

Ingeniería Inversa a los Cartuchos de Videojuegos de la Consola Leapster TV.

Edson Gerardo Magallanes Gómez [shinigami.des@gmail.com]
Yarely Baéz Lopéz [yarely.baez@gmail.com]
José Enrique Álvarez Estrada [jeae@ucaribe.edu.mx]

Universidad del Caribe, Cancún Quintana Roo, México 4 de diciembre de 2010

Resumen

Leapster TV es una plataforma para videojuegos educativos. Su simplicidad y bajo costo la hacen atractiva para micro y pequeños desarrolladores. Pero el fabricante no libera el SDK. Este trabajo presenta una propuesta de creación de un SDK alternativo mediante técnicas de ingeniería inversa. En esta primera etapa se aborda la del cartucho. Se pretende leer su contenido, y la vez descubrir el layout y su esquema de conexión. Como valor agregado se pondrán a prueba varias técnicas de ingeniería inversa de hardware parcialmente documentadas en libros, revistas e Internet.

1. Introducción

La consola Leapster se enfoca al segmento de mercado educativo infantil, cuenta con una gran variedad de juegos que ayudan a los niños a aprender las actividades escolares básicas y a desarrollar sus habilidades. Los videojuegos se distribuyen mediante cartuchos (cajas de plástico aptas para ser utilizadas repetidas veces), que contienen una memoria ROM.[1]

El precio actual de los cartuchos oscila entre los 250.00 y los 370.00 pesos, mientras que el de la consola fluctúa entre 500.00 y 1,200.00. El precio atractivo de esta última hace que muchos usuarios la adquieran, pero luego no pueden afrontar el costo de los primeros. Conscientes de esta situación, el presente proyecto pretende hacer ingeniería inversa a los cartuchos de la plataforma Leapster TV, con el fin de dotar a micro y pequeños desarrolladores con la información que les permita crear y lanzar al mercado los suyos propios.

La empresa Leapfrog no publica código o herramientas que permitan trabajar en su plataforma, por lo tanto nos enfocamos en la ingeniería inversa como método para obtener la información necesaria con vistas al desarrollo de nuevos juegos. La ingeniería inversa es el proceso de descubrir los principios tecnológicos de un dispositivo, objeto o sistema, a través del razonamiento de su estructura, función y operación, generalmente para intentar crear un dispositivo o programa que haga la misma o similar tarea sin copiar la original. Esto es importante, porque al no copiar el original, se obtiene el conocimiento necesario para desarrollar alternativas igual de eficaces sin incurrir en prácticas de piratería.[2]

Es posible que buena parte del precio de los cartuchos originales de Leapster se destine al pago de derechos de autor por los personajes que en ellos aparecen. Sin que ello beneficie a nadie, excepto a los dueños de tales derechos, quienes rara vez son artistas o productores mexicanos. Al aplicar ingeniería inversa a los cartuchos, se obtiene información que permite a desarrolladores independientes crear sus propios videojuegos, basados en sus propias ideas y personajes. Ello repercute en dos beneficios directos: se ofrece a los consumidores una alternativa más económica, al no tener que pagar derechos de autor a grandes marcas por utilizar sus personajes; y permite a creativos mexicanos popularizar los suyos propios.

En México, la ley de propiedad industrial establece en el artículo 26 que están protegidos sólo aquellos productos o procesos que cuenten con una patente de invención, o a partir del momento en que se solicite una patente de invención. Si no cuentan con una patente, no hay jurisdicción para ellos bajo las leyes mexicanas.[3] La consola Leapster TV no tiene patente de invención en México, por lo tanto no resulta ilegal en México realizar ingeniería inversa a la misma.[?]

Como antecedentes, cabe señalar que no encontramos información sobre algún otro proyecto en el que se intente hacer ingeniería inversa a la consola Leapster TV. Sin embargo existen proyectos similares para Leapster Didj y Leapster explorer, que son las sucesoras de Leapster TV. Ambas comparten la similitud de que usan Linux como sistema operativo embbebido, y los seguidores de este sistema operativo explotan esa característica para conectarlas a una consola y así extraer sus archivos y configurarlas.[4] Los procesos de ingeniería inversa que se utilizaron en esos proyectos se describen más adelante.

1.1. Acerca de la consola de videojuegos Leapster Didj



Figura 1: consola Didj

Leapfrog Didj es una consola de videojuegos educacional portátil y es una de las sucesoras de la consola de videojuegos Leapster TV, que además ofrece la capacidad de conectarse a internet y descargar los juegos de la página oficial de Leapfrog.

El primer trabajo que revisamos fue el de ingeniería inversa a la consola de videojuegos Didj. Schwarz [4] publica sus resultados y menciona que se basa en el trabajo de toda una comunidad de desarrolladores que trabajaron también sobre la misma consola, y cita todas las fuentes que usó para su desarrollo, pero la mayoría de los enlaces ya no están vigentes. Entre sus actualizaciones utiliza archivos disponibles, que Leapster ha liberado debido a que la consola Didj usa un sistema embebido Linux.

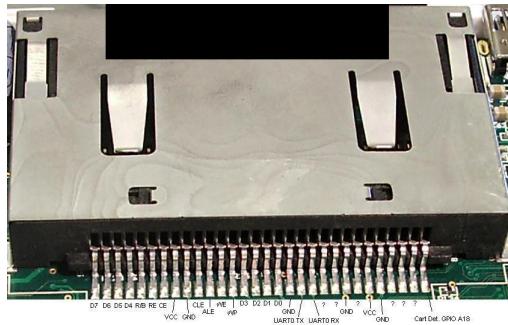


Figura 2: pines identificados de la consola Didj

El autor obtiene la configuración de los pines del cartucho en elinux [5] y con ello identifica los buses de datos, control

y dirección del cartucho. Una vez identificados se puede construir un archivo que pueda compilar, un arrancador o bootloader. El proceso de ingeniería inversa que Schwarz [4] realizó fue de la siguiente manera: Soldando un RS232 a 3.3 V a los pines del cartucho, que permiten el control del SO linux embebido. Ejecutando scripts de shell en el sistema de archivos : Lftest-lcd, Lftest-oss, Lftest-rtc, Lftest-usb realiza algunas pruebas para la manipulación de dispositivos como la pantalla y el de audio.

De los archivos liberados por Leapfrog obtuvo la mayor parte de información que necesitaba sobre el kernel del SO y el procesador del Didj. Después lo que necesitaba era obtener el archivo binario del cartucho, para esto investigó que el procesador tiene la capacidad de arrancar desde la UART (Universal Asynchronous Receiver and Transmitter). Así que utilizó los pines up/down del cartucho (D5 y D6) y conectó resistencias de 470 Ohm, provocando que la próxima vez que la consola se encendiera aparente que no arrancase, pero teniendo el procesador 100 % funcional y en espera de recibir información desde la UART. Así que utilizó un gestor de arranque volcado en una NAND y recordado a 16 kBytes, y cuando se transfirieron por completo los datos, la consola Didj inició de manera normal.

El siguiente paso fue construir su propio gestor de arranque, lo hizo en el entorno de desarrollo Linux Scratchbox en el cual configuró el paquete de herramientas para linux ARM GCC 4.1, lo compiló con una librería de C para Linux embebidos -llamada uClibc- y con esto obtuvo su propio bootloader Didj. Cabe mencionar que su trabajo no fue terminado, y por lo tanto no es funcional al 100 %. [4]

1.2. Acerca de la consola de videojuegos Leapster Explorer

Leapster Explorer también es una consola portátil desarrollada por Leapfrog, contiene juegos educativos destinados a niños de 4 a 9 años y al igual que su predecesor Didj, utiliza un sistema Linux embebido, por lo tanto el método de acceso es el mismo que el descrito en la sección 1.1.[5]

El proceso de ingeniería inversa utilizado para el Explorer y el Didj, parte principalmente de que se conoce la existencia de un Linux embebido como Sistema Operativo de la consola. Schwarz [4] enfatiza que la mayor parte de la información que obtuvo fue de elinux [5], donde encontró la configuración de pines del cartucho así como la documentación que la propia empresa Leapfrog publicó sobre los detalles



Figura 3: Consola Leapster Explorer

técnicos del Didj y el código del Linux embebido utilizado. Al contar con toda esta información pudo aprovechar la potencia del sistema operativo Linux embebido en el Didj para acceder a la consola vía software, a pesar de ratarse de un problema de ingeniería inversa de hardware. Otro punto relevante de su trabajo es el esfuerzo por la construcción del cartucho que le sirve como interfaz para seguir con el análisis del bootloader y le ayuda para mejorar y comparar los resultados. Sin embargo, no da detalles sobre los procesos de ingeniería inversa utilizados, ni hace mención sobre como llegó a sus deducciones y el porqué del interés en la realización de ese proyecto.

Para el caso de nuestro proyecto, aún no se sabe si la consola cuenta con algún tipo de Linux embebido o algo parecido, así que la propuesta de ingeniería inversa a los cartuchos de videojuegos Leapter TV se basa en conocer la estructura, operación y función de la ROM que se encuentra almacenada en los cartuchos, identificando las principales características de la misma tales como los buses de datos, dirección y control. Para ello se planea construir una interfaz que nos permita leer dicha ROM y posteriormente analizar e interpretar su contenido, y descubrir si esa información es suficiente para conocer la estructura interna de la consola y también para saber si se puede llegar a obtener el conocimiento necesario para la realización de videojuegos compatibles.

2. Desarrollo del Proyecto

2.1. Especificaciones de la consola

Para comenzar con el proyecto de Ingeniería Inversa fue necesario conocer las especificaciones técnicas de la consola Leapter TV, las cuales se enuncian a continuación:

Usa Adobe Flash

Procesador: ARCTangent 5.1, 96MHz.

Memoria: 2MB de RAM, 256 bytes no-volatil.

Tipo de Medio: Cartuchos de 4-16MB desde 2 a 512kb de almacenamiento no-volatil.

Graficos: 4Mb ATI chip.

Pantalla: 160x160 CSTN con touchscreen.



Figura 4: consola Leapster TV

A partir de estas especificaciones se comenzó a investigar más detalles de la consola y a analizar sus componentes principales.

2.2. Lectura de la Eprom 24C

Esta lectura de la EPROM se hizo porque está conectado serialmente a otros dispositivos identificados, por lo tanto una vez que se inserte el cartucho la información debe pasar por el puerto hacia los dispositivos obteniendo así los datos del cartucho. Para realizar la lectura de la EPROM



Figura 5: Memoria EPROM 24LC

24LC028, fue necesario construir una interfaz que nos permitiera leer sus datos sin desoldarla de la tarjeta principal de la consola.

La interfaz fue construida con los siguientes materiales:

1 Conector hembra DB9

2 Diodos zener de 5v 400 mW

2 Resistencias 3900 Ohms

2 m de cable de colores

Software PonyProg v1.17 H

Para conectar los materiales, nos basamos en el diagrama de la figura 6

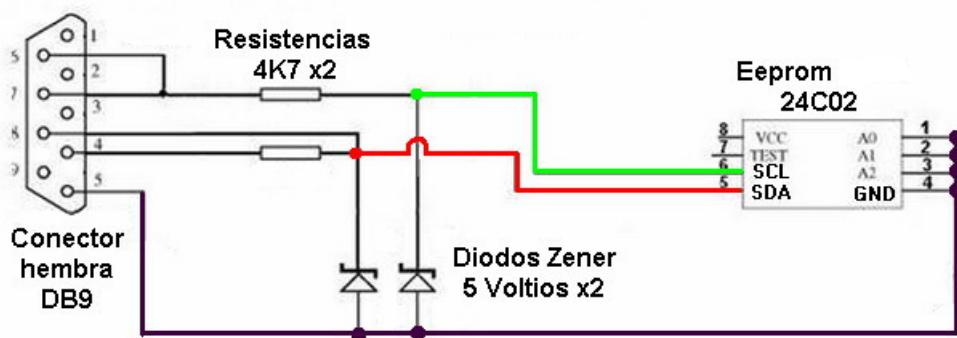


Figura 6: Diagrama de Conexión

Se conectó la resistencia 1(R1) en el pin 4 del conector hembra DB9 y la resistencia 2(R2) en los pines 6y7 del conector DB9 hembra, es decir, se puso la resistencia en el pin 7 y se hizo un puente con soldadura al pin 6.

Posteriormente se soldaron los 2 diodos (D1 y D2)al pin 5 del conector DB9, verificando que el cátodo (franja color negro) quedara en la parte alta del diodo.

Después D1 se soldó a R1 y la unión de estos se soldó al pin 8 del conector DB9 hembra y esta conexión es la que distinguimos como SDA.

D2 se soldó a R2 y esta conexión es la que distinguimos como SCL

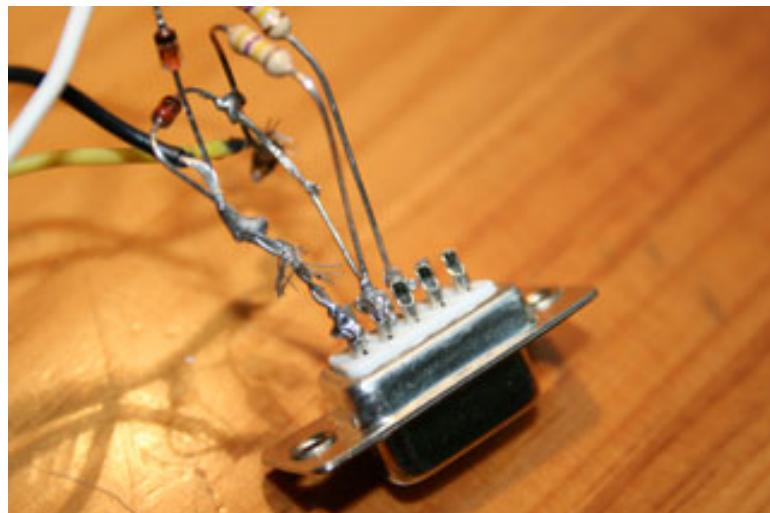


Figura 7: Interfaz

Se soldaron cables de colores diferentes a estas conexiones, se protegió el circuito de resistencias y diodos con cinta aislante y se cerró el conector DB9. En la figura 7 observamos la interfaz con los cables conectados.

Posteriormente las conexiones SDA y SCL se conectaron a la memoria EPROM 24LC en las patas 5 (SDA) y 6 (SCL) correspondientemente. Como se muestra en la figura 8.



Figura 8: Conexion de la interfaz

Una vez que se conectó la interfaz a la EPROM y al puerto serial de la PC, para leer la EPROM se utilizó el software PonyProg v1.17h, que es de libre distribución. Cuando se inicia el programa aparece la ventana de calibración, pulsamos el botón de Si para que el programa haga la calibración de manera automática. Una vez calibrado entramos en Setup y ajustamos como muestra la imagen siguiente, el puerto COM1, después arriba en Dev. Type seleccionamos 2402-16 y 2402.

Para ejecutar el programa y obtener la lectura, primero se enciende la consola conectada con la interfaz y posteriormente en el PonyProg se selecciona la opción de Read device, enseguida se ve que comienza con la lectura de la EPROM y poco después aparece la lectura obtenida. La figura 9 muestra la lectura obtenida de la EPROM de la consola Leapster TV.

000000)	50 D9 1E 3C 4C D9 1E 3C - C4 8F 80 01 00 01 00 3C	P..<L..<.....<
000010)	AC D9 1E 3C A8 00 00 - 00 00 00 00 6B 61 74 73	...<.....kats
000020)	6B 61 74 73 6B 61 74 73 - 6B 61 74 73 6B 61 74 73	katskatskatskats
000030)	6B 61 74 73 6B 61 74 73 - 6B 61 74 73 6B 61 74 73	katskatskatskats
000040)	6B 61 74 73 6B 61 74 73 - 6B 61 74 73 6B 61 74 73	katskatskatskats
000050)	6B 61 74 73 6B 61 74 73 - 6B 61 74 73 6B 61 74 73	katskatskatskats
000060)	00 00 7F 80 6B 61 74 73 - 6B 61 74 73 6B 61 74 73katskatskats
000070)	6B 61 74 FD AB 00 00 00 - FF FF FF FF 00 00 00 00	kat.....
000080)	01 20 00 00 00 00 00 00 - 02 00 00 00 00 00 00 00	-
000090)	00 00 00 00 4C 79 00 40 - E4 D8 1E 3C 28 D9 1E 3CLy.@...<(..<
0000A0)	BC EC 1F 3C A8 1D 0C 40 - 00 00 00 00 9A FE 0A 40	...<..@.....@
0000B0)	A2 FE 0A 40 3A 71 03 40 - 06 00 00 00 3A 71 03 40	...@:q.@....:q.@
0000C0)	00 00 00 00 34 0C 05 40 - 38 0C 05 40 3A 71 03 404..@8..@:q.@
0000D0)	06 00 00 00 3A 71 03 40 - 00 00 00 00 12 95 05 40:q.@.....@
0000E0)	01 00 00 00 01 00 00 00 - 0C 00 01 00 01 00 00 00
0000F0)	04 7A 1E 3C 00 00 00 00 - 0C 00 00 00 80 5C 91 0C 40	.z.<.....\..@

Figura 9: Resultado Ponyprog

Como observamos, la lectura de la EPROM (Figura 9) consta de un conjunto de datos hexadecimales del lado izquierdo

y de lado derecho un conjunto de caracteres y símbolos sin ninguna secuencia. La lectura obtenida se guardó con extensión .bin y posteriormente se le aplicó un proceso de desencriptación XOR con un script de bash, pero no obtuvimos ninguna secuencia lógica de caracteres o direcciones, ni nada que pudiera servir útil para interpretar esos datos.

2.3. Ingeniería Inversa No Destructiva



Figura 10: Macronix fabrica la ROM para Leapster

Las pruebas de ingeniería inversa no destructiva ofrecen la alternativa de no dañar los componentes de la consola de videojuegos. El proceso pretende hacer una lectura de la memoria ROM sin dañar el cartucho. Primero se investigó acerca del fabricante de las memorias ROM para estos cartuchos, ya que con esta información se puede deducir la correspondencia de pines del cartucho con la hoja de datos del posible integrado.

Se puede apreciar en la figura 10, que el cartucho de videojuego Leapster TV utiliza una ROM de Macronix [8], una empresa que se dedica a fabricar y proveer memorias ROM a nivel mundial, y esta tiene una capacidad de 64 Megabits. Con la información anterior obtuvimos la hoja de datos que se muestra en la figura 11.

48 TSOP (NORMAL TYPE)



Figura 11: Hoja de datos de la ROM

Con la hoja de datos técnicos de la ROM (figura 11) se pueden relacionar los pines de los cartuchos con las salidas del integrado y esto se hace con una prueba de continuidad entre éstos.

En las figuras 12 y 13 se puede apreciar las líneas identificadas y los pines correspondientes. Las líneas amarillas corresponden al bus de direcciones, mientras que las rojas al de datos. Con esta información el siguiente paso es crear una interfaz que nos permita enviar diferentes señales a las líneas de direcciones para ubicar en qué posición de memoria se encuentran, obteniendo así la estructura del archivo ROM.

2.4. Ingeniería Inversa Destructiva

Las pruebas de ingeniería inversa destructiva las hicimos sobre otros componentes que están hechos del mismo material epóxico y estas consistieron en aplicar ácidos, exposición de componentes a fuego directo (ejemplo Figura 14) y someterlos al calor a altas temperaturas. Se realizaron estas pruebas porque ofrecían la ventaja de saber con exactitud la distribución de las líneas de buses con la salida de pines en el cartucho. Las pruebas con ácidos se realizaron en el laboratorio de Fisicoquímica en la Universidad del Caribe, con la colaboración de Juan Francisco Bárcenas. La idea original se tomó de un video [6] donde se describen ciertos pasos para aplicar ingeniería inversa a unas smartcards, y

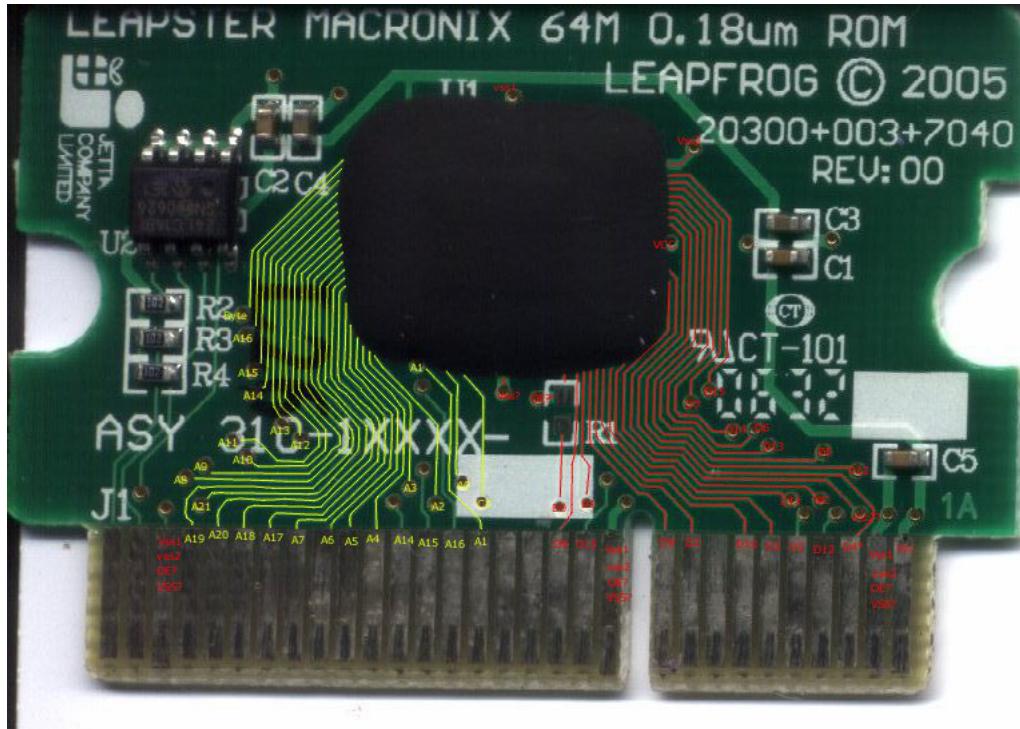


Figura 12: Pines identificados cartucho de Cars

básicamente explica como someten las tarjetas a ácido nítrico para después limpiarlas con acetona y posteriormente ver los resultados en un microscopio. También en un blog[7] se mencionan una serie de pasos para retirar epóxico a base de ácidos.

Los ácidos utilizados para las pruebas son: Ácido Nítrico (HNO_3) con una concentración de 61.5 %, Ácido Clorhídrico (HCl) con una concentración de 36.5 % y Ácido Sulfúrico (H_2SO_4) con una concentración de 95.5 %. Al terminar con estas pruebas, la del ácido sulfúrico muestra que se puede remover el epoxico, la muestra estuvo expuesta a 72 horas. A pesar de esto el ácido tambien daño los buses y la placa.

Posteriormente se hacen pruebas destructivas a un componente metiéndolo a una mufla, un horno eléctrico que trabaja a altas temperaturas y se usan para hacer pruebas de residuos de ignición. La prueba inicial se hizo a 500

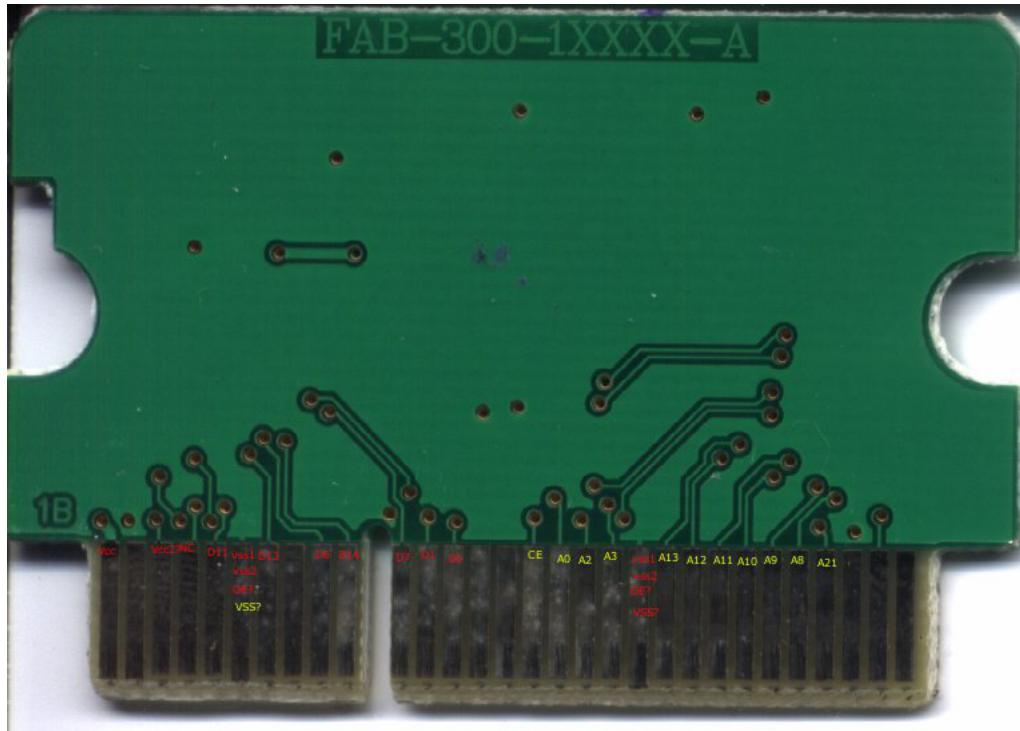


Figura 13: Pines identificados en la parte posterior del cartucho de Cars

grados centígrados. Posteriormente se duplica la temperatura a 1000 grados por casi mas de 2 horas, obteniendo el resultado que se muestra en la Figura 15. Esta prueba logró carbonizar el epóxico, dejando expuesto el núcleo del integrado, sin embargo, la alta temperatura también dañó el núcleo del circuito.

2.5. Conclusiones

Para la realización de ingeniería inversa a hardware, aún no existe una metodología con pasos bien establecidos a seguir, que garanticen el éxito, sino que de acuerdo al tipo de dispositivo varían los métodos y procedimientos que se pueden aplicar para lograrlo. Los procesos de ingeniería inversa aplicados a la consola Leapster TV han servido para identificar sus dispositivos principales dentro de la tarjeta interna y con el análisis e investigación de los mismos, tener una idea sobre el rol de importancia de cada uno, su funcionamiento específico, la conexión entre ellos, el funcionamiento general de la consola y sobre todo dar una idea de la interacción consola-cartucho.



Figura 14: Resultado de la tarjeta



Figura 15: Resultado del horno a 1000 grados

En lo correspondiente a resultados, podemos decir que no se logró implementar la forma de comprobar si la configuración de los pines obtenidos con las lecturas de continuidad en el cartucho o la lectura de la EMPROM 24LC de la consola, resultaron información suficiente para poder hacer cartuchos compatibles con Leapster TV, por lo tanto no se llegó a aprobar o desaprobar nuestra hipótesis planteada en el protocolo de investigación.

Referencias

- [1] Definicion de ROM recuperada de
<http://www.kennetholsen.cl/ROM>
- [2] Definicion de Ingenieria Inversa, recuperado de
<http://www.alegsa.com.ar/Dic/ingenieria%20inversa.php>
- [3] Ley de la propiedad industrial, recuperado de
http://www.sice.oas.org/int_prop/nat_leg/Mexico/lipmexsa.asp#tit1
- [4] Trabajo de ingenieria inversa sobre Didj, Recuperado de
<http://sites.google.com/site/claudeschwarz/didjhacking2>
- [5] Foro de la comunidad de desarrolladores, recuperada de
http://elinux.org/Leapster_Explorer)
- [6] Video sobre Ingenieria inversa a unas smart cards, recuperado de
<http://www.youtube.com/watch?v=tnY7UVyaFiQ>
- [7] Retirar epoxico a base de acidos, recuperado de
<http://travisgoodspeed.blogspot.com/2009/06/cold-labless-hno3-decapping-procedure.html>
- [8] Pagina oficial de Macronix, recuperado de
<http://www.mxic.com.tw/QuickPlace/hq/Main.nsf>
- [9] Gaceta del IMPI, Recuperado de
http://sigainfo.impi.gob.mx/wb/SIGA/SIGA_detalles_ficha?&ID_FICHA=1990588