EPROM PROGRAMMER

Enclosed are the circuits and software for the Henelec 2708 EPROM Programmer. The device is designed for use with Nascoms, having expanded and memory and using a P10 device of the MK3881 type. The software is written for Nascoms without IO expansion and uses PORTS 4, 5, 6 and 7. The software is written from ØEØØ to ØF63, which on an expanded Nascom would normally be scratch pad area. More comprehensive software could be written, but it was considered that as users wishing to programme their own EPROMS would be conversant with machine language routines, they would be able to expand this programme if the need arose.

Design Features

No mains ON/OFF is included, as it has been found that mains transients caused by the powering ON and OFF of adjacent equipment could cause corruption of data within the Nascom. Power switching is accomplished by software control of the four power rails and has been designed to fail 'safe'. Should, for instance, mains power be applied before software initialisation, the power and RD/WR automatically go to a 'safe' state.

SKT A of the Nascom is initialised to mode 3 (bit control) and is the control for the unit. Again, the unit fails safe if plugged into an un-initialised PORT. SKT B is the data I/O and is initialised as appropriate.

Pin out:-

SKT A		SKT B	
Pin 1	CLK	Pin 1	Data I/O Ø
Pin 2	PULSE	Pin 2	1
Pin 3	RESET	Pin 3	2
Pin 4	RD/WR	Pin 4	3
Pin 5	O/F	Pin 5	4
Pin 6	PWROUT	Pin 6	5
Pin 7	PWRON	Pin 7	6
Pin 9	GND rail	Pin 8	7
		Pin 9	linked to Pin 11

PARTS LIST FOR NASCOM / HENELEC PROGRAMMER

1	$5\frac{3}{4}$ x $3\frac{1}{4}$ 0.1" Veroboard		
1	16 pin Dil Skt.		
1	14 pin Dil Skt.		
1	24 pin Dil Skt.		
2	16 pin Dil Header plug		
1	250mA 20mm Fuse & open chassis F/Holder		
1	Douglas MT 235cs Mains Tx.		
1	CD4040	1 1K5 $\frac{1}{2}$w	1 470u 16v Min
1	7406	1 3K3 $\frac{1}{2}$w	1 470u 25v Min
8	BC107	1 68R $\frac{1}{2}$w	1 1000uF 16v Min
1	2N3053	1 33R $\frac{1}{2}$w	2 220uF 40v Min
3	BC177	1 1K $\frac{1}{8}$-$\frac{1}{4}$w	1 1n5 cer.
2	IN4003	4 470R $\frac{1}{8}$ - $\frac{1}{4}$w	
1	IN4002	2 47R "	
6	IN4001	12 4K7 "	
1	BZY88 c 27v	5 10K "	
1	BZY88 c 13v	2 22K "	
2	BZY88 c 5v6	1 150R "	
1	BZY88 c 4v7	1 6K8 "	
1	0.2" Red L.E.D.	2 33K "	
1	0.2" Green L.E.D.	1 100R "	

Complete Set of Parts £11.95

The programmer is almost totally software controlled in that the program controls the pulse timing, reset, data I/O select, RD/WR mode select and clock pulse are all determined by the program. Hardware is used to count the address increments and to switch the programming and RD/WR lines.

Mode of Operation

On initialisation (at Label INZ), the PORTS are set such that PORT A switches the programmer to a 'safe' condition and PORT B is set to mode 1 (input). After initialisation, control is passed back to the monitor via a forced breakpoint (E7). Provided the green LED is lit, insertion and removal of EPROMS can take place.

Programming is initiated by a two argument execute command at Label PROG; the first argument being the execution point of the program, the second argument pointing to the start of the data field to be programmed. On execution the red LED is lit indicating that the EPROM power is on, and a check is made to ensure that all bits in the EPROM have been set to 1. Should a EPROM fail this test, indicating faulty or already programmed EPROM, or partial erasure, the programme displays the error message "2708 ERROR" then branches back to INZ, thus terminating the programme.

Given that the EPROM passes this test, it then enters the programming loop, looping the data field 128 times. At the end of each loop, the loop number is printed on the screen. There is no purpose in this save to given an indication of progress.

On completion of the programming loop, the control PORT switches back to the read mode and the I/O PORT is reset to mode 1 (input). The program then waits for about 2 seconds to allow the EPROM to settle. The programmed EPROM is then compared with the data field 128 times to check for floating bits. If an error is found this is displayed and the verification terminated, the message "FINISHED" is displayed and control is passed to the monitor via INZ. If verification is completed successfully, no errors are displayed and the message "FINISHED" is displayed, control being passed to the monitor via INZ.

IMPORTANT

DO NOT USE RESET DURING PROGRAMMING AS THIS WOULD MOST LIKELY DESTROY THE EPROM. If an error has been made in entering the second argument, allow the programmer to continue to its natural end then erase the EPROM and start again.

Additional Facilities

Programming parts of EPROMS. It is important to realise that during erasure of a 2708 EPROM, all bits are set to 1 and that during programming, selected bits are set to 0, bits required to be set to 1 are not affected. It follows then that if spare space within an EPROM was initially left as 1's, then this space may be subsequently programmed without affecting the rest of the EPROM.

It must be borne in mind that only a complete 1K can be programmed, therefore, the data field must contain a copy of the original data plus the new data to be programmed. Under these circumstances, bits in the EPROM which were originally left as 1's are unaffected, likewise 0's are similarly unaffected. The only area affected is the new data which is programmed as normal.

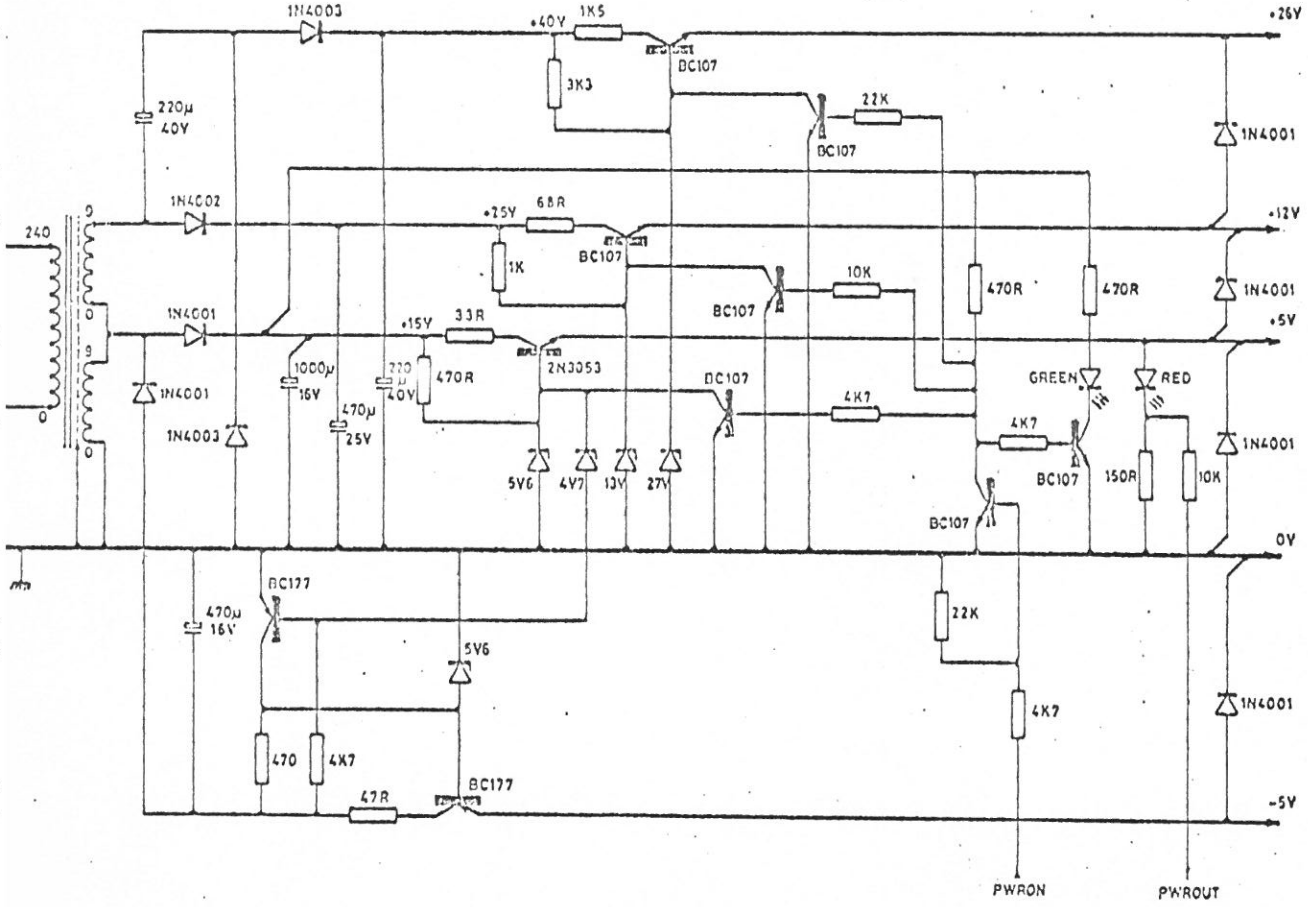
Note that the software is designed to reject a ready programmed EPROM, it is therefore necessary to start the programme after this check has been performed, hence programming commences at Label FPROG. A two argument command is used as before.

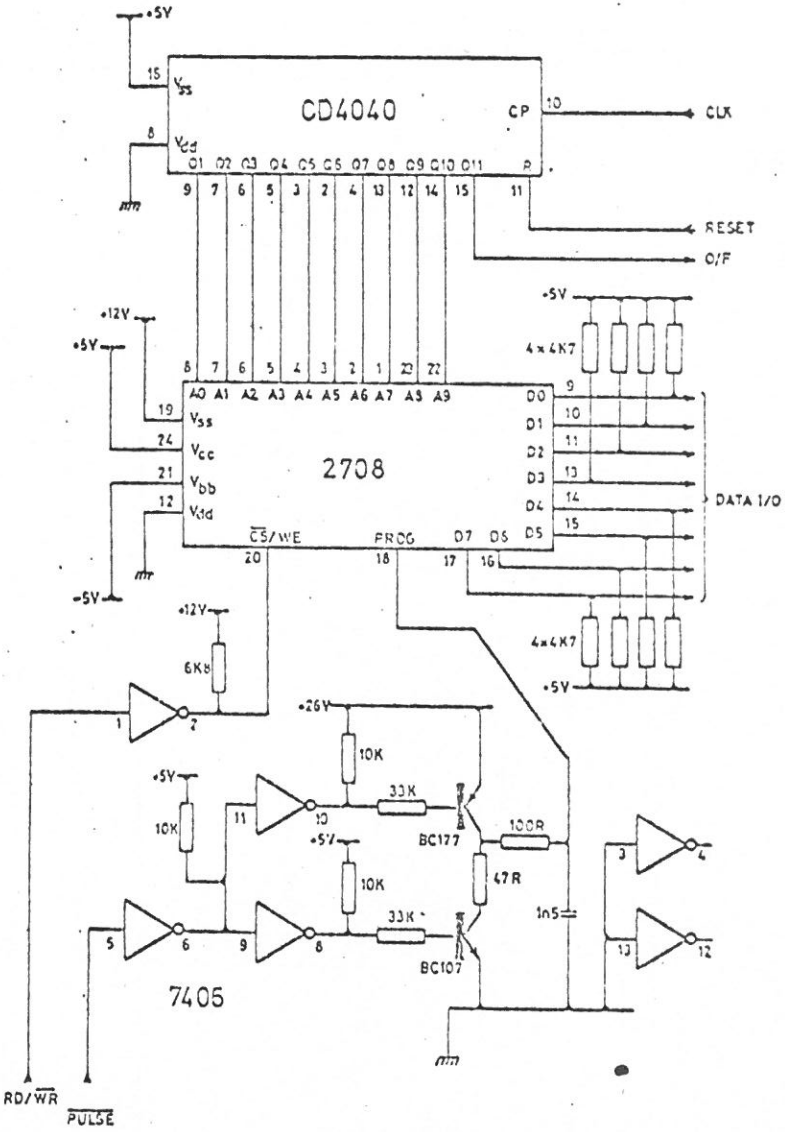
Localised heating of the EPROM during programming has been known to cause errors during verification (hence the 2 second delay), this is rare, however, in the event of errors being indicated it would be wise to re-verify the EPROM a few moments later before rejecting the EPROM as being wrongly programmed. The programme should be re-run from Label VERIFY using a two argument command as before.

A further feature is provided which enables an EPROM to be copied to RAM. This executes from Label READ using a two argument command; the second argument pointing to the start point in user RAM

Label		
INZ	ØEØØ	
PROG	ØE4Ø	dddd
FPROG	ØE7Ø	dddd
VERIFY	ØECØ	dddd
READ	ØF2Ø	dddd

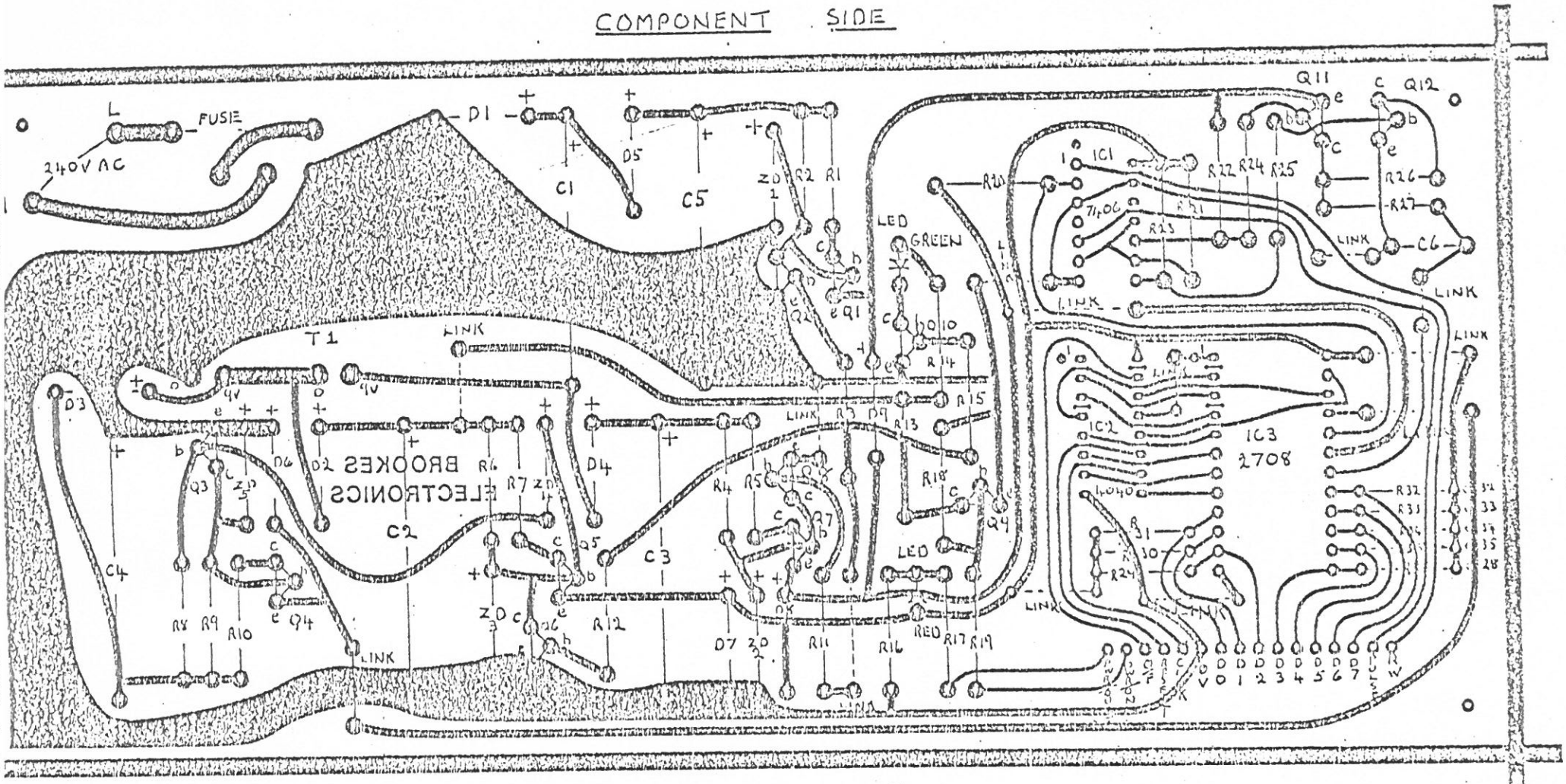
HENELEC® NASCOM EPROM PROGRAMMER





DRAWN D. R. HUNT. 20-11-78 COPYRIGHT DONDENE LTD 1978
HENRY'S RADIO, 404, EDGWARE ROAD, LONDON W2 1BN.

COMPONENT SIDE



R1. 1K5	R9. 470R	R17. 10K	R25. 33K	R 33. 4K7	C6. 1n5			
R2. 3K3	R10. 47R	R18. 22K	R26. 47R	R 34. 4K7	D1. IN4003	D8. IN4001	LEDS	Q1,2,6,7,8,9,10,12, BC107
R3. 22K	R11. 10K	R19. 4K7	R27. 100R	R 35. 4K7	D2. IN4001	D9. IN4001	1 RED	Q3,4,11, = BC177
R4. 1K0	R12. 4K7	R20. 6K8	R28. 4K7	C1. 220µ 40V	D3. IN4001	ZD1. 27v	1 GREEN	Q5, = 2N3053
R5. 68R	R13. 470R	R21. 10K	R29. 4K7	C2. 1000µ 16V	D4. IN4002	ZD2. 13V	IC1 7406	F1 = 250 m/A
R6. 470R	R14. 470R	R22. 10K	R30. 4K7	C3. 470µ 25V	D5. IN4003	ZD3. 5.6V	IC2 4040	T1 = IN 0-240 AC
R7. 33R	R15. 4K7	R23. 10K	R31. 4K7	C4. 470µ 16V	D6. IN4001	ZD4. 4.7V	IC3 2708	OUT 9-0 9-0
R8. 4K7	R16. 150R	R24. 33K	R32. 4K7	C 5. 220µ 40V	D7. IN4001	ZD5. 5.6V		