value

`statusprint` **(*`*args`*)**
print a status string on stdout.

This output can be enabled through self.config.debugstatus

- **parameters**, **types**, **return** and **return types**:

Parameters: **arg1** -- argument list

`tbot_expect_prompt` **(*`c`*)**
searches for prompt, endless

- **parameters**, **types**, **return** and **return types**:

Parameters: **arg1** -- connection

`tbot_expect_string` **(*`c`*, *`string`*)**
expect a string

- **parameters**, **types**, **return** and **return types**:

Parameters:
- **arg1** -- connection
- **arg2** -- string expected

Returns: 'prompt' if prompt found, True if string is found, else False

`tbot_fakult` **(*n*)**

`tbot_get_password` **(*user*, *board*)**

get the password for the user/board

The passwords are in the password.py file in the working directory. For example: if (user == 'passwordforuserone'):

password = 'gnlmpf'

if (user == 'anotheruser'):

password = 'passwordforanotheruser'

- **parameters**, **types**, **return** and **return types**:

Parameters:
- **arg1** -- user
- **arg2** -- board

Returns: return password if found end tc if not

`tbot_read_line_and_check_strings` **(*c*, *strings*)**

read a line and search, if it contains a string in strings.
If found, return index if read some chars, but no line, check if it is a prompt, return 'prompt' if it is a prompt. if a string in strings found return index else return None

- **parameters**, **types**, **return** and **return types**:

Parameters:
- **arg1** -- connection
- **arg2** -- a list of strings

Returns: index of string which is found 'prompt' if prompt found

`tbot_rup_check_all_strings` **(*c*, *strings*, *endtc=False*)**

read until prompt, and check if all strings in list strings are found

- **parameters**, **types**, **return** and **return types**:

Parameters:
- **arg1** -- connection
- **arg2** -- a list of strings

Returns: returns False, if not all strings in list are

found, or end tbot if endtc = True.

`tbot_rup_error_on_strings` **(*c*, *strings*, *endtc=False*)**

read until prompt and check, if a string in list is found.
If a string is found, end False.

- **parameters**, **types**, **return** and **return types**:

Parameters:
- **arg1** -- connection
- **arg2** -- list of strings
- **arg3** -- if endtc = True end with calling end_tc(True/False)

Returns: True if prompt and no string is found.

`tbot_start_wdt` **()**

start the WDT process

`tbot_trigger_wdt` **()**
trigger the WDT

`verboseprint` **(*`*args`*)**
print a verbose string on stdout.

This output can be enabled through self.config.debug

- **parameters**, **types**, **return** and **return types**:

Parameters: **arg1** -- argument list

`write_cmd_check` **(*`c`*, *`cmd`*, *`string`*)**
send a cmd and check if a string is read.

- **parameters**, **types**, **return** and **return types**:

Parameters:
- **arg1** -- connection
- **arg2** -- command send over connection
- **arg3** -- string which must be read

Returns: True if prompt and string is read

else False

`write_lx_cmd_check` **(*`c`*, *`command`*, *`endTC=True`*)**
write a linux command to console.

- **parameters**, **types**, **return** and **return types**:

Parameters:
- **arg1** -- connection
- **arg2** -- commandstring
- **arg3** -- if True and linux cmd ended False end TC with end_tc(False), else return True

Returns: if linux cmd ended successful True, else False

`write_stream` **(*`c`*, *`string`*)**
write a string to connection

If stream is not open, try to open it

- **parameters**, **types**, **return** and **return types**:

Parameters:
- **arg1** -- connection
- **arg2** -- string

Returns: True: if write was successful None: not able to open the stream

`write_stream_con` **(*`string`*)**
write a string to console connection

If stream is not open, try to open it

- **parameters**, **types**, **return** and **return types**:

Parameters: **arg1** -- string

Returns: True: if write was successful None: not able to open the stream

`write_stream_ctrl` **(*`string`*)**
write a string to the ctrl connection

If stream is not open, try to open it

- **parameters**, **types**, **return** and **return types**:

Parameters: **arg1** -- string

Returns: True: if write was successful None: not able to open the stream

`write_stream_passwd` **(*c*, *user*, *board*)**

write a passwd for user to connection

If stream is not open, try to open it Do not log it.

- **parameters**, **types**, **return** and **return types**:

Parameters:

- **arg1** -- connection
- **arg2** -- user
- **arg3** -- board

Returns: return: True: if write was successful None: not able to open the stream

event

class `tbot_event.`**`events`** (*tb*, *logfile*)

Bases: **`object`**

The event class

`create_event` **(*pname*, *name*, *id*, *value*)**

create an event

- **parameters**, **types**, **return** and **return types**:

Parameters:

- **arg1** -- parent name
- **arg2** -- function name
- **arg3** -- Event ID
- **arg4** -- value for event ID

`create_event_log` **(*c*, *dir*, *string*)**

create a log event

- **parameters**, **types**, **return** and **return types**:

Parameters:

- **arg1** -- connection
- **arg2** -- direction (r or w)
- **arg3** -- log string

`event_flush` **()**

`list_backend` **()**

list all registered backends.

ToDo

`register_backend` **()**

register a backend.

ToDo

Documentation of tbot event backends

dashboard

class `dashboard.`**`dashboard`** (*`tb`*, *`host`*, *`user`*, *`pw`*, *`dbname`*, *`tname`*)

Bases: **`object`**

extract tbot results to a mysql database after tbot has finished

Prerequisites:

MySQLdb python module is needed, install it for example on the raspberry pi with:

apt-get install python-mysqldb apt-get install sshpass

> (needed fpr passing the password to scp)

If tb.config.create_dot == 'yes' then you need the dot command, please install this, see for example:

http://askubuntu.com/questions/97552/how-to-install-dot-provided-by-graphviz

If tb.config.create_statistic == 'yes' you need the gnuplot command. See an example for installing gnuplot here:

http://askubuntu.com/questions/340579/how-to-install-gnuplot-in-ubuntu

The dashboard backend also collects information from other backends (if they are enabled) and stores them in "webdir". Currently this is a fix place, need here some work to make this configurable. Currently it is placed at "/var/www/html", and subdir "tbot" plus current MYSQL ID "id_%d" ...

- **parameters**, **types**, **return** and **return types**:

  Parameters:
  - **arg1** -- tb
  - **arg2** -- host
  - **arg3** -- username
  - **arg4** -- pw
  - **arg5** -- dbname
  - **arg5** -- tname

`insert_test_into_db` **()**

starts with filling the DB

documentation

class `documentation.`**`doc_backend`** (*`tb`*, *`ignorelist`*)

Bases: **`object`**

extract from all executed testcases the logs from tbots connection.

Format of the created filenames:

testcasename_connectionname_index_incnumber.txt

> testcasename: Name of TC connectionname: Name of the tbot connection index: counts, how often the TC was called, starts with 1 incnumber: each switch to another connection increments this number
>
> > starts with 1

This files could be used to create documentation files, which contains logs.

enable this backend with "create_documentation = 'yes'"

created files are stored in tb.workdir + '/logfiles/ so, be sure you have created this directory.

Problem: First prompt is not visible

see https://github.com/hsdenx/tbot/blob/testing/scripts/demo/documentation_backend/README for a demo, how you can create a html/pdf/man page, which contains content of tbot logfiles and text around it.

- **parameters**, **types**, **return** and **return types**:

  Parameters:
  - **arg1** -- tb
  - **arg2** -- list of strings, containing testcasesnames, which get ignored

`create_docfiles` **()**

create the files

dot

class `dot.`**`dot`** (*`tb`*, *`dotfile`*, *`ignorelist`*)

Bases: **`object`**

create a dot description file from the executed testcases.

after tbot hs finsihed create a png with:

dot -Tpng tc.dot > tc.png

or create a postscript with:

dot -Tps tc.dot > tc.ps

- **parameters**, **types**, **return** and **return types**:

  Parameters:
  - **arg1** -- tb
  - **arg2** -- filename, which contains the dot description data
  - **arg3** -- list of strings, containing testcasesnames, which get ignored

`create_dotfile` **()**

create the dot file

html_log

class `html_log.`**`html_log`** (*`tb`*, *`htmlfile`*)

Bases: **`object`**

create a html log file after tbot hs finished
create a nicer log ... see for an example:
http://xeidos.ddns.net/tbot/id_189/html_log.html
the created html file needs the css file:
https://github.com/hsdenx/tbot/blob/testing/log/multiplexed_tbotlog.css

- **parameters**, **types**, **return** and **return types**:

  Parameters:
  - **arg1** -- tb
  - **arg2** -- filename which gets created, place tb.workdir

`create_htmlfile` **()**

create the html file

statistic_plot

class `statisitic_plot.`**`statistic_plot_backend`** (*`tb`*, *`fdfile`*, *`ignorelist`*)

Bases: **`object`**

create a statistic of called testcases
create a stat.dat file for creating a TC statistic image with gnuplot
call "gnuplot balkenplot.sem" in tbot workdir after a TBot is finsihed, so you need gnuplot installed on your system.
used balkenplot.sem file: # set terminal png transparent nocrop enhanced size 450,320 font "arial,8" # set output 'histograms.4.png'
set boxwidth 0.75 absolute set style fill solid 1.00 border lt -1 set key outside right top vertical Left reverse noenhanced autotitles columnhead nobox set key invert samplen 4 spacing 1 width 0 height 0 set style histogram rowstacked title offset character 0, 0, 0 set datafile missing '-' set style data histograms set xtics border in scale 0,0 nomirror rotate by -45 offset character 0, 0, 0 autojustify set xtics norangelimit font ",8" set xtics () set title "TC statistic"
set grid ytics set terminal jpeg enhanced size 2048,768 set output "output.jpg"
i = 2 plot 'stat.dat' using 2:xtic(1), for [i=3:3] '' using i

- **parameters**, **types**, **return** and **return types**:

Parameters:

- **arg1** -- tb
- **arg2** -- filename which gets created
- **arg3** -- list of strings, containing testcasesnames, which get ignored

`create_statfile` **()**
create the statistic file

Documentation of all Testcases

src/tc/board/tc_board_aristainetos2.py

```
# start with
# tbot.py -s lab_denx -c aristainetos2 -t tc_board_aristainetos2.py
# start all testcases for the aristainetos2 board
# tc_board_aristainetos2_linux_tests.py
# tc_workfd_set_toolchain.py
```

used Testcases:

src/tc/board/tc_board_aristainetos2.py. *src/tc/board/tc_board_aristainetos2_linux_tests.py*. *src/tc/tc_workfd_set_toolchain.py*.

https://github.com/hsdenx/tbot/tree/master/src/tc/board/tc_board_aristainetos2.py

src/tc/board/tc_board_aristainetos2_linux.py

```
# start with
# tbot.py -s lab_denx -c aristainetos2 -t tc_board_aristainetos2_linux.py
# start all linux testcases for the aristainetos2 board
```

used Testcases:

src/tc/board/tc_board_aristainetos2_linux.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/board/tc_board_aristainetos2_linux.py

src/tc/board/tc_board_aristainetos2_linux_bisect.py

```
# start with
# tbot.py -s lab_denx -c aristainetos2 -t tc_board_aristainetos2_linux_bisect.py
# start a git bisect for the aristainetos2 board
```

used Testcases:

src/tc/board/tc_board_aristainetos2_linux_bisect.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/board/tc_board_aristainetos2_linux_bisect.py

src/tc/board/tc_board_aristainetos2_linux_tests.py

```
# start with
# tbot.py -s lab_denx -c aristainetos2 -t tc_board_aristainetos2_linux_tests.py
# start all linux testcases for the aristainetos2 board
```

used Testcases:

src/tc/board/tc_board_aristainetos2_linux_tests.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/board/tc_board_aristainetos2_linux_tests.py

src/tc/board/tc_board_ccu1_tests.py

```
# start with
# tbot.py -s lab_denx -c ccu1 -t tc_board_ccu1_tests.py
# start all testcases for the ccu1 board
```

used Testcases:

src/tc/board/tc_board_ccu1_tests.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/board/tc_board_ccu1_tests.py

src/tc/board/tc_board_corvus.py

```
# start with
# tbot.py -s lab_denx -c corvus -t tc_board_corvus.py
# start all testcases for the corvus board
```

used Testcases:

src/tc/board/tc_board_corvus.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/board/tc_board_corvus.py

src/tc/board/tc_board_dxr2.py

```
# start with
# tbot.py -s lab_denx -c dxr2 -t tc_board_dxr2.py
# start all testcases for the dxr2 board
```

used Testcases:

src/tc/board/tc_board_dxr2.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/board/tc_board_dxr2.py

src/tc/board/tc_board_dxr2_linux.py

```
# start with
# tbot.py -s lab_denx -c dxr2 -t tc_board_dxr2_linux.py
# start all linux testcases for the dxr2 board
```

used Testcases:

src/tc/board/tc_board_dxr2_linux.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/board/tc_board_dxr2_linux.py

src/tc/board/tc_board_dxr2_lx_ubi_tests.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_board_dxr2_lx_ubi_tests.py
# more dxr2 specific ubi tests, maybe make them common
```

used Testcases:

src/tc/board/tc_board_dxr2_lx_ubi_tests.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/board/tc_board_dxr2_lx_ubi_tests.py

src/tc/board/tc_board_dxr2_ub.py

```
# start with
# tbot.py -s lab_denx -c dxr2 -t tc_board_dxr2_ub.py
# start all u-boot testcases for the dxr2 board
```

used Testcases:

src/tc/board/tc_board_dxr2_ub.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/board/tc_board_dxr2_ub.py

src/tc/board/tc_board_dxr2_ub_ubi.py

```
# start with
# tbot.py -s lab_denx -c dxr2 -t tc_board_dxr2_ub_ubi.py
# start all ubi testcases for the dxr2 board
```

used Testcases:

src/tc/board/tc_board_dxr2_ub_ubi.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/board/tc_board_dxr2_ub_ubi.py

src/tc/board/tc_board_dxr2_uboot_patchwork.py

```
# start with
# python2.7 src/common/tbot.py -c tbot_dxr2_uboot.cfg -t tc_board_dxr2_uboot_patchwork.py
# dxr2 check all patches with patchworknumber > default_nr
# in patchwork, if it is checkpatch clean and applies to
# current mainline without errors
```

used Testcases:

src/tc/board/tc_board_dxr2_uboot_patchwork.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/board/tc_board_dxr2_uboot_patchwork.py

src/tc/board/tc_board_fipad.py

```
# start with
# tbot.py -s lab_denx -c fipad -t tc_board_fipad.py
# start all U-Boot/linux testcases for the fipad board
```

used Testcases:

src/tc/board/tc_board_fipad.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/board/tc_board_fipad.py

src/tc/board/tc_board_fipad_linux.py

```
# start with
# tbot.py -s lab_denx -c fipad -t tc_board_fipad_linux.py
# start all linux testcases for the fipad board
```

used Testcases:

src/tc/board/tc_board_fipad_linux.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/board/tc_board_fipad_linux.py

src/tc/board/tc_board_fipad_ub_tests.py

```
# start with
# tbot.py -s lab_denx -c fipad -t tc_board_fipad_ub_tests.py
# start all U-Boot testcases for the fipad board
```

used Testcases:

src/tc/board/tc_board_fipad_ub_tests.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/board/tc_board_fipad_ub_tests.py

src/tc/board/tc_board_fipad_upd_ub.py

```
# start with
# tbot.py -s lab_denx -c fipad -t tc_board_fipad_upd_ub.py
# update SPL and u-boot.img on the SPI NOR or the MMC0
# card, and boot it ...
```

used Testcases:

src/tc/board/tc_board_fipad_upd_ub.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/board/tc_board_fipad_upd_ub.py

src/tc/board/tc_board_fipad_upd_ub_mmc.py

```
# start with
# tbot.py -s lab_denx -c fipad -t tc_board_fipad_upd_ub_mmc.py
# update SPL and u-boot.img on the MMC0
```

used Testcases:

src/tc/board/tc_board_fipad_upd_ub_mmc.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/board/tc_board_fipad_upd_ub_mmc.py

src/tc/board/tc_board_fipad_upd_ub_spi.py

```
# start with
# tbot.py -s lab_denx -c fipad -t tc_board_fipad_upd_ub_spi.py
# update SPL and u-boot.img on the SPI NOR
```

used Testcases:

src/tc/board/tc_board_fipad_upd_ub_spi.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/board/tc_board_fipad_upd_ub_spi.py

src/tc/board/tc_board_flea3.py

```
# start with
# tbot.py -s lab_denx -c flea3 -t tc_board_flea3.py
# start all testcases for the flea3 board
# currently only test the nor unprotect with linux
```

used Testcases:

src/tc/board/tc_board_flea3.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/board/tc_board_flea3.py

src/tc/board/tc_board_mcx.py

```
# start with
# tbot.py -s lab_denx -c mcx -t tc_board_mcx.py
# start all testcases for the mcx board linux stable and linux-ml
```

used Testcases:

src/tc/board/tc_board_mcx.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/board/tc_board_mcx.py

src/tc/board/tc_board_mcx_tests.py

```
# start with
# tbot.py -s lab_denx -c mcx -t tc_board_mcx_tests.py
# start all testcases for the mcx board
```

used Testcases:

src/tc/board/tc_board_mcx_tests.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/board/tc_board_mcx_tests.py

src/tc/board/tc_board_shc.py

```
# start with
# tbot.py -s lab_denx -c shc -t tc_board_shc.py
# start all testcases for the shc board linux and linux-stable
```

used Testcases:

src/tc/board/tc_board_shc.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/board/tc_board_shc.py

src/tc/board/tc_board_shc_compile_ml.py

```
# start with
# tbot.py -s lab_denx -c shc -t tc_board_shc_compile_ml.py
# compile ML linux kernel for the shc board
```

used Testcases:

src/tc/board/tc_board_shc_compile_ml.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/board/tc_board_shc_compile_ml.py

src/tc/board/tc_board_shc_tests.py

```
# start with
# tbot.py -s lab_denx -c shc -t tc_board_shc_tests.py
# start all testcases for the shc board
```

used Testcases:

src/tc/board/tc_board_shc_tests.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/board/tc_board_shc_tests.py

src/tc/board/tc_board_shc_ub_create_regdump.py

```
# start with
# tbot.py -s lab_denx -c shc -t tc_board_shc_ub_create_regdump.py
# create a uboot regdump for all interesting registers
# on the shc board
```

used Testcases:

src/tc/board/tc_board_shc_ub_create_regdump.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/board/tc_board_shc_ub_create_regdump.py

src/tc/board/tc_board_shc_ub_tests.py

```
# start with
# tbot.py -s lab_denx -c shc -t tc_board_shc_ub_tests.py
# start all U-Boot testcases for the shc board
```

used Testcases:

src/tc/board/tc_board_shc_ub_tests.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/board/tc_board_shc_ub_tests.py

src/tc/board/tc_board_shc_upd_ub.py

```
# start with
# tbot.py -s lab_denx -c shc -t tc_board_shc_upd_ub.py
# update MLO and u-boot.img on the SD card or the eMMC
# card, and boot it ...
```

used Testcases:

src/tc/board/tc_board_shc_upd_ub.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/board/tc_board_shc_upd_ub.py

src/tc/board/tc_board_sigmatek-nand.py

```
# start with
# tbot.py -s lab_denx -c sigmatek-nand -t tc_board_sigmatek-nand.py
# On the sigmatek-nand board we have problems with a crash in U-boot
# We do:
# - wait until linux state is reached
# - wait random seconds (3 -10)
# - power off the board
# - wait 3 seconds for powering really of the board
# - loop this 50 times
```

used Testcases:

src/tc/board/tc_board_sigmatek-nand.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/board/tc_board_sigmatek-nand.py

src/tc/board/tc_board_sirius_dds.py

```
# start with
# python2.7 src/common/tbot.py -c tbot_sirius_dds.cfg -t tc_board_sirius_dds.py
# On the sirius board we have problems with ubifs
# on nand flash and power cuts. So this is a special
# testcase for this board. We do:
# - go into statte u-boot
# - start linux with ubifs as rootfs
# - wait until Userspace APP SiriusApplicat is started
# - wait random seconds (3 -10)
# - power off the board
# - wait 3 seconds for powering really of the board
# - loop this 50 times
# if we have an ubifs error, testcase ends with error
```

used Testcases:

src/tc/board/tc_board_sirius_dds.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/board/tc_board_sirius_dds.py

src/tc/board/tc_board_smartweb.py

```
# start with
# tbot.py -s lab_denx -c smartweb -t tc_board_smartweb.py
# start all testcases for the smartweb board
```

used Testcases:

src/tc/board/tc_board_smartweb.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/board/tc_board_smartweb.py

src/tc/board/tc_board_taurus.py

```
# start with
# tbot.py -s lab_denx -c taurus -t tc_board_taurus.py
# start all testcases for the taurus board
```

used Testcases:

src/tc/board/tc_board_taurus.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/board/tc_board_taurus.py

src/tc/board/tc_board_tqm5200s_try_cur_ub.py

```
# start with
# tbot.py -s lab_denx -c tqm5200s -t tc_board_tqm5200s_try_cur_ub.py
# remove current u-boot code on the lab PC
# then call tc tc_board_tqm5200s_ub_comp_install.py
```

used Testcases:

src/tc/board/tc_board_tqm5200s_try_cur_ub.py. *src/tc/board/tc_board_tqm5200s_ub_comp_install.py*.

https://github.com/hsdenx/tbot/tree/master/src/tc/board/tc_board_tqm5200s_try_cur_ub.py

src/tc/board/tc_board_tqm5200s_ub_comp_install.py

```
# start with
# tbot.py -s lab_denx -c tqm5200s -t tc_board_tqm5200s_ub_comp_install.py
# compile and install U-Boot for the tqm5200s board
# install U-Boot with BDI
```

used Testcases:

src/tc/board/tc_board_tqm5200s_ub_comp_install.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/board/tc_board_tqm5200s_ub_comp_install.py

src/tc/board/tc_linux_create_reg_file_am335x.py

```
# start with
# tbot.py -s lab_denx -c aristainetos2 -t tc_linux_create_reg_file_am335x.py
# create a regfile for am335x SoC registers
```

used Testcases:

src/tc/board/tc_linux_create_reg_file_am335x.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/board/tc_linux_create_reg_file_am335x.py

src/tc/board/tc_linux_create_reg_file_at91sam9g15.py

```
# start with
# python2.7 src/common/tbot.py -c tbot_wivue2.cfg -t tc_linux_create_reg_file_at91sam9g15.py
# create a regfile for at91sam9g15 SoC registers
```

used Testcases:

src/tc/board/tc_linux_create_reg_file_at91sam9g15.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/board/tc_linux_create_reg_file_at91sam9g15.py

src/tc/board/tc_linux_create_reg_file_imx6qdl.py

```
# start with
# tbot.py -s lab_denx -c aristainetos2 -t tc_linux_create_reg_file_imx6qdl.py
# create a regfile for am335x SoC registers
```

used Testcases:

src/tc/board/tc_linux_create_reg_file_imx6qdl.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/board/tc_linux_create_reg_file_imx6qdl.py

src/tc/debugger/bdi/tc_lab_bdi_connect.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_lab_bdi_connect.py
# connect to the BDI if tb.config.board_has_debugger != 0
# - send to workfd tb.config.lab_bdi_upd_uboot_bdi_cmd
# - set BDI prompt tb.config.lab_bdi_upd_uboot_bdi_prompt
# - wait for BDI prompt
```

used Testcases:

src/tc/debugger/bdi/tc_lab_bdi_connect.py.

used config variables:

board_has_debugger. *lab_bdi_upd_uboot_bdi_cmd*. *lab_bdi_upd_uboot_bdi_prompt*.

https://github.com/hsdenx/tbot/tree/master/src/tc/debugger/bdi/tc_lab_bdi_connect.py

src/tc/debugger/bdi/tc_lab_bdi_disconnect.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_lab_bdi_disconnect.py
# disconnect from the BDI
# - send bdi command "quit"
# - set tb.config.linux_prompt
```

used Testcases:

src/tc/debugger/bdi/tc_lab_bdi_disconnect.py.

used config variables:

tb_config_linux_prompt.

https://github.com/hsdenx/tbot/tree/master/src/tc/debugger/bdi/tc_lab_bdi_disconnect.py

src/tc/debugger/bdi/tc_lab_bdi_run.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_lab_bdi_upd_uboot.py
# BDI run
# - send "res halt" to workfd
# - send BDI cmd tb.config.lab_bdi_upd_uboot_bdi_run
```

used Testcases:

src/tc/debugger/bdi/tc_lab_bdi_upd_uboot.py.

used config variables:

lab_bdi_upd_uboot_bdi_run.

https://github.com/hsdenx/tbot/tree/master/src/tc/debugger/bdi/tc_lab_bdi_run.py

src/tc/debugger/bdi/tc_lab_bdi_upd_uboot.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_lab_bdi_upd_uboot.py
# update u-boot with BDI
# - send BDI cmd: "res halt"
# - send BDI cmd: "era"
# - send BDI cmd:
#   tb.config.lab_bdi_upd_uboot_bdi_prog + ' ' + tb.config.lab_bdi_upd_uboot_bdi_file + ' BI
# - send BDI cmd: tb.config.lab_bdi_upd_uboot_bdi_run
```

used Testcases:

src/tc/debugger/bdi/tc_lab_bdi_upd_uboot.py.

used config variables:

lab_bdi_upd_uboot_bdi_prog. *lab_bdi_upd_uboot_bdi_file*. *lab_bdi_upd_uboot_bdi_run*.

https://github.com/hsdenx/tbot/tree/master/src/tc/debugger/bdi/tc_lab_bdi_upd_uboot.py

src/tc/default/tc_def_tbot.py

```
# start with
# tbot.py -s lab_denx -c cfgfile -t tc_def_tbot.py
# simple set default values for tbot
```

used Testcases:

src/tc/default/tc_def_tbot.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/default/tc_def_tbot.py

src/tc/default/tc_def_ub.py

```
# start with
# tbot.py -s lab_denx -c cfgfile -t tc_def_ub.py
# simple set default values for U-Boot testcases
```

used Testcases:

src/tc/default/tc_def_ub.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/default/tc_def_ub.py

src/tc/demo/tc_demo_can_part1.py

```
# start with
# python2.7 src/common/tbot.py -c tbot_board.cfg -t tc_demo_can_part1.py
# start tc:
# starts a can demo
# For this demo the fipad board in the denx lab is used.
# To test the CAN bus we have in the DENX lab installed a PC, called
# CANPC to which a PEAK CAN adapter is attached, which then is connected
# to the CAN bus the fipad board is also connected.
#
# We use tc_workfd_can.py for testing
#
# We open a new connection to the LabPC, called canm and then we ssh
# to the CANPC, from where we then start candump, while on the console
# connection a cansend was started. So we can read from the canm
# connection, the bytes we send with cansend on the console connection.
#
# If we got the same bytes as we send -> TC True
# else the TC returns False
#
# Only one cansend call is tested ... room for more.
```

used Testcases:

src/tc/demo/tc_demo_can_part1.py. *src/tc/linux/tc_workfd_can.py*.

https://github.com/hsdenx/tbot/tree/master/src/tc/demo/tc_demo_can_part1.py

src/tc/demo/tc_demo_compile_install_test.py

```
# start with
# tbot.py -c -s lab_denx -c demo -t tc_demo_compile_install_test.py
# start tc:
# - compile source tree
# - install bin on board
# - call board uboot testcase
```

used Testcases:

src/tc/demo/tc_demo_compile_install_test.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/demo/tc_demo_compile_install_test.py

src/tc/demo/tc_demo_get_ub_code.py

```
# start with
# python2.7 src/common/tbot.py -c tbot_board.cfg -t tc_demo_get_ub_code.py
# start tc:
# - rm old u-boot tree (if there is one)
# - tc_lab_get_uboot_source.py
# -
```

used Testcases:

src/tc/demo/tc_demo_get_ub_code.py. *src/tc/tc_lab_get_uboot_source.py*.

https://github.com/hsdenx/tbot/tree/master/src/tc/demo/tc_demo_get_ub_code.py

src/tc/demo/tc_demo_part1.py

```
# start with
# tbot.py -s lab_denx -c smartweb -t tc_demo_part1.py
# start tc:
# - call tc_demo_get_ub_code.py
# - call tc_demo_compile_install_test.py
```

used Testcases:

src/tc/demo/tc_demo_part1.py. *src/tc/demo/tc_demo_get_ub_code.py*. *src/tc/demo/tc_demo_compile_install_test.py*.

https://github.com/hsdenx/tbot/tree/master/src/tc/demo/tc_demo_part1.py

src/tc/demo/tc_demo_part2.py

```
# start with
# tbot.py -s lab_denx -c smartweb -t tc_demo_part2.py
# start tc:
# - call tc_demo_get_ub_code.py
# - call tc_demo_compile_install_test.py
```

used Testcases:

src/tc/demo/tc_demo_part2.py. *src/tc/demo/tc_demo_get_ub_code.py*. *src/tc/demo/tc_demo_compile_install_test.py*.

https://github.com/hsdenx/tbot/tree/master/src/tc/demo/tc_demo_part2.py

src/tc/demo/tc_demo_part3.py

```
# start with
# tbot.py -s lab_denx -c smartweb -t tc_demo_part3.py
# start tc:
```

used Testcases:

src/tc/demo/tc_demo_part3.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/demo/tc_demo_part3.py

src/tc/lab/denx/tc_lab_denx_connect_to_board.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_lab_denx_connect_to_board.py
# connect to board with connect
```

used Testcases:

src/tc/lab/denx/tc_lab_denx_connect_to_board.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/lab/denx/tc_lab_denx_connect_to_board.py

src/tc/lab/denx/tc_lab_denx_disconnect_from_board.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_lab_denx_disconnect_from_board.py
# disconnect from board in denx vlab
```

used Testcases:

src/tc/lab/denx/tc_lab_denx_disconnect_from_board.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/lab/denx/tc_lab_denx_disconnect_from_board.py

src/tc/lab/denx/tc_lab_denx_get_power_state.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_lab_denx_get_power_state.py
# get the power state of the board, and save it in
# tb.power_state
```

used Testcases:

src/tc/lab/denx/tc_lab_denx_get_power_state.py.

used config variables:

tb_power_state.

https://github.com/hsdenx/tbot/tree/master/src/tc/lab/denx/tc_lab_denx_get_power_state.py

src/tc/lab/denx/tc_lab_denx_power.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_lab_denx_power.py
# power on/off the board
```

used Testcases:

src/tc/lab/denx/tc_lab_denx_power.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/lab/denx/tc_lab_denx_power.py

src/tc/lab/denx/tc_lab_interactive_get_power_state.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_lab_denx_get_power_state.py
# get the power state of the board through user input,
# and save it in tb.power_state
```

used Testcases:

src/tc/lab/denx/tc_lab_denx_get_power_state.py.

used config variables:

tb_power_state.

https://github.com/hsdenx/tbot/tree/master/src/tc/lab/denx/tc_lab_interactive_get_power_state.py

src/tc/lab/denx/tc_lab_interactive_power.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_lab_denx_power.py
# power on/off the board from hand
```

used Testcases:

src/tc/lab/denx/tc_lab_denx_power.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/lab/denx/tc_lab_interactive_power.py

src/tc/linux/ubi/tc_lx_ubi_attach.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_lx_ubi_attach.py
```

used Testcases:

src/tc/linux/ubi/tc_lx_ubi_attach.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/ubi/tc_lx_ubi_attach.py

src/tc/linux/ubi/tc_lx_ubi_detach.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_lx_ubi_detach.py
# detach ubi device tb.config.tc_ubi_mtd_dev
```

used Testcases:

src/tc/linux/ubi/tc_lx_ubi_detach.py.

used config variables:

tc_ubi_mtd_dev.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/ubi/tc_lx_ubi_detach.py

src/tc/linux/ubi/tc_lx_ubi_format.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_lx_ubi_format.py
# ubiformat tb.config.tc_ubi_mtd_dev with tb.config.tc_lx_ubi_format_filename
```

used Testcases:

src/tc/linux/ubi/tc_lx_ubi_format.py.

used config variables:

tc_ubi_mtd_dev. *tc_lx_ubi_format_filename*.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/ubi/tc_lx_ubi_format.py

src/tc/linux/ubi/tc_lx_ubi_info.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_lx_ubi_info.py
# ubinfo tb.config.tc_ubi_ubi_dev
```

used Testcases:

src/tc/linux/ubi/tc_lx_ubi_info.py.

used config variables:

tc_ubi_ubi_dev.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/ubi/tc_lx_ubi_info.py

src/tc/linux/ubi/tc_lx_ubi_tests.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_lx_ubi_tests.py
# - install mtd utils if needed with tc_lx_mtdutils_install.py
# - attach ubi device with tc_lx_ubi_attach.py
# - get info with tc_lx_ubi_info.py
# - get parameters with tc_lx_get_ubi_parameters.py
```

used Testcases:

src/tc/linux/ubi/tc_lx_ubi_tests.py. *src/tc/linux/tc_lx_mtdutils_install.py*. *src/tc/linux/ubi/tc_lx_ubi_attach.py*. *src/tc/linux/ubi/tc_lx_ubi_info.py*. *src/tc/linux/tc_lx_get_ubi_parameters.py*.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/ubi/tc_lx_ubi_tests.py

src/tc/linux/tc_lx_bonnie.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_lx_bonnie.py
# run a bonnie test, if timer tc_workfd_check_tc_time.py timed out
# - try to install bonnie if not is installed tc_lx_bonnie_install.py
# - start bonnie on device tb.config.tc_lx_bonnie_dev with
#   size tb.config.tc_lx_bonnie_sz
```

used Testcases:

src/tc/linux/tc_lx_bonnie.py. *src/tc/linux/tc_workfd_check_tc_time.py*. *src/tc/linux/tc_lx_bonnie_install.py*.

used config variables:

tc_lx_bonnie_dev. *tc_lx_bonnie_sz*.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_lx_bonnie.py

src/tc/linux/tc_lx_bonnie_install.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_lx_bonnie_install.py
# get bonnie source and install it
```

used Testcases:

src/tc/linux/tc_lx_bonnie_install.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_lx_bonnie_install.py

src/tc/linux/tc_lx_check_reg_file.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_lx_check_reg_file.py
# checks if the default values in reg file tb.config.tc_lx_create_reg_file_name
# on the tbot host in tb.workdir have the same values, as the
# registers on the board. Needs devmem2 installed.
# format of the regfile:
# regaddr mask type defval
# ToDo: use the file from the lab host, not the tbot host
```

used Testcases:

src/tc/linux/tc_lx_check_reg_file.py.

used config variables:

tc_lx_create_reg_file_name. *tb_workdir*.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_lx_check_reg_file.py

src/tc/linux/tc_lx_check_usb_authorized.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_lx_check_usb_authorized.py
# check if usb device tb.config.tc_lx_check_usb_authorized needs authorizing
```

used Testcases:

src/tc/linux/tc_lx_check_usb_authorized.py.

used config variables:

tc_lx_check_usb_authorized.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_lx_check_usb_authorized.py

src/tc/linux/tc_lx_cpufreq.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_lx_cpufreq.py
# check if frequencies in tb.config.tc_lx_cpufreq_frequences
# are possible to set with cpufreq-info
```

used Testcases:

src/tc/linux/tc_lx_cpufreq.py.

used config variables:

tc_lx_cpufreq_frequences.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_lx_cpufreq.py

src/tc/linux/tc_lx_create_dummy_file.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_lx_create_dummy_file.py
# create a random dummy file tb.tc_lx_dummy_file_tempfile in linux
# on tb.c_con with bs = tb.tc_lx_dummy_file_bs and
# count = tb.tc_lx_dummy_file_count
```

used Testcases:

src/tc/linux/tc_lx_create_dummy_file.py.

used config variables:

tb_tc_lx_dummy_file_tempfile. *tb_c_con*. *tb_tc_lx_dummy_file_bs*. *tb_tc_lx_dummy_file_count*.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_lx_create_dummy_file.py

src/tc/linux/tc_lx_create_reg_file.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_lx_create_reg_file.py
# creates a reg file tb.config.tc_lx_create_reg_file_name on the tbot host
# in tb.workdir
# read from tb.config.tc_lx_create_reg_file_start to tb.config.tc_lx_create_reg_file_stop
# and writes the results in the regfile
# format of the regfile:
# regaddr mask type defval
# This reg file can be used as a default file, how the
# registers must be setup, check it with testcase
# tc_lx_check_reg_file.py
# ToDo: use the file from the lab host, not the tbot host
```

used Testcases:

src/tc/linux/tc_lx_create_reg_file.py. *src/tc/linux/tc_lx_check_reg_file.py*.

used config variables:

tc_lx_create_reg_file_name. *tb_workdir*. *tc_lx_create_reg_file_start*. *tc_lx_create_reg_file_stop*.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_lx_create_reg_file.py

src/tc/linux/tc_lx_devmem2_install.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_lx_devmem2_install.py
# get devmem2 source from www.lartmaker.nl/lartware/port/devmem2.c
# and install it
```

used Testcases:

src/tc/linux/tc_lx_devmem2_install.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_lx_devmem2_install.py

src/tc/linux/tc_lx_dmesg_grep.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_lx_dmesg_grep.py
# check if string tb.config.tc_lx_dmesg_grep_name is in dmesg output.
```

used Testcases:

src/tc/linux/tc_lx_dmesg_grep.py.

used config variables:

tc_lx_dmesg_grep_name.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_lx_dmesg_grep.py

src/tc/linux/tc_lx_eeprom.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_lx_eeprom.py
# Test an eeprom:
# - read the content from eeprom @ tb.config.tc_lx_eeprom_tmp_dir
#   with "cat" into tmpfile
# - check tb.config.tc_lx_eeprom_wp_gpio != 'none'
#   if WP pin works
# - generate random file with tb.config.tc_lx_eeprom_wp_sz size
# - write it into eeprom
# - reread it
# - compare it with original
# - restore original eeprom content at end
```

used Testcases:

src/tc/linux/tc_lx_eeprom.py.

used config variables:

tc_lx_eeprom_tmp_dir. *tc_lx_eeprom_wp_gpio*. *tc_lx_eeprom_wp_sz*.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_lx_eeprom.py

src/tc/linux/tc_lx_get_ubi_parameters.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_lx_get_ubi_parameters.py
# get ubi parameters of ubi device tb.config.tc_ubi_mtd_dev
# save them into:
# - tb.config.tc_ubi_max_leb_cnt
# - tb.config.tc_ubi_min_io_size
# - tb.config.tc_ubi_leb_size
```

used Testcases:

src/tc/linux/tc_lx_get_ubi_parameters.py.

used config variables:

tc_ubi_mtd_dev. *tc_ubi_max_leb_cnt*. *tc_ubi_min_io_size*. *tc_ubi_leb_size*.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_lx_get_ubi_parameters.py

src/tc/linux/tc_lx_get_version.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_lx_get_version.py
# get the linux version and create event LINUX_VERSION
# save the linux version in tb.config.tc_return
```

used Testcases:

src/tc/linux/tc_lx_get_version.py.

used config variables:

tc_return.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_lx_get_version.py

src/tc/linux/tc_lx_gpio.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_lx_gpio.py
# set in linux gpio tb.config.tc_lx_gpio_nr to direction tb.config.tc_lx_gpio_dir
# and value tb.config.tc_lx_gpio_val
```

used Testcases:

src/tc/linux/tc_lx_gpio.py.

used config variables:

tc_lx_gpio_nr. *tc_lx_gpio_dir*. *tc_lx_gpio_val*.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_lx_gpio.py

src/tc/linux/tc_lx_mount.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_lx_mount.py
# mount device tb.config.tc_lx_mount_dev with fs type tb.config.tc_lx_mount_fs_type
# to tb.config.tc_lx_mount_dir
```

used Testcases:

src/tc/linux/tc_lx_mount.py.

used config variables:

tc_lx_mount_dev. *tc_lx_mount_fs_type*. *tc_lx_mount_dir*.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_lx_mount.py

src/tc/linux/tc_lx_mtdutils_install.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_lx_mtdutils_install.py
# check if mtdutils are installed. If not, clone the code with
# git clone git://git.infradead.org/mtd-utils.git mtd-utils
# and install it
```

used Testcases:

src/tc/linux/tc_lx_mtdutils_install.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_lx_mtdutils_install.py

src/tc/linux/tc_lx_partition_check.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_lx_partition_check.py
# cp a dummy file into a partiton umount/mount it and
# compare it.
# - Mount tb.config.tc_lx_mount_dir with tc_lx_mount.py
```

used Testcases:

src/tc/linux/tc_lx_partition_check.py. *src/tc/linux/tc_lx_mount.py*.

used config variables:

tc_lx_mount_dir.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_lx_partition_check.py

src/tc/linux/tc_lx_printenv.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_lx_printenv.py
# simple printenv linux command
```

used Testcases:

src/tc/linux/tc_lx_printenv.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_lx_printenv.py

src/tc/linux/tc_lx_regulator.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_lx_regulator.py
# check if regulators in tb.config.tc_lx_regulator_nrs exist, and have
# the correct microvolts settings.
```

used Testcases:

src/tc/linux/tc_lx_regulator.py.

used config variables:

tc_lx_regulator_nrs.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_lx_regulator.py

src/tc/linux/tc_lx_trigger_wdt.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_lx_trigger_wdt.py
# simple trigger wdt with command tb.config.tc_lx_trigger_wdt_cmd
```

used Testcases:

src/tc/linux/tc_lx_trigger_wdt.py.

used config variables:

tc_lx_trigger_wdt_cmd.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_lx_trigger_wdt.py

src/tc/linux/tc_lx_uname.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_lx_uname.py
# simple linux "uname -a" command
```

used Testcases:

src/tc/linux/tc_lx_uname.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_lx_uname.py

src/tc/linux/tc_workfd_apply_local_patches.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_workfd_apply_local_patches.py
# apply patches from directory tb.config.tc_workfd_apply_local_patches_dir
# with 'git am -3' to the source in current directory.
# if tb.config.tc_workfd_apply_local_patches_checkpatch_cmd != 'none'
# check the patches with the checkpatch cmd tb.config.tc_workfd_apply_local_patches_checkpat
# before applying.
```

used Testcases:

src/tc/linux/tc_workfd_apply_local_patches.py.

used config variables:

tc_workfd_apply_local_patches_dir. *tc_workfd_apply_local_patches_checkpatch_cmd*. *tc_workfd_apply_local_patches_checkpatch_cmd*.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_workfd_apply_local_patches.py

src/tc/linux/tc_workfd_apply_patchwork_patches.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_workfd_apply_patchwork_patches.py
# apply patchworkpatches from list:
# tb.config.tc_workfd_apply_patchwork_patches_list:
# to source in current directory.
# creates event:
# - PW_NR: which patchwork number used
# - PW_CLEAN: is it checkpatch clean
# - PW_AA: already applied
# - PW_APPLY: apply it clean to source
```

used Testcases:

src/tc/linux/tc_workfd_apply_patchwork_patches.py.

used config variables:

tb_config_tc_workfd_apply_patchwork_patches_list:.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_workfd_apply_patchwork_patches.py

src/tc/linux/tc_workfd_can.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_workfd_can.py
#
# minimal can test:
# starts a new connection named tb_canm. This connection runs
# on board/PC which has a can conncetion to the board tbot
# tests, named CAN PC.
# If necessary (tb.config.tc_workfd_can_ssh != 'no'), tc connects first
# to ssh (if the CAN PC is not the lab PC). Also if necessary
# (tb.config.tc_workfd_can_su != 'no', switch to superuser on the CAN PC.
#
# Set on the CAN PC, with the "ip" command the bitrate
# tb.config.tc_workfd_can_bitrate for the can device tb.config.tc_workfd_can_dev
# and activate the interface.
#
# Now on the board, go into tb.config.tc_workfd_can_iproute_dir
# (which contains the "ip" command ...
# Set the bitrate with it and activate the can interface.
# Goto into tb.config.tc_workfd_can_util_dir which contains canutils
# Send '123#DEADBEEF' with cansend
#
# check if the CAN PC gets this string.
# End True if this is the case, False else
#
# ToDo:
# - get rid of tb.config.tc_workfd_can_iproute_dir and tb.config.tc_workfd_can_util_dir
#   (add the commands to rootfs ...)
# - support different can devices on the CAN PC and board
```

used Testcases:

src/tc/linux/tc_workfd_can.py.

used config variables:

tc_workfd_can_bitrate. *tc_workfd_can_dev*. *tc_workfd_can_iproute_dir*. *tc_workfd_can_util_dir*. *tc_workfd_can_iproute_dir*. *tc_workfd_can_util_dir*.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_workfd_can.py

src/tc/linux/tc_workfd_cd_to_dir.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_workfd_cd_to_dir.py
# simple cd into directory tb.config.tc_workfd_cd_name
```

used Testcases:

src/tc/linux/tc_workfd_cd_to_dir.py.

used config variables:

tc_workfd_cd_name.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_workfd_cd_to_dir.py

src/tc/linux/tc_workfd_check_cmd_success.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_workfd_check_cmd_success.py
# simple check if previous shell command was succesful
```

used Testcases:

src/tc/linux/tc_workfd_check_cmd_success.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_workfd_check_cmd_success.py

src/tc/linux/tc_workfd_check_if_cmd_exist.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_workfd_check_if_cmd_exist.py
# check if a command exists
```

used Testcases:

src/tc/linux/tc_workfd_check_if_cmd_exist.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_workfd_check_if_cmd_exist.py

src/tc/linux/tc_workfd_check_if_dir_exist.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_workfd_check_if_dir_exist.py
# check if a dir in tbot workdir exist
# this tc returns always true, but sets
# tb.config.tc_return True or False, because we may not
# want to end testcase failed, if dir not exists.
```

used Testcases:

src/tc/linux/tc_workfd_check_if_dir_exist.py.

used config variables:

tc_return.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_workfd_check_if_dir_exist.py

src/tc/linux/tc_workfd_check_if_file_exist.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_workfd_check_if_file_exist.py
# check if a file in tbot workdir exist
```

used Testcases:

src/tc/linux/tc_workfd_check_if_file_exist.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_workfd_check_if_file_exist.py

src/tc/linux/tc_workfd_check_tc_time.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_workfd_check_tc_time.py
# check if time for a special testcase is expired.
# some testcases (like writting in a flash) are not good for
# execute them every day, so give them a timeout. This testcase
# checks, if the testcases is ready for a new run.
# False means time is not expired
# True means time is expired
```

used Testcases:

src/tc/linux/tc_workfd_check_tc_time.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_workfd_check_tc_time.py

src/tc/linux/tc_workfd_compile_linux.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_workfd_compile_linux.py
# compile linux:
# - set toolchain with tc_lab_set_toolchain.py
# - if tb.config.tc_workfd_compile_linux_clean == 'yes'
#   call "make mrproper"
# - tb.config.tc_workfd_compile_linux_load_addr != 'no'
#   add LOAD_ADDR=tb.config.tc_workfd_compile_linux_load_addr to make
# - make tb.config.tc_workfd_compile_linux_boardname defconfig
# - make tb.config.tc_workfd_compile_linux_makeoptions tb.config.tc_workfd_compile_linux_mak
# - if tb.config.tc_workfd_compile_linux_modules != 'none'
#   compile modules
# - if tb.config.tc_workfd_compile_linux_dt_name != 'none'
#   compile DTB from list tb.config.tc_workfd_compile_linux_dt_name
# - if tb.config.tc_workfd_compile_linux_fit_its_file != 'no'
#   create FIT image
#   mkimage path: tb.config.tc_workfd_compile_linux_mkimage
#   fit description file: tb.config.tc_workfd_compile_linux_fit_its_file
#   tb.config.tc_workfd_compile_linux_fit_file
# - if tb.config.tc_workfd_compile_linux_append_dt != 'no'
#   append dtb to kernel image
# tb.config.tc_workfd_compile_linux_boardname _defconfig
```

used Testcases:

src/tc/linux/tc_workfd_compile_linux.py. *src/tc/tc_lab_set_toolchain.py*.

used config variables:

tc_workfd_compile_linux_clean. *tc_workfd_compile_linux_load_addr*. *tc_workfd_compile_linux_boardname*. *tc_workfd_compile_linux_makeoptions*. *tc_workfd_compile_linux_make_target*. *tc_workfd_compile_linux_modules*. *tc_workfd_compile_linux_dt_name*. *tc_workfd_compile_linux_dt_name*. *tc_workfd_compile_linux_fit_its_file*. *tc_workfd_compile_linux_mkimage*. *tc_workfd_compile_linux_fit_its_file*. *tc_workfd_compile_linux_fit_file*. *tc_workfd_compile_linux_append_dt*. *tc_workfd_compile_linux_boardname*.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_workfd_compile_linux.py

src/tc/linux/tc_workfd_connect_with_conmux.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_workfd_connect_with_conmux.py
# connect to console with conmux
# Never tested !!!
```

used Testcases:

src/tc/linux/tc_workfd_connect_with_conmux.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_workfd_connect_with_conmux.py

src/tc/linux/tc_workfd_connect_with_kermit.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_workfd_connect_with_kermit.py
# connect with kermit to serials board console
# - if tb.config.tc_workfd_connect_with_kermit_ssh != 'none'
#   connect first with ssh to another PC (where kermit is started)
# - start kermit
# - if tb.config.tc_workfd_connect_with_kermit_rlogin == 'none'
#   connect with command in tb.config.tc_workfd_connect_with_kermit_rlogin
#   else
#   set line tb.config.kermit_line and speed tb.config.kermit_speed and
#   connect to serial line.
```

used Testcases:

src/tc/linux/tc_workfd_connect_with_kermit.py.

used config variables:

tc_workfd_connect_with_kermit_ssh. *tc_workfd_connect_with_kermit_rlogin*. *tc_workfd_connect_with_kermit_rlogin*. *kermit_line*. *kermit_speed*.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_workfd_connect_with_kermit.py

src/tc/linux/tc_workfd_cp_file.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_workfd_cp_file.py
# simple copy file from tb.tc_workfd_cp_file_a to tb.tc_workfd_cp_file_b
```

used Testcases:

src/tc/linux/tc_workfd_cp_file.py.

used config variables:

tb_tc_workfd_cp_file_a. *tb_tc_workfd_cp_file_b*.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_workfd_cp_file.py

src/tc/linux/tc_workfd_create_ubi_rootfs.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_workfd_create_ubi_rootfs.py
# create a ubifs rootfs
# ubi rootfs path: tb.config.tc_workfd_create_ubi_rootfs_path
# ubi parameters:
# tb.config.tc_ubi_min_io_size tb.config.tc_ubi_leb_size tb.config.tc_ubi_max_leb_cnt
# output path: tb.config.tc_workfd_create_ubi_rootfs_target
```

used Testcases:

src/tc/linux/tc_workfd_create_ubi_rootfs.py.

used config variables:

tc_workfd_create_ubi_rootfs_path. *tc_ubi_min_io_size*. *tc_ubi_leb_size*. *tc_ubi_max_leb_cnt*. *tc_workfd_create_ubi_rootfs_target*.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_workfd_create_ubi_rootfs.py

src/tc/linux/tc_workfd_disconnect_with_kermit.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_workfd_connect_with_kermit.py
# disconnect from a kermit connection
```

used Testcases:

src/tc/linux/tc_workfd_connect_with_kermit.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_workfd_disconnect_with_kermit.py

src/tc/linux/tc_workfd_generate_random_file.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_workfd_generate_random_file.py
# simple create a random file tb.tc_workfd_generate_random_file_name
# with tb.tc_workfd_generate_random_file_length length.
```

used Testcases:

src/tc/linux/tc_workfd_generate_random_file.py.

used config variables:

tb_tc_workfd_generate_random_file_name. *tb_tc_workfd_generate_random_file_length*.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_workfd_generate_random_file.py

src/tc/linux/tc_workfd_get_linux_source.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_workfd_get_linux_source.py
# get Linux source tb.config.tc_lab_get_linux_source_git_repo with "git clone"
# and go into the source tree. Apply patches if needed with:
# tc_lab_apply_patches.py and tc_workfd_apply_local_patches.py
```

used Testcases:

src/tc/linux/tc_workfd_get_linux_source.py. *src/tc/tc_lab_apply_patches.py*. *src/tc/linux/tc_workfd_apply_local_patches.py*.

used config variables:

tc_lab_get_linux_source_git_repo.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_workfd_get_linux_source.py

src/tc/linux/tc_workfd_get_list_of_files_in_dir.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_workfd_get_list_of_files_in_dir.py
# get a list of files from directory tb.tc_workfd_get_list_of_files_dir
# tb.config.tc_workfd_get_list_of_files_mask
```

used Testcases:

src/tc/linux/tc_workfd_get_list_of_files_in_dir.py.

used config variables:

tb_tc_workfd_get_list_of_files_dir. *tc_workfd_get_list_of_files_mask*.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_workfd_get_list_of_files_in_dir.py

src/tc/linux/tc_workfd_get_patchwork_number_list.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_workfd_get_patchwork_number_list.py
# get a list of patchworknumbers
# which are delegated to specific user
# tb.config.workfd_get_patchwork_number_user
# currently, this testcase reads "http://patchwork.ozlabs.org/project/uboot/list/"
# and filters out the patches, which are for
# tb.config.workfd_get_patchwork_number_user
# It would be better to login and look for the users
# ToDo list, but I did not find out, how to login ...
# ignore patches on blacklist:
# tb.config.tc_workfd_apply_patchwork_patches_blacklist
# also you can set the patch order with:
# tb.config.tc_workfd_get_patchwork_number_list_order
```

used Testcases:

src/tc/linux/tc_workfd_get_patchwork_number_list.py.

used config variables:

workfd_get_patchwork_number_user. *workfd_get_patchwork_number_user*. *tc_workfd_apply_patchwork_patches_blacklist*. *tc_workfd_get_patchwork_number_list_order*.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_workfd_get_patchwork_number_list.py

src/tc/linux/tc_workfd_get_uboot_config_hex.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_workfd_get_uboot_config_hex.py
# get a hex parameter from U-Boot configuration
# Input:
# tb.config.uboot_get_parameter_file_list: list of files, where TC searches
#   for the define
# tb.uboot_config_option: config option which get searched
#
# return value:
# TC ends True, if hex value found, else False
# tb.config_result: founded hex value, else 'undef'
```

used Testcases:

src/tc/linux/tc_workfd_get_uboot_config_hex.py.

used config variables:

tb_config_uboot_get_parameter_file_list:. *tb_uboot_config_option:*. *tb_config_result:*.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_workfd_get_uboot_config_hex.py

src/tc/linux/tc_workfd_get_uboot_config_string.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_workfd_get_uboot_config_string.py
# get a string parameter from U-Boot configuration
# Input:
# tb.config.uboot_get_parameter_file_list: list of files, where TC searches
#   for the define
# tb.uboot_config_option: config option which get searched
#
# return value:
# TC ends True, if string value found, else False
# tb.config_result: founded string value, else 'undef'
```

used Testcases:

src/tc/linux/tc_workfd_get_uboot_config_string.py.

used config variables:

tb_config_uboot_get_parameter_file_list:. *tb_uboot_config_option:*. *tb_config_result:*.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_workfd_get_uboot_config_string.py

src/tc/linux/tc_workfd_goto_lab_source_dir.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_workfd_goto_lab_source_dir.py
# switch into lab PC source directory tb.config.tc_lab_source_dir
```

used Testcases:

src/tc/linux/tc_workfd_goto_lab_source_dir.py.

used config variables:

tc_lab_source_dir.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_workfd_goto_lab_source_dir.py

src/tc/linux/tc_workfd_goto_linux_code.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_workfd_goto_linux_code.py
# switch into linux source tb.config.tc_lab_source_dir + "/linux-" + tb.config.boardlabname
```

used Testcases:

src/tc/linux/tc_workfd_goto_linux_code.py.

used config variables:

tc_lab_source_dir. *boardlabname*.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_workfd_goto_linux_code.py

src/tc/linux/tc_workfd_goto_tbot_workdir.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_workfd_goto_tbot_workdir.py
# go into the tbot work dir tb.config.tc_workfd_work_dir
# if not exist, create it
```

used Testcases:

src/tc/linux/tc_workfd_goto_tbot_workdir.py.

used config variables:

tc_workfd_work_dir.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_workfd_goto_tbot_workdir.py

src/tc/linux/tc_workfd_goto_uboot_code.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_workfd_goto_uboot_code.py
# switch into U-Boot source tb.config.tc_lab_source_dir + "/u-boot-" + tb.config.boardlabnam
```

used Testcases:

src/tc/linux/tc_workfd_goto_uboot_code.py.

used config variables:

tc_lab_source_dir. *boardlabname*.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_workfd_goto_uboot_code.py

src/tc/linux/tc_workfd_grep.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_workfd_grep.py
# search string tb.tc_workfd_grep_string in file tb.tc_workfd_grep_file
```

used Testcases:

src/tc/linux/tc_workfd_grep.py.

used config variables:

tb_tc_workfd_grep_string. *tb_tc_workfd_grep_file*.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_workfd_grep.py

src/tc/linux/tc_workfd_hdparm.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_workfd_hdparm.py
# make a minimal hdparm check
# call hdparm -t tb.config.tc_workfd_hdparm_dev
# and check if read speed is greater than tb.config.tc_workfd_hdparm_min
# It is possible to add a PATH tb.config.tc_workfd_hdparm_path
# where hdparm is installed
# Testcase fails if readen speed is <= tb.config.tc_workfd_hdparm_min
```

used Testcases:

src/tc/linux/tc_workfd_hdparm.py.

used config variables:

tc_workfd_hdparm_dev. *tb_config_tc_workfd_hdparm_min*. *tc_workfd_hdparm_path*. *tb_config_tc_workfd_hdparm_min*.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_workfd_hdparm.py

src/tc/linux/tc_workfd_insmod.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_workfd_insmod.py
# insmod module tb.tc_workfd_insmod_module with
# module path tb.tc_workfd_insmod_mpath and
# tb.tc_workfd_insmod_module_path
# check if the strings in list tb.tc_workfd_insmod_module_checks
# come back when inserting the module.
```

used Testcases:

src/tc/linux/tc_workfd_insmod.py.

used config variables:

tb_tc_workfd_insmod_module. *tb_tc_workfd_insmod_mpath*. *tb_tc_workfd_insmod_module_path*. *tb_tc_workfd_insmod_module_checks*.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_workfd_insmod.py

src/tc/linux/tc_workfd_iperf.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_workfd_iperf.py
# make a minimal iperf check
# starts an iperf server on tb.tc_workfd_c_sr connection
#   with ip addr tb.tc_workfd_iperf_sip
# starts an iperf "slave" on tb.tc_workfd_c_sl
# waiting for the first result of iperf measure and
# check if the resulting speed is bigger then
# tb.tc_workfd_iperf_minval
```

used Testcases:

src/tc/linux/tc_workfd_iperf.py.

used config variables:

tb_tc_workfd_c_sr. *tb_tc_workfd_iperf_sip*. *tb_tc_workfd_c_sl*. *tb_tc_workfd_iperf_minval*.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_workfd_iperf.py

src/tc/linux/tc_workfd_md5sum.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_workfd_md5sum.py
# calculate md5sum of file tb.tc_workfd_md5sum_name , and store it in
# tb.tc_workfd_md5sum_sum
```

used Testcases:

src/tc/linux/tc_workfd_md5sum.py.

used config variables:

tb_tc_workfd_md5sum_name. *tb_tc_workfd_md5sum_sum*.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_workfd_md5sum.py

src/tc/linux/tc_workfd_rm_file.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_workfd_rm_file.py
# simple rm directory tb.config.tc_workfd_rm_file_name on the lab
```

used Testcases:

src/tc/linux/tc_workfd_rm_file.py.

used config variables:

tc_workfd_rm_file_name.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_workfd_rm_file.py

src/tc/linux/tc_workfd_rm_linux_code.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_workfd_rm_linux_code.py
# rm linux source tb.config.tc_lab_source_dir + '/linux-' + tb.config.boardlabname
```

used Testcases:

src/tc/linux/tc_workfd_rm_linux_code.py.

used config variables:

tc_lab_source_dir. *boardlabname*.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_workfd_rm_linux_code.py

src/tc/linux/tc_workfd_rm_uboot_code.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_workfd_rm_uboot_code.py
# rm U-Boot source tb.config.tc_lab_source_dir + '/u-boot-' + tb.config.boardlabname
```

used Testcases:

src/tc/linux/tc_workfd_rm_uboot_code.py.

used config variables:

tc_lab_source_dir. *boardlabname*.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_workfd_rm_uboot_code.py

src/tc/linux/tc_workfd_ssh.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_workfd_ssh.py
# login with ssh to tb.workfd_ssh_cmd and set new ssh prompt
# tb.config.workfd_ssh_cmd_prompt
```

used Testcases:

src/tc/linux/tc_workfd_ssh.py.

used config variables:

tb_workfd_ssh_cmd. *workfd_ssh_cmd_prompt*.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_workfd_ssh.py

src/tc/linux/tc_workfd_sudo_cp_file.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_workfd_sudo_cp_file.py
# simple copy file from tb.tc_workfd_cp_file_a to tb.tc_workfd_cp_file_b
# with sudo rights
```

used Testcases:

src/tc/linux/tc_workfd_sudo_cp_file.py.

used config variables:

tb_tc_workfd_cp_file_a. *tb_tc_workfd_cp_file_b*.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_workfd_sudo_cp_file.py

src/tc/linux/tc_workfd_switch_su.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_workfd_switch_su.py
# switch to superuser
```

used Testcases:

src/tc/linux/tc_workfd_switch_su.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/linux/tc_workfd_switch_su.py

src/tc/uboot/duts/tc_ub_basic.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_ub_basic.py
# convert duts tests from:
# http://git.denx.de/?p=duts.git;a=blob;f=testsystems/dulg/testcases/02_UBootBasic.tc;h=5503
```

used Testcases:

src/tc/uboot/duts/tc_ub_basic.py.

links:

http://git.denx.de/?p=duts.git;a=blob;f=testsystems/dulg/testcases/02_UBootBasic.tc;h=5503cc6c716d2748732d30d63b0801e651fe1706;hb=101ddd5dbd547d5046363358d560149d873b238a

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/duts/tc_ub_basic.py

src/tc/uboot/duts/tc_ub_bdinfo.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_ub_bdinfo.py
# convert duts tests from:
# http://git.denx.de/?p=duts.git;a=blob;f=testsystems/dulg/testcases/10_UBootBdinfo.tc;h=aa7
```

used Testcases:

src/tc/uboot/duts/tc_ub_bdinfo.py.

links:

http://git.denx.de/?p=duts.git;a=blob;f=testsystems/dulg/testcases/10_UBootBdinfo.tc;h=aa794a93ac5c8d2c3aea4aa5d02433ca2ee0f010;hb=101ddd5dbd547d5046363358d560149d873b238a

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/duts/tc_ub_bdinfo.py

src/tc/uboot/duts/tc_ub_boot.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_ub_boot.py
# convert duts tests from:
# http://git.denx.de/?p=duts.git;a=blob;f=testsystems/dulg/testcases/10_UBootBoot.tc;h=f679f
```

used Testcases:

src/tc/uboot/duts/tc_ub_boot.py.

links:

http://git.denx.de/?p=duts.git;a=blob;f=testsystems/dulg/testcases/10_UBootBoot.tc;h=f679ff09cdb1e1393829c32dc5aa5cf299e9af07;hb=101ddd5dbd547d5046363358d560149d873b238a

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/duts/tc_ub_boot.py

src/tc/uboot/duts/tc_ub_coninfo.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_ub_coninfo.py
# convert duts tests from:
# http://git.denx.de/?p=duts.git;a=blob;f=testsystems/dulg/testcases/10_UBootConinfo.tc;h=2d
```

used Testcases:

src/tc/uboot/duts/tc_ub_coninfo.py.

links:

http://git.denx.de/?p=duts.git;a=blob;f=testsystems/dulg/testcases/10_UBootConinfo.tc;h=2d028f74ba791343b8a70a97885eabe8b5944017;hb=101ddd5dbd547d5046363358d560149d873b238a

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/duts/tc_ub_coninfo.py

src/tc/uboot/duts/tc_ub_date.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_ub_date.py
# convert duts tests from:
# http://git.denx.de/?p=duts.git;a=blob;f=testsystems/dulg/testcases/15_UBootDate.tc;h=03b7d
```

used Testcases:

src/tc/uboot/duts/tc_ub_date.py.

links:

http://git.denx.de/?p=duts.git;a=blob;f=testsystems/dulg/testcases/15_UBootDate.tc;h=03b7d53fd93bd61381db4095a4bff58b1d1efff7;hb=101ddd5dbd547d5046363358d560149d873b238a

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/duts/tc_ub_date.py

src/tc/uboot/duts/tc_ub_diskboothelp.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_ub_diskboothelp.py
# convert duts tests from:
# http://git.denx.de/?p=duts.git;a=blob;f=testsystems/dulg/testcases/15_UBootIde.tc;h=03c2a0
```

used Testcases:

src/tc/uboot/duts/tc_ub_diskboothelp.py.

links:

http://git.denx.de/?p=duts.git;a=blob;f=testsystems/dulg/testcases/15_UBootIde.tc;h=03c2a05b75c6f9f6fc257fa84a2220f965567699;hb=101ddd5dbd547d5046363358d560149d873b238a

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/duts/tc_ub_diskboothelp.py

src/tc/uboot/duts/tc_ub_download.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_ub_download.py
# convert duts tests from:
# http://git.denx.de/?p=duts.git;a=blob;f=testsystems/dulg/testcases/15_UBootCmdGroupDownloa
```

used Testcases:

src/tc/uboot/duts/tc_ub_download.py.

links:

http://git.denx.de/?p=duts.git;a=blob;f=testsystems/dulg/testcases/15_UBootCmdGroupDownload.tc;h=8e58d53add90b680ef7a1300894d2392f90d9824;hb=101ddd5dbd547d5046363358d560149d873b238a

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/duts/tc_ub_download.py

src/tc/uboot/duts/tc_ub_dtt.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_ub_dtt.py
# convert duts tests from:
# http://git.denx.de/?p=duts.git;a=blob;f=testsystems/dulg/testcases/15_UBootDtt.tc;h=e420c7
```

used Testcases:

src/tc/uboot/duts/tc_ub_dtt.py.

links:

http://git.denx.de/?p=duts.git;a=blob;f=testsystems/dulg/testcases/15_UBootDtt.tc;h=e420c7b45cd73b00465d69f969039222868f4cc7;hb=101ddd5dbd547d5046363358d560149d873b238a

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/duts/tc_ub_dtt.py

src/tc/uboot/duts/tc_ub_environment.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_ub_environment.py
# convert duts tests from:
# http://git.denx.de/?p=duts.git;a=blob;f=testsystems/dulg/testcases/10_UBootEnvironment.tc;
```

used Testcases:

src/tc/uboot/duts/tc_ub_environment.py.

links:

http://git.denx.de/?p=duts.git;a=blob;f=testsystems/dulg/testcases/10_UBootEnvironment.tc;h=18d235f427e3efe9e6a04f870a3c5426d719ec58;hb=101ddd5dbd547d5046363358d560149d873b238a

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/duts/tc_ub_environment.py

src/tc/uboot/duts/tc_ub_flash.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_ub_flash.py
# convert duts tests from:
# http://git.denx.de/?p=duts.git;a=blob;f=testsystems/dulg/testcases/15_UBootFlashTest.tc;h=
```

used Testcases:

src/tc/uboot/duts/tc_ub_flash.py.

links:

http://git.denx.de/?p=duts.git;a=blob;f=testsystems/dulg/testcases/15_UBootFlashTest.tc;h=6eea72c8e9f3f4739a76ff59bb2e9a7c693aedd9;hb=101ddd5dbd547d5046363358d560149d873b238a

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/duts/tc_ub_flash.py

src/tc/uboot/duts/tc_ub_flinfo.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_ub_flinfo.py
# convert duts tests from:
# http://git.denx.de/?p=duts.git;a=blob;f=testsystems/dulg/testcases/10_UBootFlinfo.tc;h=f5b
```

used Testcases:

src/tc/uboot/duts/tc_ub_flinfo.py.

links:

http://git.denx.de/?p=duts.git;a=blob;f=testsystems/dulg/testcases/10_UBootFlinfo.tc;h=f5b728258250211d86dc9c6a9208639d8542b845;hb=101ddd5dbd547d5046363358d560149d873b238a

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/duts/tc_ub_flinfo.py

src/tc/uboot/duts/tc_ub_i2c_help.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_ub_i2c_help.py
# simple prints "help i2c" cmd
```

used Testcases:

src/tc/uboot/duts/tc_ub_i2c_help.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/duts/tc_ub_i2c_help.py

src/tc/uboot/duts/tc_ub_ide.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_ub_ide.py
# convert duts tests from:
# http://git.denx.de/?p=duts.git;a=blob;f=testsystems/dulg/testcases/15_UBootIde.tc;h=03c2a0
```

used Testcases:

src/tc/uboot/duts/tc_ub_ide.py.

links:

http://git.denx.de/?p=duts.git;a=blob;f=testsystems/dulg/testcases/15_UBootIde.tc;h=03c2a05b75c6f9f6fc257fa84a2220f965567699;hb=101ddd5dbd547d5046363358d560149d873b238a

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/duts/tc_ub_ide.py

src/tc/uboot/duts/tc_ub_memory.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_ub_memory.py
# convert duts tests from:
# http://git.denx.de/?p=duts.git;a=blob;f=testsystems/dulg/testcases/10_UBootMemory.tc;h=f5f
```

used Testcases:

src/tc/uboot/duts/tc_ub_memory.py.

links:

http://git.denx.de/?p=duts.git;a=blob;f=testsystems/dulg/testcases/10_UBootMemory.tc;h=f5fb055499db17c322859215ab489cefb063ac47;hb=101ddd5dbd547d5046363358d560149d873b238a

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/duts/tc_ub_memory.py

src/tc/uboot/duts/tc_ub_run.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_ub_run.py
# convert duts tests from:
# http://git.denx.de/?p=duts.git;a=blob;f=testsystems/dulg/testcases/10_UBootRun.tc;h=44f8a0
```

used Testcases:

src/tc/uboot/duts/tc_ub_run.py.

links:

http://git.denx.de/?p=duts.git;a=blob;f=testsystems/dulg/testcases/10_UBootRun.tc;h=44f8a0a0de256afdd95b5ec20d1d4570373aeb7d;hb=101ddd5dbd547d5046363358d560149d873b238a

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/duts/tc_ub_run.py

src/tc/uboot/duts/tc_ub_start_all_duts.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_ub_start_all_duts.py
# start all DUTS tests
```

used Testcases:

src/tc/uboot/duts/tc_ub_start_all_duts.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/duts/tc_ub_start_all_duts.py

src/tc/uboot/tc_ub_aristainetos2_ubi.py

```
# start with
# tbot.py -s lab_denx -c aristainetos2 -t tc_ub_aristainetos2_ubi.py
# ubi testcases for the aristainetos2 board
```

used Testcases:

src/tc/uboot/tc_ub_aristainetos2_ubi.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/tc_ub_aristainetos2_ubi.py

src/tc/uboot/tc_ub_check_reg_file.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_ub_check_reg_file.py
# checks if the default values in reg file tb.tc_ub_create_reg_file_name
# on the tbot host in tb.workdir have the same values, as the
# registers on the board
# format of the regfile:
# regaddr mask type defval
# ToDo: use the file from the lab host, not the tbot host
```

used Testcases:

src/tc/uboot/tc_ub_check_reg_file.py.

used config variables:

tb_tc_ub_create_reg_file_name. *tb_workdir*.

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/tc_ub_check_reg_file.py

src/tc/uboot/tc_ub_check_version.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_ub_check_version.py
# check if the current running U-Boot vers == tb.uboot_vers
# and SPL vers == tb.spl_vers
```

used Testcases:

src/tc/uboot/tc_ub_check_version.py.

used config variables:

tb_uboot_vers. *tb_spl_vers*.

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/tc_ub_check_version.py

src/tc/uboot/tc_ub_cmp.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_ub_cmp.py
# - compare 2 the contents of tb.tc_ub_cmp_addr1 with tb.tc_ub_cmp_addr2
# bytes tb.tc_ub_cmp_len length
```

used Testcases:

src/tc/uboot/tc_ub_cmp.py.

used config variables:

tb_tc_ub_cmp_addr1. *tb_tc_ub_cmp_addr2*. *tb_tc_ub_cmp_len*.

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/tc_ub_cmp.py

src/tc/uboot/tc_ub_create_reg_file.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_ub_create_reg_file.py
# creates a reg file tb.tc_ub_create_reg_file_name on the tbot host
# in tb.workdir
# read from tb.tc_ub_create_reg_file_start to tb.tc_ub_create_reg_file_stop
# and writes the results in the regfile tb.tc_ub_create_reg_file_name
# format of the regfile:
# regaddr mask type defval
# This reg file can be used as a default file, how the
# registers must be setup, check it with testcase
# tc_ub_check_reg_file.py
# ToDo: use the file from the lab host, not the tbot host
```

used Testcases:

src/tc/uboot/tc_ub_create_reg_file.py. *src/tc/uboot/tc_ub_check_reg_file.py*.

used config variables:

tb_tc_ub_create_reg_file_name. *tb_workdir*. *tb_tc_ub_create_reg_file_start*. *tb_tc_ub_create_reg_file_stop*. *tb_tc_ub_create_reg_file_name*.

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/tc_ub_create_reg_file.py

src/tc/uboot/tc_ub_dfu.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_ub_dfu.py
# test dfu
# - use dfu-util in tb.config.tc_ub_dfu_dfu_util_path
# - upload file tb.config.tc_ub_dfu_dfu_util_alt_setting to
#   tb.config.tc_ub_dfu_dfu_util_downloadfile
# - download it back to the board
# - upload it again
# - and compare the two files
```

used Testcases:

src/tc/uboot/tc_ub_dfu.py.

used config variables:

tc_ub_dfu_dfu_util_path. *tc_ub_dfu_dfu_util_alt_setting*. *tb_config_tc_ub_dfu_dfu_util_downloadfile*.

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/tc_ub_dfu.py

src/tc/uboot/tc_ub_dfu_random.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_ub_dfu_random.py
# test a U-Boot dfu alt setting tb.config.tc_ub_dfu_dfu_util_alt_setting
# Therefore write a random file with size tb.config.tc_ub_dfu_rand_size
# to it, reread it and compare it. TC fails if files differ
# (If readen file is longer, this is no error!)
```

```
#
# If dfu-util is not installed on the lab PC, set
# tb.config.tc_ub_dfu_dfu_util_ssh for connecting with ssh to a PC
# which have it installed, and a USB cable connected to the board.
# Set tb.config.tc_ub_dfu_dfu_util_path to the path of dfu-util, if
# you have a self compiled version of it.
# Set tb.config.tc_ub_dfu_rand_ubcmd for the executed command on
# U-Boot shell for getting into DFU mode
```

used Testcases:

src/tc/uboot/tc_ub_dfu_random.py.

used config variables:

tc_ub_dfu_dfu_util_alt_setting. *tb_config_tc_ub_dfu_rand_size*. *tb_config_tc_ub_dfu_dfu_util_ssh*. *tc_ub_dfu_dfu_util_path*. *tb_config_tc_ub_dfu_rand_ubcmd*.

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/tc_ub_dfu_random.py

src/tc/uboot/tc_ub_dfu_random_default.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_ub_dfu_random_default.py
# test a U-Boot dfu alt setting tb.config.tc_ub_dfu_dfu_util_alt_setting
# with reading / writing different sizes
```

used Testcases:

src/tc/uboot/tc_ub_dfu_random_default.py.

used config variables:

tc_ub_dfu_dfu_util_alt_setting.

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/tc_ub_dfu_random_default.py

src/tc/uboot/tc_ub_get_filesize.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_ub_get_filesize.py
# simple get the content of U-Boot env variable filesize
# and store it in tb.ub_filesize
```

used Testcases:

src/tc/uboot/tc_ub_get_filesize.py.

used config variables:

tb_ub_filesize.

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/tc_ub_get_filesize.py

src/tc/uboot/tc_ub_get_version.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_ub_get_version.py
# get the U-Boot and/or SPL version from a binary
# and save it in tb.uboot_vers or tb.spl_vers
```

used Testcases:

src/tc/uboot/tc_ub_get_version.py.

used config variables:

tb_uboot_vers. *tb_spl_vers*.

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/tc_ub_get_version.py

src/tc/uboot/tc_ub_help.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_ub_help.py
# - test U-Boots help cmd
#   may we add a list as parameter, so we can adapt it board
#   specific.
```

used Testcases:

src/tc/uboot/tc_ub_help.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/tc_ub_help.py

src/tc/uboot/tc_ub_load_board_env.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_ub_load_board_env.py
# load U-Boot Environment for the board tb.config.tftpboardname
# tb.config.ub_load_board_env_addr and tb.config.ub_load_board_env_subdir
```

used Testcases:

src/tc/uboot/tc_ub_load_board_env.py.

used config variables:

tftpboardname. *ub_load_board_env_addr*. *ub_load_board_env_subdir*.

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/tc_ub_load_board_env.py

src/tc/uboot/tc_ub_reset.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_ub_reset.py
# simple U-Boot reset command.
```

used Testcases:

src/tc/uboot/tc_ub_reset.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/tc_ub_reset.py

src/tc/uboot/tc_ub_setenv.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_ub_setenv.py
# set U-Boot Environmentvariable tb.config.setenv_name with value
# tb.config.setenv_value
```

used Testcases:

src/tc/uboot/tc_ub_setenv.py.

used config variables:

setenv_name. *setenv_value*.

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/tc_ub_setenv.py

src/tc/uboot/tc_ub_test_py.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_ub_test_py.py
# call test/py from u-boot source
# - disconnect console
# - call test/py
# - connect back to console
# test/py hookscript directory:
# tb.config.tc_ub_test_py_hook_script_path
```

used Testcases:

src/tc/uboot/tc_ub_test_py.py.

used config variables:

tc_ub_test_py_hook_script_path.

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/tc_ub_test_py.py

src/tc/uboot/tc_ub_tftp_file.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_ub_tftp_file.py
# load file tb.config.tc_ub_tftp_file_name to tb.config.tc_ub_tftp_file_addr
# with tftp command in uboot
```

used Testcases:

src/tc/uboot/tc_ub_tftp_file.py.

used config variables:

tb_config_tc_ub_tftp_file_name. *tc_ub_tftp_file_addr*.

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/tc_ub_tftp_file.py

src/tc/uboot/tc_ub_ubi_check_volume.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_ub_ubi_check_volume.py
# - checks if ubi volume tb.config.tc_ub_ubi_load_name exists
```

used Testcases:

src/tc/uboot/tc_ub_ubi_check_volume.py.

used config variables:

tc_ub_ubi_load_name.

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/tc_ub_ubi_check_volume.py

src/tc/uboot/tc_ub_ubi_create_volume.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_ub_ubi_create_volume.py
# - create ubi volume tb.config.tc_ub_ubi_create_vol_name with size
# tb.config.tc_ub_ubi_create_vol_sz
```

used Testcases:

src/tc/uboot/tc_ub_ubi_create_volume.py.

used config variables:

tc_ub_ubi_create_vol_name. *tc_ub_ubi_create_vol_sz*.

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/tc_ub_ubi_create_volume.py

src/tc/uboot/tc_ub_ubi_erase.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_ub_ubi_erase.py
# - erase an ubi device
#   execute U-Boot Env tbot_ubi_erase
```

used Testcases:

src/tc/uboot/tc_ub_ubi_erase.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/tc_ub_ubi_erase.py

src/tc/uboot/tc_ub_ubi_info.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_ub_ubi_info.py
# - simple print ubi info
```

used Testcases:

src/tc/uboot/tc_ub_ubi_info.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/tc_ub_ubi_info.py

src/tc/uboot/tc_ub_ubi_prepare.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_ub_ubi_prepare.py
# - ubi prepare
#   execute "ubi part" ith tb.config.tc_ub_ubi_prep_partname
#   if tb.config.tc_ub_ubi_prep_offset != 'none'
#   with offset tb.config.tc_ub_ubi_prep_offset
```

used Testcases:

src/tc/uboot/tc_ub_ubi_prepare.py.

used config variables:

tc_ub_ubi_prep_partname. *tc_ub_ubi_prep_offset*. *tc_ub_ubi_prep_offset*.

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/tc_ub_ubi_prepare.py

src/tc/uboot/tc_ub_ubi_read.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_ub_ubi_read.py
# - read ubi volume tb.config.tc_ub_ubi_prep_offset to tb.tc_ub_ubi_read_addr
# with len tb.tc_ub_ubi_read_len
```

used Testcases:

src/tc/uboot/tc_ub_ubi_read.py.

used config variables:

tc_ub_ubi_prep_offset. *tb_tc_ub_ubi_read_addr*. *tb_tc_ub_ubi_read_len*.

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/tc_ub_ubi_read.py

src/tc/uboot/tc_ub_ubi_write.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_ub_ubi_write.py
# - write image @ tb.config.tc_ub_ubi_write_addr to ubi volume
#   tb.config.tc_ub_ubi_write_vol_name with len tb.config.tc_ub_ubi_write_len
```

used Testcases:

src/tc/uboot/tc_ub_ubi_write.py.

used config variables:

tc_ub_ubi_write_addr. *tc_ub_ubi_write_vol_name*. *tc_ub_ubi_write_len*.

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/tc_ub_ubi_write.py

src/tc/uboot/tc_ub_ubifs_ls.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_ub_ubifs_ls.py
# - ls ubifs tb.config.tc_ub_ubifs_ls_dir
```

used Testcases:

src/tc/uboot/tc_ub_ubifs_ls.py.

used config variables:

tc_ub_ubifs_ls_dir.

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/tc_ub_ubifs_ls.py

src/tc/uboot/tc_ub_ubifs_mount.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_ub_ubifs_mount.py
# - mount ubifs tb.config.tc_ub_ubifs_volume_name
```

used Testcases:

src/tc/uboot/tc_ub_ubifs_mount.py.

used config variables:

tc_ub_ubifs_volume_name.

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/tc_ub_ubifs_mount.py

src/tc/uboot/tc_ub_upd_spl.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_ub_upd_spl.py
# update new spl to board
# steps:
# - load tbot u-boot env vars
# - execute "run tbot_upd_spl"
# - execute "run tbot_cmp_spl"
# - reset board
# - get u-boot
```

used Testcases:

src/tc/uboot/tc_ub_upd_spl.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/tc_ub_upd_spl.py

src/tc/uboot/tc_ub_upd_uboot.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_ub_upd_uboot.py
# update new uboot to board
# steps:
# - load tbot u-boot env vars
# - execute "run tbot_upd_uboot"
# - execute "run tbot_cmp_uboot"
# - reset board
# - get u-boot
```

used Testcases:

src/tc/uboot/tc_ub_upd_uboot.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/tc_ub_upd_uboot.py

src/tc/uboot/tc_uboot_check_kconfig.py

```
# start with
# python2.7 src/common/tbot.py -c config/tbot_uboot_kconfig_check.cfg -t tc_uboot_check_kcon
# check for all boards, if a patch changes the u-boot binary
# If U-boot binary changed by the patch this testcase fails.
# use this testcase, if you for example move a config option
# into Kconfig. As we need reproducable builds, we need to
# patch U-Boot with tb.tc_uboot_check_kconfig_preparepatch
# - rm U-Boot code with tc_workfd_rm_uboot_code.py
# - get U-Boot code with tc_lab_get_uboot_source.py
# - set SOURCE_DATE_EPOCH=0 to get reproducible builds
# - get rid of local version ToDo: find a way to disable CONFIG_LOCALVERSION_AUTO
# - if tb.tc_uboot_check_kconfig_read_sumfile is != 'none'
#     read a list of boards and md5sums from the file in
#     tb.tc_uboot_check_kconfig_read_sumfile
#   else
#   - create a list of boards (all defconfigs)
#   - do for all boards:
#     - get architecture and set toolchain
#     - compile U-Boot and calculate md5sum
#       with tc_workfd_compile_uboot.py and tc_workfd_md5sum.py
#     - if tb.tc_uboot_check_kconfig_create_sumfile != 'none'
#       save the board md5sum lists in the file
#       tb.tc_uboot_check_kconfig_create_sumfile
#       you can use this now as a reference, so no need
#       to calculate always for all boards the md5sums
#       -> saves a lot of time!
#
# - apply patch(es) with tc_workfd_apply_patches.py
# - do for all boards:
#   - compile U-Boot again (patched version)
#   - calculate md5sum again
#   - calculate md5sums
#   - check if they are the same
```

used Testcases:

src/tc/uboot/tc_uboot_check_kconfig.py. src/tc/linux/tc_workfd_rm_uboot_code.py. src/tc/tc_lab_get_uboot_source.py. src/tc/tc_workfd_compile_uboot.py. src/tc/linux/tc_workfd_md5sum.py. src/tc/tc_workfd_apply_patches.py.

used config variables:

tb_tc_uboot_check_kconfig_preparepatch. *tb_tc_uboot_check_kconfig_read_sumfile*. *tb_tc_uboot_check_kconfig_read_sumfile*. *tb_tc_uboot_check_kconfig_create_sumfile*. *tb_tc_uboot_check_kconfig_create_sumfile*.

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/tc_uboot_check_kconfig.py

src/tc/uboot/tc_uboot_get_arch.py

```
# start with
# python2.7 src/common/tbot.py -c config/tbot_dxr2_uboot_kconfig_check.cfg -t tc_uboot_get_a
# get architecture from u-boot config
```

used Testcases:

src/tc/uboot/tc_uboot_get_arch.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/uboot/tc_uboot_get_arch.py

src/tc/tc_board_git_bisect.py

```
# start with
# python2.7 src/common/tbot.py -c tbot_tqm5200s.cfg -t tc_board_git_bisect.py
# get a source code with tc tb.config.board_git_bisect_get_source_tc
# and start a "git bisect" session
# current commit is bad
# good commit id is defined through tb.config.board_git_bisect_good_commit
# tc for testing good or bad is tb.config.board_git_bisect_call_tc
# if you have some local patches, which needs to be applied
# each git bisect step, set tb.config.board_git_bisect_patches
```

used Testcases:

src/tc/tc_board_git_bisect.py.

used config variables:

board_git_bisect_get_source_tc. *board_git_bisect_good_commit*. *board_git_bisect_call_tc*. *board_git_bisect_patches*.

https://github.com/hsdenx/tbot/tree/master/src/tc/tc_board_git_bisect.py

src/tc/tc_lab_apply_patches.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_lab_apply_patches.py
# apply patches to source
```

used Testcases:

src/tc/tc_lab_apply_patches.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/tc_lab_apply_patches.py

src/tc/tc_lab_compile_uboot.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_lab_compile_uboot.py
# compile u-boot
```

used Testcases:

src/tc/tc_lab_compile_uboot.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/tc_lab_compile_uboot.py

src/tc/tc_lab_cp_file.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_lab_cp_file.py
# simple copy  file from arg.get('s')
# to arg.get('t') on the channel arg.get('ch')
```

used Testcases:

src/tc/tc_lab_cp_file.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/tc_lab_cp_file.py

src/tc/tc_lab_get_uboot_source.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_lab_get_uboot_source.py
# get U-Boot source
# and go into the source tree
```

used Testcases:

src/tc/tc_lab_get_uboot_source.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/tc_lab_get_uboot_source.py

src/tc/tc_lab_poweroff.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_lab_poweroff.py
# simple power off the board
```

used Testcases:

src/tc/tc_lab_poweroff.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/tc_lab_poweroff.py

src/tc/tc_lab_rm_dir.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_lab_rm_dir.py
# simple rm a directory on the lab
```

used Testcases:

src/tc/tc_lab_rm_dir.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/tc_lab_rm_dir.py

src/tc/tc_lab_set_toolchain.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_lab_set_toolchain.py
# set the toolchain
```

used Testcases:

src/tc/tc_lab_set_toolchain.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/tc_lab_set_toolchain.py

src/tc/tc_ub_boot_linux.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_ub_boot_linux.py
# - load u-boot environment with testcase "tc_ub_load_board_env.py"
# - execute u-boot cmd tb.config.ub_boot_linux_cmd
```

used Testcases:

src/tc/tc_ub_boot_linux.py.

used config variables:

ub_boot_linux_cmd.

https://github.com/hsdenx/tbot/tree/master/src/tc/tc_ub_boot_linux.py

src/tc/tc_workfd_apply_patches.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_lab_apply_patches.py
# apply patches to source
```

used Testcases:

src/tc/tc_lab_apply_patches.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/tc_workfd_apply_patches.py

src/tc/tc_workfd_compile_uboot.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_workfd_compile_uboot.py
# compile u-boot
```

used Testcases:

src/tc/tc_workfd_compile_uboot.py.

https://github.com/hsdenx/tbot/tree/master/src/tc/tc_workfd_compile_uboot.py

src/tc/tc_workfd_set_toolchain.py

```
# start with
# python2.7 src/common/tbot.py -c tbot.cfg -t tc_workfd_set_toolchain.py
# set the toolchain, dependend on the architecture setting in
# tb.tc_workfd_set_toolchain_arch
# supported toolchains defined in
# tb.tc_workfd_set_toolchain_t_p and tb.tc_workfd_set_toolchain_cr_co
```

used Testcases:

src/tc/tc_workfd_set_toolchain.py.

used config variables:

tb_tc_workfd_set_toolchain_arch. *tb_tc_workfd_set_toolchain_t_p*. *tb_tc_workfd_set_toolchain_cr_co*.

https://github.com/hsdenx/tbot/tree/master/src/tc/tc_workfd_set_toolchain.py

Documentation of all Variables

debug

```
debug = False
```

debugstatus

```
debugstatus = False
```

uboot_strings

```
uboot_strings = ['Autobooting in', 'noautoboot',  'autoboot', 'EOF', 'RomBOOT']
```

term_line_length

```
term_line_length = '200'
```

wdt_timeout

```
wdt_timeout = '120' # wdt timeout after 2 minutes
```

state_linux_timeout

```
state_linux_timeout = 4
```

labsshprompt

```
labsshprompt = '$ '
```

tc_return

```
tc_return = True
```

tc_workfd_check_if_cmd_exist_cmdname

```
tc_workfd_check_if_cmd_exist_cmdname = 'none'
```

setenv_name

```
setenv_name = 'tralala'
```

setenv_value

```
setenv_value = 'hulalahups'
```

tc_ub_boot_linux_load_env

```
tc_ub_boot_linux_load_env = 1
```

tc_lx_mount_dev

```
tc_lx_mount_dev = '/dev/sda1'
```

tc_lx_mount_fs_type

```
tc_lx_mount_fs_type = 'ext4'
```

tc_lx_mount_dir

```
tc_lx_mount_dir = '/home/hs/mnt'
```

tc_lx_bonnie_dev

```
tc_lx_bonnie_dev = tc_lx_mount_dev
```

tc_lx_bonnie_sz

```
tc_lx_bonnie_sz = '968'
```

ub_load_board_env_addr

```
ub_load_board_env_addr = '0x81000000'
```

ub_load_board_env_subdir

```
ub_load_board_env_subdir = 'tbot'
```

ub_boot_linux_cmd

```
ub_boot_linux_cmd = 'run tbot_boot_linux'
```

tc_lab_compile_uboot_boardname

```
tc_lab_compile_uboot_boardname = 'config.boardname'
```

tc_lab_compile_uboot_makeoptions

```
tc_lab_compile_uboot_makeoptions = '-j4'
```

do_connect_to_board

```
do_connect_to_board = True
```

tc_lab_compile_uboot_export_path

```
tc_lab_compile_uboot_export_path = 'none'
```

tftpboardname

```
tftpboardname = 'config.boardname'
```

boardlabname

```
boardlabname = 'config.boardname'
```

boardlabpowername

```
boardlabpowername = 'config.boardname'
```

tc_ub_dfu_dfu_util_path

```
tc_ub_dfu_dfu_util_path = "/home/hs/zug/dfu-util"
```

tc_ub_dfu_dfu_util_alt_setting

```
tc_ub_dfu_dfu_util_alt_setting = "Linux"
```

tc_lab_source_dir

```
tc_lab_source_dir = "/work/hs/tbot"
```

tc_lab_get_uboot_source_git_repo

```
tc_lab_get_uboot_source_git_repo = "/home/git/u-boot.git"
```

tc_lab_get_uboot_source_git_branch

```
tc_lab_get_uboot_source_git_branch = "master"
```

tc_lab_toolchain_rev

```
tc_lab_toolchain_rev = "5.4"
```

tc_lab_toolchain_name

```
tc_lab_toolchain_name = "armv5te"
```

tc_ub_ubi_load_name

```
tc_ub_ubi_load_name = "kernel"
```

tc_ub_ubi_prep_partname

```
tc_ub_ubi_prep_partname = "ubi"
```

tc_ub_ubi_prep_offset

```
tc_ub_ubi_prep_offset = "none"
```

tc_ub_ubi_load_addr

```
tc_ub_ubi_load_addr = "14000000"
```

tc_ub_ubi_create_vol_name

```
tc_ub_ubi_create_vol_name = 'config.tc_ub_ubi_load_name'
```

tc_ub_ubi_create_vol_sz

```
tc_ub_ubi_create_vol_sz = "600000"
```

tc_ub_ubi_write_len

```
tc_ub_ubi_write_len = '0xc00000'
```

tc_ub_ubi_write_addr

```
tc_ub_ubi_write_addr = 'config.tc_ub_ubi_load_addr'
```

tc_ub_ubi_write_vol_name

```
tc_ub_ubi_write_vol_name = 'config.tc_ub_ubi_create_vol_name'
```

tc_ub_ubifs_volume_name

```
tc_ub_ubifs_volume_name = 'ubi:rootfs'
```

tc_ub_ubifs_ls_dir

```
tc_ub_ubifs_ls_dir = '/'
```

tc_lx_gpio_nr

```
tc_lx_gpio_nr = '69'
```

tc_lx_gpio_dir

```
tc_lx_gpio_dir = 'out'
```

tc_lx_gpio_val

```
tc_lx_gpio_val = '1'
```

tc_lx_eeprom_file

```
tc_lx_eeprom_file = '/sys/class/i2c-dev/i2c-0/device/0-0050/eeprom'
```

tc_lx_eeprom_tmp_dir

```
tc_lx_eeprom_tmp_dir = 'config.lab_tmp_dir'
```

tc_lx_eeprom_wp_gpio

```
tc_lx_eeprom_wp_gpio = 'none'
```

tc_lx_eeprom_wp_val

```
tc_lx_eeprom_wp_val = "0"
```

tc_lx_eeprom_wp_sz

```
tc_lx_eeprom_wp_sz = "4096"
```

tc_lx_eeprom_wp_obs

```
tc_lx_eeprom_wp_obs = "32"
```

tc_lx_eeprom_wp_wc

```
tc_lx_eeprom_wp_wc = "128"
```

tc_lx_cpufreq_frequences

```
tc_lx_cpufreq_frequences = ['294']
```

tc_lx_check_usb_authorized

```
tc_lx_check_usb_authorized = 'usb 1-1'
```

tc_workfd_work_dir

```
tc_workfd_work_dir = "/work/tbot"
```

tc_workfd_check_if_file_exists_name

```
tc_workfd_check_if_file_exists_name = "bonnie++-1.03e.tgz"
```

tc_workfd_check_if_dir_exists_name

```
tc_workfd_check_if_dir_exists_name = "mtd-utils"
```

tc_lx_dmesg_grep_name

```
tc_lx_dmesg_grep_name = "zigbee"
```

tc_lx_readreg_mask

```
tc_lx_readreg_mask = "0x000000ff"
```

tc_lx_readreg_type

```
tc_lx_readreg_type = "w"
```

tc_lx_create_reg_file_name

```
tc_lx_create_reg_file_name = "pinmux.reg"
```

tc_lx_create_reg_file_start

```
tc_lx_create_reg_file_start = "0x44e10800"
```

tc_lx_create_reg_file_stop

```
tc_lx_create_reg_file_stop = "0x44e10a34"
```

tc_lx_regulator_nrs

```
tc_lx_regulator_nrs = ['0 regulator-dummy -', '1 hsusb1_vbus 5000000',
'2 vmmc 3300000', '3 pbias_mmc_omap2430 3000000',
'4 DCDC1 1200000', '5 DCDC2 3300000', '6 DCDC3 1800000',
'7 LDO1 1800000', '8 LDO2 3300000']
```

board_has_debugger

```
board_has_debugger = 0
```

lab_bdi_upd_uboot_bdi_cmd

```
lab_bdi_upd_uboot_bdi_cmd = 'telnet bdi6'
```

lab_bdi_upd_uboot_bdi_prompt

```
lab_bdi_upd_uboot_bdi_prompt = 'BDI>'
```

lab_bdi_upd_uboot_bdi_era

```
lab_bdi_upd_uboot_bdi_era = 'era'
```

lab_bdi_upd_uboot_bdi_prog

```
lab_bdi_upd_uboot_bdi_prog = 'prog 0xfc000000'
```

lab_bdi_upd_uboot_bdi_file

```
lab_bdi_upd_uboot_bdi_file = '/tftpboot/tqm5200s/tbot/u-boot.bin'
```

lab_bdi_upd_uboot_bdi_run

```
lab_bdi_upd_uboot_bdi_run = 'res run'
```

board_git_bisect_get_source_tc

```
board_git_bisect_get_source_tc = 'tc_lab_get_uboot_source.py'
```

board_git_bisect_call_tc

```
board_git_bisect_call_tc = 'tc_board_tqm5200s_ub_comp_install.py'
```

board_git_bisect_good_commit

```
board_git_bisect_good_commit = 'f9860cf'
```

board_git_bisect_patches

```
board_git_bisect_patches = 'none'
```

tc_lab_apply_patches_dir

```
tc_lab_apply_patches_dir =  'none'
```

tc_ubi_cmd_path

```
tc_ubi_cmd_path = "/work/tbot/mtd-utils"
```

tc_ubi_mtd_dev

```
tc_ubi_mtd_dev = "/dev/mtd4"
```

tc_ubi_ubi_dev

```
tc_ubi_ubi_dev = "/dev/ubi0"
```

tc_ubi_min_io_size

```
tc_ubi_min_io_size = "1024"
```

tc_ubi_max_leb_cnt

```
tc_ubi_max_leb_cnt = "100"
```

tc_ubi_leb_size

```
tc_ubi_leb_size = "126976"
```

tc_ubi_vid_hdr_offset

```
tc_ubi_vid_hdr_offset = "default"
```

tc_lx_ubi_format_filename

```
tc_lx_ubi_format_filename = "/home/hs/ccu1/ecl-image-usbc.ubi"
```

tc_workfd_apply_patchwork_patches_list

```
tc_workfd_apply_patchwork_patches_list = []
```

tc_workfd_apply_patchwork_patches_list_hand

```
tc_workfd_apply_patchwork_patches_list_hand = []
```

tc_workfd_apply_patchwork_patches_blacklist

```
tc_workfd_apply_patchwork_patches_blacklist = []
```

tc_workfd_apply_patchwork_patches_checkpatch_cmd

```
tc_workfd_apply_patchwork_patches_checkpatch_cmd = 'none'
```

tc_workfd_apply_patchwork_patches_eof

```
tc_workfd_apply_patchwork_patches_eof = 'yes'
```

tc_workfd_get_patchwork_number_list_order

```
tc_workfd_get_patchwork_number_list_order = '-delegate'
```

tc_workfd_rm_file_name

```
tc_workfd_rm_file_name = 'none'
```

tc_workfd_cd_name

```
tc_workfd_cd_name = 'none'
```

tc_lab_get_linux_source_git_repo

```
tc_lab_get_linux_source_git_repo = "/home/git/linux.git"
```

tc_lab_get_linux_source_git_repo_user

```
tc_lab_get_linux_source_git_repo_user = 'anonymous'
```

tc_lab_get_linux_source_git_branch

```
tc_lab_get_linux_source_git_branch = "master"
```

tc_lab_get_linux_source_git_reference

```
tc_lab_get_linux_source_git_reference = 'none'
```

tc_workfd_apply_local_patches_dir

```
tc_workfd_apply_local_patches_dir = 'none'
```

tc_workfd_apply_local_patches_checkpatch_cmd

```
tc_workfd_apply_local_patches_checkpatch_cmd = 'none'
```

tc_workfd_apply_local_patches_checkpatch_cmd_strict

```
tc_workfd_apply_local_patches_checkpatch_cmd_strict = "no"
```

tc_workfd_get_list_of_files_mask

```
tc_workfd_get_list_of_files_mask = '*'
```

tc_workfd_compile_linux_boardname

```
tc_workfd_compile_linux_boardname = 'config.boardname'
```

tc_workfd_compile_linux_clean

```
tc_workfd_compile_linux_clean = 'yes'
```

tc_workfd_compile_linux_modules

```
tc_workfd_compile_linux_modules = 'none'
```

tc_workfd_compile_linux_modules_path

```
tc_workfd_compile_linux_modules_path = 'none'
```

tc_workfd_compile_linux_dt_name

```
tc_workfd_compile_linux_dt_name = 'none'
```

tc_workfd_compile_linux_append_dt

```
tc_workfd_compile_linux_append_dt = 'no'
```

tc_workfd_compile_linux_load_addr

```
tc_workfd_compile_linux_load_addr = 'no'
```

tc_workfd_compile_linux_make_target

```
tc_workfd_compile_linux_make_target = 'uImage'
```

tc_workfd_compile_linux_fit_its_file

```
tc_workfd_compile_linux_fit_its_file = 'no'
```

tc_workfd_compile_linux_fit_file

```
tc_workfd_compile_linux_fit_file = 'no'
```

tc_workfd_compile_linux_mkimage

```
tc_workfd_compile_linux_mkimage = '/home/hs/i2c/u-boot/tools/mkimage'
```

tc_workfd_compile_linux_makeoptions

```
tc_workfd_compile_linux_makeoptions = ''
```

workfd_get_patchwork_number_user

```
workfd_get_patchwork_number_user = 'hs'
```

workfd_get_patchwork_number_list_order

```
workfd_get_patchwork_number_list_order = '-delegate'
```

tc_workfd_connect_with_kermit_ssh

```
tc_workfd_connect_with_kermit_ssh = "none"
```

tc_workfd_connect_with_kermit_rlogin

```
tc_workfd_connect_with_kermit_rlogin = "none"
```

kermit_line

```
kermit_line = '/dev/ttyUSB0'
```

kermit_speed

```
kermit_speed = '115200'
```

tc_ub_tftp_file_addr

```
tc_ub_tftp_file_addr = 'config.ub_load_board_env_addr'
```

tc_lab_denx_power_tc

```
tc_lab_denx_power_tc = 'tc_lab_denx_power.py'
```

tc_lab_denx_get_power_state_tc

```
tc_lab_denx_get_power_state_tc = 'tc_lab_denx_get_power_state.py'
```

tc_lab_denx_connect_to_board_tc

```
tc_lab_denx_connect_to_board_tc = 'tc_lab_denx_connect_to_board.py'
```

tc_lab_denx_disconnect_from_board_tc

```
tc_lab_denx_disconnect_from_board_tc = 'tc_lab_denx_disconnect_from_board.py'
```

tc_ub_memory_ram_ws_base

```
tc_ub_memory_ram_ws_base = 'undef'
```

tc_ub_memory_ram_ws_base_alt

```
tc_ub_memory_ram_ws_base_alt = 'undef'
```

tc_ub_memory_ram_big

```
tc_ub_memory_ram_big = 'undef'
```

tc_lx_trigger_wdt_cmd

```
tc_lx_trigger_wdt_cmd = '/home/hs/wdt &'
```

tc_workfd_create_ubi_rootfs_path

```
tc_workfd_create_ubi_rootfs_path = '/opt/eldk-5.4/armv7a-hf/rootfs-minimal-mtdutils'
```

tc_workfd_create_ubi_rootfs_target

```
tc_workfd_create_ubi_rootfs_target = '/tftpboot/dxr2/tbot/rootfs-minimal.ubifs'
```

tc_ub_i2c_help_with_bus

```
tc_ub_i2c_help_with_bus = 'no'
```

dfu_test_sizes_default

```
dfu_test_sizes_default = [
64 - 1,
64,
64 + 1,
128 - 1,
128,
128 + 1,
960 - 1,
960,
960 + 1,
4096 - 1,
4096,
4096 + 1,
1024 * 1024 - 1,
1024 * 1024,
8 * 1024 * 1024,
]
```

workfd_ssh_cmd_prompt

```
workfd_ssh_cmd_prompt = '$'
```

linux_prompt_default

```
linux_prompt_default = 'root@generic-armv7a-hf:~# '
```

labprompt

```
labprompt = 'config.linux_prompt'
```

create_dot

```
create_dot = 'no'
```

create_statistic

```
create_statistic = 'no'
```

create_dashboard

```
create_dashboard = 'no'
```

create_webpatch

```
create_webpatch = 'no'
```

create_html_log

```
create_html_log = 'no'
```

create_doc

```
create_doc = 'no'
```

tc_ub_test_py_hook_script_path

```
tc_ub_test_py_hook_script_path = '$HOME/testframework/hook-scripts'
```

switch_su_board

```
switch_su_board = 'lab'
```

tc_workfd_can_ssh

```
tc_workfd_can_ssh = 'no'
```

tc_workfd_can_ssh_prompt

```
tc_workfd_can_ssh_prompt = '$'
```

tc_workfd_can_su

```
tc_workfd_can_su = 'no'
```

tc_workfd_can_dev

```
tc_workfd_can_dev = 'can0'
```

tc_workfd_can_bitrate

```
tc_workfd_can_bitrate = '500000'
```

tc_workfd_can_iproute_dir

```
tc_workfd_can_iproute_dir = '/home/hs/iproute2'
```

tc_workfd_can_util_dir

```
tc_workfd_can_util_dir = '/home/hs/can-utils'
```

tc_workfd_hdparm_path

```
tc_workfd_hdparm_path = '/home/hs/shc/hdparm-9.50/'
```

tc_workfd_hdparm_dev

```
tc_workfd_hdparm_dev = '/dev/mmcblk1'
```

Indices and tables

- *genindex*
- *modindex*
- *search*

Index

Python Module Index