|  |
| --- |
| Macintosh HD:Users:iriark01:Documents:BLE:LargeLogo.png |
|  |
| Engineering  **mbedOS System Test Testbed Specification: High Level Design**  **CONFIDENTIAL**  Approval Status: Draft  Document Reference Number: ARM\_MBED\_TN\_0012  Date: 26 June 2015  Version: ~1.0 |
|  |
| Authors:  Simon Hughes |
|  |

Table of Contents

[Acronyms 3](#_Toc423093927)

[1 Introduction 4](#_Toc423093928)

[1.1 Document Scope 4](#_Toc423093929)

[1.2 Outstanding Issues 4](#_Toc423093930)

[1.3 Definition of Terms 5](#_Toc423093931)

[2 Requirements 5](#_Toc423093932)

[3 Testbed Architecture 6](#_Toc423093933)

[3.1 Physical Architecture 6](#_Toc423093934)

[3.2 Logical Architecture 7](#_Toc423093935)

[4 Physical Design of Test Bed 9](#_Toc423093936)

[4.1 Physical Realisation 11](#_Toc423093937)

[5 Software 11](#_Toc423093938)

[5.1 Overview 11](#_Toc423093939)

[5.2 Automation Software 12](#_Toc423093940)

[5.3 Supported Host Operating Systems 12](#_Toc423093941)

[6 Responsibilities 12](#_Toc423093942)

[6.1 Module Owners (Tools) 12](#_Toc423093943)

[6.2 System Test Squad 12](#_Toc423093944)

[6.3 Service Owner 12](#_Toc423093945)

[7 References 12](#_Toc423093946)

# Acronyms

**Build/CI** Build/Continuous Integration.

**KPI** Key Performance Metric

**Test Node** Vendor Target MCU Platform combined with RF PHY Shield.

Table 1. List of acronyms.

# Introduction

This document describes the high level design of the mbedOS System Test Testbed for end-to-end system testing of mbed software components. The testbed can also be used for unit, module, integration, functional, non-functional, regression, stress and load testing.

Section 1 provides an introduction to the document including an overview of each of the sections.

Section 2 provides a brief list of requirements for the testbed including the key requirement for critical end-to-end system testing of mbedOS software applications running on target hardware (Freescale K64F) connecting to Device Server over a 6LoWPAN RF PHY.

Section 3 outlines the solution architecture of the testbed including a description of the physical entities, the logical entities and how the testbed integrates with the Jenkins Build/Continuous Integration infrastructure.

Section 4 describes the physical design of the test bed including the hardware components used.

Section 5 describes the logical design of the testbed highlighting the communication channels between the target software and Device Server.

Section 6 outlines the software that will be run on the test bed including the tools for building and testing software, Test PC host operating systems, and software to integrate the testbed with the Jenkins Build/CI infrastructure.

Section 7 describes the responsibilities for construction and on-going maintenance/operation of the test bed from supporting parties.

Section 8 provides references used throughout this document.

## Document Scope

This document describes the hardware and software high level design of the system test testbed for end-to-end testing and for generating KPIs. The interfaces between key software and hardware entities are within scope of this document.

The following items are out of scope:

1. The Build/CI Infrastructure.
2. The specification of the KPIs to be collected, and the tests used for generating the KPIs.
3. The specification of the dashboard for presentation of the KPIs.
4. The specification of the test suite used to perform testing.

## Outstanding Issues

The following items remain outstanding issues with this version of the document:

1. Drawings have been prepared as freehand sketches and should be captured in a drawing package.
2. The details of a host OS specific, common location (e.g. shared network driver) for compilation and test tools to reduced testbed maintenance is not specified.
3. The document should be made available in a markdown via a GitHub repository.

## Definition of Terms

The acronyms used in this document are defined in Table 1.

# Requirements

The mbedOS system test testbed is a hardware and software environment for running automation tests on physical devices. The requirements for the testbed are as follows:

1. To provide an environment for performing system testing including:
   1. Critical end-to-end system testing. For example, testing target hardware running software applications successfully interoperates with Device Server, can update firmware over the air and establish/de-establish TLS connections.
   2. Integration testing of key module compositions.
   3. Unit testing.
   4. Module testing.
   5. Functional testing.
   6. Non-functional testing.
   7. Single device stress and load testing.
2. To provide an environment for generating critical, end-to-end system tests Key Performance Indicators (KPIs) measuring software quality.
3. To test the mbedOS software build and test systems on the supported host operating systems and a minimal subset of support vendor target MCU platforms including the Freescale FRDM K64F equipped with sub 2.4GHz and 2.4GHz 6LoWPAN shields.

See reference [1] for more information.

# Testbed Architecture

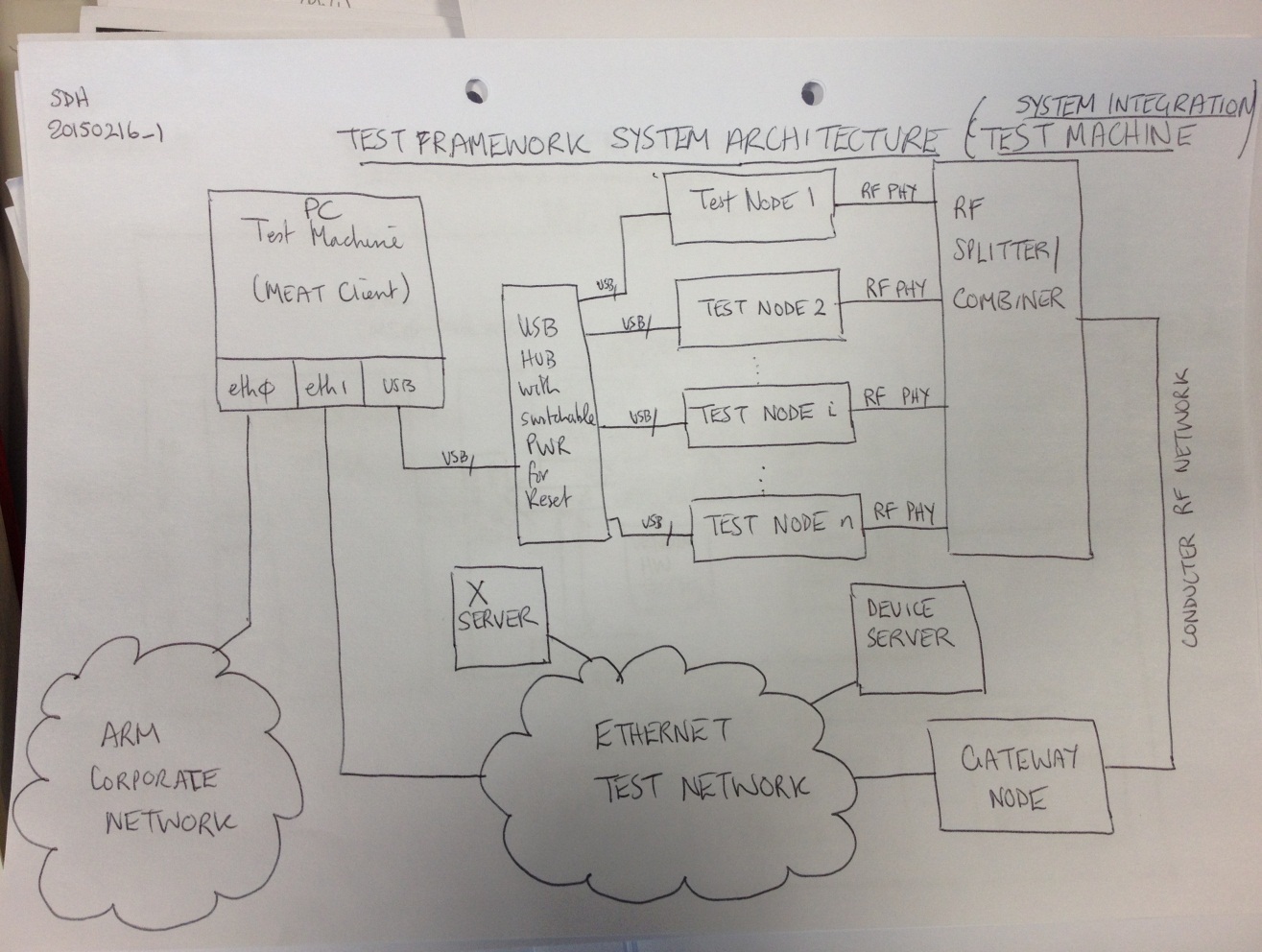


Figure 1. The figure shows the physical design of the mbedOS integration test subsystem. See reference [2] for more information.

## Physical Architecture

Figure 1 provides an overview of the main components in the testbed.

1. The testbed is composed of a Test PC (top left) connected to {1..n} vendor target MCU platforms using {1..m} USB hubs.
2. The target platforms are fitted with RF 6LoWPAN shields for network communication with the Device Server. This composite is called a test node and runs the software under test. The antennae from the test nodes RF shields are connected into an RF conducted network using splitter-combiners to minimise the impact of electromagnetic interference and improve reliability.
3. 6LoWPAN/IPv6 packets from the test nodes are received by the Gateway Node and bridged onto Ethernet for receipt by Device Server.
4. The Ethernet Test Network is a private network for testing locally to the PC and does not connect to the ARM corporate network. The network consists of a switch interconnecting the Test PC, the Device Server PC and the Ethernet ports of each of the test nodes (should they have one). The Test PC has a second Ethernet interface for receiving packets from the test network but does not route packets between the test networks and the corporate network.
5. The Device Server receives packets from the target platform and provides service as required.
6. The Test PC also connects to the ARM corporate network (eth0) for connecting to GitHub on the internet, and interconnecting with the Build/CI infrastructure.

## Logical Architecture

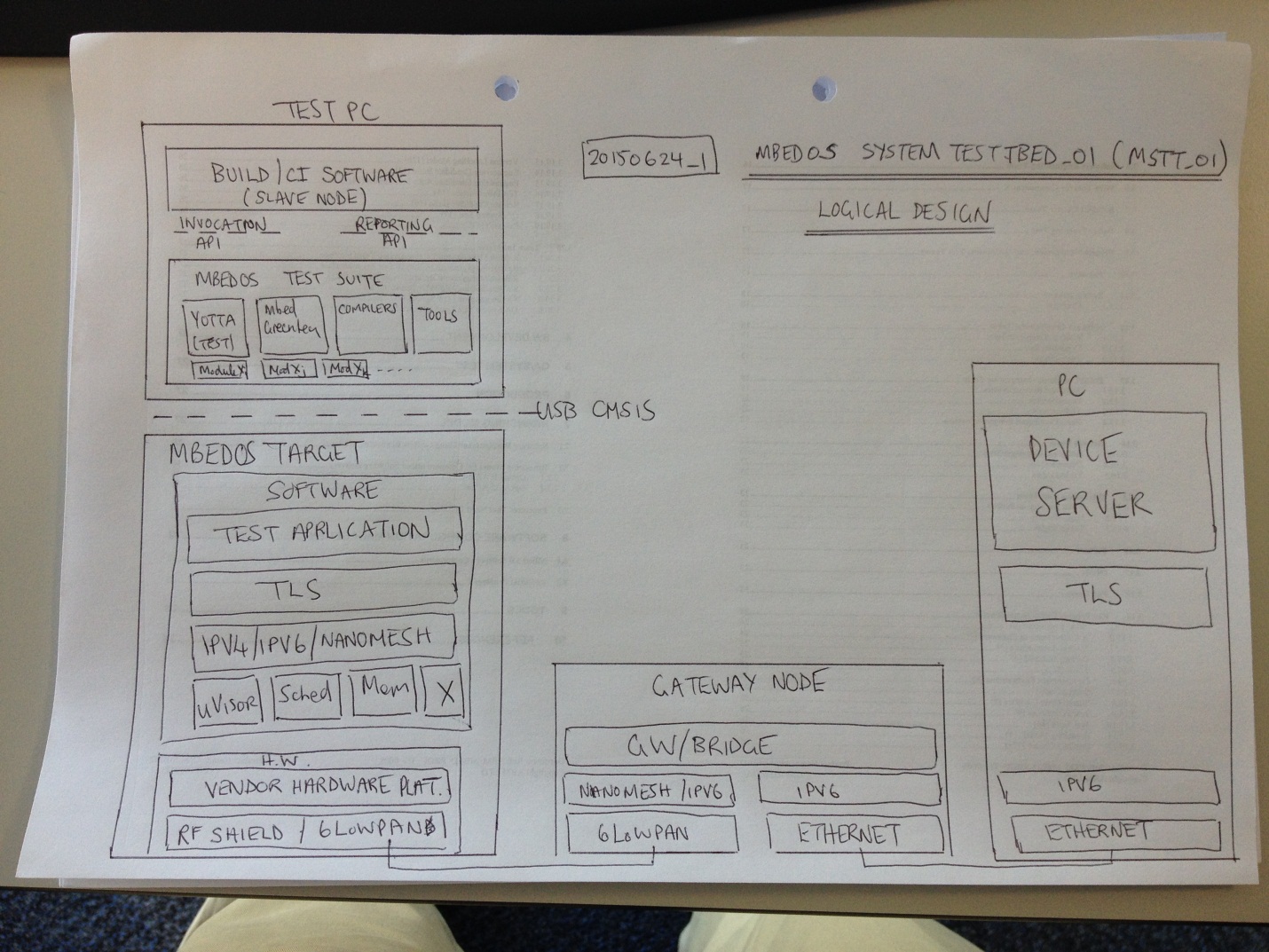


Figure 2.The testbed components with software stacks are shown in the figure.

The logical software entities running on the testbed are shown in Figure 2:

1. The Test PC (top left) has 2 essential functions:
   1. Running the mbedOS test suite. This includes using the yotta, CMake, Ninja, ARM compiler, GCC compiler, IAR compiler toolchains to build the test binaries and then mbed-greentea for automating the copying of the test.bin to the target and collecting results. The output results are in junit xml.
   2. Integrating with the Build/CI infrastructure. When triggered to run a test job, the Build/CI software on the Test PC uses the “Invocation API” to run the mbedOS test suite and the “Reporting API” to collect the junit xml output results.
2. The mbedOS Target test node (bottom left) RF PHY establishes logical transport connections with the Gateway Node for forwarding IPv6 packets to the test network.
3. The 6LoWPAN Gateway Node (centre bottom) bridges IPv6 packets from the RF network to the Ethernet test network for receipt by Device Server.
4. The Device Server (bottom right) is the COAP enabled server providing device services to the target under test, including LWM2M services. Target test binaries using COAP requests/responses establish UDP sessions with the Device Server listening on port 5683, or compressed 61616-61631 for compressed 6LoWPAN.
5. Target test binaries using secure COAP establish logical Datagram TLS sessions with the Device Server.
6. The Test PC is logically a Jenkins Slave Node.

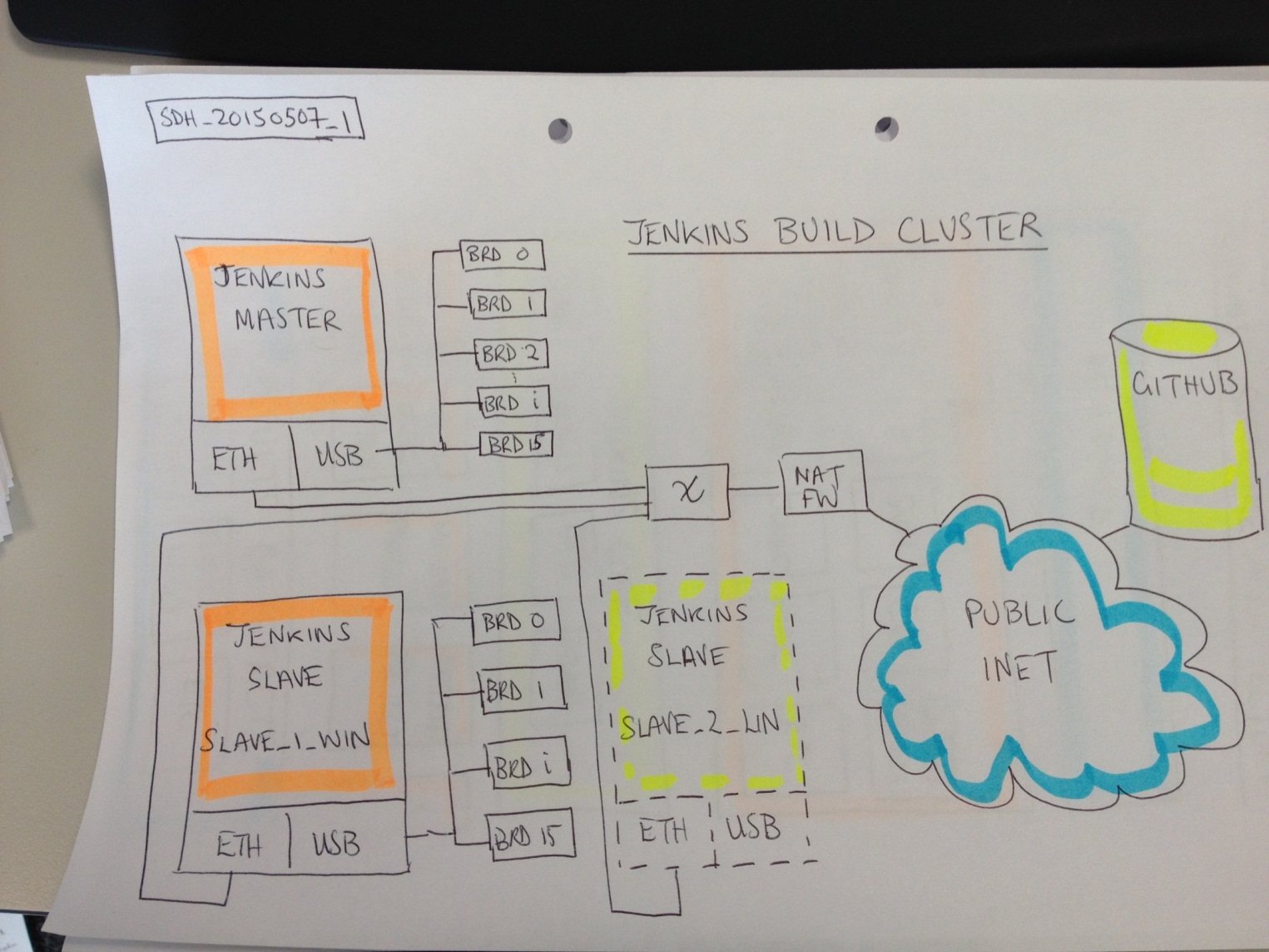


Figure 3. The Jenkins distributed build/test infrastructure consisting of a master and slaves.{todo: redraw more accurately with MSST\_01, MSST\_02, MSST\_03 labels}

The testbed Test PC integrates with the Build/Continuous Integration test infrastructure by being configured as a Jenkins Slave Node (see Figure 3). Furthermore, there is at least one testbed per supported host operating system, as enumerated below:

1. mbedos System Test Testbed number 01 (MSST\_01) has a Test PC running Windows 7. This is Jenkins Slave MSST\_01\_WIN7.
2. mbedos System Test Testbed number 02 (MSST\_02) has a Test PC running Ubuntu 14.04. This is Jenkins Slave MSST\_02\_UBU14.
3. mbedos System Test Testbed number 03 (MSST\_03) has a Test PC running MacOS 10.x. This is Jenkins Slave MSST\_03\_MAC10.

# Physical Design of Test Bed

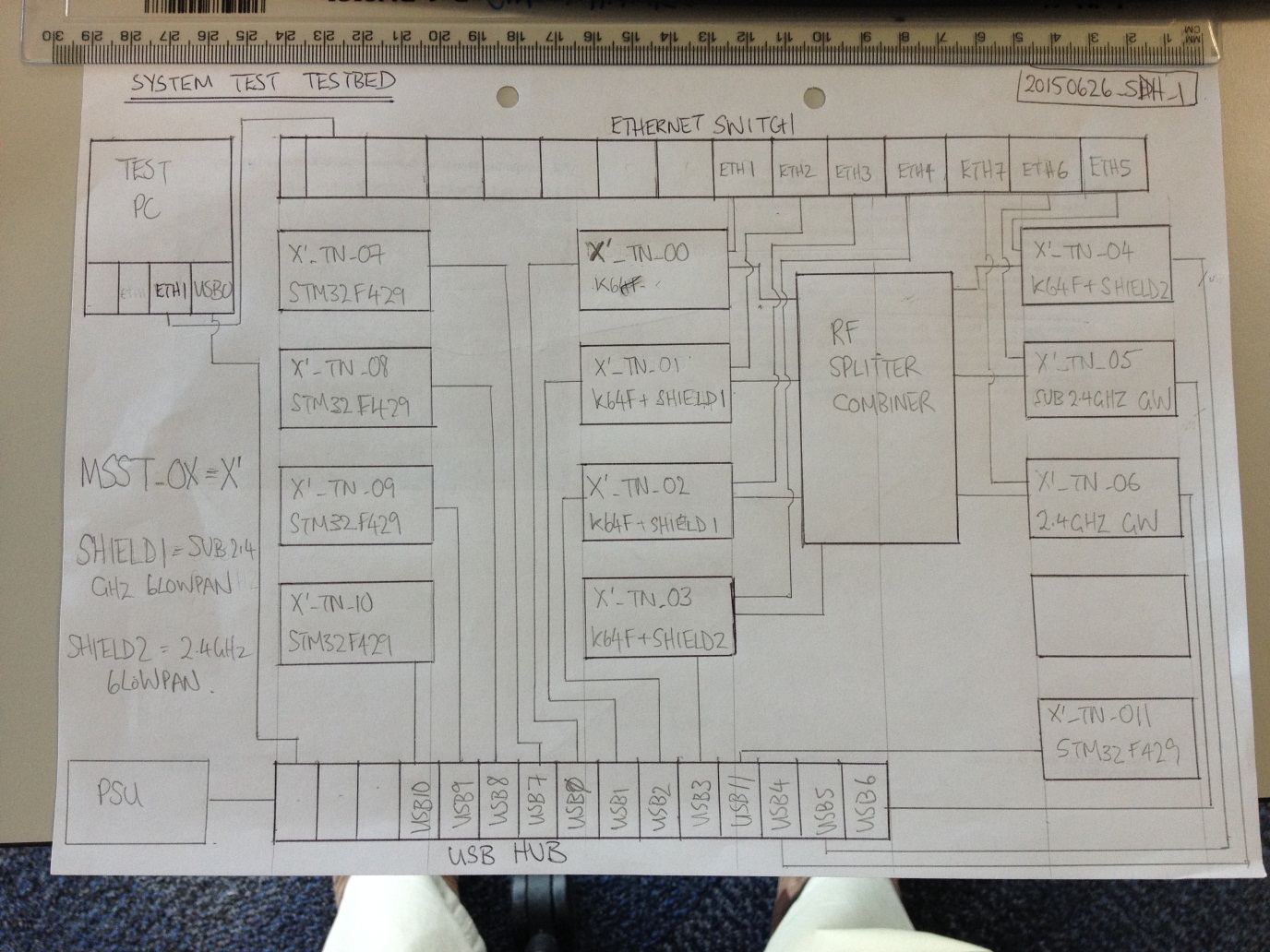


Figure 4. Physical wiring diagram for the testbed.

This section provides a discussion of the physical design of the testbed (see Figure 4). The key features are as follows:

1. Figure 4 shows the testbed which is replicated for each of the supported host OSes. The host OS is running on the Test Machine (top left).
2. The Test Machine has the following interfaces:
   1. It has at least 2 USB 3.0 ports for connecting to {1..n} Cambrionix 12 port USB power switchable hubs.
   2. It has at least 2 Ethernet interfaces, one for connecting to the ARM corporate network and a second for connecting to the private Ethernet test network.
3. In summary, the 12 test nodes in the testbed are as follows (see Figure 4):
   1. 5 Freescale FRDM K64F, 1 without shield (MSST\_0X\_TN\_00), 2 with sub-2.4 GHz 6LoWPAN shields (MSST\_0X\_TN\_01, MSST\_0X\_TN\_02) and 2 with 2.4 GHz 6LoWPAN shields (MSST\_0X\_TN\_03, MSST\_0X\_TN\_04).
   2. 5 ST Microelectronics STM32F429[[1]](#footnote-1) all without shields (MSST\_0X\_TN\_07-MSST\_0X\_TN\_11).
   3. 1 sub-2.4GHz 6LoWPAN/IPv6 to Ethernet/IPv6 Gateway Node (MSST\_0X\_TN\_05).
   4. 1 2.4GHz 6LoWPAN/IPv6 to Ethernet/IPv6 Gateway Node (MSST\_0X\_TN\_06).
4. The test nodes are connected to the Cambrionix USB ports as follows:
   1. USB Port 0: Freescale FRDM K64F with no shield. This device can be used with a custom shield should one be required. The Ethernet port is connected to the Ethernet switch.
   2. USB Port 1: Freescale FRDM K64F with sub-2.4GHz 6LoWPAN shield. The RF antenna is connected to an RF Splitter-Combiner and the Ethernet port is connected to the Ethernet switch.
   3. USB Port 2: Freescale FRDM K64F with sub-2.4GHz 6LoWPAN shield. The RF antenna is connected to an RF Splitter-Combiner and the Ethernet port is connected to the Ethernet switch.
   4. USB Port 3: Freescale FRDM K64F with 2.4GHz 6LoWPAN shield. The RF antenna is connected to the RF Splitter-Combiner and the Ethernet port is connected to the Ethernet test network.
   5. USB Port 4: Freescale FRDM K64F with 2.4GHz 6LoWPAN shield. The RF antenna is connected to the RF Splitter-Combiner and the Ethernet port is connected to the Ethernet test network.
   6. USB Port 5: Mbed sub-2.4GHz 6LoWPAN Gateway Node. The gateway bridges packets received over the RF PHY onto the Ethernet Test Network.
   7. USB Port 6: Mbed 2.4GHz 6LoWPAN Gateway Node. The gateway bridges packets received over the RF PHY onto the Ethernet Test Network.
   8. USB Port 7: This connects to an ST Microelectronics STM32F429.
   9. USB Port 8: This connects to an ST Microelectronics STM32F429.
   10. USB Port 9: This connects to an ST Microelectronics STM32F429.
   11. USB Port 10: This connects to an ST Microelectronics STM32F429.
   12. USB Port 11: This connects to an ST Microelectronics STM32F429.
5. The RF-Splitter Combiner is used for connecting both the sub-2.4GHz and 2.4GHz test nodes in a conducted RF network.
6. The Ethernet switch for the private test network is to provide:
   1. Connectivity between the K64F nodes and the Test PC.
   2. Connectivity between the 6LoWPAN gateway nodes, the Device Server (if installed on a separate PC) and the Test PC.
7. The Ethernet port connectivity is as follows:
   1. ETH Port 0: Connects to the eth1 interface on the Test PC.
   2. ETH Port 1: Connects to the K64F test node MSST\_0X\_TN\_00.
   3. ETH Port 2: Connects to the K64F test node MSST\_0X\_TN\_01 port on the K64F.
   4. ETH Port 3: Connects to the K64F test node MSST\_0X\_TN\_02 port on the K64F.
   5. ETH Port 4: Connects to the K64F test node MSST\_0X\_TN\_03 port on the K64F.
   6. ETH Port 5: Connects to the K64F test node MSST\_0X\_TN\_04 port on the K64F.
   7. ETH Port 6: Connects to the sub 2.4 GHz GW test node MSST\_0X\_TN\_05.
   8. ETH Port 7: Connects to the 2.4 GHz GW test node MSST\_0X\_TN\_06.
   9. ETH Port 8: Reserved.
   10. ETH Port 9: Reserved.
   11. ETH Port 10: Reserved.
   12. ETH Port 11: Reserved.
   13. ETH Port 12: Connects to the Device Server PC Ethernet port (if present as separate PC).
8. The Device Server provides connectivity services for software under test e.g. mbed-client, lwm2m-client.

## Physical Realisation

The test-bed physical equipment is installed in 7U Eurocard form factor chassis. Four or more blades are mounted in the chassis, and the chassis mounted in a 19inch rack.

The following components will typically be mounted on a chassis blade for test nodes (for example):

1. Up to 12 vendor target MCU platforms and/or 6LoWPAN gateway nodes with RF shields.
2. 1 x Cambrionix USB Power Switchable Hub. The PSU for the USB hub is located on a separate blade.
3. Up to 2 RF Splitter Combiners for interconnecting the 6LoWPAN nodes and gateways.

The Test PCs are of 1U or 3U form factor server blades suitable for mounting in the 19inch rack.

# Software

## Overview

The following software is installed on Test PC:

1. ARM Compiler.
2. CMake.
3. CodeSourcery Compiler
4. gcc-arm-none-eabi Compiler.
5. Git.
6. GnuWin32.
7. IAR Systems.
8. IAR\_License\_Server\_Tools. This is used to License the IAR compiler.
9. Java. This is used by Device Server and Jenkins.
10. lwm2m-CONF. This is the Device Server installation subdirectory.
11. mbed-Greentea test suite including mbedls, mbed-hosttests.
12. Ninja. Used by Yotta.
13. NXP LPCXpresso Compiler.
14. Python27.
15. SafeNet\_Sentinel.
16. Scripts.
17. Virtualenv
18. Yotta.

As far as practicable, the tools are collectively managed and installed on a shared network drive in a mbed\_tools directory allowing multiple machines to share the same tools installation.

Virtualenv should be used for the simultaneous use of different version of Python and associated libraries. A single standard version of Python will be installed on the system PATH.

## Automation Software

The Test PC in the test bed will be configured as a Jenkins slave node.

## Supported Host Operating Systems

The supported host operating systems are as follows:

1. Windows 7.
2. Ubuntu 14.04 .
3. Mac OS 10.x.

# Responsibilities

Multiple parties undertake responsibility for the generation of mbedOS KPIs and the operation of the Build/CI test infrastructure used to generate the KPIs.

## Module Owners (Tools)

In relation to the ongoing support of the System Test Testbed, Module Owners (containing tools) are responsible for:

1. Ensuring that the tool is installed in the correct place on the Build/CI for use by test jobs, and the correct version is installed.

## System Test Squad

In relation to the ongoing support of the System Test Testbed, the System Test Squad is responsible for:

1. Building, maintaining and the on-going operational support of the test infrastructure hardware and software platforms.

## Service Owner

Service Owners are responsible for the deployment of test services on the Build/CI Infrastructure (e.g. Device Server), and the correct version of the test service.

# References

The references used throughout this document are recorded in the following table.

| Ref | Title |  |
| --- | --- | --- |
| 1 | ARM\_MBED\_TN\_0011 The mbedOS Way, [http://teamsites.arm.com/sites/IOTBU/Engineering/Shared%20Documents/ Mbed%20Document%20Library/Technical%20Notes/ The%20mbed%20OS%20way.docx](http://teamsites.arm.com/sites/IOTBU/Engineering/Shared%20Documents/%20Mbed%20Document%20Library/Technical%20Notes/%20The%20mbed%20OS%20way.docx) |  |
| 2 | ARM\_MBED\_TN\_0010 Test Framework System Architecture v1.0, [http://teamsites.arm.com/sites/IOTBU/Engineering/Shared%20Documents/ Mbed%20Document%20Library/Technical%20Notes/ Test%20Framework\_System\_Architecture\_v1.0.docx](http://teamsites.arm.com/sites/IOTBU/Engineering/Shared%20Documents/%20Mbed%20Document%20Library/Technical%20Notes/%20Test%20Framework_System_Architecture_v1.0.docx) |  |
| 3 |  |  |
| 4 |  |  |
| 5 |  |  |
| 6 |  |  |
| 7 |  |  |
| 8 |  |  |
| 9 |  |  |

1. Does it make sense to include STM32F429 in the testbed? [↑](#footnote-ref-1)