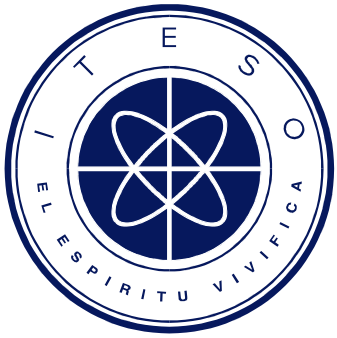
INSTITUTO TECNOLÓGICO Y DE ESTUDIOS SUPERIORES DE OCCIDENTE

Reconocimiento de validez oficial de estudios de nivel superior según acuerdo secretarial 15018, publicado en el Diario Oficial de la Federación el 29 de noviembre de 1976.

Departamento de Electrónica, Sistemas e Informática

ESPECIALIDAD EN SISTEMAS EMBEBIDOS



Control Remoto Sobre Ethernet a través de Twitter

Trabajo final que para obtener el diploma de

ESPECIALISTA EN SISTEMAS EMBEBIDOS

Presenta: CARLOS ALBERTO MUSICH CUEVAS

Asesor: LUIS ENRIQUE GARABITO SIORDIA

Tlaquepaque, Jalisco, Mayo de 2017.

AGRADECIMIENTOS

El autor agradece sinceramente a las siguientes personas e instituciones:

- A Dios por brindarme la oportunidad de superar este reto y completar mis estudios de posgrado en esta institución;

- A mi padre y a mis tías Lucrecia, Esperanza y Martha, por ser el pilar fundamental en todo lo que soy, en toda mi educación, tanto académica, como de la vida y por su incondicional apoyo, a mi hermana Carla por ser mi compañera más querida en cada momento de mi vida y a mi esposa por su apoyo incondicional. Todo este trabajo ha sido posible gracias a ellos;

- Que nos apoyamos mutuamente durante esta especialidad y que seguimos siendo amigos. Con especial mención a Paolo Alcántara y Luis Chaídez;

- Al ITESO por proporcionar las instalaciones, equipos y laboratorios, además del seguimiento y apoyo de los profesores del programa;

- Finalmente, pero no menos importante, a mi asesor Luis garabito, por el apoyo técnico en la resolución del problema de mi trabajo de grado, así como en la revisión de este documento.

RESUMEN

Hoy en día es posible acceder a cualquier tipo de información pública, comunicarnos con cualquier persona, revisar cuentas de banco, hacer compras, entre muchas otras cosas, todo esto a través de Internet, que se ha vuelto la fuente más popular de información y de comunicación. A su vez, dentro del mundo creado por la Internet, el uso de las redes sociales ha incrementado de manera exorbitante es por esto que para el desarrollo de esta aplicación se toma ventaja de dicha tendencia.

Este trabajo se enfoca en el diseño e implementación de un sistema de control y monitoreo remoto utilizando la red social Twitter. Se utiliza la tarjeta de desarrollo TWR-K60N512 de Freescale la cual se conecta al servidor de Twitter utilizando el paquete de desarrollo de TCP sobre IP que provee el sistema operativo en tiempo real MQX. El microcontrolador hace peticiones periódicas de HTTP (GET) para leer el último mensaje de la cuenta de twitter FSL_K60 que fue creada específicamente para este proyecto. La aplicación interpreta el mensaje de texto, y si este mensaje es un comando conocido por el sistema, se ejecuta alguna acción.

Así mismo se hacen peticiones HTTP (POST) para enviar un mensaje desde la misma cuenta cada que se presiona un botón en la tarjeta o cada que el potenciómetro integrado a esta misma tarjeta cambia de valor.

La primera parte de esta tesis establece los objetivos y el marco teórico, así como el estado del arte de las soluciones actuales. La segunda parte muestra los detalles de la implementación de la solución embebida y finalmente la tercera parte se refiere al trabajo futuro.

ABSTRACT

This work obeys to the crescent need on communicating, having access to information and controlling everything in any moment taking advantage of the world tendency of social networks. This document tries to address an effective implementation of an embedded control system, as part of a major system, to control lights and motors and monitoring sensors simulating an automated building.

The development was made using TWR-K60N512 board from Freescale which connects to the Twitter server using TCP/IP stack provided by MQX RTOS. The microcontroller performs periodic HTTP (GET) requests to read the last message in the Twitter account FSL_K60 which was created specifically for this project. The application parses the text message, if this message is a command known by the system, some action is executed.

It is also makes HTTP (POST) requests to send a message from the same account each time an on-board button is pressed or each time that the on-board potentiometer changes its value.

First part of this Thesis states the objectives and theoretical framework as well as state of the art of existing solutions. Second part will get into details of the embedded system implementation scoping the hardware proposal, as well as the technical Embedded Software Engineering, design, validation and code implementation phases, and finally a third part will focus on the future work.

CONTENIDO

LISTA DE FIGURAS

LISTA DE TABLAS

ACRONIMOS Y SIMBOLOS

TCP	Protocolo de Control de Transferencias
IP	Protocolo de Internet
HTTP	Protocolo de Transferencia de Hyper Texto
K60	Microcontrolador de la familia Kinetis
TWR-K60N512	Tarjeta de desarrollo Tower Sytem con microcontrolador K60
MQX	Ejecutor de Colas de Mensajes
RTOS	Sistema Operativo de Tiempo Real
RTCS	Conjunto de Soluciones para Comunicaciones en Tiempo Real
GET	Petición para obtener información de Internet
POST	Petición para enviar información de Internet
USB	Puerto Serial Universal
WiFi	Marca registrada de mecanismo de conexión inalámbrica
ARM	Maquina RISC Avanzada
AVR	Marca registrada de Microcontroladores de Atmel
LWGPIO	Controlador de pines de propósito general de entrada y salida
RFC	Requisición de Comentarios
URL	Localizador de Recursos Uniforme
XML	Lenguaje de Marcado Extensible
API	Interfaz de Programación de Aplicaciones
RTCS	Conjunto de Soluciones para Comunicaciones en Tiempo Real

1. INTRODUCCION

En la actualidad existe una infinidad de sistemas que se pueden controlar y monitorear de manera remota, ya sea en la vida cotidiana o en la industria. Conforme pasa el tiempo esta situación va pasando de ser una posibilidad a una necesidad.

Hoy en día las redes sociales son un tema muy popular y gran parte de la población está familiarizada con ellas. Este trabajo se enfoca en el diseño e implementación de un cliente HTTP embebido en un microcontrolador de 32 bits con la finalidad de controlar y monitorear un sistema a través de Ethernet. En particular, el cliente se conecta al servidor de la red social Twitter la cual se utiliza como plataforma para el sistema de control y monitoreo.

Este trabajo Puede ser ubicado en el área industrial y de consumidores.

Las siguientes líneas describen como se organiza el resto de este documento.

El Capítulo 1 describe el planteamiento del problema y objetivos del trabajo, así como su justificación.

El capítulo 2 menciona antecedentes y los trabajos realizados en este tema.

El Capítulo 3 ofrece una descripción funcional de sistema y la fundamentación teórico/técnica sobre los modelos cliente servidor y como aplican estos conceptos para trabajar con la red social Twitter en este proyecto. También presenta la arquitectura de la solución al problema, describiendo tanto los elementos de hardware, de firmware y software. En este capítulo también se detallan aspectos de la implementación de la solución como los diagramas de flujo de las tareas que se utilizan en la aplicación y la estructura de las peticiones de HTTP hacia el servidor de Twitter.

El Capítulo 4 describe las herramientas de implementación usadas para la solución.

El Capítulo 5 contiene las pruebas aplicadas al sistema y sus resultados.

El Capítulo 6 incluye las conclusiones y un panorama general del trabajo futuro de este proyecto de grado.

1.1. Introducción al problema y su contexto

El problema abordado en este trabajo de grado consiste en la implementación de un cliente HTTP embebido en in microcontrolador de 32 bits que se conecte al servidor de la red social Twitter con la finalidad de controlar y monitorear un sistema simulando un edificio automatizado.

1.2. Justificación/Motivación

Ya que las redes sociales se han vuelto parte de la vida cotidiana de las personas, aprovecharse de su uso es una buena oportunidad para llevar el servicio más allá de la comunicación entre los usuarios. Por otro lado, la facilidad, conveniencia e incluso necesidad de controlar y monitorear nuestra casa desde cualquier en el que estemos es lo que da vida a esta propuesta que incluye la posibilidad de encender y apagar las luces de un edificio, abrir y cerrar puertas o ventanas además de tener el conocimiento de lo que está pasando en el edificio, todo a través de una cuenta personal de la red social Twitter.

1.3. Problema a resolver

El problema específico que se pretende resolver consiste en crear una aplicación embebida en la que un cliente que este leyendo la cuenta de un usuario de Twitter y que pueda interpretar los mensajes enviados a esta cuenta, de modo que al recibir un mensaje que pueda ser decodificado como un comando se ejecuten acciones específicas. Por otro lado, el cliente debe ser capaz de recibir estímulos externos y enviar un mensaje a la cuenta de Twitter para reportar dichos estímulos.

1.4. Objetivos

El objetivo general de esta tesis es implementar un sistema que sea capaz de controlar algunos dispositivos como motores de corriente directa y bombillas de 60 Watt. Así mismo el sistema debe monitorear el estado de algunos sensores y atender interrupciones aperiódicas como el presionar un botón.

El objetivo específico es montar un sistema de control sobre la red social Twitter con la finalidad de obtener las siguientes ventajas:

- No se necesita la implementación de una interfaz de monitoreo.
- No se necesita montar un servidor.
- No necesitamos una dirección IP pública.

Facilidad de uso. Cualquier usuario de Twitter puede tener un sistema de control y monitoreo.

2. ANTECEDENTES

Resumen: En este capítulo se presenta los antecedentes del problema a resolver, así como su contexto

1.5. Antecedentes y Preliminares

La industria electrónica se mueve a un paso muy veloz. Para los diseñadores y fabricantes de dispositivos electrónicos es imprescindible contar con herramientas que les permitan desarrollar sus proyectos de la manera más rápida y sencilla, ya que el tiempo que tarda en salir un producto al mercado desde el momento de su concepción es sumamente crítico para ser competitivo.

La intención del proyecto Control Remoto sobre Ethernet a través de Twitter es que sea un modelo de referencia para el mercado industrial y de consumidores. Éste, al ser capaz de controlar y monitorear varios dispositivos de forma remota a través de Ethernet y permitir portabilidad a WiFi, puede ser utilizado por diseñadores y tomarlo como referencia para desarrollar aplicaciones más complejas sin tener que partir de ceros. A su vez la compañía Freescale puede usar este sistema para demostrar las capacidades que ofrecen estas herramientas.

1.6. Trabajos Relacionados

En la actualidad existen aplicaciones similares. Dentro de la investigación que se realizó, las aplicaciones desarrolladas con las plataformas Arduino[3] y Raspberry Pi[4] son las más populares, sin embargo, hay significativas diferencias y ventajas en la forma en que está desarrollada esta aplicación. En principio las plataformas Arduino y Raspberry Pi fueron desarrolladas con el objetivo de estimular la enseñanza en ciencias de la ingeniería y computación, por lo que ninguna de las dos se consideran plataformas de desarrollo para productos comerciales. Hablando ya específicamente de Raspberry Pi, utiliza procesadores de Broadcom basados en ARM11 lo que implica un costo mucho más elevado, mayor consumo de potencia y la necesidad de un sistema operativo basado en Linux ya que su implementación está hecha en Python que es un lenguaje de alto nivel. Por otro lado, la implementación en Arduino es mucho más similar a la del proyecto que se describe en este documento ya que Arduino se basa en un microcontrolador AVR de Atmel que se puede considerar dentro de la misma gama de los microcontroladores Kinetis, sin embargo, utiliza una librería específica para publicar mensajes en Twitter lo cual limita el alcance de la aplicación y su flexibilidad pues no tiene disponible ningún servicio genérico de TCP/IP a nivel de usuario.

Tomando lo anterior en cuenta se puede considerar que la integración de las herramientas y los servicios utilizados en este proyecto es una combinación optima y única ya que incluye el equilibrio entre precio, procesamiento y consumo de potencia que ofrece un microcontrolador Kinetis, más el

alcance que ofrece el sistema operativo en tiempo real MQX con su librería de TCP/IP para manejar libremente la comunicación por Ethernet y la facilidad de uso de un tercer servicio llamado SuperTweet para poder encriptar la información y enviar un mensaje al servidor de Twitter. Todo esto está desarrollado en la tarjeta de evaluación TWR-K60N512 que es una de las muchas tarjetas que Freescale manufactura para que sus clientes implementen prototipos de sus aplicaciones finales.

3. DISEÑO PROPUESTO

Resumen: En este capítulo se presentan el diseño propuesto como solución al problema de estudio en este trabajo.

1.7. Visión Global del Sistema

Para la solución al problema planteado en este trabajo se realizó el diseño e implementación de un cliente HTTP utilizando el conjunto de soluciones de TCP/IP que provee el Sistema Operativo en Tiempo Real MQX. El desarrollo de esta implementación se describe en las siguientes sub secciones.

1.8. Fundamentación teórica

La aplicación será desarrollada utilizando la red social Twitter. Esta permite mandar mensajes de texto plano de bajo tamaño con un máximo de 140 caracteres, llamados tweets, que se muestran en la página principal del usuario. Los usuarios pueden enviar un mensaje de Twitter (Tweet) desde una página web, desde aplicaciones para teléfonos inteligentes, o mediante el Servicio de mensajes cortos (SMS) disponible en ciertos países.

La interfaz web de Twitter está escrita en Ruby on Rails, y los mensajes se mantienen en un servidor que funciona con software programado en Scala y además dispone de una API abierta para todo tipo de desarrolladores, lo cual supone una gran ventaja para todos aquellos que quieran integrar Twitter como un servicio tanto en otras aplicaciones web como en aplicaciones de escritorio o móviles.

El protocolo que se utilizara es TCP sobre IP por lo cual se requiere un conjunto de soluciones que provea soporte de estos protocolos. La familia de protocolos de Internet es un conjunto de protocolos de red en los que se basa Internet y que permiten la transmisión de datos entre computadoras. En ocasiones se le denomina conjunto de protocolos TCP/IP, en referencia a los dos protocolos más importantes que la componen: Protocolo de Control de Transmisión (TCP por sus siglas en inglés) y Protocolo de Internet (IP), que fueron dos de los primeros en definirse, y que son los más utilizados de la familia. Existen tantos protocolos en este conjunto que llegan a ser más de 100 diferentes, entre ellos se encuentra el popular protocolo HTTP, que es el que se utiliza para acceder a las páginas web.

En el microcontrolador se desarrollará una aplicación que consiste en un cliente http para acceder a Twitter. La arquitectura cliente-servidor es un modelo de aplicación distribuida en el que las tareas se reparten entre los proveedores de recursos o servicios, llamados servidores, y los demandantes, llamados clientes. Un cliente realiza peticiones a otro programa, el servidor, que le da respuesta. Esta idea también se puede aplicar a programas que se ejecutan sobre una sola computadora, aunque es más ventajosa en un sistema operativo multiusuario distribuido a través de una red de computadoras.

1.9. Diagrama a Bloques (Diagrama Funcional)

La Figura 1 muestra la arquitectura de solución propuesta en este trabajo. El usuario puede conectarse desde cualquier dispositivo a su cuenta de Twitter. El bloque compuesto por el K60 con MQX embebido se encarga de leer e interpretar los mensajes enviados por el usuario. Utilizando comandos predefinidos el usuario puede controlar una bombilla de 60 Watts y un motor de corriente directa de 12V interfazados con el microcontrolador a través de una etapa de potencia. También se cuenta con un potenciómetro y un botón para obtener entradas asíncronas las cuales son notificadas al usuario a través de la misma cuenta de Twitter.

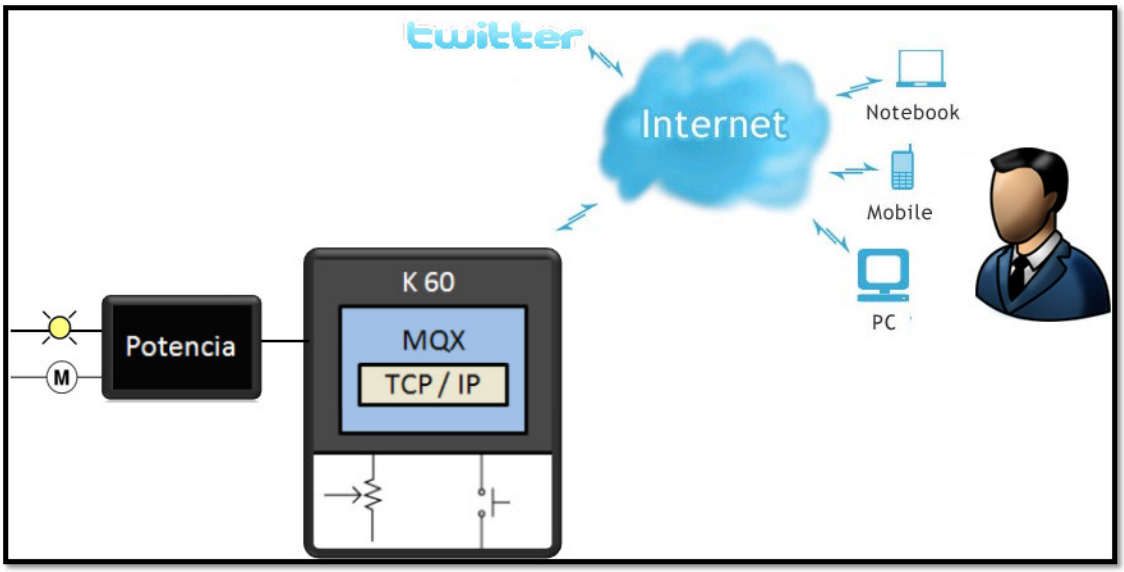


Ilustración 1. Diagrama de bloques

1.10. Arquitectura de Software

Tarea Main

Inicialmente, los pines de propósito general de entrada y salida son configurados para controlar 4 diodos luminosos en la tarjeta TWR-K60N512-KIT. El driver de LWGPIO es utilizado para este fin. Estos diodos luminosos se encenderán y apagarán según los comandos leídos desde la cuenta de Twitter. Después de que los pines de propósito general de entrada y salida son inicializados la tarea Main llama a la función InitializeNetworking() para iniciar servicio de TCP/IP provisto por la librería RTCS de MQX. Este puede ser configurado para fijar una dirección de IP estática o utilizar DHCP para obtener una dirección IP dinámica.

Después de concluirse las inicializaciones, la tarea principal entra en un ciclo infinito donde se monitorean el convertidor analógico a digital y el botón, a su vez se lleva la cuenta del tiempo para iniciar periódicamente la tarea HTTP_Client_Get.

La figura 2 muestra el flujo de la tarea principal.

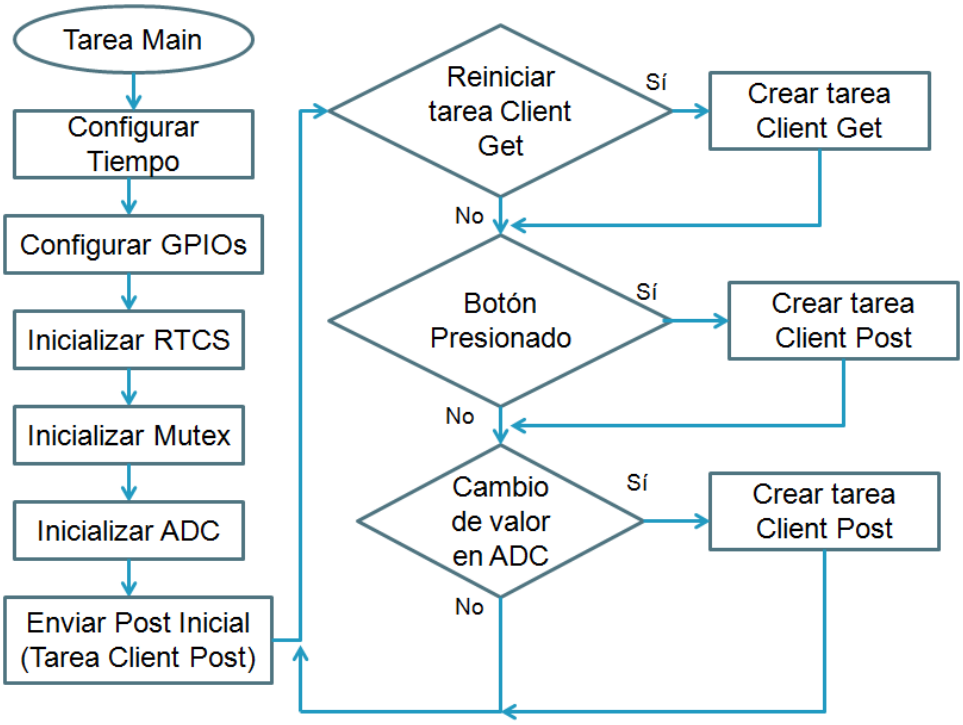


Ilustración 2. Tarea Main

Cliente HTTP Get

El propósito de esta tarea es llevar a cabo la comunicación con el servidor de Twitter y leer los comandos a ser ejecutados. Para obtener los comandos de entrada la función httpclient() los lee desde el ultimo tweet publicado por una cuenta especifica de Twitter. El comando es después analizado y ejecutado según la implementación.

El tiempo para reiniciar la tarea es controlado por un valor que puede ser configurado por el usuario.

La siguiente figura muestra el diagrama de flujo de HTTP Client Get Task.

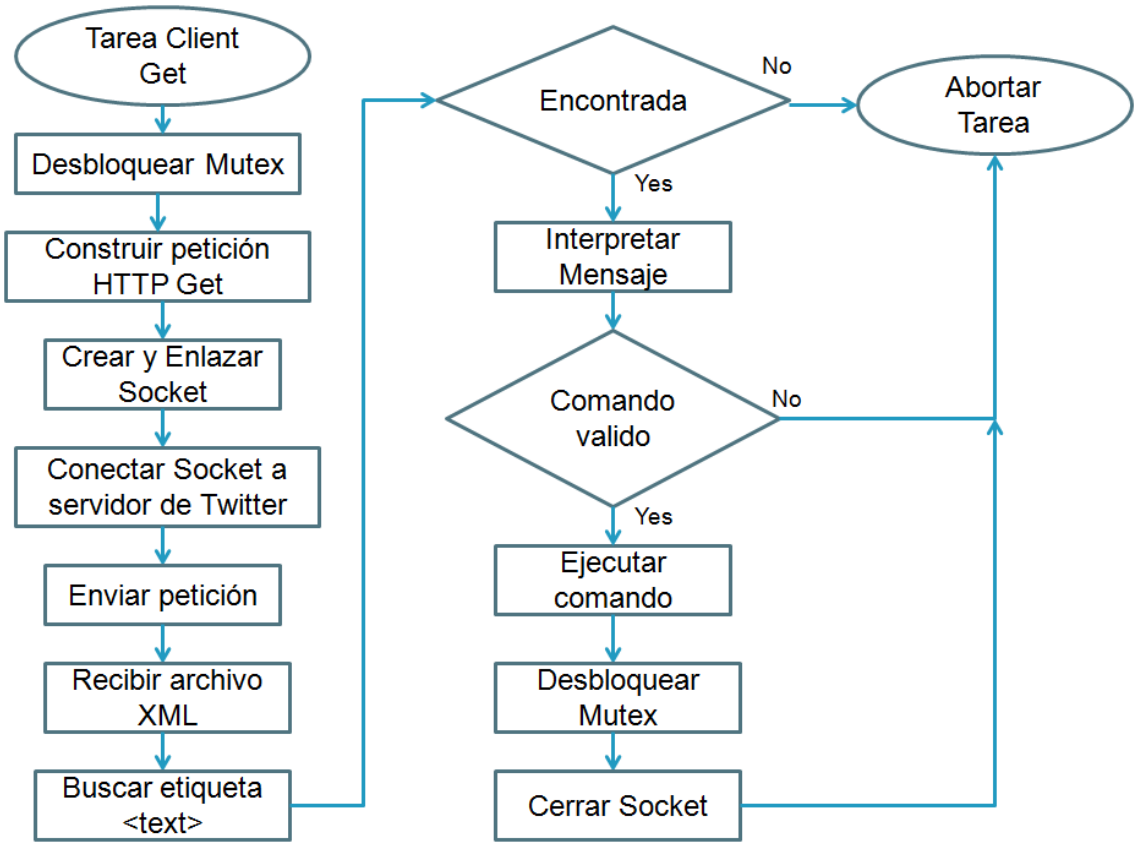


Ilustración 3. Tarea Client Get

La tarea httpclient crea un socket TCP que utiliza el puerto 80 que a su vez pertenece al protocolo HTTP. Este socket se conecta a al servidor de Twitter. Para obtener el último tweet es necesario construir una petición de HTTP. Esta petición debe realizarse como lo indica la siguiente figura.

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

Ilustración 4. Petición HTTP Get

El protocolo HTTP se describe en el RFC 2616 (Request for Comments por sus siglas en inglés). A continuación, se describen cada una de las cadenas.

GET /users/FSL_K60.xml HTTP/1.0 – Se utiliza el método GET para obtener cualquier información que contenga el URL (Uniform Resource Locator por sus siglas en inglés). En este caso el URL devuelve el archivo FSL_K60.xml que corresponde a la cuenta de Twitter creada para este proyecto. También se envía la versión del protocolo HTTP. Este caso se usa la versión HTTP/1.0.

Host: twitter.com – La segunda cadena especifica el Host al que se hace la petición. Debe representar el URL del servidor al que se hace la solicitud.

User-Agent: HTMLGET 1.0 – La cadena User-Agent contiene la información del usuario. Se puede componer de múltiples palabras simbólicas a las que llamamos símbolo que el agente puede usar para identificarse. En este caso se utiliza un símbolo para identificar a la aplicación de control y monitoreo. Dicha cadena es "HTMLGET" versión "1.0".

El servicio de Twitter proporciona un archivo XML, el cual es analizado para obtener información de la cuenta de un usuario. Los tweets que se envían desde la cuenta de Twitter "FSL_K60" son utilizados para representar los comandos de entrada del microcontrolador. El archivo FSL_K60.xml consiste en una serie de datos que corresponden a la cuenta de Twitter "FSL_K60". La etiqueta <text> contiene el último tweet que el usuario envió al servicio de Twitter. Esta etiqueta puede encontrarse en el archivo XML.

Para obtener el archivo XML es necesario establecer una conexión TCP a través del puerto 80 con servidor HTTP de Twitter. La petición HTTP descrita en la figura 1 es enviada al servidor y el resultado es el archivo XML. El contenido del archivo es analizado, una vez que la etiqueta <text> es encontrada el análisis concluye y el contenido de esta etiqueta es interpretado y ejecutado en caso de que se identifique un comando valido.

El comando tiene una estructura especifica; este debe comenzar con un punto (.) que indica que es el inicio del comando. La primera palabra debe de ser "spark", esto le dice al analizador que el comando debe de ser evaluado. Otro punto (.) es necesario para separar el siguiente indicador. La aplicación implementa cuatro objetos; "led", "samplerate", "motor" y "light". El objeto "led" recibe dos parámetros. El primero indica el número de diodo luminoso a controlar, los valores pueden ir de 1 a 4. El segundo parámetro indica el nuevo estado del diodo luminoso, solo dos valores son válidos; "on" y "off". Estos valores le indican al microcontrolador si debe encender o apagar un diodo luminoso en específico.

La implementación del objeto "samplerate" se utiliza para modificar el tiempo de espera para que la aplicación vaya y cheque si hay nuevos mensajes en la cuenta de Twitter. Por default la aplicación inicia con 5 minutos como periodo.

También se implementaron los objetos "motor" y "light". Para implementar estos comandos se editó el driver de LWGPIO para mandar tres señales de salida. Una que controle una luz de 60 Watts y las otras dos se utilizan para controlar el sentido del giro del motor de corriente directa los cuales están conectados con su respectiva etapa de potencia.

El objeto "motor" recibe como parámetros los comandos "left", "right" y "off". El objeto light solamente recibe "on" y "off".

En la siguiente tabla se pueden observar algunos ejemplos:

.spark.led.N.on.	Enciende el diodo luminoso N del TWR-K60N512-kit, donde N es u numero entero del 1 al 4.
.spark.samplerate.N.	Cambia el periodo para reiniciar la tare HTTP Get al número de minutos indicado por N.
.spark.light.ACCION.	Envía la instrucción para encender o apagar una bombilla de 60W que es controlada a través de un pin de propósito general de entrada y salida, donde ACCION puede ser 'on' u 'off'.

.spark.motor.DIRECCION.	Envía las instrucciones para girar el motor de corriente directa de 12V a la derecha o a la izquierda, donde DIRECCION puede ser 'right' o 'left'.

Tabla 1 Ejemplos de comandos

Como se muestra en la tabla los puntos (.) son los símbolos que se utilizan para identificar cada parte del comando. Es necesario que le comando inicie y termine con punto (.).

La lista de comandos puede ser tan extensa como la aplicación lo demande. Con esta implementación solo se pretende mostrar la funcionalidad de los comandos.

HTTP Client Post Task

El propósito de esta tarea es establecer la comunicación con el servidor de Twitter con el fin de enviar un mensaje desde el microcontrolador. Para llevar a cabo esta tarea se utiliza un tercer elemento que es el servicio de Super Tweet[2].

El servicio de SuperTweet.net ofrece una alternativa para los conjuntos de soluciones de TCP/IP que no cuentan con el mecanismo de Capa de Sockets Seguros (SSL por sus siglas en inglés). La autenticación básica de HTTP se utiliza para iniciar sesión en el servidor SuperTweet.net. Para empezar, es necesario firmarse en una sesión en Twitter para autorizar a la aplicación API MyAuth Proxy SuperTweet.net. el acceso a la cuenta de Twitter. Después elegir una nueva contraseña (diferente a la contraseña real para twitter) que las aplicaciones pueden utilizar con la API de http://api.supertweet.net. Después que estos pasos han sido completados es posible abrir comunicación entre MQX y Supertweet.net para que sea posible postear un tweet en la cuenta de Twitter.

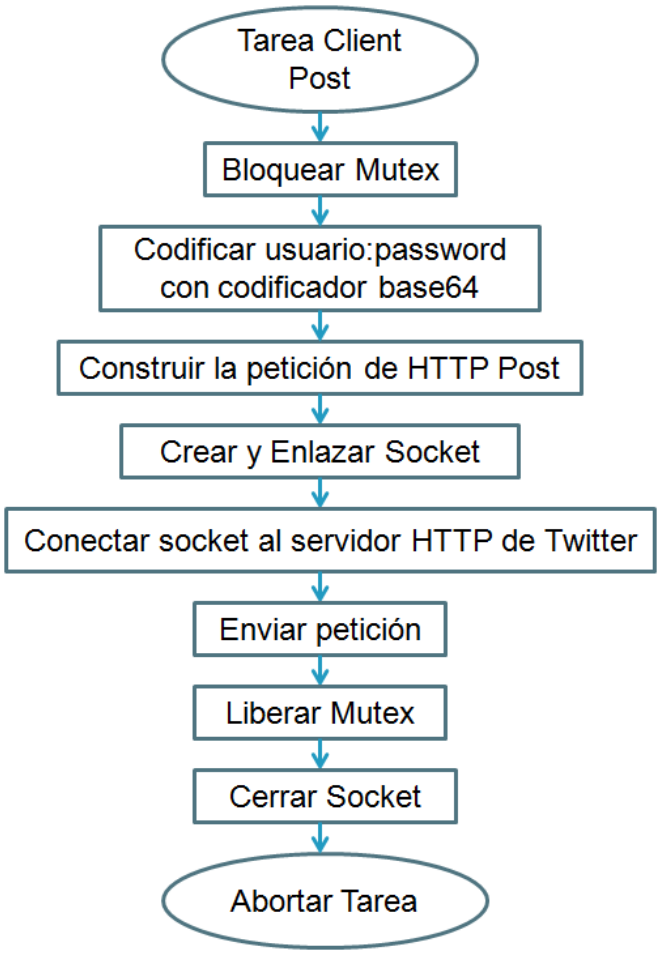


Ilustración 5. Tarea Client Post

La tarea httpclient crea un socket TCP que utiliza el puerto 80 el cual a su vez pertenece al protocolo HTTP. Este socket se conecta al servidor de SuperTweet. La API de SuperTweet.net proporciona un método que toma ventaja de la tecnología de autenticación OAuth de Twitter, sin el costo ni la complejidad de OAuth, en una simple aplicación en Twitter. Para publicar un tweet en una cuenta de Twitter, es necesario construir una petición HTTP y enviarla al servidor de SuperTweet. La solicitud debe de ser construida como se muestra en la siguiente figura.

```
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
```

Ilustración 6. Peticion HTTP Post

A continuación, se describen cada uno de los encabezados de la petición de HTTP mostrada en la figura 4 de acuerdo al RFC 2616.

POST /1/statuses/update.xml HTTP/1.1 – El método POST está diseñado para enviar un bloque de datos de manera uniforme. Un ejemplo puede ser el enviar de una forma llenada en algún sitio web para hacer un pago, o darse de alta para obtener algún servicio. En esta aplicación el método POST utiliza la API "/1/statuses/update.xml". Esta API se utiliza para enviar un tweet desde la cuenta indicada.

Authorization: Basic ZnNsX21jdTp4c3cyMXFheg== -- El acceso por autenticación básica es un método de seguridad utilizado por los clientes HTTP en los cuales los usuarios proveen un nombre de usuario y una contraseña para poder hacer alguna petición desde alguna cuenta. El nombre de usuario y la contraseña se concatenan utilizando dos puntos (:). La cadena construida es codificada usando el algoritmo Base 64. En esta aplicación el usuario 'FSL_K60' y la contraseña 'xsw21qaz', son concatenados formando la cadena 'FSL_K60:xsw21qaz' la cual es codificada usando el algoritmo Base64. La cadena resultante de este proceso es 'ZnNsX21jdTp4c3cyMXFheg==.

Usuario	Contraseña	Antes de Base64	Después de Base64
FSL_K60	xsw21qaz	FSL_K60:xsw21qaz	ZnNsX21jdTp4c3cyMXFheg==

Tabla 2. Decodificación Base64

User-Agent: HTMLPOST 1.0 – El campo User-Agent contiene información acerca del agente de usuario (cliente) que origina la petición. Este campo puede contener varios símbolos y comentarios que identifican el agente. En este caso solamente un símbolo es utilizado; nombre "HTMLPOST" y versión "1.0".

Host: api.supertweet.net – Este campo especifica la Internet del recurso que se solicita. Este campo debe representar el nombre del servidor origen que está dado por el URL original. Esto le permite al servidor distinguir entre URLs internos y ambiguos de un servidor de múltiples conexiones.

Accept: */* -- El campo Accept se utiliza para especificar ciertos "media types" que son aceptados en la respuesta.

Content-length: 35 – El encabezado Content-Length indica el tamaño del mensaje que será enviado y está dado en número decimal.

Content-Type: application/x-www-form-urlencoded -- Indica el "media type" del contenido enviado.

status=Tweet from TWR-K60N512 board – Esta cadena es enviada a la API "/1/statuses/update.xml". Este texto representa un comando que indica el mensaje que será posteado en la cuenta de Twitter.

Una vez enviada la petición, Twitter regresa un archivo XML que incluye el código de la API ejecutada y el mensaje enviado. De esta manera se envía un mensaje a la cuenta de Twitter cada vez que el potenciómetro tiene una variación o cada vez que se presiona un botón de la tarjeta que en una aplicación real podríamos interpretar como notificaciones de lecturas de sensores.

4. IMPLEMENTACIONES

Resumen: En este capítulo se presentan las implementaciones desarrolladas en este trabajo para solucionar el problema planteado como objeto de estudio.

1.11. Herramientas de Desarrollo

Para desarrollar el sistema se utiliza el micro controlador K60N512 de la familia Kinetis de Freescale. Sobre este dispositivo se monta el Sistema Operativo en Tiempo Real (RTOS por sus siglas en inglés) MQX.

Entre los múltiples servicios que provee MQX, para el desarrollo de esta aplicación se utilizaron los siguientes:

- Manejador de tareas
- Mutex
- Funciones de peso ligero de pines de Propósito General de Entrada y Salida
- Driver de Convertidor Analógico a Digital
- Funciones de fecha y hora

MQX provee librerías que dan soporte de TCP/IP y USB entre otras. En esta aplicación se utilizó la Conjunto de Soluciones de Comunicaciones en Tiempo Real (RTCS por sus siglas en inglés), para implementar la parte de comunicaciones. Dicha Suite provee los servicios de TCP sobre IP y HTTP que son los protocolos de comunicaciones que se utilizan en la aplicación.

En la figura 7 se muestran todos los protocolos soportado por RTCS.

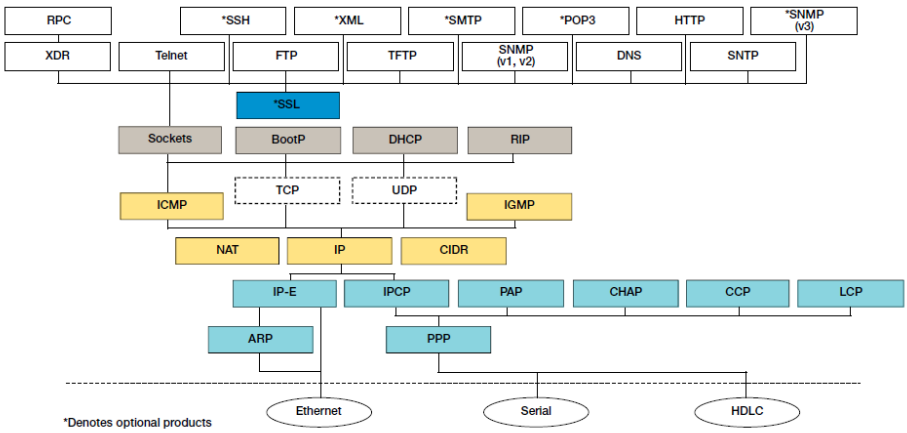


Ilustración 7. Servicios que provee RTCS

5. RESULTADOS

Resumen: En este capítulo se presentan los resultados obtenidos del desarrollo de este trabajo y una discusión sobre su impacto en el problema estudiado.

1.1. Pruebas y Resultados

Para probar la solución implementada, se configuro para leer el último tweet cada 30 segundos y se enviaron todos los comandos reconocidos por el sistema repetidas veces. Después de varias sesiones de prueba y depurar algunos errores finalmente todos los comandos fueron ejecutados correctamente.

Lo comandos enviados fueron los siguientes:

Control de diodos luminosos	**Control de muestreo**	**Control de bombilla 60W**	**Control de motor de CD**
.spark.led.1.on.	.spark.samplerate.1.	.spark.light.on.	.spark.motor.left.
.spark.led.2.on.		.spark.light.off.	.spark.motor.right.
.spark.led.3.on.			
.spark.led.4.on.			

Tabla 3. Pruebas

También se comprobó que el botón y el potenciómetro envían un tweet cada que el botón es presionado o cada que el potenciómetro cambia su valor.

6. CONCLUSIONES

1.2. Conclusiones

En este proyecto se diseñó, implementó y probó un cliente HTTP embebido en un microcontrolador de 32 bits de Freescale el cual se comunica con el servidor de Twitter. La aplicación implementada es capaz de leer los mensajes de la cuenta de Twitter "FSL_K60". También es capaz de interpretar los mensajes y ejecutar los comandos como encender y apagar diodos luminosos, una luz de 60 Watts, controlar el sentido del giro de un motor de corriente directa y cambiar el periodo en que se hace una nueva petición para verificar si hay comandos nuevos.

1.3. Trabajo Futuro

El trabajo futuro de este trabajo consiste en las siguientes actividades:

1. Incluir la opción para escoger ya sea conectividad a través de cable de Ethernet o por WiFi. Esto se realizaría con alguna de las tarjetas TWR-WiFi de Freescale (Atheros, RedPine o GainSpan)
2. Agregar soporte de USB para implementar un host que pueda guardar un archivo con una lista de todos los comandos ejecutados con la respectiva fecha y hora.
3. Implementar un driver que controle la velocidad del motor y la intensidad de la luz.
4. Conectarse a un servidor de fecha y hora para sincronizar la aplicación con los tweets enviados desde cualquier parte del mundo.
5. Diseñar un pequeño PCB para implementar un modelo de referencia.

REFERENCIAS

[1] Hypertext Transfer Protocol -- HTTP/1.1 disponible en: https://www.w3.org/Protocols/rfc2616/rfc2616.html

[2] Supertweet.net API, disponible en: https://www.supertweet.net/about/documentation

[3] Tweet Library for Arduino, disponible en: https://twitter.com/arduinoorg?lang=es

[4] Raspberry Pi Learning Resources, disponible en: https://www.raspberrypi.org/learning/getting-started-with-the-twitter-api/