

# MANUAL AT89C51 PROGRAMMER

■ R. JEYARAMAN AND R. LAKSHMANAN

For hand coding of AT89C51 microcontroller, here is a simple programmer to program binary data into the microcontroller. It doesn't require use of a computer and all the control signals and data are entered manually. The programmer can erase, read and write data into the flash memory of the microcontroller. It can also read the signature byte of the microcontroller.

## Circuit description

Fig. 1 shows the circuit of the manual programmer. It uses timer NE555 (IC1) wired as a monostable to increment the address location one by one in conjunction with counter CD4040.

When switch S1 is pressed, IC1 generates a clock pulse, which is given to clock pin 10 of 12-bit binary counter CD4040 (IC2). The 12-bit counter is used for generating the 12-bit binary address for AT89C51 microcontroller IC. The microcontroller to be programmed is inserted into the ZIF socket for programming, reading and erasing. During flash programming and verification, port 1 receives low-order address bytes, while port 2 receives high-order address bits and some control signals.



Q0 through Q11 outputs of IC2 are

connected to pins 1 through 7 of port 1 and pins 21 through 24 of port 2 of the microcontroller. These pins act as the address pins (A0 through A11) of the microcontroller. The outputs of IC2 are also connected to LED1 through LED12 via current-limiting resistors R4 through R15, respectively. LED1 through LED12 indicate address location for the microcontroller IC. Switch S2 is used to reset the counter (IC2).

Port 0 receives code bytes during flash programming and outputs them during program verification. Port-0 pins 39 down to 32 are used as data pins for the microcontroller, and connected to pins 1 through 8 of DIP switch SW1 as well as LED13 through LED20 via current-limiting resistors R16 through R23. The LEDs indicate data at/for the addressed location of the microcontroller. Port pins 16, 17, 27 and 28 are controlled by slide switches S6, S7, S8 and S9, respectively, as per the table given below.

Programming pulse ( $\overline{ALE}/\overline{PROG}$ ) and programming voltage ( $\overline{EA}/V_{pp}$ ) are derived with the help of IC3 and transistors T1 through T3. Timer IC3 generates and determines the programming and erasing pulse time. It is configured as a monostable whose time period is decided by resistor R24 and capacitors C6 and C7. Capacitors C6 and C7 are selected by slide switch S4 for programming and erasing,

## Flash Programming Modes

Mode	Pin 9	Pin 29	Pin 30	Pin 31	Pin 27	Pin 28	Pin 16	Pin 17
	RST	PSEN	ALE/ $\overline{PROG}$	$\overline{EA}/V_{pp}$	P2.6	P2.7	WR	RD
Write code data	H	L		H/12V	L	H	H	H
Read code data	H	L	H	H	L	L	H	H
Chip erase	H	L	 (1)	H/12V	H	L	L	L
Read signature byte	H	L	H	H	L	L	L	L

Note: 1. Chip erase requires a 10ms PROG pulse

## PARTS LIST

### Semiconductors:

IC1, IC3	- NE555 timer
IC2	- CD4040 12-bit binary counter
IC4	- 7812 12V regulator
IC5	- 7805 5V regulator
IC6	- AT89C51 microcontroller
T1-T3	- BC548 npn transistor
D1-D4	- 1N4007 rectifier diode
D5	- 1N4148 switching diode
LED1-LED12,	
LED22-LED29	- Red LED
LED13-LED20	- Green LED
LED21	- Yellow LED

Resistors (all 1/4-watt,  $\pm 5\%$  carbon, unless mentioned otherwise):

R1-R3,	
R31-R33	- 10-kilo-ohm
R4-R23	- RNW1, RNW2, RNW3
R35-R42	- 220-ohm
R24, R26	- 1-kilo-ohm
R27, R29	- 2.7-kilo-ohm
R28	- 15-kilo-ohm
R30	- 100-ohm
R34	- 4.7-kilo-ohm
R25	- 12-kilo-ohm
RNW4	- 4.7-kilo-ohm $\times 8$ (SIP9)

### Capacitors:

C1	- 0.47 $\mu$ F, 16V electrolytic
C2, C8	- 0.01 $\mu$ F ceramic disk
C3, C4, C12,	
C13	- 33pF ceramic disk
C5, C9, C10	- 0.1 $\mu$ F ceramic disk
C6	- 10 $\mu$ F, 16V electrolytic
C7	- 1 $\mu$ F, 16V electrolytic
C11	- 1000 $\mu$ F, 35V electrolytic

### Miscellaneous:

X1	- 230V AC to 15V, 300mA secondary transformer
X <sub>TAL</sub>	- 4MHz crystal
S1-S3	- Push-to-on switch
S4-S9	- Slide switch
SW1	- 8-way DIP switch
	- 40-pin ZIF socket

respectively. The triggering pulse is applied through a high-pass R-C network (comprising C5 and R28) and diode D5.

Output pin 3 of IC3 is connected to the base of transistor T1 via resistor R27. The high pulse output of IC3 drives transistor T1 and provides low pulse to pin 30 of the microcontroller. LED21 glows to indicate application

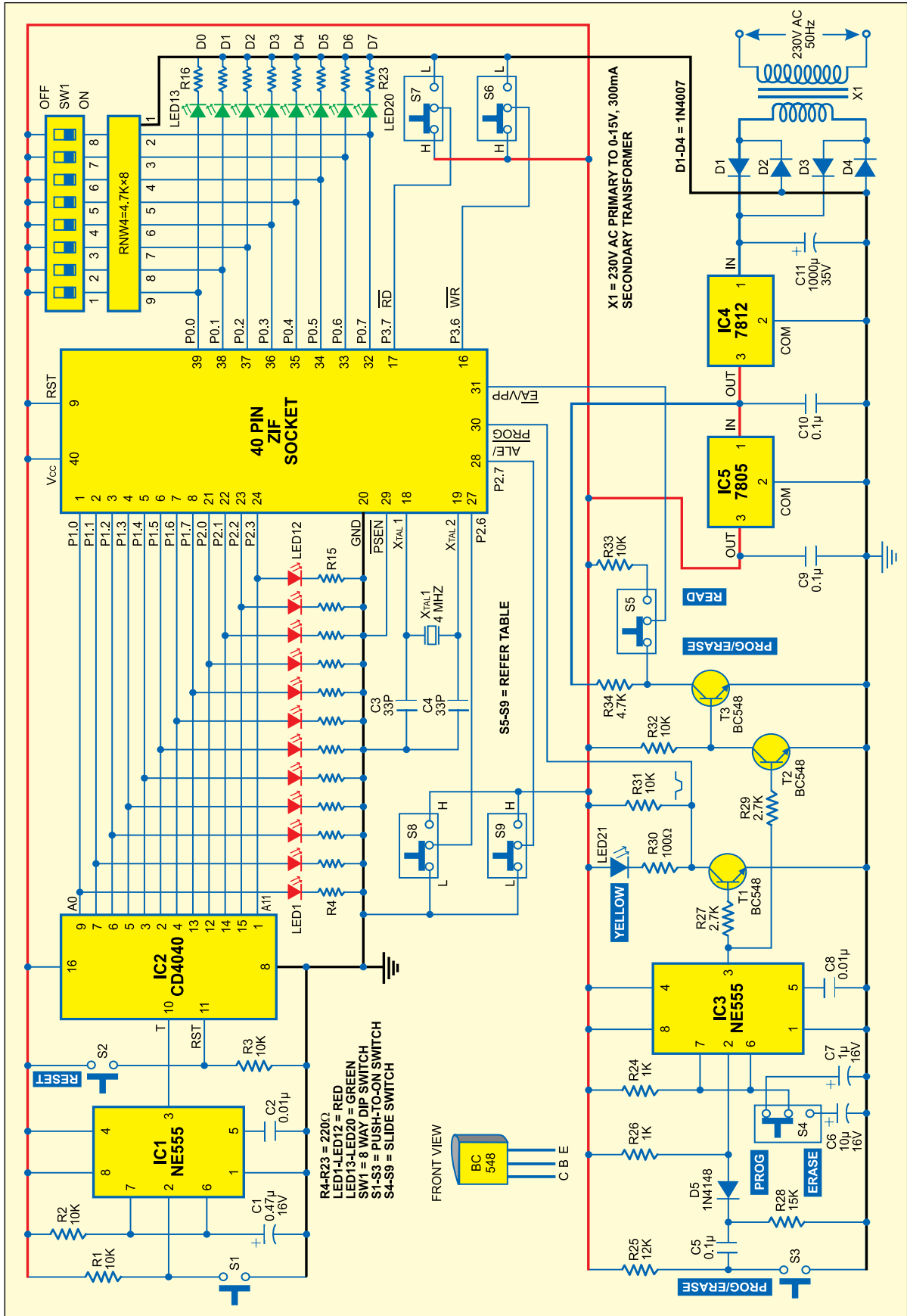


Fig. 1: Circuit diagram of manual AT89C51 programmer

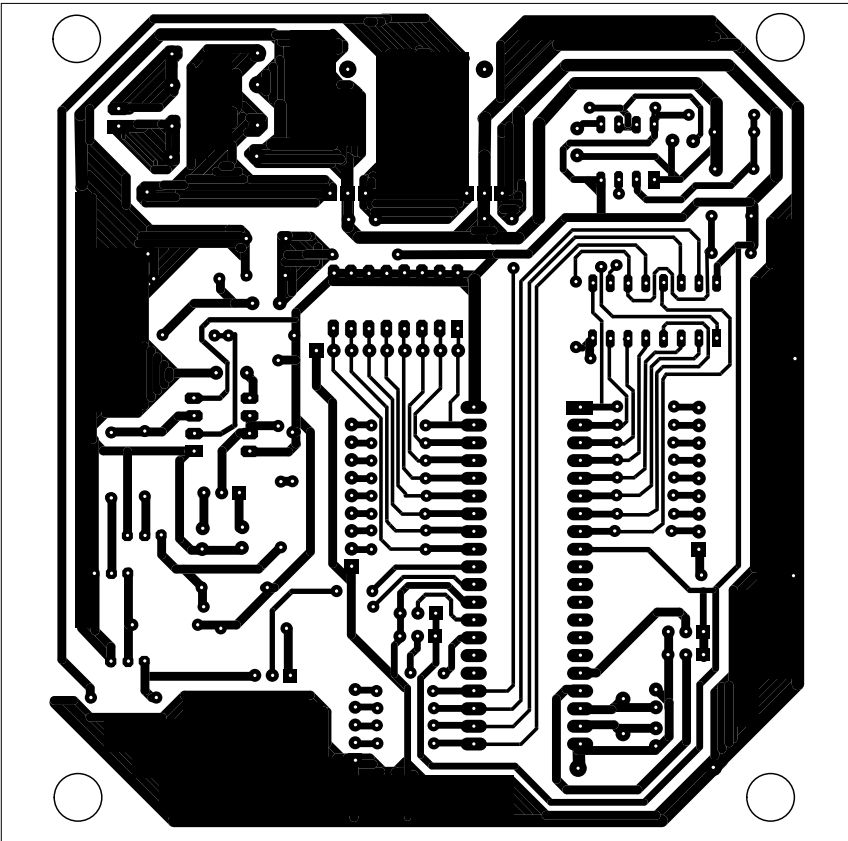


Fig. 2: Actual-size, single-side PCB layout for manual AT89C51 programmer of Fig. 1

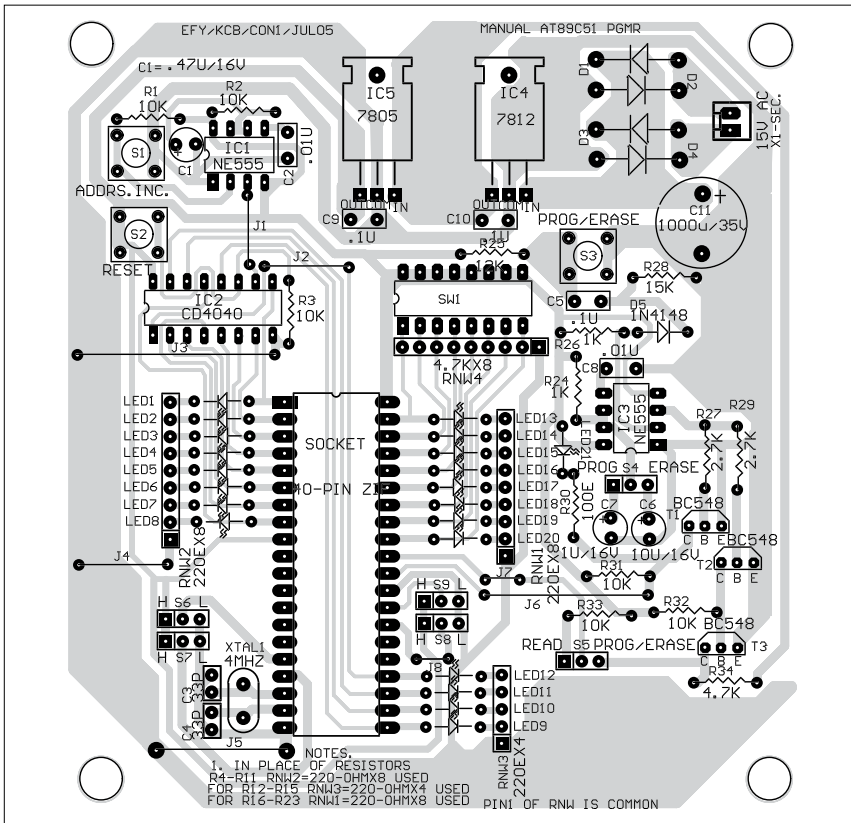


Fig. 3: Component layout for the PCB in Fig. 2

of the programming/erasing pulse. IC3 also determines the time period for application of the programming/erasing voltage (12V) to the microcontroller. When its output goes high, npn transistor T2 conducts and its collector goes low. As a result, transistor T3 is cut off and its collector voltage rises to around 12V. Slide switch S5 is used for selecting 12V or 5V as required.

The AC mains is stepped down by transformer X1 to deliver a secondary output of 15V at 300 mA. The transformer output is rectified by a full-wave bridge rectifier comprising diodes D1 through D4, filtered by capacitor C11, and regulated by IC4 and IC5 to provide regulated 12V and 5V supplies, respectively. Capacitors C9 and C10 bypass any ripple in the regulated outputs.

## Programming

The AT89C51 microcontroller is programmed byte by byte. Before programming, the flash memory must be erased completely and control signals should be set according to the table for erasing, programming and reading the microcontroller as follows.

**Chip erase.** Move slide switch S4 towards Erase position. The high output at pin 3 of IC3 provides a low pulse of around 10 milliseconds (ms) available at pin 30 of the microcontroller (ALE/PROG). Now move slide switch S5 towards Prog/Erase position for programming voltage (12V). The entire flash array can be erased electrically by setting pins 28, 16 and 17 to low level using switches S9, S6 and S7, respectively, and pin 27 at high level using switch S8. Insert the microcontroller into the ZIF socket and press switch S3 (LED21 glows to indicate application of the programming pulse) to erase the flash array (reset to all 1's).

**Write code.** Press switch S2 to reset the address counter to address '0000H.' In this state, LED1 through LED12 are off. Move slide switch S4 towards Prog position and switch S5 towards Prog/Erase position. Make control signal

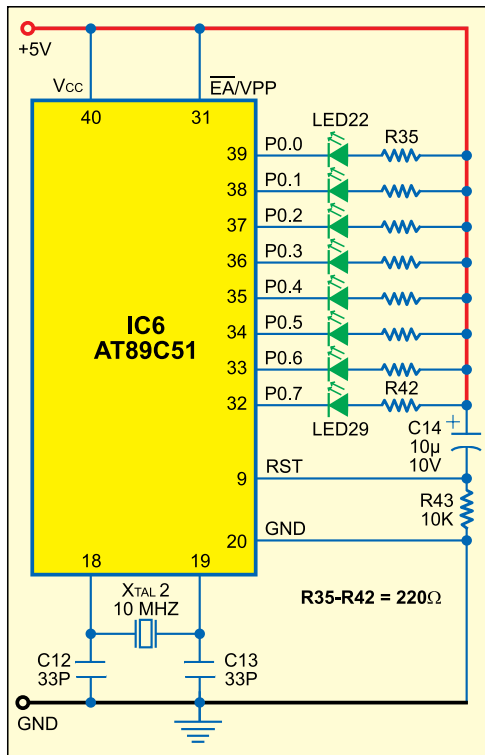


Fig. 4: Circuit of the example ring counter

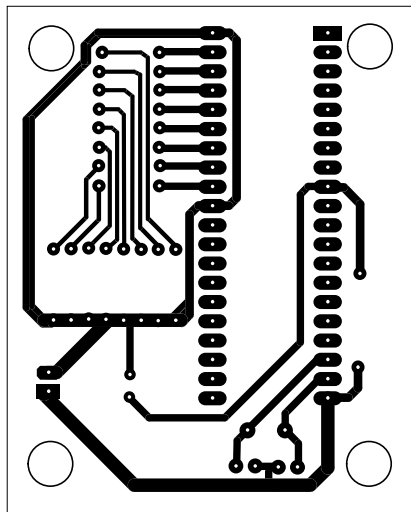


Fig. 5: Actual-size, single-side PCB layout for the ring counter

pins 28, 16 and 17 high using switches S9, S6 and S7, respectively, and pin 27 low using switch S8. Set the code data using DIP switch SW1. The data (D0 through D7) being set is indicated by LED13 through LED20. Program the code data by pressing switch S3. (LED21 glows to indicate application of the programming pulse.)

For programming the next data,

press switch S1 momentarily to increment the address (indicated by LEDs). Set the code data using DIP switch SW1 and program it by pressing switch S3. Repeat this process until the program code completes.

**Read/verification.** Move slide switch S5 towards Read position and DIP switch SW1 towards Off position. Make control signal pins 27 and 28 low using switches S8 and S9, and pins 16 and 17 high using switches S6 and S7, respectively. Reset the counter using switch S2 and read/verify data at location '0000H.' Increment the address counter using switch S1. The data at the incremented location (say, 0001H) can be seen on LED13 through LED20. Again press switch S1 and see the data at the incremented address on the LEDs.

**Reading the signature bytes.**

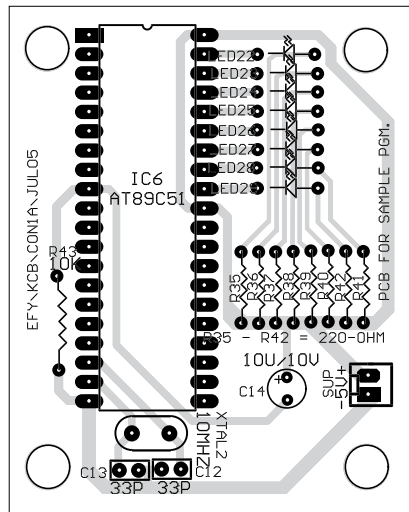


Fig. 6: Component layout for the PCB in Fig. 5

Signature bytes can be read using the same procedure as used for normal verification of locations 030H, 031H and 032H, except that all the control pins (pins 27, 28, 16 and 17) must be pulled low using switches S8, S9, S6 and S7, respectively. The values returned would be:

(030H) = 1EH indicates 'manufactured by Atmel'

(031H) = 51H indicates '89C51'

## RING.LST

Addr.	OPCode	Line	Mnemonics
		1	\$MOD52
		2	ORG 0000H
0000	747F	3	MOV A, #07FH
0002	F580	4	MOV 80H,A
0004	1140	5	ACALL DELAY
0006	74BF	6	MOV A, #0BFH
0008	F580	7	MOV 80H,A
000A	1140	8	ACALL DELAY
000C	74DF	9	MOV A, #0DFH
000E	F580	10	MOV 80H,A
0010	1140	11	ACALL DELAY
0012	74EF	12	MOV A, #0EFH
0014	F580	13	MOV 80H,A
0016	1140	14	ACALL DELAY
0018	74F7	15	MOV A, #0F7H
001A	F580	16	MOV 80H,A
001C	1140	17	ACALL DELAY
001E	74FB	18	MOV A, #0FBH
0020	F580	19	MOV 80H,A
0022	1140	20	ACALL DELAY
0024	74FD	21	MOV A, #0FDH
0026	F580	22	MOV 80H,A
0028	1140	23	ACALL DELAY
002A	74FE	24	MOV A, #0FEH
002C	F580	25	MOV 80H,A
002E	1140	26	ACALL DELAY
0030	0100	27	AJMP 0000H
		28	
		29	ORG 0040H
0040	7FFF	30	DELAY: MOV R7, #0FFH
0042	7DFE	31	LOOP1: MOV R5, #0FFH
0044	1D	32	LOOP: DEC R5
0045	ED	33	MOV A, R5
0046	70FC	34	JNZ LOOP
0048	1F	35	DEC R7
0049	EF	36	MOV A, R7
004A	70F6	37	JNZ LOOP1
004C	22	38	RET
		39	END

VERSION 1.2k ASSEMBLY COMPLETE, 0 ERRORS FOUND

(032H) = FFH indicates '12V programming'

(032H) = 05H indicates '5V programming'

**Note.** The present circuit is meant for 12V programming.

## Construction

A single-side PCB layout for the manual programmer (Fig. 1) is shown in Fig. 2 and its component layout in Fig. 3.

## Sample program

To test the working of the programmer kit, here's a program (RING.LST) for a continuously running ring counter:

The circuit of the ring counter is shown in Fig. 4. Its PCB layout is shown in Fig. 5 and component layout in Fig. 6.

**Download source code:** <http://www.efymag.com/admin/issuepdf/Manual%2089c51%20Programmer.zip>