Request for an Ethernet library to support MKR1000 using MKR ETH shield

# Introduction

This document was prepared in response to a suggestion. It is intended for review by Arduino development staff to provide support for future extensions to Arduino IoT Cloud solutions.

# Background

There are many industrial environments where WiFi is impractical or LAN connectivity is well established. Some examples of these environments are very diverse indeed from petrochemical plants to multi-storied buildings with robust infrastructure. In order to facilitate rapid deployment of Arduino boards it is desirable to leverage the existing infrastructure from simple proof-of-concept to full-fledged production use.

This approach has been proven in the use basic Arduino libraries by porting sketches from WiFi examples to Ethernet demonstrations.

# Goals/Objectives

The primary objective is to support the Arduino MKR ETH shield in Arduino IoT Cloud solutions. A key goal would be to support TCP and UDP connectivity along the same lines as supported presently with MKR1000 board for use external to Arduino IoT Cloud.

### Limitations to Goals/Objectives

The request is limited to support for MKR ETH shield. There are many chip-sets that may merit future consideration but since these chip-sets do not meet the security considerations for Arduino IoT Cloud, there is no requirement to support these chip-sets (*e.g.* Wiznet W5500 series with Arduino Nano).

# Use Cases

The following use cases are annotated below:

* Basic
* UDP
* MQTT

## Basic

The basic scenario requires support for:

* MKR ETH equivalent of the following WiFi instantiation:
	+ WiFiConnectionHandler ArduinoIoTPreferredConnection(SSID, PASS);
	+ ArduinoCloud.begin(ArduinoIoTPreferredConnection);

The assumption being that **ArduinoCloud** methods (*e.g.* update, printDebugInfo) would recognize the new interface. Any initialization required to use these methods would be the responsibility of user code (*e.g.* static IP address allocation or DHCP lease) subject to availability of corresponding methods/functions.

Currently a standard example has the following declaration:

WiFiClient client;

The ensuing client entity is used in the remaining code extensively. Changing the instantation to the following does not require complete change to the remaining code:

Ethernet client;

The thinking here is that Arduino IoT Cloud “*could*” support MKR ETH in a similar way. (pardon my ignorance here, please)

## UDP

There is a simple example in the Arduino Ethernet library that illustrates the use UDP for NTP synchronization. The author has embed the concepts from this example to support a Stratum peer server in a local infrastructure. While this use has no bearing on Arduino IoT Cloud directly it is hoped that UDP support should not be a handicap for any enhancements.

On a personal note, using a stationary GPS with PPS signal will improve time synchronization tasks in my infrastructure. I am using an Arduino board to leverage a low cost ublox GPS chip-set that does not have PPS signal. The IoT applications will benefit from this data.

### Code concepts

EthernetClient LANclient; // instantiate Ethernet network client

PubSubClient MQTTclient(LANclient); // partially initialized client instance

EthernetUDP Udp; // for NTP use to avoid TCP overhead

…

 Udp.beginPacket(address, 123); // send request using default port 123

 Udp.write(packetBuffer, NTP\_PACKET\_SIZE);

 Udp.endPacket();

## MQTT

For some sites it may be preferable to route messages to Arduino IoT Cloud through an on-site MQTT service broker. The library should permit the equivalent of the following declaration:

Ethernet LANclient;

PubSubClient MQTTclient(LANclient);

such that MQTTclient can be used freely for user’s custom purposes. There may not be a need to do anything here specific towards MQTT as long as Arduino IoT Cloud libraries support the instantiation of LANclient in the above example. In the simple case, interaction between Arduino IoT Cloud and a local MQTT service broker is *not* being requested (but may be something to think about for Edge Computing purposes for the roadmap).

# Acknowledgment

This document was prepared owing to a note from Alexander Etinger. Errors and omissions in this document arise solely from the author's limited experience in these matters.