Freescale Semiconductor
Application Note

Document Number: AN4417
Rev. 0, 12/2011

MCU control and monitor with Twitter

by: Carlos Musich, Luis Garabito

Contents

1 Introduction

This application note is intended to demonstrate http client implementation using Freescale MQX RTOS capabilities.

The hardware used to illustrate this is the TWR-K60N512-KIT. The remote controlling and monitoring have become a need rather than an option in the embedded world. This application note takes advantage of two social media interfaces for these purposes. One is used to enter commands to the MCU; http://twitter.com/. The other one is used to pull out data from the MCU; https://www.supertweet.net/.

It is important to focus in the fact that with these methods the MCU is reachable through Internet without the need of a public IP address or without mounting a HTTP server in the MCU.

The application source code described in this document can be found in the AN4417SW.zip file. For a full description of Freescale MQX RTOS, please visit https://www.freescale.com

For more information about HTTP standard, refer to Reference 1. For more information about SuperTweet.net API, refer to Reference 2.

2 Overview

The examples developed to illustrate the http client functionality are divided into two applications. The first application shows the user how to read text from a Twitter's account. The second application explains how to send data to be published in a specific Twitter's account by using a third service; SuperTweet.net.

3 Enter commands to the MCU

3.1 Architecture

This application defines two tasks in MQX. The first task is main. It is meant to configure GPIO, the RTCS and create the second task. The name of this second task is httpclient. The purpose of this task is to carry out the communication with the Twitter server and read the commands to be executed. To retrieve the input commands, the httpclient task reads them from the last tweet published by a specific Twitter account.

The command then is parsed and executed according to the implementation. The main task enters into an infinite loop where the httpclient is restarted in each loop to allow a cycle behavior for reading commands. The time for each loop is controlled by a sample rate value that can be configured by the user via a command. The figure 1 shows the flow chart of each one of the tasks.

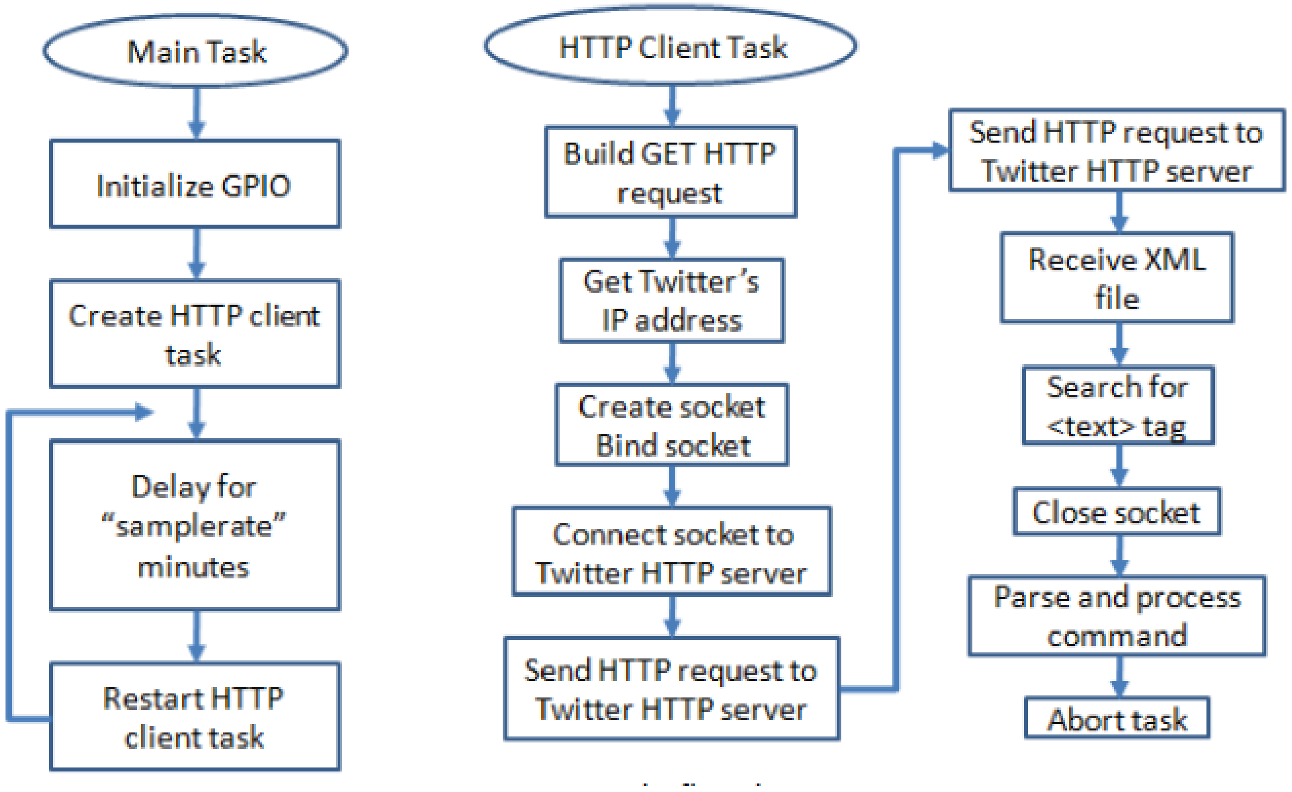
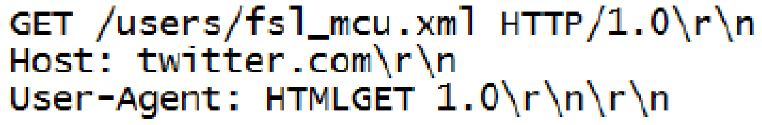

Main Task
Initialize GPIO
Create HTTP client task
Delay for "samplerate" minutes
Restart HTTP client task
HTTP Client Task
Build GET HTTP request
Get Twitter's IP address
Create socket Bind socket
Connect socket to Twitter HTTP server
Send HTTP request to Twitter HTTP server
Send HTTP request to Twitter HTTP server
Receive XML file
Search for <text> tag
Close socket
Parse and process command
Abort task

Figure 1. Tasks Flowchart

3.2 Implementation

First, the GPIO is setup to control the 4 LEDs in the TWR-K60N512-KIT. The LWGPIO driver is used to do this. These LEDs turn on and off by the commands read from the Twitter account. After the GPIO is initialized the main task calls the function InitializeNetworking() to initialize the RTCS. It can be configured to set a static IP address or use the DHCP client to get a dynamic IP address. The next step is to create the httpclient task.

The httpclient task creates a TCP socket that uses the port 80 which belongs to the HTTP protocol. This socket connects to the Twitter server. To get the latest tweet is necessary to build a HTTP request. The request must be build like the Figure below shows.



```
GET /users/fsl_mcu.xml HTTP/1.0\r\n
Host: twitter.com\r\n
User-Agent: HTMLGET 1.0\r\n\r\n
```

Figure 2. Get HTTP request

The HTTP protocol is described by the Request for Comments (RFC) 2616. Following is a description of each of the headers in the HTTP request from figure 1 according with the RFC 2616.

GET /users/fsl_mcu.xml HTTP/1.0 -- The GET method means retrieve whatever information is identified by the Uniform Resource Locators (URL). In this case the URL retrieves the fsl_mcu.xml file. The HTTP protocol version is also set in this header. The protocol HTTP/1.0 is used.

Host: twitter.com -- The Host request header field specifies the Internet of the resource being requested. The Host field value must represent the naming authority of the origin server given by the original URL. This allows the origin server or gateway to differentiate between internally ambiguous URLs, such as the root "/" URL of a server for multiple host names on a single IP address.

User-Agent: HTMLGET 1.0 -- The User-Agent request header field contains information about the user agent originating the request. The field can contain multiple tokens and comments identifying the agent and any sub-product. In this case a simple token is used to identify the application; name “HTMLGET” and version “1.0”.

The Twitter service provides a XML file that can be consulted to get information from an user’s account. A tweet from the Twitter account “fsl_mcu” is used to represent the input commands to the MCU. The file fsl_mcu.xml consists of a series of data from the Twitter account “fsl_mcu”. The <text> tag contains the latest tweet the user sent to the Twitter service. This tag can be found in the XML file.

To get the XML file it is necessary to establish a TCP connection to the port 80 of the Twitter http server. The HTTP request described in figure 2 is sent to the http server and the result is the XML file. The content of the file is analyzed while it is received. Once the <text> tag is detected the analyze finishes and the tag with its content is sent to the parser for the evaluation and execution.

The command has a specific structure. It must start with a period (.) that indicates the beginning of the command. The first word has to be “spark”. This tells to the parser that it’s a command that must be evaluated. Another period is needed to separate the following indicator. The application implements only two objects; “led” and “samplerate”.

The “led” objects receive two parameters. The first one indicates the number of the LED to control. The possible values go from 1 to 4. The second parameter indicates the new state of the LED. Only two values are valid; “on” and “off”. These values indicate the MCU to turn on or off a specific LED. An example of this command is as follows:

“.spark.led.1.on.”

The “samplerate” object implementation is used to modify the rate of occurance at which the application goes and checks for another command. It receives only one parameter. This parameter is the regularity at which the microcontroller reads a new command. The time is given in minutes. The default sample rate is 5 minutes.

An example is shown below:

".spark.samplerate.5."

As the example shows; the periods (.) are the tokens used to parse the command string. It is necessary that each command starts and ends with a period (.).

The commands set can be increased. The implementation can be done as simple or as complex as the design requires it. These two examples are intended to demonstrate how to create commands that can be executed by the MCU.

4 Pull out data from the MCU

4.1 Architecture

This application defines two tasks in MQX. The first task; main, is meant to configure GPIO, the RTCS and create the second task. The name of this second task is httpclient. The purpose of this task is to carry out the communication with the Twitter server to post a tweet. To accomplish this task, a third element is used to reach Twitter server;https://www.supertweet.net/.

The SuperTweet.net service offers an alternative for TCPIP stacks that does not support Secure Sockets Layer (SSL) mechanism. The HTTP Basic Authentication is used to log in SuperTweet.net server. First, it is necessary to Sign-up with Twitter credentials to authorize the MyAuth API Proxy SuperTweet.net application to access a Twitter account. Next, chose a new password (not the real Twitter password) that the applications can use with the http://api.supertweet.net API. After these steps are completed it is possible to have MQX opening a communication with Supertweet.net and then be able to post a tweet in Twitter's account.

The main task enters into an infinite loop where the httpclient is restarted in each loop to allow a cycle behavior for publishing information. The time is controlled by a sample rate value that can be configured by the user. The figure 3 shows the flow chart of each one of the tasks.

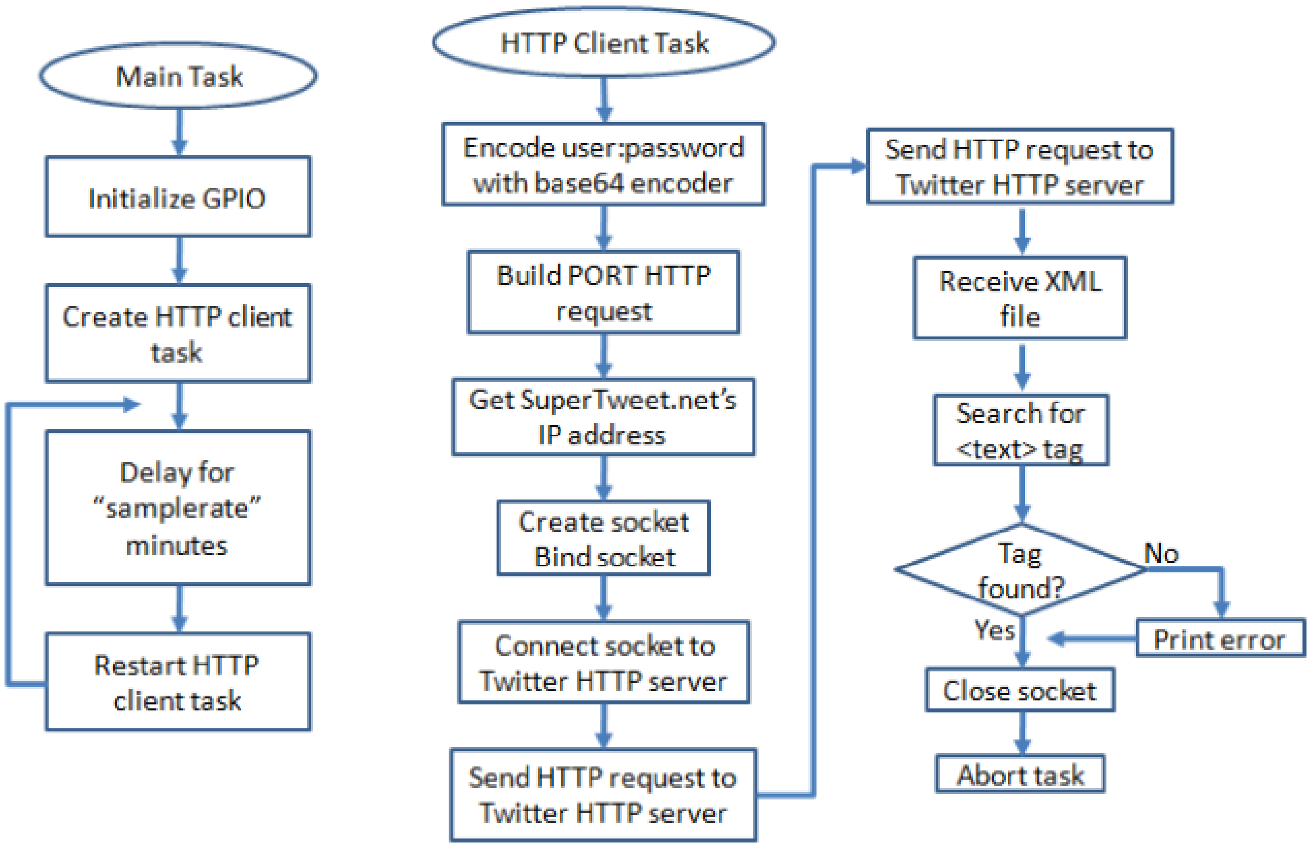
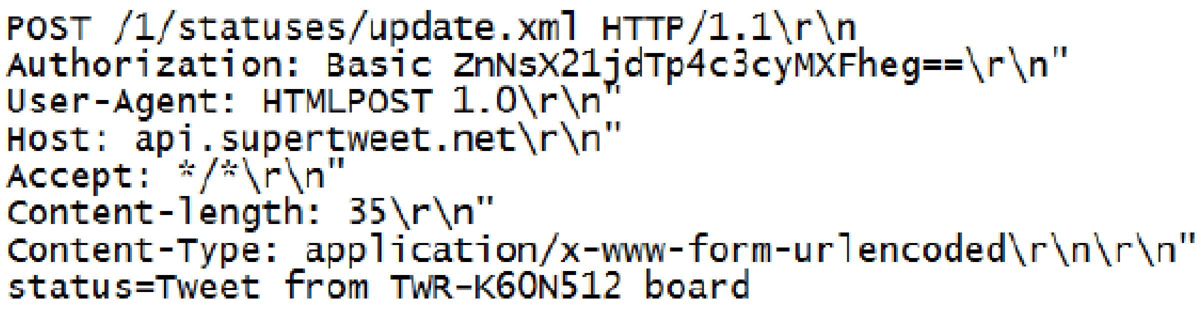

Main Task
Initialize GPIO
Create HTTP client task
Delay for "samplerate" minutes
Restart HTTP client task
HTTP Client Task
Encode user:password with base64 encoder
Build PORT HTTP request
Get SuperTweet.net's IP address
Create socket Bind socket
Connect socket to Twitter HTTP server
Send HTTP request to Twitter HTTP server
Send HTTP request to Twitter HTTP server
Receive XML file
Search for <text> tag
Tag found?
No
Yes
Print error
Close socket
Abort task

Figure 3. Tasks Flowchart

4.2 Implementation

The main task calls the function InitializeNetworking() to initialize the RTCS. It can be configured to set a static IP address or use the DHCP client to get a dynamic IP address. The next step is to create the httpclient task.

The httpclient task creates a TCP socket that uses the port 80 which belongs to the HTTP protocol. This socket connects to the SuperTweet server. The SuperTweet.net API provides a method to take advantage of Twitter's OAuth authentication technology, without the cost and complexity of OAuth,in a simple tweeting application. The SuperTweet.net Twitter MyAPI Proxy implements some of the Twitter API. To post a tweet in a Twitter account it is necessary to build a HTTP request and POST it to the SuperTweet server. The request must be build like the Figure below shows.



```
POST /1/statuses/update.xml HTTP/1.1\r\n
Authorization: Basic ZnNsX21jdTp4c3cyMXFheg==\r\n"
User-Agent: HTMLPOST 1.0\r\n"
Host: api.supertweet.net\r\n"
Accept: */*\r\n"
Content-length: 35\r\n"
Content-Type: application/x-www-form-urlencoded\r\n\r\n"
status=Tweet from TWR-K60N512 board
```

Figure 4. Post HTTP Request

Following is a description of each of the headers in the HTTP request from the figure 4 according with the RFC 2616.

POST /1/statuses/update.xml HTTP/1.1 -- The POST method is designed to allow a uniform method to provide a block of data, such as the result of submitting a form to a data-handling process. For the application purpose the POST is using the API “/1/statuses/update.xml”. This API is used to tweet in a Twitter account.

Authorization: Basic ZnNsX21jdTp4c3cyMXFheg== -- The basic access authentication is a method for an http client application used to provide the user name and password for an account when making a request. The user name and password are concatenated with a colon. The resulting string is encoded using the Base64 algorithm. For instance, with the user name 'fsl_mcu' and password 'xsw21qaz', the string 'fsl_mcu:xsw21qaz' is encoded with the Base64 algorithm. The result is 'ZnNsX21jdTp4c3cyMXFheg=='. The USER and PASSWORD encoded here are the one created for the SuperTweet.net account. The Table 1 shows in details the previous explanation.

Table 1. Base64 Encoding

User	Password	Before Base64 Encoding	After Base64 Encoding
fsl_mcu	xsw21qaz	fsl_mcu:xsw21qaz	ZnNsX21dTp4c3cyMXFheg==

User-Agent: HTMLPOST 1.0 -- The User-Agent request header field contains information about the user agent originating the request. The field can contain multiple tokens and comments identifying the agent and any sub-product. In this case a simple token is used to identify the application; name “HTMLPOST” and version “1.0”.

Host: api.supertweet.net -- The Host request header field specifies the Internet of the resource being requested. The Host field value must represent the naming authority of the origin server given by the original URL. This allows the origin server or gateway to differentiate between internally ambiguous URLs, such as the root "/" URL of a server for multiple host names on a single IP address.

Accept: */* -- The Accept request-header field can be used to specify certain media types which are acceptable for the response.

Content-length: 35 -- The Content-Length header field indicates the size of the body message to be post. It is given in decimal number. Content-Type: application/x-www-form-urlencoded --

The Content-Type header field indicates the media type of the body sent to the recipient.

status=Tweet from TWR-K60N512 board -- This is the string that is sent to the API “/1/statuses/update.xml”. This text represents a command that indicates which is the tweet to publish in the Twitter account. This string is defined by the user in the http request.

The Twitter returns an XML file that includes the return code for the API executed. If the <text> tag is there then it is an indication that the tweet posting was successful. This tag returns the string sent to be tweeted.

5 Conclusion

The application note explains the basics for an application that reads commands and provides feedback in the same or different Twitter account. This document shows a method that can set a base for many applications that require remote control and monitoring through Internet.

6 References

Hypertext Transfer Protocol -- HTTP/1.1: https://www.w3.org/Protocols/rfc2616/rfc2616.html

Supertweet.net API:https://www.supertweet.net/about/documentation

How to Reach Us:

Home Page:
www.freescale.com

Web Support:
http://www.freescale.com/support

USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
+1-800-521-6274 or +1-480-768-2130
www.freescale.com/support

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
www.freescale.com/support

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor China Ltd.
Exchange Building 23F
No. 118 Jianguo Road
Chaoyang District
Beijing 100022
China
+86 10 5879 8000
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
1-800-441-2447 or +1-303-675-2140
Fax: +1-303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Document Number: AN4417
Rev. 0, 12/2011