1. INTRODUCTION

1.1. PURPOSE

The purpose of this document is to identify the requirements of the Hardware Abstraction Layer (HAL) Framework version 1.0. The HAL Framework provides generic functionality across multiple embedded hardware platforms. This generic functionality can be selectively changed by user code, thus providing application specific software. This software framework will be included in all major software projects. This document will provide a complete set of definitions for the entire framework embedded software solution.

1.2. PROJECT SCOPE

The HAL Framework provides a generic interface that supports multiple hardware platforms. The major benefit to this is that projects utilizing the framework will only contain application specific code. Hardware can be interchanged without effecting how the program logically functions. If the hardware is supported by the framework, a user will be able to write an application using one specific set of hardware and port it to another set just by changing minor configuration and settings. The final goal of this framework is to provide enough flexibility to allow for support of multiple hardware components by different manufacturers.

Hardware abstraction will allow internal and external projects to get up and running faster and more reliably. Customers will have more time to develop their applications and will spend less time monotonously configuring board components such as the microcontroller and radio. This will lead to more productivity and higher customer satisfaction.

1.3. PRODUCT FEATURES

The HAL Framework provides an abundance of features that range from abstracting the hardware from a user's application to keeping memory requirements to a minimum. Since this framework is targeted towards embedded solutions, optimization must be in the forefront. Frameworks add overhead and this should to be kept to a minimum.

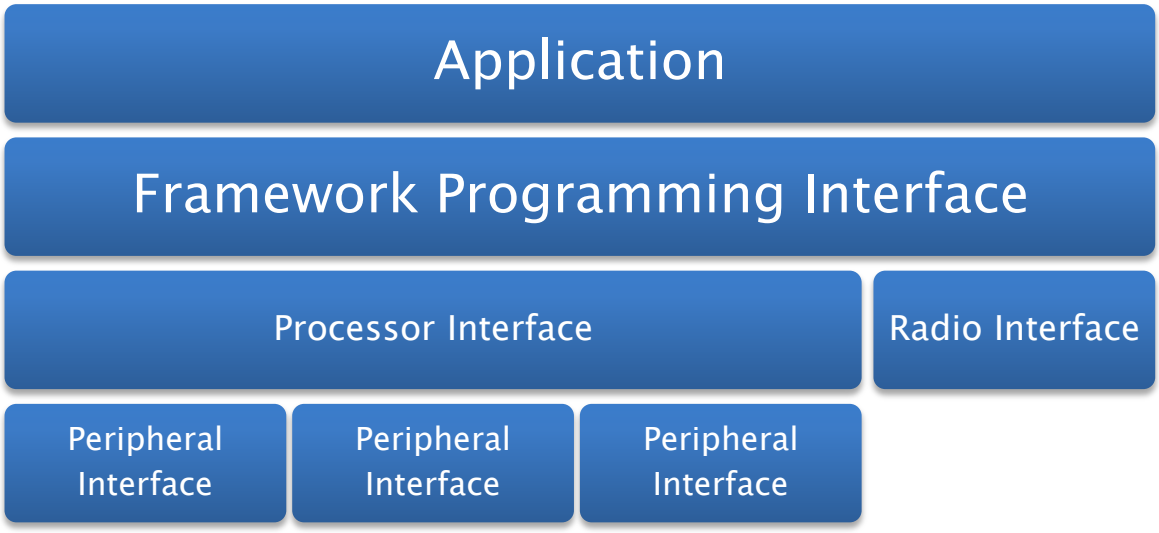

Figure 2-2. HAL Framework architectural overview.

The framework acts like a library to the user. It provides a public interface to communicate with all hardware supported.

The following diagram demonstrates how the HAL Framework would be included into an application project:

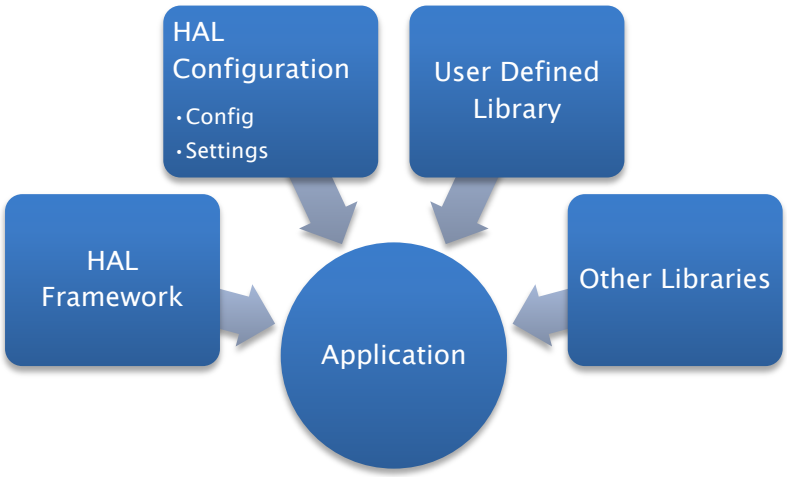

Figure 2-3. HAL Framework integration with application projects.

Below are the major product features that the HAL Framework provides.

1.3.1. Architecture of the framework is module based

Keeping the architecture module based allows for easy integration and unit testing. This increases reliability and helps to create a scalable product.

1.3.2. Support for multiple compilers

Multiple compilers are supported by the framework. The list of supported compilers will continue to expand as the framework is revised. This will be based on internal and external needs.

1.3.3. Support for multiple processor series chips and models

Multiple processor series are supported by the framework. The list of supported processor series and models will continue to expand as the framework is revised. This will be based on internal and external needs. The framework must therefore provide the capability of easily adding a new processor series or model when required. It must also be easy to create and include a new Peripheral Interface if one is not already provided for a processor peripheral.

1.3.4. Support for multiple radio series chips and models

Multiple radio series are supported by the framework. The list of supported radio series and models will continue to expand as the framework is revised. This will be based on internal and external needs. The framework must therefore provide the capability of easily adding a new radio series or model when required.

1.3.5. Support for other component chips and models

Other components may need hardware abstraction from the application by the framework. These components will be defined in the board definition with their appropriate connections to the processor running the framework. Some examples of other components include, but are not limited to, EEPROM, an LCD connected via SPI, or another processor connected by UART. These components will be supported based on the internal and external needs.

1.3.6. Provides a board definition

The framework will provide a board definition for each of the commonly used boards internally and with external customers. A customer may define their own board definition based on the strict set of rules set forth by the document, “PLACE DOCUMENT TITLE HERE”. *(TODO: Create HAL Framework Creating a Board Definition instruction document)*

1.3.7. Optimized for embedded systems

The target for this framework is an embedded system. Since embedded systems can potentially have tight memory restrictions, the framework must be optimized to create the least amount of overhead to the system.

1.3.8. Easily configurable using a set of configuration definitions and settings

A user will only need to change a configuration and settings file to begin utilizing the framework in their project. The user will never need to change any files located in the HAL Framework directory structure as these files are considered to be protected.

1.3.9. Provides well-defined Peripheral Interfaces for supported processors

The framework will include peripheral interfaces for supported processors. This list of supported peripherals will continue to expand as the framework is revised. This will be based on internal and external needs. A Peripheral Interface is used by the Processor Interface directly. No other interface may access the Peripheral Interface directly without having access to the Processor Interface. No application code may use the Peripheral Interface if full hardware abstraction is to be achieved.

1.3.10. Provides a well-defined Processor and Radio Interface

A Processor and Radio Interface must be defined and provided to the Framework Programming Interface (FPI), if one exists. If the FPI does not exist, for reasons of optimization, the user may make calls to these interfaces in their application. Each processor series will have its own implementation for the defined Processor Interface. Each radio series will have its own implementation for the defined Radio Interface.

1.3.11. Provides a well-defined Framework Programming Interface

A robust and well-defined Framework Programming Interface (FPI) will be provided to the user to perform all necessary hardware operations required in an application. It will simplify monotonous tasks for the user in an efficient manner without including application specific implementations.