

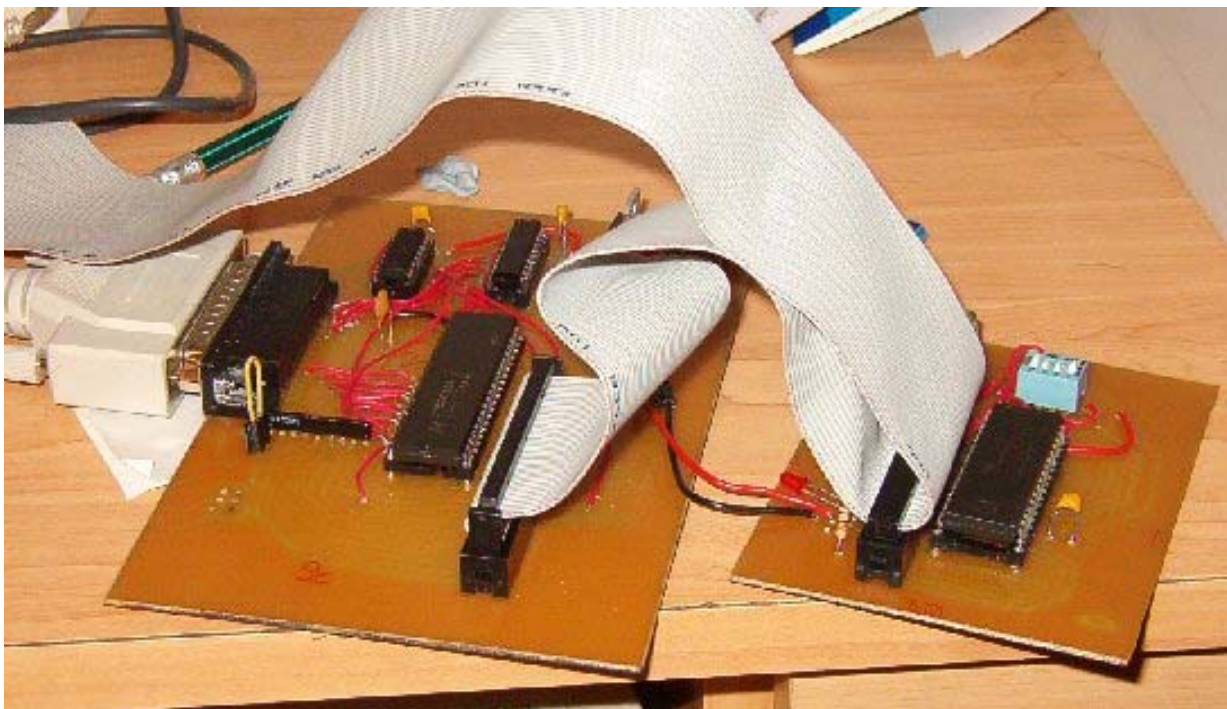
Programador Flash

Por : David Senabre (Patroclus), 2004 -2006

Versión 2. 20/10/07

Introducción:

Este proyecto nació como una necesidad de poder grabar memorias de algún tipo. Después de mucho pensarlo, leer y aprender todo lo que pude sobre memorias, y analizar varios programadores, decidí que lo mejor era diseñar un programador desde cero para memorias flash tipo 29F0x0, de AMD y Atmel, principalmente. Al usar sólo un tipo de memoria, el diseño se simplificaba, y también el software que debía programar para controlarlo. Escogí estas memorias por tener un precio asequible, ser no volátiles, regrabables, y requerir sólo 5V para programarlas. Aunque parezca muy artesanal, la implementación final fue en placa de cobre, con pistas por la cara inferior, unos pocos cables por la parte superior, y unos resultados excelentes de funcionamiento.



Este documento sirve como guía para todo aquel que quiera construir un programador para memorias flash, completamente probado y con software para windows XP. Todo de cosecha propia :)

Como ya he comentado, el diseño es genuino.

No daré todos los detalles paso por paso, pero daré la suficiente información como para que su construcción pueda llevarse a cabo sin muchos problemas.

Doy por hecho que quién se enfrente a este documento sabe lo básico sobre memorias, que sabe lo que es un bus de datos, de direcciones, y está familiarizado con la electrónica digital, sabe lo que es una resistencia, voltaje, y cosas así.

Vista general de mi diseño.

Decidí realizar el programador de forma modular, dejando toda la lógica de control en una placa, y conectando los dispositivos a grabar a esta placa "madre" a través de un conector de 40 vías, como el que usan los dispositivos IDE en los PC (*ver foto*).

De este modo el programador podría valer para programar diversos dispositivos, o acceder a memorias SRAM, o a las ROMs de cartuchos de consolas (con el conector

adecuado), para realizar backups de los originales, etc. En definitiva, sirve para leer y grabar dispositivos de memoria de 8 bits de hasta 4 Mb de capacidad (o 32Mbits).

En el proyecto actual sólo fabriqué un módulo con zócalo DIP de 32 pines, para programar memorias 29F0x0 de AMD y Atmel, de hasta 4 Mbits (*placa pequeña en la foto*), y he probado ambas con éxito y excelentes resultados. Esta placa se conecta a la placa de control a través de un cable plano de 40 hilos muy fácil de conseguir o hacer.

El tiempo de lectura ronda menos de un minuto para leer una 29F040 (de 512Kb, la mayor memoria que acepta este módulo), y poco más de 6 minutos para grabarla. Aunque dependiendo del contenido de lo que se quiera grabar puede tardar menos (el software evita hacer ciertas lecturas redundantes). Aún se podría mejorar este tiempo revisando el software, que escribí rápidamente, aunque su funcionamiento básico está garantizado (detectar dispositivo, leer, borrar y grabar). He visto otros programadores “caseros” de estas características por Internet, y a excepción de uno, que es ligeramente más rápido, el resto tardaban incluso más del doble en grabar el mismo dispositivo. No está nada mal, ya que esté ha sido mi primer proyecto serio diseñado desde cero, hardware y software. Así que estoy contento con él ;-)

Sólo comentar que no es un proyecto fácil y asequible para novatos, o gente sin experiencia. Hay que estar dispuestos a tomárselo con calma, y saber que tardarás un tiempo en construirlo. Pero te ahorras un dineral, ya que los programadores de eproms son carisimos.

Comenzando su construcción.

Algunos detalles.

La placa madre, y toda la lógica del programador, está contenida en 3 chips.

- * Un 8255 (para generar direcciones).
- * Un decodificador 3:8 (para generar las señales de lectura y escritura).
- * Biestables D (para generar señales de control del 8255, excepto de lectura y escritura).

El corazón del circuito es el **chip de entrada/salida programable 8255**. Es un chip con un puerto de entrada de 8 bits, y 3 puertos de salida de 8 bits. Permite mandar datos metidos por el puerto de entrada a cualquiera de los puertos de salida, o al revés, entre otras funciones.

Así pues, el 8255 se conecta al puerto paralelo del PC, de 8 bits, y permite mandar los datos enviados desde el PC, a cualquiera de tres puertos de salida del chip de 8 bits cada uno, lo que suma 24 bits en total (8x3). Esto lo uso para direccionar la memoria a leer/grabar. Así se consigue dar una dirección de 24 bits con sólo 8 bits del puerto paralelo, cargando de 8 bits en 8 bits la dirección, por el puerto de entrada. Con 24 bits podemos direccionar 16Mb. Pero en mi diseño sólo utilicé 22 líneas, que dan un máximo de 4Mb.

Para usar así el 8255, hay que configurarlo en ese modo, y decirle que los 3 puertos de salida serán de salida, ya que podrían no serlo. Es decir, el 8255 se puede configurar en varios modos de funcionamiento. Quien quiera aprender más del 8255, que busque en Internet. En este proyecto, su funcionamiento es muy sencillo, y muy instructivo a la hora de aprender a manejarlo.

Para controlar las señales de lectura y escritura de la memoria flash, y a la vez controlar las del chip 8255, usando las líneas de control del puerto paralelo, utilicé un **decodificador 3 a 8**. Así puedo controlar las señales de lectura y escritura de la memoria (/RD y /WR), la de escritura del 8255, y una señal de reloj, usando sólo 3 líneas en lugar de 4. Esto fue posible porque me di cuenta que podía construir el circuito de manera que sólo una de estas señales (/RD, /WR de la Flash, /WR del 8255 y el reloj) estuviera activada a la vez (ej: no se puede leer a la vez la memoria y el 8255, o dar un pulso de reloj a la vez que se lee o escribe). Este requisito no es algo muy restrictivo.

El objetivo de hacer esto es dejar libre una de las líneas de control del puerto paralelo para futuros usos. La señal de reloj se usa para cargar un último chip de **biestables D** que controlan el 8255 (en concreto, su señal de reset, y el puerto activo). Estos biestables se cargan desde 3 de los bits del bus de datos, a la señal del reloj. Y esto lo hago así porque mientras se lee y se escribe la memoria, que se hace con el bus de datos también, no se debe afectar al contenido de los biestables que controlan el 8255.

Además, al usar chips como intermediarios, evito que la excesiva longitud del cable del puerto paralelo pueda generar ruido y un incorrecto funcionamiento del programador (los chips actúan como “repetidores” de la señal que llega del puerto paralelo).

A pesar de todo, se debe tener cuidado de no usar cables largos, porque los datos leídos y escritos en la memoria flash se hacen directamente desde y hacia el puerto paralelo, sin chip intermediario. Por lo tanto, recomiendo usar un cable de puerto paralelo de 1 metro a ser posible, y un cable de 40 vías para conectar los módulos, lo más corto posible.

El esquemático de la placa de control es el siguiente;

<http://www.consolasparasiempre.net/taller/programador%20sch.png>

Comentarios sobre el esquemático

Las señales que dicen DATA BUS, y todas las de direcciones que salen del 8255 (las del lado derecho), además de la de /WR y /RD, deben ir al conector de 40 vías de la placa de control. Es decir, éstas serán las líneas que se llevarán a las placas de programación a través de un cable plano, por ejemplo.

Pero si sólo deseas programar un tipo concreto de memorias, por ejemplo Flash roms del tipo 29F0x0, de un mismo formato (DIP o PLCC), que es el objetivo primordial de este programador, puedes no querer hacer 2 placas, sino meter todo en una placa y olvidarte del conector de 40 vías. Como he dicho, el objetivo de tener 2 placas, es que manteniendo la misma placa de control, se pueden hacer y conectar otras placas de programación para leer o escribir otros tipos de memorias (eso sí, de momento el software sólo soporta Flash roms del tipo 29F0x0). Es decir, las placas pequeñas sólo llevan un zócalo o conector para un tipo de memoria, mientras la placa de control lleva todo el circuito y la lógica de programación.

Para quienes quieran usar el diseño modular de varias placas, debo decir que por facilitar la construcción, no pude incluir las líneas de alimentación y tierra en el conector de 40 pines, por lo que se deben llevar a las placas de programación por otro medio. Yo elegí el sencillo método de 2 cables soldados a un pequeño conector, como puede verse en las fotos.

No daré más detalles sobre la construcción de la placa de programación, ya que creo que cada uno puede hacerle a su conveniencia, y no tiene ninguna complicación adicional. Es sólo conectar pin a pin las señales del conector de 40 pines, a los pines del zócalo o conector de la memoria. Sólo recuerda añadir un condensador cerámico de 100nF cerca del zócalo o conector.

En las fotos de mi placa, se ve un interruptor DIP de 4 interruptores (azul y blanco), que sirve para poder leer eproms de 28 pines en el zócalo de 32 pines. Hace un sencillo recableado, nada más. Basta con comparar el pinout de una eprom de 32 pines con el de una de 28 pines e idear una manera de llevar a los pines que difieren, las señales correctas. Tampoco daré más detalles porque en principio es algo opcional que cada uno puede encontrar útil o no.

Si alguien necesita ayuda adicional, que me lo consulte.

Consideraciones prácticas

Aconsejo usar condensadores cerámicos de 100nF cerca de cada integrado, entre las patas de alimentación y tierra. Sirven para filtrar ruido, que podría hacer funcionar mal tanto algún integrado como el programador en sí. Está comprobado científicamente, no es una chorrada. De hecho, cuando el programador estaba en estadios de desarrollo, y después de romperme la cabeza para solucionar diversos problemas, sólo cuando puse los condensadores, se dignó a funcionar correctamente.

Para mejorar la compatibilidad de ciertos puertos paralelos, que por regla general son bastante quisquillosos, recomiendo usar una resistencia SIP de 8 resistencias de 4.2Kohmios. Ésta se deberá colocar en las 8 líneas de datos del puerto paralelo, y el pin común de la resistencia SIP a tierra.

Para la alimentación se requieren 5V. Así que para poder usar un adaptador de corriente de 9 o 12V, muy comunes, hay que usar un regulador de voltaje de 5V, como el 7805, para estabilizar el voltaje. El 7805 es un encapsulado con 3 patas, una de entrada de voltaje, a una tensión de 7.5V a unos 20V, la pata central se conecta a tierra, y por la pata restante se obtienen 5V. Eso sí, recomiendo no usar voltajes de entrada de más de 12V porque el 7805 se calienta bastante.

Pero para garantizar un funcionamiento perfecto, hay que añadir algún elemento más al regulador. Hay muchos ejemplos en Internet de circuitos con el 7805, bastante sencillos. Pero para el programador yo probé con uno especialmente sencillo, que como va de lujo, es el que usé. Básicamente, coloca un condensador, electrolítico de 100uF y al menos 25V, entre la pata de tierra y de entrada de voltaje del 7805. También puedes añadir un diodo como el 1N4004 como medida de seguridad. Recomiendo usar un adaptador de corriente de 9V o 12V con este esquema.

Notas sobre las líneas de dirección:

Para simplificar el cableado en el PCB final que yo diseñé, he usado las siguientes conexiones. La columna de la izquierda corresponde con los pines del bus de dirección de la memoria Flash, y la columna de la derecha, los pines del 8255 a los que están unidos. Recordar que es el 8255 el que genera las direcciones.

Teniendo en cuenta que para el 8255;

A00 – A07 se refiere a P00-P07 (puerto 0, bits 0 a 7)

A08 – A15 se refiere a P10-P17 (puerto 1, bits 0 a 7)

A16 – A23 se refiere a P20-P27 (puerto 2, bits 0 a 7)

ROM	<----->	8255
A0	-	A15
A1	-	A14
A2	-	A13
A3	-	A12
A4	-	A11
A5	-	A10
A6	-	A09
A7	-	A08
A8	-	A01
A9	-	A02
A10	-	A04
A11	-	A03
A12	-	A19
A13	-	A00
A14	-	A21
A15	-	A18
A16	-	A17
A17	-	A20
A18	-	A16

Con este aparentemente caótico cableado, he conseguido trazar casi todas las pistas de datos y dirección en una sola cara de circuito impreso. Si quieres usar mi programa para flashear, deberás conectar de este modo las salidas del 8255 a las líneas de dirección del zócalo de la ROM. Si decides fabricar el circuito de otra manera y unir las líneas de dirección en su orden natural, entonces ponte en contacto conmigo y te suministraré el programa modificado para ello, si aún no lo he colgado en la Web. Es decir, no estás obligado a hacer este cableado. Pero sobretodo si vas a hacer un PCB con pistas de cobre, plantéatelo, porque simplifica mucho.

Pinout del 8255:

	+--	--+	
P03	1 \ / 40		P04
P02	2 39		P05
P01	3 38		P06
P00	4 37		P07
RD	5 36		WR
CS	6 35		RESET
GND	7 34		D0
A1	8 33		D1
A0	9 32		D2
P27	10 31		D3
P26	11 30		D4
P25	12 29		D5
P24	13 28		D6
P20	14 27		D7
P21	15 26		Vcc
P22	16 25		P17
P23	17 24		P16
P10	18 23		P15
P11	19 22		P14
P12	20 21		P13
	+-----+		

A0 y A1 sirven para seleccionar el puerto activo.

P0x se refiere al puerto 0.

P1x se refiere al puerto 1.

P2x se refiere al puerto 2.

Para más información sobre el 8255 buscar en Internet. El uso que le doy en este proyecto es muy sencillo e intuitivo cuando se entiende el 8255.

Pinout del conector

Como ves, este programador se basa en una placa madre, o de control, que contiene la electrónica necesaria para generar las direcciones y las señales de control a partir de un puerto paralelo. A esta placa se le conectan módulos, es decir, otras placas que contienen el dispositivo en cuestión que vamos a programar. Esto lo elegí así para facilitar la ampliación de las capacidades del programador.

Pues bien, para conectar la placa madre a los módulos hay que usar algún tipo de conexión. Decidí usar un conector de 40 pines en 2 hileras, de 2.54mm de separación. Es decir, exactamente el mismo que el que usan los discos duros y las unidades CD/DVD IDE. De esta forma, para conectar la placa madre a los módulos, se puede usar un cable plano IDE estándar. A este conector lo vamos a llamar a partir de ahora **FlashBUS**.

¿Cómo conectar cada pin del 8255 en la placa de control a su conector de 40 pines, que vamos a llamar *FlashBUS*? Así.

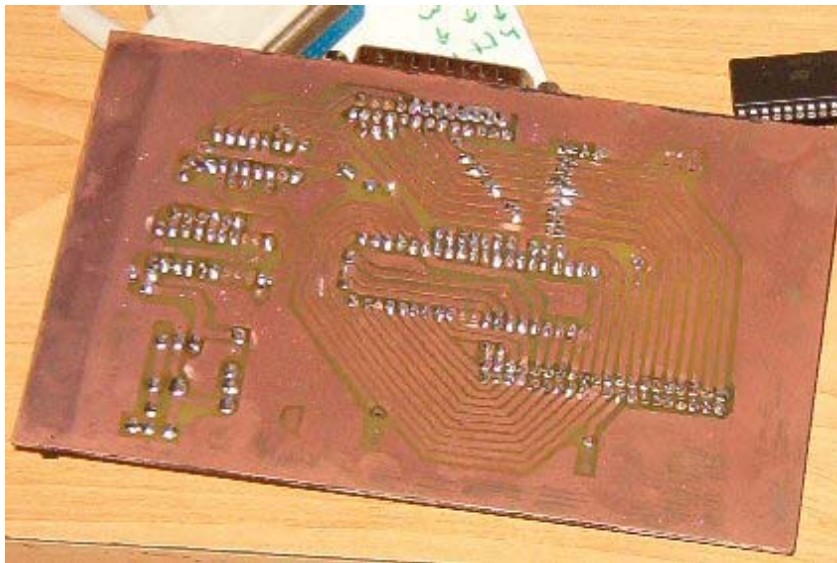
A16	A17	A18	A19	A8	A9	A10	A11	A12	A13	A14	A15	-	-	-	-	-	* Ctr	* A22	* A23
/OE	/WR	A20	A21	A0	A1	A2	A3	A4	A5	A6	A7	D0	D1	D2	D3	D4	D5	D6	D7

En el módulo de programación de memorias Flash, el pinout será el siguiente. En él se muestran qué pines del zócalo deben ir a cada pin del *FlashBUS*.

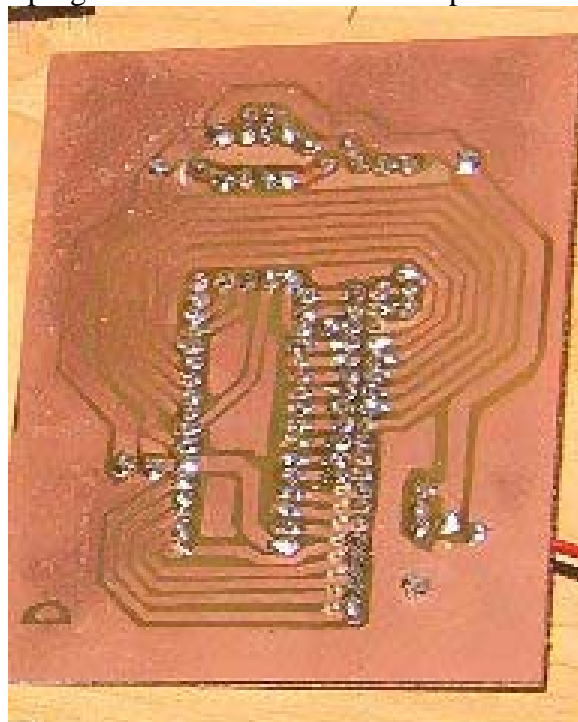
Placa DIP32 para memorias 29F0x0

A18	A16	A15	A12	A7	A6	A5	A4	A3	A2	A1	A0	-	-	-	-	-	-	-	-
/OE	/WR	A17	A14	A13	A8	A9	A11	A10	-	-	-	D0	D1	D2	D3	D4	D5	D6	D7

La placa de control acabada me quedó así (la calidad de la fotos es regular):



Y la placa con el zócalo para programar flash roms DIP de 32 pines:



Versión 2 de la placa de control

En este documento describo la forma de construir un módulo para programar memorias Flash. Pero posteriormente a la construcción de la placa madre, diseñé otro módulo para leer/escribir cartuchos de Super Nintendo, que requería de ciertas capacidades hardware que mi placa de control no tenía, por lo que decidí modificarla ligeramente. No necesitas tenerlo en cuenta si únicamente quieres grabar memorias Flash de hasta 512 Kbytes.

Pinout de FlashBUS de la placa madre v.2.

A16	A17	A18	A19	A8	A9	A10	A11	A12	A13	A14	A15	-	-	-	-	-	* Ctr	* A22	* A23
/OE	/WR	A20	A21	A0	A1	A2	A3	A4	A5	A6	A7	D0	D1	D2	D3	D4	D5	D6	D7

(*) líneas añadidas en la versión 2.

La señal *Ctr* no proviene del 8255 sino del puerto paralelo.

Lista de componentes para la placa de control.

- 1 x zócalo de 40 pines ancho.
- 1 x zócalo de 20 pines estrecho.
- 1 x zócalo de 16 pines estrecho.

- 1 x 8255 (chip I/O)
- 1 x 74LS138 (decodificador)
- 1 x 74LS374 (biestables D)
- 1 x 7805 (regulador de tensión)

- 1 x condensador 220 uF electrolítico de 16 o más voltios.
- 3 x condensadores 100 nF cerámicos.
- 1 x resistencia 330 ohms.
- 1 x led de 3 o de 5 mm.
- 1 x diodo 1N4004 o equivalente.

- 1 x conector DB25 macho (puerto paralelo)
- 1 x conector de 40 pines en 2 hileras de separación 2.54mm (0.1 pulgada)
- 1 x conector de alimentación del mismo tipo que del adaptador que vayáis a usar para alimentar la placa.

Lista de componentes para la placa de programación Flash DIP32 básica.

- 1 x zócalo de 32 pines ancho.
- 1 x condensadores 100 nF cerámicos.
- 1 x resistencia 330 ohms.
- 1 x led de 3 o de 5 mm.
- 1 x conector de 40 pines en 2 hileras de separación 2.54mm (0.1 pulgada)