Synchronous IoT Remote Control

with Arduino MEGA and AppMyProduct

By **Carsten Gregersen**, Nabto

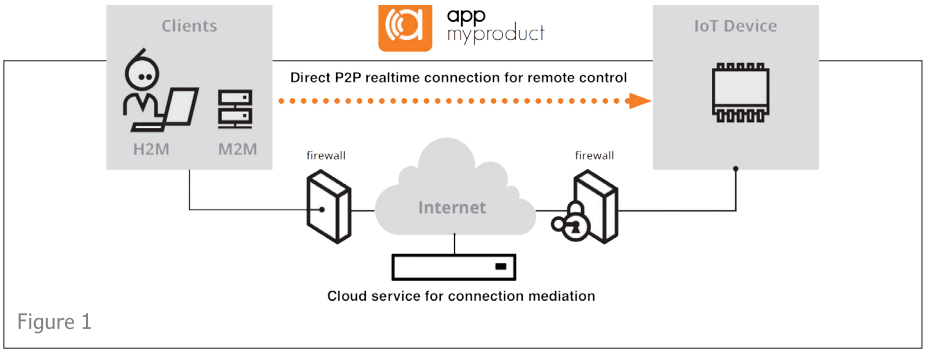

Figure 1

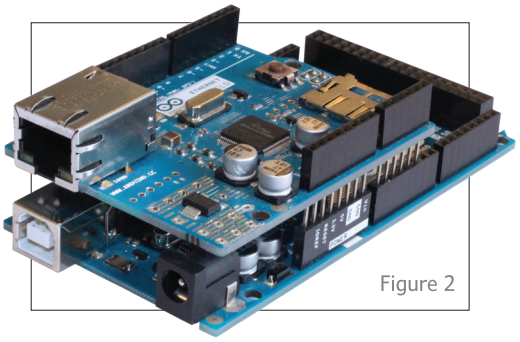

Figure 2

One framework to support synchronous "realtime" operation for IoT is AppMyProduct. It focuses on delivering an easy way to remote control IoT devices directly via apps on smartphone and tablet (**Figure 1**). One such use case could be turning your summerhouse heating on and off remotely. Or viewing a live video stream from a camera, although that's much more complex and out of scope of this brief article.

Remote-control an Arduino MEGA board + Ethernet shield

The Arduino Mega 2560 has 54 digital input/output pins, 16 analogue inputs, 4 UARTs, a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It is not as widely deployed as the Arduino UNO boards, but for our IoT project which requires both an IP stack and encryption, the extra-large Flash Memory of 128 KB/256 KB and extra 8 KB SRAM comes in handy.

AppMyProduct is built on the Nabto software stack. You can sign-up to the service for free and get 10 IoT device ids and matching keys for free. This will enable you to deploy and remote control 10 IoT devices using the platform. Three 'adapters' needs to be plugged into the framework. A UDP stack adapter needs to be interfaced with the IP stack of the native platform. A DNS adapter needs to be interfaced with the platform DNS resolver service. The framework also needs to be able to make decisions based upon the time.

With the above mentioned hardware ready and Arduino IDE installed on your PC you need to sign-up with the AppMyProduct.com SaaS service. Next, create a product, for example "MyTestProduct". Don't register as a "Free Product", since you currently can get up to 10 production licenses for free. Next, Generate licenses by clicking the "Generate licenses" button. Then go to the new domain device list. Note one of the "Device IDs" and the matching "License Key" needed later (copy it to a text file on your computer or similar).

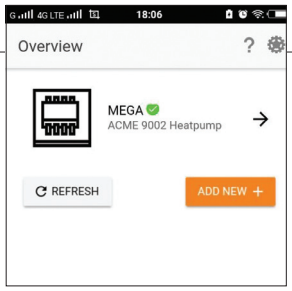

Figure 3

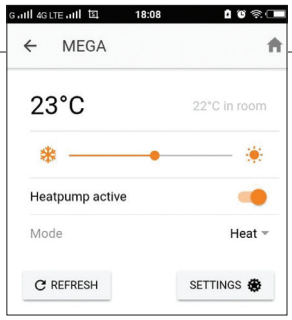

Figure 4

Using the s/w adapters

The adapters come ready programmed. Fetch them from: https://github.com/nabtodaemon/heatcontrol-arduinomega. Download the .zip file for the Arduino IDE. Next add the .zip file to the IDE and start. Add the library to the Arduino IDE via these steps: Sketch → Include Library → Add .ZIP Library. Browse to the folder containing the downloaded library file and add the unabto-arduinomega-sdk-2.1.1.zip. Open the HeatPump.ino example like so: File → Examples → Nabto-Mega2560 → HeatPump.

The sample sketch includes the Nabto class, which encapsulates the Nabto setup. For the sketch to work, the below settings are to be changed. The setting should specify the board's MAC address (found on the Ethernet board) followed by the unique Device ID and pre-shared key of the device obtained from: portal.appmyproduct.com

You need to adjust the line:

```
// Enter device id and pre-shared
key from portal.appmyproduct.com
const char* DEVICE_ID = "abc.xyz.
appmyproduct.com";
const char* PRE_SHARED_KEY =
"4f2a03f29f509035c03bc229ae870849";
```

with the device id and key that you noted from the AppMyProduct portal.

After compiling and uploading your HeatPump sketch to the Arduino Mega, it establishes a connection to your Ethernet network and starts the uNabto server. In operation the system shows screens like in Figures 3 and 4. Now you can start your journey to the summerhouse.

Web Links

App for iOS devices: https://itunes.apple.com/dk/app/appmy-product-heat-control-client/id1213447922

App for Android devices: https://play.google.com/store/apps/details?id=com.ionicframework.nabtoionicstarter

App sourcecode: https://play.google.com/store/apps/details?id=com.ionicframework.nabtoionicstarter