EVAJIG

Turnkey PCBA testing & provisioning

Getting started

by

EVABITS

Purpose of this document

This document describes how to define functional tests for the EVAjig testing and provisioning platform and how to provide the necessary information to EVAbits. The EVAjig portal guides product developers through the same steps that are described in this document. This document is intended for internal use by product developers and/or communication with EVAbits via other means than the portal. The portal can be found at portal.evajig.com and requires a user to register before use.

Revision history

DATE	VERSION	CHANGES
4-10-2023	1.0	Initial document

Contact information

Find more information on:
evabits.com
evajig.com
tel (EVAbits): +31 50 211 38 45

Disclaimer

No rights can be derived from the information in this document. The information contained in this document is subject to change without prior notice. Always contact EVAbits when considering the EVAjig test platform.

Table of contents

Functional Test Template

EVAjig

The EVAjig is a bed-of-nails device for functional testing, programming and provisioning PCBAs. Up to 12 devices can be handled in parallel. We refer to EVAjig.com for the most recent specifications and design rules for placing test points.

Approach

Ideally, the product to be produced is designed with a set of functional requirements in mind and a document describing these requirements is available. These probably must be augmented with some of the more basic tests (e.g. intermediate power rail check). This document gives examples of the type of tests that can be performed by the EVAjig and outlines how to describe these tests in a uniform manner.

Gold reference

A gold reference panel or single PCB is really useful to make adjustments to tests. For instance, upper and lower limits for acceptance must be defined for each test. These limits must be set stringent enough to prevent faulty PCBAs from passing, but at the same time allow for some variation among PCBAs and components and possibly different vendors for components used in different production runs.

1 - General information

The first thing to do is to provide a summary of the equipment and revisions of designs and firmware. An example of such a summary is provided in the table below. Make sure that this table is always up to date, as it will probably require frequent updates.

category	item	value	version
Test fixture	EVAjig		version 1.0*
DUT / Panel	Gerber file	unique_filename.grbrs	1.0.2
	Panel size	240x160x1.60 mm	n.a.
	DUTs per Panel	12	
	Testpoints per DUT	10	
	Testpoint position file	TP_pos_bottom.csv	1.0.2
	components on bottom?	yes	
	● if yes, maximum height?	2.45 mm	
	Testpoints on top?	no	
	Testpoint position file	n/a	n.a.
Firmware	Version	unique_filename.hex	1.2.21

** Please note that all values given throughout this document are examples.*

Components on the bottom side that are higher than 3.0mm require an additional cutout in the cassette; please provide the coordinates and shape of the protruding component. Please contact EVAbits when components on the bottom side are higher than 8.0mm.

For convenience, provide a list of test points in the document (copy or export from csv). Since testpoint names do not change as readily as references, use the testpoint name in this document. This list can actually be made when describing tests (see next chapters), prior to design implementation.

Reference	Name	Description
TP01	TP_UART_RX	debug info in ascii
TP02	TP_BATV	battery voltage
TP03	TP_LDO	regulated voltage

2 - Top-level test description (workflow)

To perform a test, execute the steps outlined in this chapter in order. In general, each row in the table below corresponds to a step in the operator interface. A detailed work description for each step is provided in the following chapter and in the EVAjig portal.

Overview

step	action	test	note
1	operator: insert PCBA	-	correct alignment by alignment pins in reference holes
2	scan device IDs	-	
3	power check	check power voltages	Current limits for power rails are set. Use this test phase to test several voltages in parallel, e.g. power rails, reference voltages, I/O lines that are pulled high or low, etc.
4	program	program component(s)	
5	power up sequence	check startup sequence, communications	
6	I/O tests	provide digital commands or analog voltages, test response	
7	provisioning	program unique keys etc.	
8	lock device	lock, then try to access DAP port	
9	power off	-	
10	generate report, operator: take PCBA	-	

3 - Detailed test descriptions

The steps outlined in the previous chapter will be described in more detail below. For each step, it will be specified whether an action is required (operator), whether the EVAjig will perform an action (test), and what the result of performing the actions pertaining to each step will be (result). Please refer to the EVAjig portal for a more detailed description of each step.

1 insert PCBA

Operator:
    Place panel in the EVAjig. Check the orientation. Close the lid. Upon closing, the next phase will be started automatically.
Test:
    None

2 scan device IDs

Operator:
    Start scanning of IDs. When scanning is done, click s*tart test* to advance.
Test:
    None
Result:
    DUT identification numbers are stored.

3 power check

Operator:

No action required. Typically, all the tests follow each other automatically unless there is a failure.

Test:

EVAjig turns on power rail voltage and observes the maximum current setting. Voltages are semi-static.

Result:

Pass if all voltages are within range (i.e. no shorts that affect DUT power)

		testpoint(s)	max current	settling time	target	min	max
3.1	output voltage 5V	TP_PWR	250 mA	100 ms	5.0		
3.2	read Vsup	TP_VSUP	-	200 ms	5.0	4.5	5.1
3.3	read 1V8	TP_1V8	-	200 ms	1.8	1.75	1.85

4 program

Operator:

No action required.

Test:

Bitfiles are downloaded and flashed to programmable devices.

Result:

The programmable devices are now functional.

		testpoint(s)	fw	max duration
4.1	program file	TP_SWDIO TP_SWDCLK TP_SDWRST	hexfile name	5s
4.2	toggle reset	TP_NRST		

5 power-up sequence

Operator:

No action required.

Test:

Monitor a communication channel (e.g. UART) that provides information about a microcontroller startup sequence. The tests are positively defined, e.g. *the string “init procedure completed successfully” is observed within 10 seconds*. Any negatively defined message, such as a debug message like “*error in modem_init()*” should be changed into something like: *“modem setup successfully” within 10 seconds*.

Result:

Pass if there are one or more strings observed within a given time.

		testpoint	pass conditions
5.1	monitor UART rx	TP_UART_RX	A: “system init success” within 2 seconds B: “zephyr console ready” within 4 seconds C: “SD card ready” within 4 seconds
5.1			pass if A&B&C

6 I/O tests

Operator:

No action required.

Test:

Command/response type of tests. Provide a command, test vector of DIOs or analog voltage to the DUT and observe the response. Values can be read from UART for this purpose, e.g. the number 222.26 can be extracted from “longitude: 222.26\n\r”.

Result:

Pass when values are within range, or output otherwise is as expected.

		testpoint	expected result or conditions	
6.1	UART tx	TP_UART_TX	“get ”	
6.2	monitor UART rx	TP_UART_RX	“fix acquired” within 20 seconds	
6.3	get value UART rx	TP_UART_RX	“longitude: “	5 < value < 6
6.4	out INV	TP_INV	0.153	
6.5	read OUTV	TP_OUTV	2.0*0.153	+/- 0.01

7 Provisioning

Operator:
    No action required.
Test:
    Program unique keys for the device to access the end-customer IT infrastructure.
    Perform an (OTA) firmware update if necessary.
Result:
    Device is now known by the end-customer's network. The device can access the network in a secure manner.

8 Lock device

Operator:
    No action required.
Test:
    Lock the device via JTAG (see programming pins).
Result:
    Chip is locked. Always lock programmable devices, thus making sure the keys cannot be copied by anyone.

		testpoint(s)	fw	max duration
4.1	lock	TP_SWDIO TP_SWDCLK TP_SDWRST	hexfile name	5s
4.2	toggle reset	TP_NRST		
4.3	try to access DAP	TP_SWDIO TP_SWDCLK TP_SDWRST	expected: fail to access DAP	

9 Power down

Operator:
    No action required.
Test:
    None
Result:
    DUT is shut down devices gracefully before ending the test procedure.

10 Show results

Operator:

> Check the overview of test results. Test results are automatically stored in the EVAjig portal. Click next to test another PCBA.

Result:

> The test report is uploaded to the backend and can be accessed from the EVAjig portal.

Providing information

Name the testpoints according to the instructions outlined in this document. When delivering a complete panel, the use of the <DUT_idx>_TP_<description> naming convention is strongly recommended, where DUT_idx would go from 0 (upper left DUT on panel) to a maximum of 12 (the bottom right). Always make sure that the origin of the coordinates is at the top left corner of the panel, as a result of which all coordinates will be positive.
Deliver testpoint information in the following formats (in order of preference):

1. If you are using Kicad, the .kicad_pcb file (of the complete panel) is the most concise.
2. Gerber files of edge.cuts and bottom layers (with testpoint names and positions).
3. A position file (usually <your filename>_pos_bottom.csv), like the example provided below. The mandatory columns are marked with *.

Ref *	**Val**	**Package**	**PosX ***	**PosY ***	**Rot**	**Side**
Board_0-TP_UARTTx1	TestPoint	TestPoint_Pad_D1.5mm	36.6774	51.88	0	bottom
Board_0-TP_SWDio1	TestPoint	TestPoint_Pad_D1.5mm	21.9024	30.13	0	bottom