Test Plan Reference

PLT

0.5.1

2019-04-16

Revision History

Revision History	
Revision 0.5.1	2019-04-16

- Add `CC1352` target
- Document `define` command

Revision 0.5.0	2019-03-31

- Document `freq` command
- Document `measure` command
- Document `mux` command
- Document `pin` command
- Document `power` command
- Document `short` command
- Document `uartCfg` command

Revision 0.4.9	2019-03-28

- Document `serial request` command
- Update UART port names

Revision 0.4.8	2019-03-08
Renamed to Test Plan Reference	
Revision 0.4.7	2019-03-08
Split off from System Manual	

Table of Contents

Preface

Document describing the test suite definition for use with the Production Line Tool.

1. Test Suite Reference

1.1 Test Suites

Example: A minimal test suite, with a single test item, scanning a bar code.

```
title: "PLT demo: Scan"
suite:
  - ident: SCAN-T1
    title: Scan MAC address
    steps:
      - command: scan MAC_ADDRESS
```

1.1.1 Test Suite Structure

PLT test suites are encoded as YAML text files, starting with the test suite title.

```
title: "PLT demo: Scan"
```

The `title:` is followed by the `suite:` section, containing the test items in the test plan.

1.1.1.1 Test Items

Test items are identified with an `ident:` line, can contain a descriptive `title`, and should contain one or more test item steps.

During test execution, all test item steps need to complete successfully for the test item to succeed.

1.1.1.2 Test Item Steps

Test item steps consist of `command:` or `uartcmd:` blocks. A Test Item can contain multiple Test Item steps.

1.1.2 Basic Example

A basic In-Circuit Test suite for the PLT demo board.

```
title: "v0.1.8 (Green)"
suite:
  - ident: ICT-T1
    title: Identify DUT
    steps:
      - command: identify nRF52
  - ident: ICT-T2
    title: Erase nRF52 with FT2232H Mini Module
    steps:
      - command: erase nRF52
  - ident: ICT-T3
    title: Program DEMO-BOARD FW nRF52 with FT2232H Mini Module
    steps:
      - command: program nRF52 s132_nrf52_6.0.0_softdevice.hex,ly10-demo-fw-
0.1.8.hex,none
  - ident: ICT-T4
    title: BLE discovery
    steps:
      - command: bledis %BLEMAC% 30  # Wait up to 30 seconds for BLE discovery to
complete
```

1.2 Test Commands

1.2.1 bledis - Test BLE Discovery

Establishes a BLE connection to the DUT and discovers GATT services.

Usage:

```
bledis %BLEMAC%|<name> [<timeout>] [<minRSSI>]
```

Argument	Description
name	GAP name advertised by DUT. **%BLEMAC%** to specify the DUT's BLE MAC address instead
timeout	Timeout, in seconds
minRSSI	RSSI treshold (optional)

Example: BLE discovery of identified BLE HW MAC address.

```
- ident: ICT-T1
  title: Identify DUT
  steps:
    - command: identify nRF52
- ident: ICT-T2
  title: BLE Discovery
  steps:
    - command: bledis %BLEMAC% 30 -60
```

1.2.2 define - Define user key

Manually defines a test plan key, which will be emedded in the test report and serial number requests performed as part of the current test plan.

Usage:

```
define <key> <value>
```

Argument	Description
key	Name of the test plan key.
value	Value for the key; can be any kind of string value.

```
- ident: ICT-T0
  title: Set variables
  steps:
    - command: define work_order 1011X02
- ident: ICT-T1
  title: Request serial
  steps:
    - command: request serial
```

1.2.3 erase - Erase DUT Flash

Erase DUT MCU on-board flash.

Usage:

```
erase <target> [UART0|UART1]
```

Argument	Description
target	Target to erase
UART0, **UART1**	Port for UART targets

Supported *target* values:

target	Description
CC1352	TI CC1352 RFSoC
DA14580	Dialog DA14580 RFSoC
ESP32	Espressif ESP32 RFSoC (JTAG)
ESP32_HomeKit	Espressif ESP32 RFSoC (JTAG; HomeKit)
ESP32_JTAG	Espressif ESP32 RFSoC (JTAG)
ESP32_UART	Espressif ESP32 RFSoC (esptool)
nRF52	Nordic nRF52 RFSoC (SWD)
nRF52_DevKit	Nordic nRF52 RFSoC (USB, JLink)
STM32F4	ST STM32F4xx MCU (SWD)
STM32F4_DevKit	ST STM32F4xx MCU (USB, JLink)
STM32L4	ST STM32L4xx MCU (SWD)
STM32L4_DevKit	ST STM32L4xx MCU (USB, JLink)

Example: Erase STM32L4 on-board flash.

```
- ident: ICT-T1
  title: Erase
  steps:
  - command: erase STM32L4
```

1.2.4 extflash_write - Write Peripheral Flash

Write DUT periperal flash.

Usage:

```
extflash_write UART0|UART1 <filename>
```

Argument	Description
UART0, UART1	Port for UART targets
filename	Firmware Element Filename

1.2.5 freq - Frequency Monitor Control

Set the channel to use for frequency measurements.

Usage:

```
freq 0|1
```

Argument	Description
0	Use CLK0
1	Use CLK1

Example: Measure CLK1 frequency

```
suite:
  - ident: ICT-T1
    title: Measure CLK1 frequency
    steps:
    - command: freq 1
    - command: sleepms 1000
    - command: measure frequency 7.90-8.10MHz
```

1.2.6 identify - Identify DUT

Identify DUT MCU and/or RF peripherals.

Usage:

```
identify <target> [UART0|UART1]
```

Argument	Description
target	Target to identify
`UART0`, `UART1`	Port for UART targets

Supported *target* values:

target	Description
`CC1352`	TI CC1352 RFSoC
`DA14580`	Dialog DA14580 RFSoC
`ESP32`	Espressif ESP32 RFSoC (JTAG)
`ESP32_HomeKit`	Espressif ESP32 RFSoC (JTAG; HomeKit)
`ESP32_JTAG`	Espressif ESP32 RFSoC (JTAG)
`ESP32_UART`	Espressif ESP32 RFSoC (esptool)
`nRF52`	Nordic nRF52 RFSoC (SWD)
`nRF52_DevKit`	Nordic nRF52 RFSoC (USB, JLink)
`STM32F4`	ST STM32F4xx MCU (SWD)
`STM32F4_DevKit`	ST STM32F4xx MCU (USB, JLink)
`STM32L4`	ST STM32L4xx MCU (SWD)
`STM32L4_DevKit`	ST STM32L4xx MCU (USB, JLink)

1.2.7 measure - Probe Measurement

Usage:

```
measure <channel> [<signal>] <range>
```

Argument	Description
channel	Measurement channel
signal	DDTPxx or RDTPxx probe for `pin` measurement
range	Acceptable range

The `measure` command supports the following channels:

channel	Description
`current3V3`	3V3 current draw
`current5V`	5V current draw
`currentVARV`	VARV current draw
`frequency`	Frequency (CLK0 or CLK1)
`impedance`	Impedance measurement (RVREF/RA)
`pin`	Digital pin measurement (DDTPxx/RDTPxx)
`voltageDATP07`	DATP07 voltage
`voltageDATP08`	DATP08 voltage
`voltageDATP09`	DATP09 voltage
`voltageDATP10`	DATP10 voltage
`voltageDATP11`	DATP11 voltage

channel	Description
voltageDATP12	DATP12 voltage
voltageMUX0	MUX0 voltage
voltageMUX1	MUX1 voltage
voltageMUX2	MUX2 voltage
voltageMUX3	MUX3 voltage

Example: Electrical measurements

```
suite:
- ident: ICT-T1
  title: Measure CLK0 frequency
  steps:
  - command: freq 0
  - command: sleepms 1000
  - command: measure frequency 32.75-32.78kHz
- ident: ICT-T2
  title: Measure Impedance
  steps:
  - command: measure impedance 2-10Ohm
- ident: ICT-T3
  title: Measure Currents
  steps:
  - command: measure current3V3 <1A
  - command: measure current5V 0.1-0.5A
  - command: measure currentVARV >100mA
- ident: ICT-T4
  title: Measure Voltages
  steps:
  - command: measure voltageMUX0 >1V
  - command: measure voltageMUX1 <3300mV
  - command: measure voltageMUX2 >100mV
  - command: measure voltageMUX3 1500-1800mV
  - command: measure voltageDATP07 200-6000mV
  - command: measure voltageDATP08 -0.2-0.1V
  - command: measure voltageDATP09 0-4V
  - command: measure voltageDATP10 -0.1-3.3V
  - command: measure voltageDATP11 0-3.4V
  - command: measure voltageDATP12 -0.1-3.4V
```

Example: Digital pin measurement

```
suite:
- ident: ICT-T1
  title: Set Digital pins
  steps:
  - command: pin RDTP21 input pullup
  - command: pin DDTP04 input
- ident: ICT-T2
  title: Read Digital pins
  steps:
  - command: measure pin RDTP21 low
  - command: measure pin DDTP04 high
```

1.2.8 mux - Multiplex Control

Select a probe or signal for a multiplex channel.

Usage:

```
mux <channel> <signal>
```

Argument	Description
channel	Multiplex channel (0..3)

Argument	Description
signal	Probe or signal

The following signals and probes can be assigned to a multiplex channel:

signal	MUX	Description
`DATP00`..`DATP06`	0, 1	Direct Analog Test Probes
`RATP00`..`RATP31`	0, 1	Routed Analog Test Probes
`DDTP00`..`DDTP08`	2, 3	Direct Digital Test Probes
`RVREF`	all	Impedance measurement, reference voltage
`RA`	all	Impedance measurement, test voltage
`GND`	all	Ground
`3V3OUT`	all	3.3V power rail
`5VOUT`	all	5.0V power rail
`VARVDIV`	all	VARV power rail, after 1/3 voltage divider

Example: Multiplex control

```
suite:
- ident: ICT-T1
  title: Setup MUX
  steps:
  - command: mux 0 DATP02
  - command: mux 1 DATP03
  - command: mux 2 RA
  - command: mux 3 RVREF
  - command: short 0 2 set
  - command: short 1 3 set
- ident: ICT-T2
  title: Measure impedance DATP02..DATP03
  steps:
  - command: measure impedance 10-200kOhm
```

1.2.9 operator - Operator Test

Instruct operator to perform a manual test step.

Usage:

```
operator <message>
```

Example: Instruct operator inspect housing.

```
- ident: ICT-T1
  - ident: FAT-T1
    title: Visual Inspection (manual)
    steps:
    - command: operator "Inspect Housing"
```

1.2.10 pin - Digital pin control

Configure a Digital probe.

Usage:

```
pin <probe> input [pullup]
pin <probe> output [low|high]
```

Argument	Description
probe	Probe (`DDTPxx` or `RDTPxx`)

Argument	Description
input	Configure as input
pullup	Enable pull-up (only for RDTPxx pins)
output	Configure as output
low	Set low
high	Set high

Example: Control digital pins

```
suite:
  - ident: ICT-T1
    title: Configure digital inputs
    steps:
    - command: pin DDTP05 input
    - command: pin DDTP03 input
    - command: pin RDTP04 input pullup
  - ident: ICT-T2
    title: Control digital outputs
    steps:
    - command: pin DDTP00 output
    - command: pin DDTP02 output low
    - command: pin RDTP01 output high
```

1.2.11 power - Power Control

Control power applied to the Device Under Test.

Usage:

```
power off
power <rail> [on|off|<level>]
```

Argument	Description
rail	Power rail
on	Turn on the specified power rail
off	Turn off all or specified power rail
level	Voltage level for VARV: 2..12.0

Supported *rail* values:

rail	Description
3V3	3.3V power rail
5V	5V power rail
VARV	Variable power rail
12V	12V power rail
SWD	SWD (3.3V) power rail
JTAG	JTAG (3.3V) power rail
UART0	UART0 (3.3V) power rail
UART1	UART1 (3.3V) power rail

Example: Apply power

```
suite:
  - ident: ICT-T1
    title: Apply power
    steps:
    - command: power 3V3 on
    - command: power 5V on
    - command: power VARV 10.2
  - ident: ICT-T2
    title: Wait
    steps:
    - command: sleepms 1000
  - ident: ICT-T3
    title: Power off
    steps:
    - command: power off
```

1.2.12 program - Program DUT

Erase and Program DUT MCU on-board flash.

Usage:

```
program <target> [UART0|UART1] [offset1:]<img1>,[offset2:]<img2>,[offset3:]<img3>[,...]
```

Argument	Description
target	Target to program
UART0, UART1	Port for UART targets
offset1 Opti	onal: Offset for 1st image
img1	Firmware Element (Bootloader)
offset2 Opti	onal: Offset for 2nd image
img2	Firmware Element (Application)
offset3 Opti	onal: Offset for 3rd image
img3	Firmware Element (Partitioning)
…	Optional: Additional offsets, images

Supported *target* values:

target	Description
CC1352	TI CC1352 RFSoC
DA14580	Dialog DA14580 RFSoC
ESP32	Espressif ESP32 RFSoC (JTAG)
ESP32_HomeKit	Espressif ESP32 RFSoC (JTAG; HomeKit)
ESP32_JTAG	Espressif ESP32 RFSoC (JTAG)
ESP32_UART	Espressif ESP32 RFSoC (esptool)
nRF52	Nordic nRF52 RFSoC (SWD)
nRF52_DevKit	Nordic nRF52 RFSoC (USB, JLink)
STM32F4	ST STM32F4xx MCU (SWD)
STM32F4_DevKit	ST STM32F4xx MCU (USB, JLink)
STM32L4	ST STM32L4xx MCU (SWD)
STM32L4_DevKit	ST STM32L4xx MCU (USB, JLink)

1.2.13 scan - Scan Barcode

Scan a barcode using USB-attached barcode scanner.

Usage:

```
scan <format>
```

Argument	Description
format	Format of code to scan

Supported *format* values:

format	Description
%MAC_ADDRESS%	48-bit MAC address (XX:XX:XX:XX:XX:XX)

1.2.14 serial - Request serial number

Request a serial number through PLTcloud. Under Project Settings in PLTcloud, serial number allocation can be configured to use either monotonic counters, or through a WebHook.

Usage:

```
serial request
```

Argument	Description
request	Request serial number through PLTcloud backend

When a webhook is selected in PLTcloud, a JSON request will be sent containing:

```
{
    "dut_id": "<DUT IDENTIFIER>",
    "serial_number": "<SERIAL NUMBER>",
    "ble_mac": "<BLE MAC>",
    "mcu_id": "<MCU ID>",
    "MYCUSTOMKEY": "my custom value"
}
```

And PLTcloud should return something like:

```
{
    "serial_number": "<SERIAL NUMBER>"
}
```

1.2.15 short - Connect Multiplex channels

Usage:

```
short <muxA> <muxB> set|release
```

Argument	Description
muxA	Multiplex channel (0..3)
muxB	Multiplex channel (0..3)
set	Short specified multiplex channels
release	Release short between specified multiplex channels

Example: connect multiplex channels

```
suite:
  - ident: ICT-T1
    title: Control shorts
    steps:
    - command: short 0 1 set
    - command: short 1 3 set
    - command: sleepms 1000
  - ident: ICT-T1
    title: 0-2 instead of 0-1
    - command: short 0 1 release
    - command: short 0 2 set
```

1.2.16 sleepms - Delay

Temporarily suspend test suite execution.

Usage:

```
sleepms <duration>
```

Argument	Description
duration	Duration, in milliseconds

1.2.17 uart - Send and Extract UART response

Extract data from UART.

Usage:

```
uartcmd: uart UART0|UART1
  [[expect: <expectText>]
   [extract: <extractText>
    extractKey: <extractKey>]]
  [send: <sendText>]
```

Argument	Description
`UART0`, `UART1`	UART port
expectText	Text to expect, prior to extraction
extractText	Regular expression to extract
extractKey	Key in which to store extracted text
sendText	Text to send prior to extaction

Example: extract ICCID from cellular modem, storing in the `ICCID` key.

```
- uartcmd: uart UART0
  expect: "+CCID:"
  extract: "CCID: (\\d{20})\r\n"
  extractKey: ICCID
  send: "AT+ICCID\r\n"
```

1.2.18 uartAwait - Await UART response

Wait for a specific UART response.

Usage:

```
uartAwait UART0|UART1 <seconds>
```

Argument	Description
`UART0`, `UART1`	UART port
`seconds`	Time to await response, in seconds

Example:

```
- command: uartExpect UART0 Pressed
- command: operator "Press button"
- command: uartAwait UART0 1
```

1.2.19 uartCfg - Configure UART

Configure a UART port.

Usage:

```
uartCfg UART0|UART1 <speed> [<triplet>]
uartCfg UART0 tp
```

Argument	Description
UART0, UART1	UART port
speed	Baud rate
triplet	UART configuration triplet: 8N1 or 7E1
tp	Use alternate test points instead of UART. (Only for UART0)

Example:

```
- command: uartCfg UART1 9600 8N1
- command: uartCfg UART0 tp
```

1.2.20 uartExpect - Set expectation for uartAwait

Set a UART response to wait for with the `uartAwait` command.

Usage:

```
uartExpect UART0|UART1 <expect>
```

Argument	Description
UART0, UART1	UART port
expect	String to expect with subsequent `uartAwait` command

Example:

```
- command: uartExpect UART0 Pressed
- command: operator "Press button"
- command: uartAwait UART0 1
```

1.2.21 uartReadTimeout - Test if UART is not transmissing

Test if nothing is received from UART.

Usage:

```
uartReadTimeout UART0|UART1 <seconds> [<sendText>]
```

Argument	Description
UART0, UART1	UART port
seconds	Number of seconds to wait for incoming data
sendText	Text to send before waiting

Example: Test if modem is shut down.

```
- command: uartReadTimeout UART0 1 "AT"
```