IoT Software Requirements Specification

Andrew Bonneville

Table of Contents

1 Introduction 3
  1.1 Purpose 3
  1.2 References 3
2 Overall Description 4
  2.1 Product Perspective 4
  2.2 Design & Implementation Constraints 4
    2.2.1 Operations 5
  2.3 External Interface Requirements 5
    2.3.1 User Interface 5
    2.3.2 Hardware Interface 5
      2.3.2.1 Processor 5
      2.3.2.2 Sensors 5
      2.3.2.3 Storage 5
      2.3.2.4 Connectivity 5
    2.3.3 Communication Interface 5
      2.3.3.1 USB Class 5
      2.3.3.2 USB Command Interface 6
      2.3.3.3 USB Response Interface 7
      2.3.3.4 WI-FI 7
  2.4 Other Requirements 7
    2.4.1 Performance 7
      2.4.1.1 Processor Load 7
      2.4.1.2 Memory Utilization 7
    2.4.2 Safety 8
    2.4.3 Security 8
    2.4.4 Software Quality Attributes 8
3 Glossary 8

1 Introduction

1.1 Purpose

Define software requirements specification (SRS) for the IoT Sensor Project. The SRS captures the requirements between users and design.

Any and all requirements defined in here will be tested and validated under a separate test plan.

1.2 References

STMicroelectronics, https://www.st.com:

- UM2153 User manual, Discovery kit for IoT node (B-L475E-IOT01A).
- UM1734 User manual, STM32Cube USB device library.
- RM0351 Reference manual, STM32L4x5
- DS10969 Datasheet, STM32L475xx

Other:

- Amazon FreeRTOS, device registration and security requirements, https://aws.amazon.com/freertos/

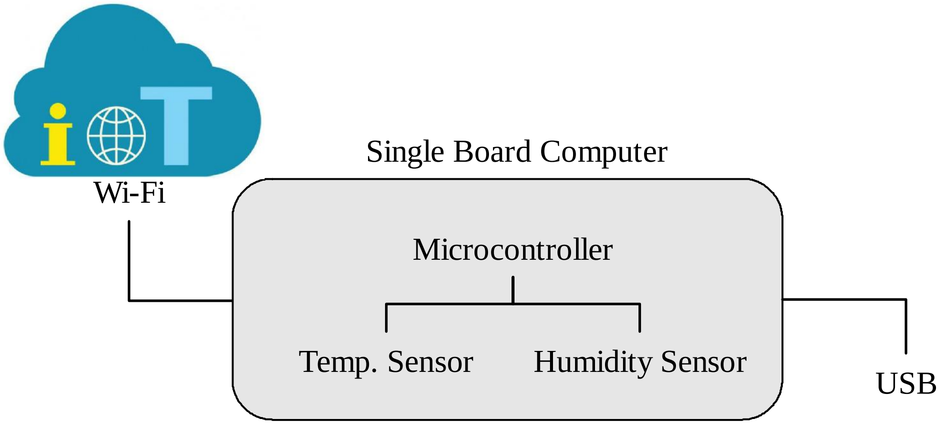
2 Overall Description

2.1 Product Perspective

Overall product will upload sensor information unto cloud servers. This project will demonstrate the capability as a stepping stone to develop more advance products.

Project consists of the embedded software that will be developed for installation on a commercial of the shelf (COTS) single board computer that contains a; microcontroller, sensors, USB connectivity, and Wi-Fi connectivity.

- The USB interface will implement a command and response interface that connects directly to a serial terminal (e.g. YAT, HyperTerminal). This will provide the ability to configure the embedded software for different Wi-Fi installations, cloud server configuration, as well the ability to directly connect and monitor system behavior.
- The Wi-Fi interface will provide an Internet connection to periodically deliver sensor information to cloud servers (e.g. AWS, Azure)



2.2 Design & Implementation Constraints

The primary application will be implemented in C++, and leverage vendor and open source software typically implemented in C.

Software build tools and development will utilize GNU tools. Note, known distributions deliver the Libstdc++ in a less than ideal configuration for bare metal systems that results in some unused code remaining in the executable after final link and optimization.

2.2.1 Operations

Any and all open source software being considered must be reviewed and approved by contracts before use. In general all copyleft licensing (e.g. GPL) is not permitted, while permissive licensing models (e.g. MIT, BSD, Apache) will be considered.

2.3 External Interface Requirements

2.3.1 User Interface

No user specific or graphical interface will be implemented for this project, see communication interface.

2.3.2 Hardware Interface

2.3.2.1 Processor

A single processor shall host the embedded software, STM32L475VG.

2.3.2.2 Sensors

On board digital sensor for relative humidity and temperature shall be HTS221.

2.3.2.3 Storage

All storage shall use on chip memory (STM32L475VG), no external memory devices.

2.3.2.4 Connectivity

USB connectivity shall use on the chip (STM32L475VG) USB OTG FS controller.

Wi-Fi connectivity shall use the on board module from Inventek Systems (ISM43362). Software will implement a SPI interface to exchange AT commands with module.

2.3.3 Communication Interface

2.3.3.1 USB Class

USB interface shall implement a communication device class (CDC) that operates at full-speed (FS), 12 mpbs. USB class shall conform to:

- Universal Serial Bus Class Definitions for Communications Devices Revision 1.2 November 16, 2007
- Sub-protocol specification of “Universal Serial Bus Communications Class Subclass Specification for PSTN Devices Revision 1.2 February 9, 2007"

The implementation shall use the STM32 default vendor provided vendor ID (VID) and product ID (PID), and vendor provided host drivers for a virtual communication port (VCP).

2.3.3.2 USB Command Interface

All commands and related fields shall be case sensitive. Following list is all of the user commands available via the USB interface:

Command	Description	Stored	Re-sponse
cloud cert <field>	Shall set the device certificate for connecting to a cloud server. <field> is variable length cert value, and is limited to 2048-bytes or less	Yes	No
cloud key <field>	Shall set the private key for connecting to a cloud server. <field> is variable length key value, and is limited to 2048-bytes or less	Yes	No
cloud name <field>	Shall set the thing name for connecting to a cloud server. <field> is variable length name value, and is limited to 64-bytes or less	Yes	No
cloud url <field>	Shall set the hostname / URL for connecting to a cloud server. <field> is variable length url value, and is limited to 64-bytes or less	Yes	No
cloud status	Shall report status for the cloud connection.	No	Yes
help	Shall report a list and brief description of each command supported.	No	Yes
reset	Full processor reset; core and peripherals, as well as external plug-in modules.	No	No
status	Shall report high level system information and status.	No	Yes
version	Shall report application and library version numbers	No	Yes
wifi off	Immediately turns off the WIFI radio.	Yes	No
wifi on	Immediately turns on the WIFI radio.	Yes	No
wifi password <field>	Shall set the password for connecting to a particular WIFI network. <field> is variable length password value, and is limited to 32-bytes or less	Yes	No
wifi ssid <field>	Shall set the SSID for connecting to a particular WIFI network. <field> is variable length SSID value, and is limited to 32-bytes or less	Yes	No
wifi status	Shall report status for the WIFI connection.	No	Yes

Stored = setting is retained in non-volatile memory and is applied when the system initializes during reset/power-up.

Response = command generates a response message

2.3.3.3 USB Response Interface

Not all commands generate a command specific response message. For a list of commands that do, see USB Command Interface.

When an invalid command is received, the system will generate a response message stating an invalid command was received.

When the system is ready to accept a new command, a command prompt message will be sent and consist of two '>' followed by a single space ' ' character: ">> "

Content of a response message is not mandated, but shall be human readable and address the command description as stated in the USB Command Interface.

2.3.3.4 WI-FI

Wi-Fi interface shall utilize the external module Inventek ISM43362-M3G-L44 (802.11 b/g/n compliant), which handles all radio communication as well as OSI Layers 1- 4 (fully contained TCP/IP stack).

This project shall be responsible for developing and maintaining OSI Layers 5-7 and shall be stored and executed on the processor (STM32L475VG). Both UDP and TCP/IP clients shall be supported.

2.4 Other Requirements

2.4.1 Performance

2.4.1.1 Processor Load

A single processor (STM32L475VG) utilization goal is:

- 75% or less peak load

CPU load / utilization will be monitored and reviewed during each software release cycle.

2.4.1.2 Memory Utilization

On chip memory (STM32L475VG) utilization goals are:

- Contiguous non-volatile memory, 80% or less
- Contiguous volatile memory (includes stack, heap), 75% or less

Memory utilization will be monitored and reviewed during each software release cycle.

2.4.2 Safety

Project is limited to monitoring sensors and uploading data to the cloud. No safety hazards are anticipated at this time that require special user or operate protection.

2.4.3 Security

No sensitive information will be exchanged under this proof of concept.

However, all IoT devices including this project shall implement the following security measures over Wi-Fi to comply with cloud storage:

- Connection to a cloud server is authenticated with X.509 certificate, the private key and root certificate are stored on this device.
- TLS - OSI layer 6, handles encryption/decryption of data exchanged with cloud server.

Information exchanged via the USB interface will be unsecured and readable using a simple text editor.

2.4.4 Software Quality Attributes

Availability: Checking that the system always has something to function and always pop up error messages in case of component failure. In that case the error messages appear when something goes wrong so to prevail availability problems.

Usability: Checking that the system is easy to handle and navigates in the most expected way with no delays. In that case the system program reacts accordingly and transverses quickly between its states.

Functionality: Checking that the system provide the right tools for editing question databases,creating session tests and analyzing the test sessions. In that case the tools that the Database editor provide are the ones that provide that attribute.

3 Glossary

AWS - Amazon Web Services

CDC - Communication Class Device

FS - Full Speed

IoT - Internet of Things

MQTT - Message Queuing Telemetry Transport, data message protocol

OSI - Open Systems Interconnection model

OTG - On the Go

PKCS #11 - defines a platform-independent API to cryptographic tokens,

Sockets - in general, specify an IP address and read/write data from the location

SPI - Serial Peripheral Interface

SRS - Software Requirements Specification

TLS - Transport Layer Security (encrypt/decrypt data)

USB - Universal Serial Bus

USB OTG - USB On The Go

VCP - virtual communication port

Wi-Fi - technology for radio wireless local area networking of devices based on the IEEE 802.11 standards. Wi-Fi is a trademark of the Wi-Fi Alliance

YAT - Yet Another Terminal