Centro de Telemedicina
Facultad de Medicina

BioIngenium
Research Group

UNIVERSIDAD NACIONAL DE COLOMBIA
SEDE BOGOTÁ

Dispositivo de Autenticación y Firma Digital con Interfaz USB utilizando un Algoritmo de Conocimiento Nulo

Alfredo Espitia Beltrán, Karen Ortega Marín

Universidad Nacional de Colombia

25 de marzo de 2009

OBJETIVO

Proporcionar una herramienta a los usuarios del sistema *SARURO* que les permita ingresar a la plataforma de una manera más segura. Esta seguridad se representa en el hecho de minimizar el riesgo de que su contraseña sea descifrada en beneficio de un tercero.

SEGURIDAD INFORMÁTICA

"Algo que soy", "algo que sé", "algo que poseo".

Figura: Biometría

Figura: Smart Cards

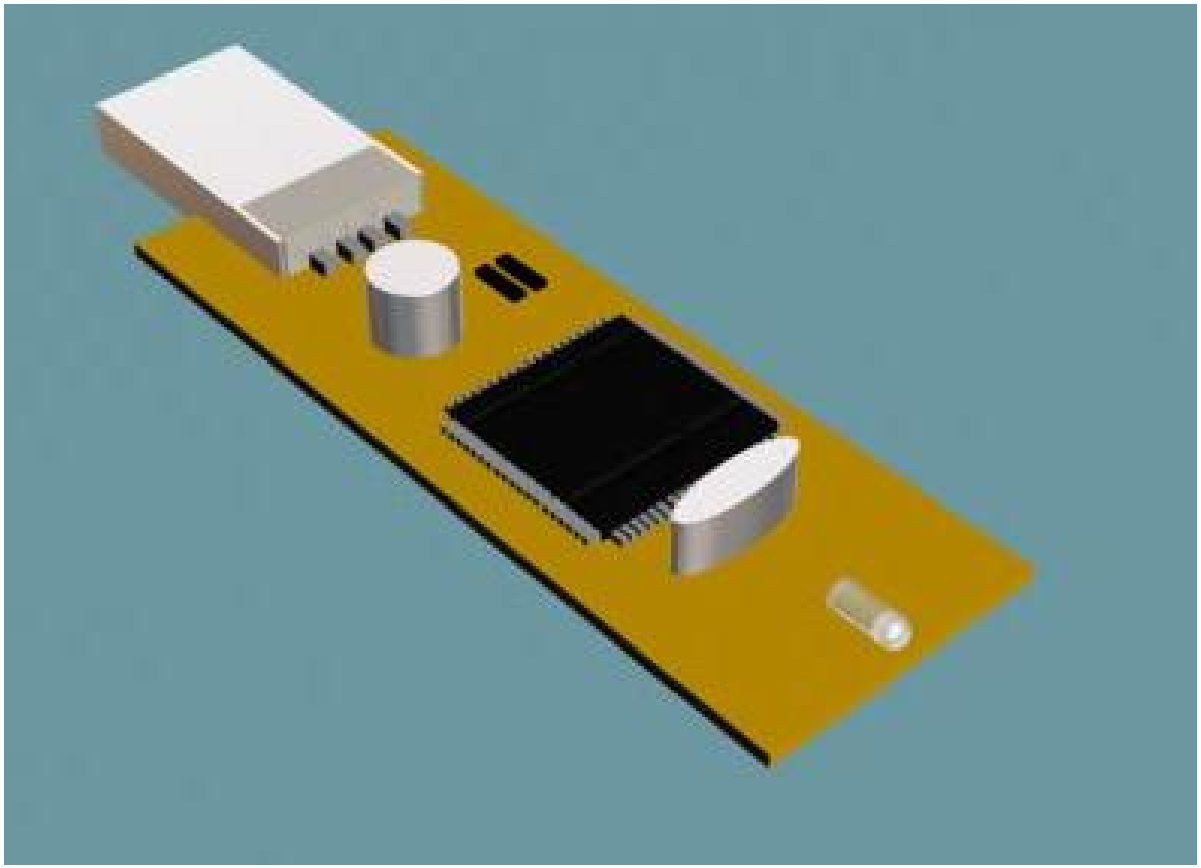

Figura: Token USB

RESUMEN

- Se diseñó e implementó una llave o token USB que realiza un proceso de autenticación/verificación con el computador para acceder a la red SARURO.
- La autenticación se lleva a cabo con un método de encriptación de la identidad de la llave, que denominamos **algoritmo de conocimiento nulo**, porque en el canal de transmisión nunca se revela dicha identidad.

- El token tiene como base un microcontrolador con las capacidades suficientes para llevar a cabo una implementación que soporte una seguridad considerable. Hemos escogido un AT90USB1287.
- Se creó un driver en el PC, bajo Windows con Visual C++, para efectuar una comunicación USB con el microcontrolador y para hacer las negociaciones computacionales con el token.
- Finalmente los resultados se acoplaron al servidor del sistema de información, el cual está hecho en Java, con las herramientas JNI (la cual implementa funciones de C++ en Java) y RMI (que ejecuta procesos cliente-servidor a través de internet).

OPERACIÓN

Figura: Ingreso al Sistema

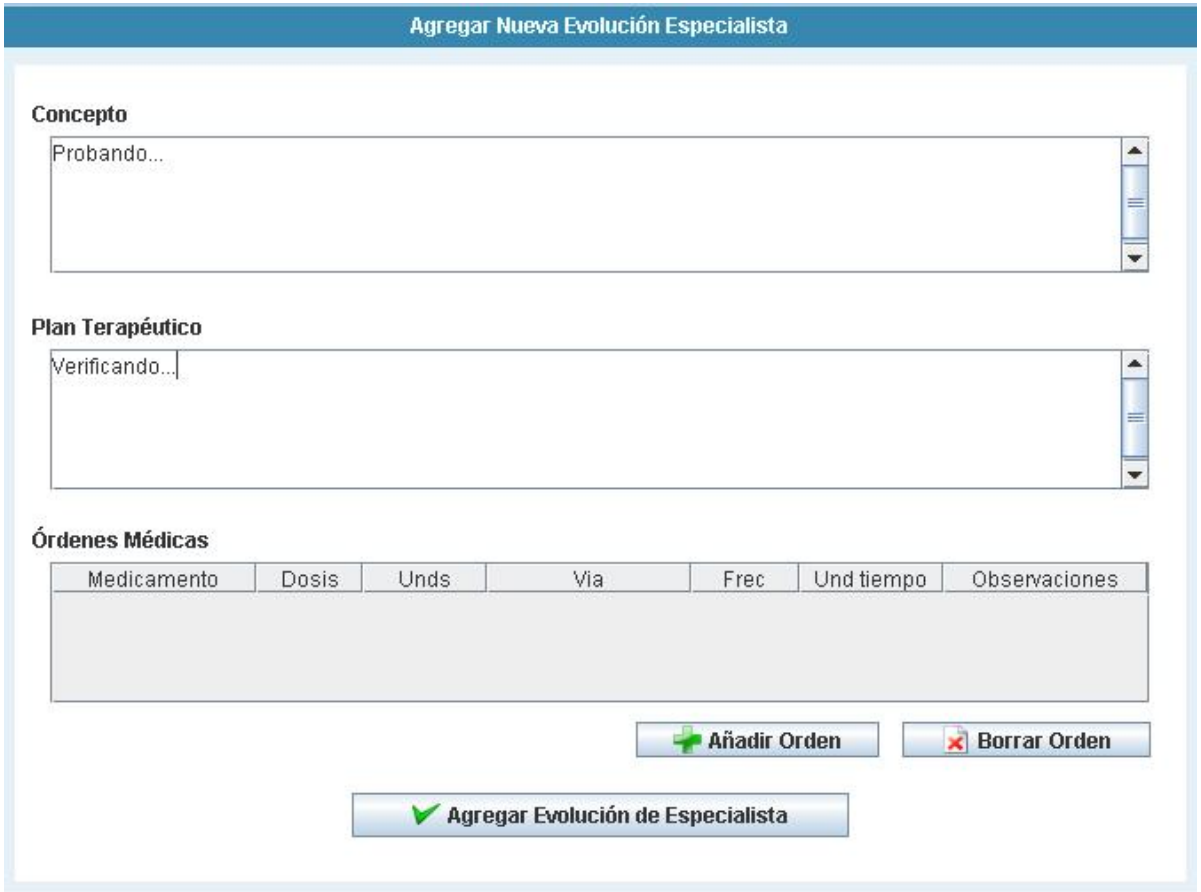

Figura: Applet

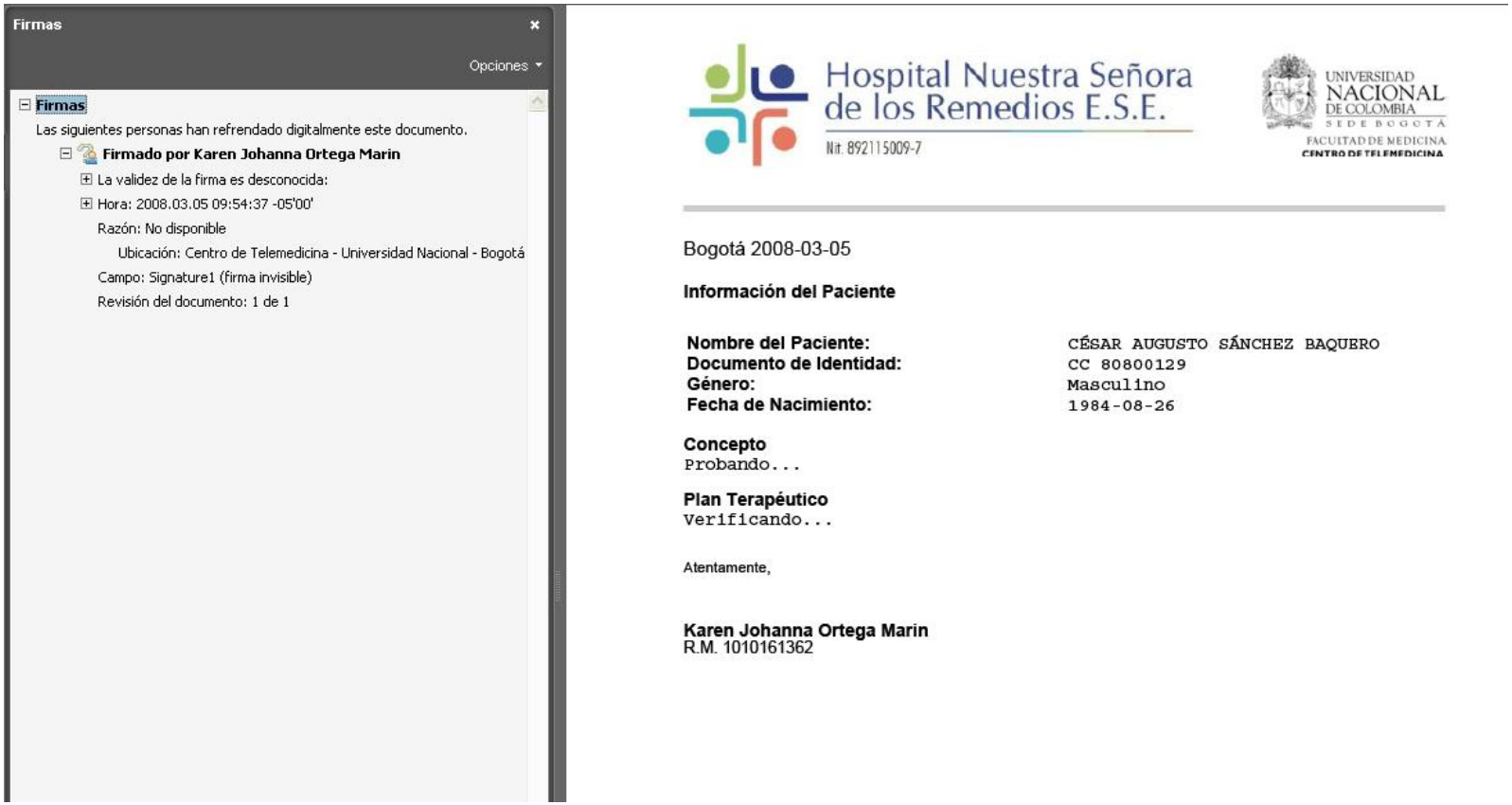

Figura: Documento firmado digitalmente

ETAPAS

1. Recopilación y lectura de la información concerniente a los recursos de hardware necesarios y a la implementación del algoritmo de conocimiento nulo.
2. Escogencia del microcontrolador a utilizar en el dispositivo.
3. Realización de pruebas al microcontrolador, en cuanto a la interfaz USB y a la capacidad de cómputo en las operaciones matemáticas necesarias en el algoritmo.
4. Implementación del algoritmo en el microcontrolador.

5. Implementación del software complementario para la ejecución del algoritmo entre el dispositivo y un computador personal.
6. Verificación del funcionamiento y posibles mejoras.
7. Diseñoo de ataques especializados en intentar violar la seguridad que presta el dispositivo.
8. Generación de la documentación y presentación final con resultados y conclusiones.

CONOCIMIENTO NULO

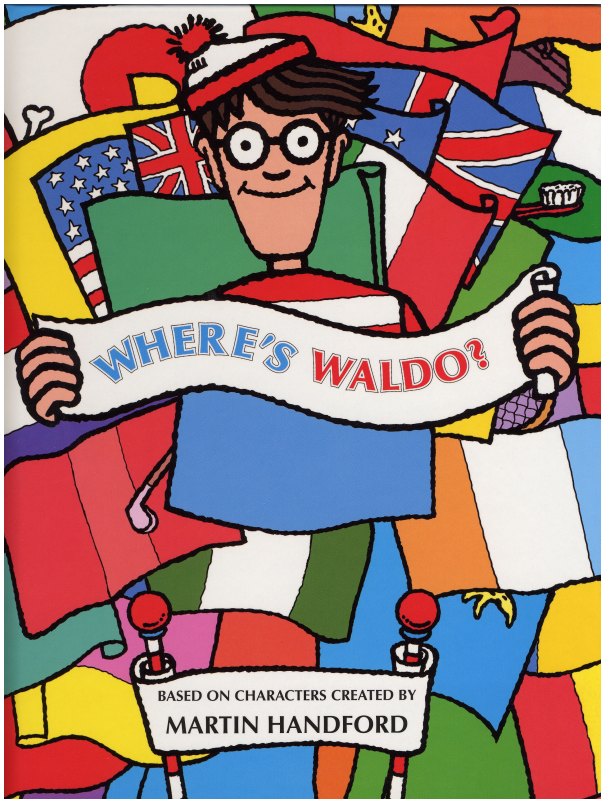

Figura: ¿Dónde está Waldo?

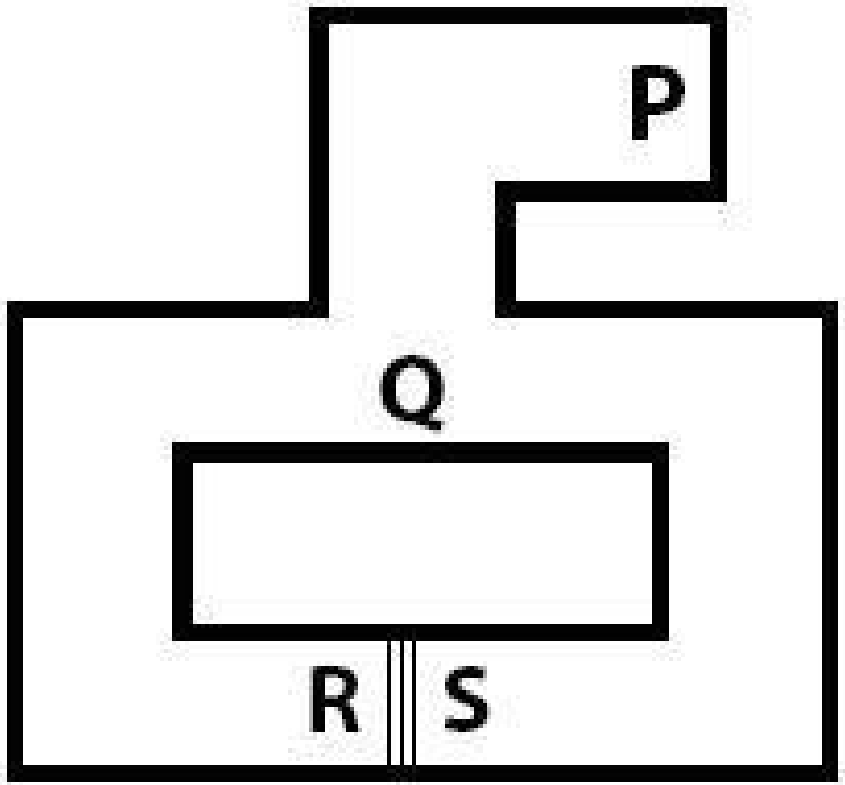

Figura: La cueva de Alí Babá

Los pasos típicos en una prueba de conocimiento – cero son:

- El probador envía un mensaje de **compromiso** aleatorio.
- El verificador le devuelve un **desafío** aleatorio.
- Finalmente, dependiendo del compromiso y del desafío, el probador envía una **respuesta**.

Este protocolo se puede hacer varias veces y, dependiendo de las respuestas obtenidas, el verificador puede aceptar o no la prueba.

Aritmética Modular

Consiste en realizar cálculos usando sólo una cantidad finita de números enteros, limitados por una constante que se escoge de acuerdo a la conveniencia con dichos cálculos.

Si son las 13h, ¿qué hora será dentro de 15h?
$(13 + 15) \mod 24 = 28 \mod 24 = 4 \mod 24$, es decir, las 4 de la madrugada

Problema de los Residuos Cuadráticos

$$y \equiv x^2 \mod n$$

Problema del Logaritmo Discreto

$$y \equiv a^x \mod n$$

MICROCONTROLADOR

Una vez escogido el microcontrolador que pudiese resultar más adecuado, se inició una serie de pruebas técnicas con el mismo para:

- Confirmar la información brindada por el fabricante.
- Verificar el comportamiento de la interfaz USB.
- Estimar los alcances y limitaciones para la implementación del algoritmo, es decir la capacidad de cómputo de la ALU de 8 bits con la que cuenta el microcontrolador.
- Decidir una posible y rápida solución de reemplazo en caso de no encontrar satisfacción con el dispositivo.

INTERFAZ USB

Los ensayos con el puerto USB se dirigieron a dos aspectos fundamentales:

- La programación del microcontrolador. Se refiere a la instalación de los drivers necesarios y a la aplicación del software de programación.
- La ejecución de la función programada. Hace referencia a la configuración adecuada del puerto, de tal forma que la función implementada en el microcontrolador se ejecute estableciendo una correcta comunicación con el PC.

CAPACIDAD COMPUTACIONAL

Por otra parte, a causa del algoritmo ZKP implementado, se derivó la necesidad de verificar la capacidad de la ALU y determinar si es propicia para ejecutar el algoritmo con valores suficientemente grandes que garanticen una mayor seguridad.

Además, fue menester hacer algunas pruebas misceláneas en temas como la generación de números aleatorios, uso de variables adecuadas, transformación de variables y uso de operadores, entre otros.

USB - UNIVERSAL SERIAL BUS

- **USB** (Universal Serial Bus): Es una arquitectura de comunicación que dota a equipos y dispositivos con la habilidad de interconectar una gran variedad de componentes que usan un cable simple de cuatro hilos: dos de alimentación y dos de datos, por los cuales puede fluir información a tasas de 1.5, 12 ó 480 Mbits/s (Mbps).

  El protocolo USB permite instalar componentes al iniciar el dispositivo o sobre la marcha. Estos componentes se clasifican en clases según su conducta y su función. Algunas de ellas son: visualización, comunicación, almacenamiento masivo, audio y de interfaz humana (HID).

- **HID** (Human Interface Device): HID es una clase de dispositivos USB utilizados por seres humanos para controlar alguna operación en el software del PC. Algunos ejemplos son: teclado, mouse, joystick y tablero táctil.

  Sin embargo, existen dispositivos HID que no requieren interacción humana directa, pero que por su naturaleza se incluyen en esta clase, como son los lectores de códigos de barras, medios de seguridad biométrica e instrumentos de medición.

DRIVER Y PROGRAMACIÓN

Una ventaja importante de utilizar este microcontrolador radica en las herramientas gratuitas que ofrece Atmel en su página Web. Dos de ellas son el AVR Studio y el WinAVR, que constituyen el software necesario para realizar, en lenguaje C++, la descripción de la función a implementar.

AVR Studio además permite programar físicamente al microcontrolador, siempre y cuando se cuente con alguna de las tarjetas de desarrollo que soporta, lo cual significa una desventaja, pues se hace necesaria la adquisición de hardware adicional, aumentando los costos del diseño.

Otra herramienta disponible es el FLIP 3.2, dirigida a programar, entre otros, al AT90USB1287 sin que exista la necesidad de hardware adicional.

Este software programa al microcontrolador mediante el método **In-System Programming** que consiste en aprovechar el puerto USB, tanto del PC como del microcontrolador, para programarlo usando un simple cable USB.

Esto es posible gracias a que el AT90USB1287 trae por defecto un pequeño programa (**Bootloader**), que funciona como interfaz entre el USB y la memoria a ser programada (Flash o EEPROM).

CONFIGURACIÓN DEL PUERTO USB

- **Device Configuration.** Se comportará como dispositivo y no como Host.
- **USB 2.0 Full Speed.** Versión y velocidad.
- **Bus Powered Device.** El dispositivo no cuenta con alimentación externa. Utiliza la fuente del bus USB.

- **Interrup Tranfer.** En lugar de realizar la transferencia de datos de forma periódica, se generan interrupciones cuando es necesario responder a una petición.
- **Device Class HID.** Clase de dispositivo. HID por defecto.
- **NumEndpoints = 2.** Número de paquetes de informaciósn. 1 de salida y 1 de entrada.
- **Length = 8.** Tamaño de los paquetes de información en Bytes.

ALU

Inicialmente se consideró la posibilidad de utilizar un microcontrolador que contara con una ALU de por lo menos 16 bits, por el tamaño de las variables a utilizar en el algoritmo. Pero dado que no se consiguió ningún modelo con esta característica y de bajo costo, se renunció a esta posibilidad.

El AT90USB1287 cuenta con una **ALU de 8 bits**, lo que llevó a pensar en la necesidad de diseñar un algoritmo que permitiera realizar en esta ALU, operaciones con números con más de 8 bits. Contrario a esta idea, únicamente utilizando la librería math.h contenida en el software de diseño, se logró realizar operaciones comprobadas con variables de **64 bits**.

Gracias a lo expuesto, es viable utilizar en el dispositivo de autenticación llaves privadas de 64 bits que en decimal se representan en números de hasta 20 dígitos. Es claro que una llave de este tamaño no es comparable con las llaves de 512 bits utilizadas en sistemas como el RSA o ElGamal pero teniendo en cuenta que (al contrario de estos) el dispositivo en desarrollo utiliza una prueba dinámica e interactiva, podemos decir que la llave de 64 bits representa una seguridad bastante aceptable.

CONSIDERACIONES

- Al final del desarrollo, los detalles del algoritmo no deben ser conocidos, excepto por un número muy reducido y necesario de personas.
- El verificador no debe tener intenciones de conocer la llave privada.
- Se pueden introducir errores intencionalmente para dificultar aún más el espionaje de un tercero.
- En caso de que se presenten falsas respuestas en cualquier paso del protocolo, no se deben dar a conocer inmediatamente. Únicamente al final se informará que la prueba no ha sido válida.

PRODUCTOS

- Dispositivo implementado e integrado en SARURO.
- Publicación internacional:
  Authentication and Digital Signature USB Device for Telemedicine Applications
  Publicado en: *Proceedings of the 7th International Caribbean Conference on Devices, Circuits and Systems, Mexico, Apr. 28-30, 2008.*
  ISBN: 978-1-4244-1956-2.

IMÁGENES

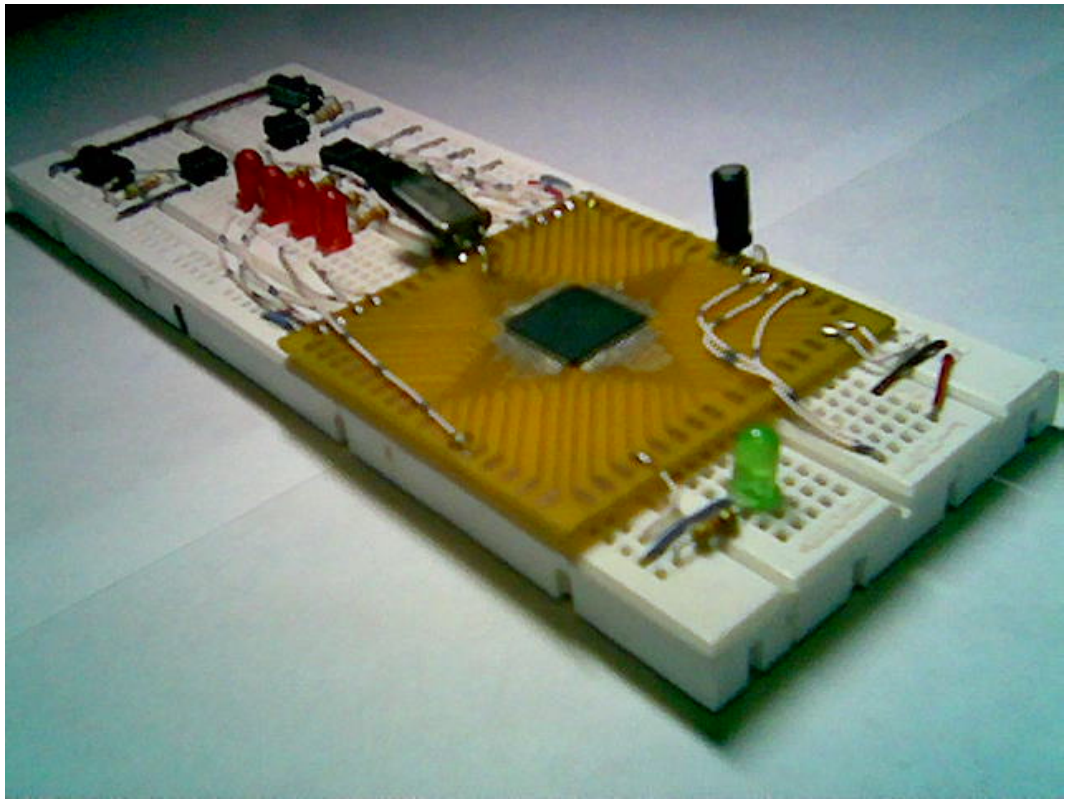

Figura: Prototipo de pruebas

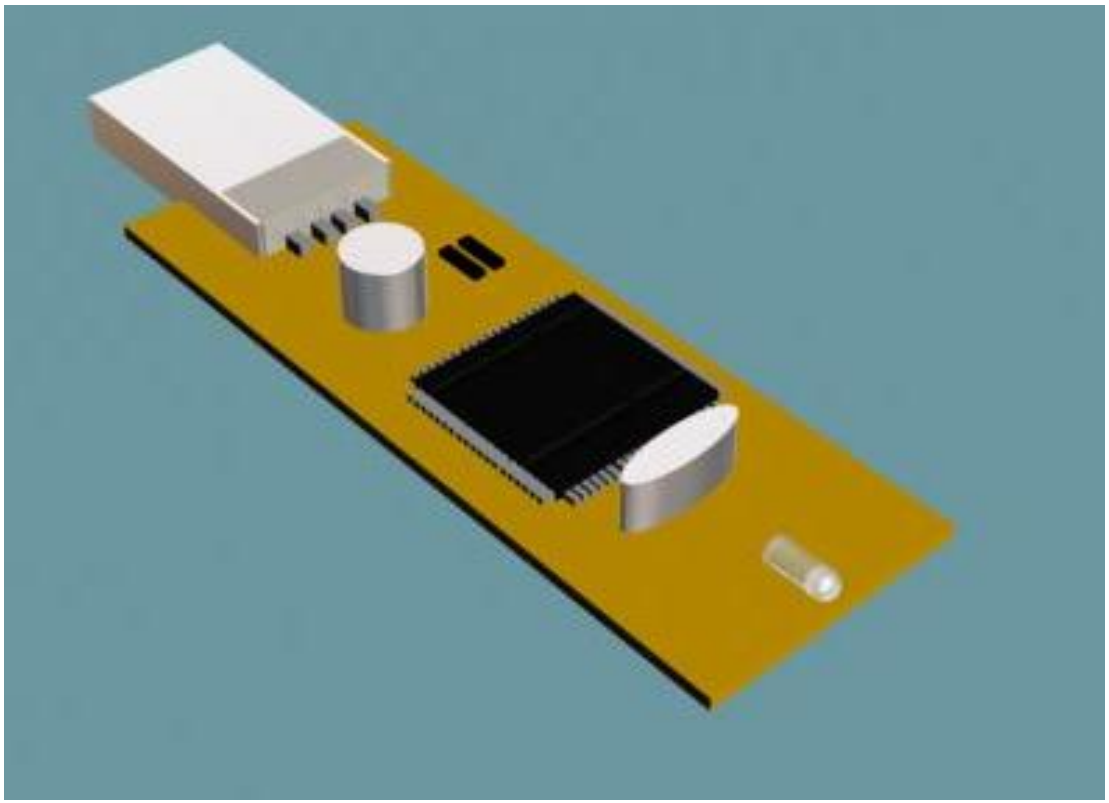

Figura: Diseño 3D de la tarjeta

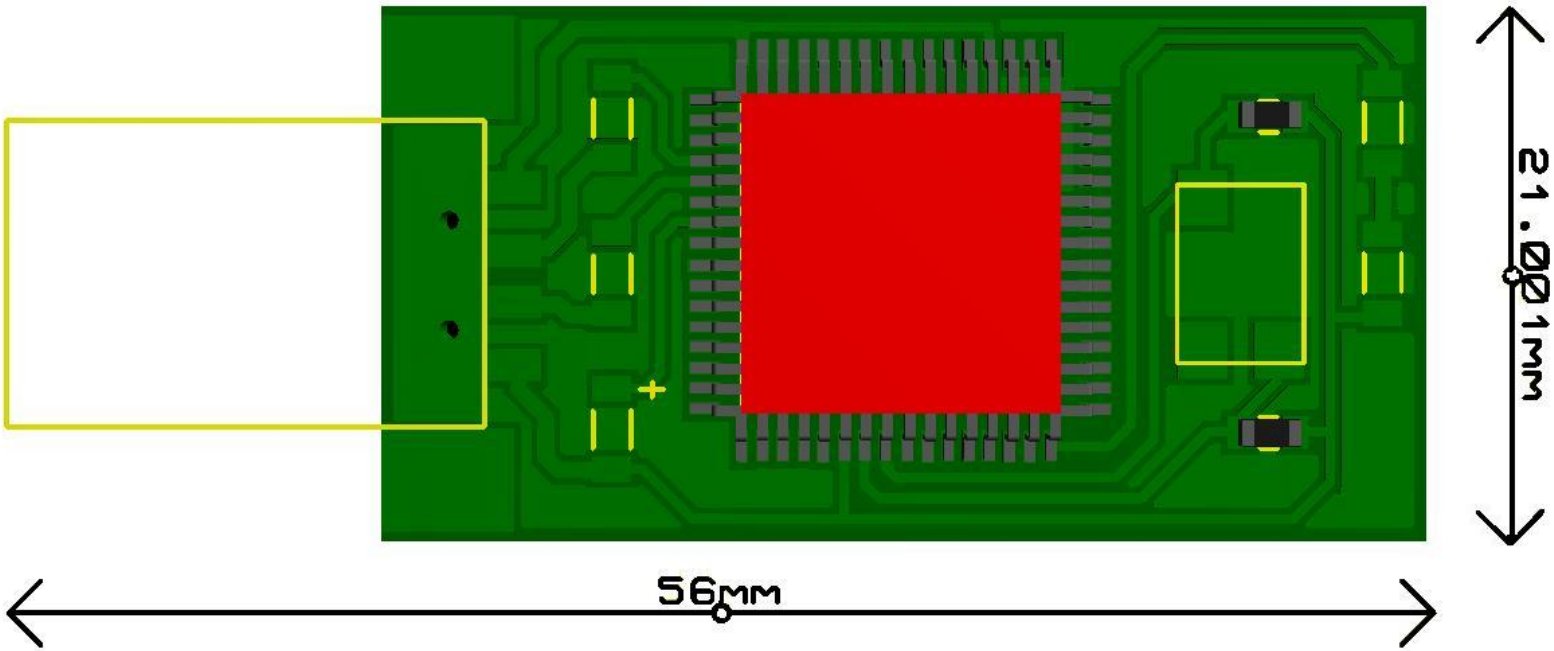

Figura: Layout de la tarjeta

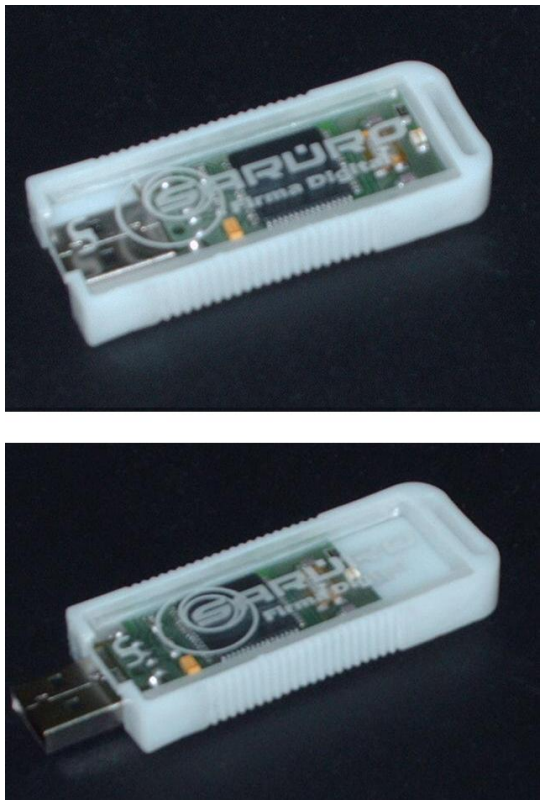

Figura: Fotografía del dispositivo

Figura: Driver verificador