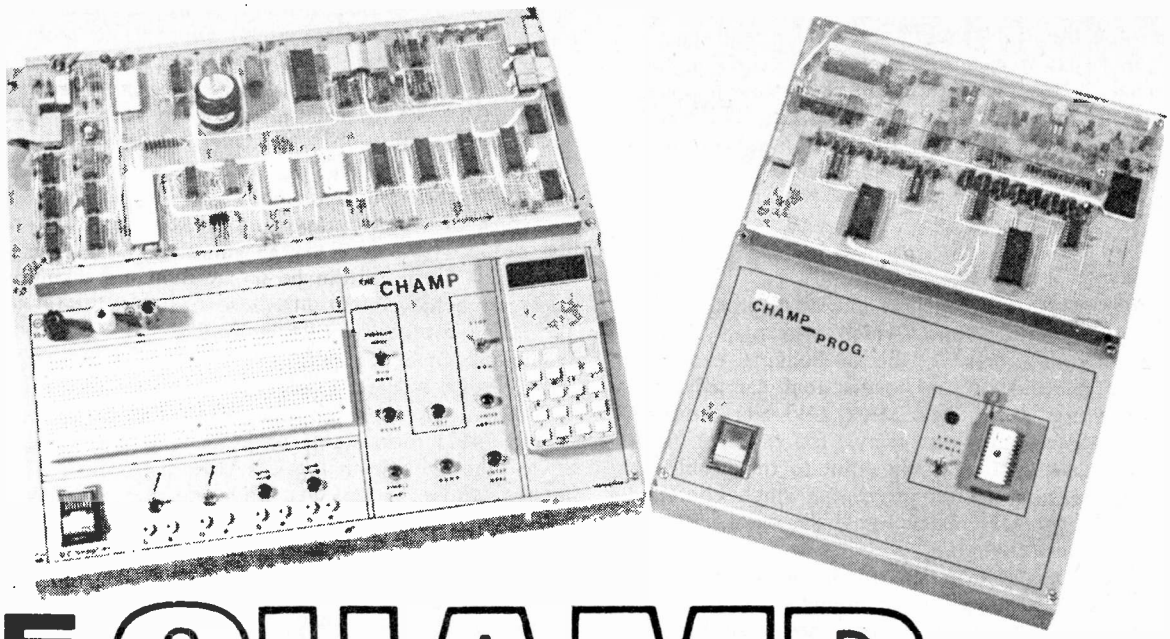

PE CHAMP

R.W.COLES
B.CULLEN

PART SEVEN

With the construction of CHAMP and its keyboard behind us, and with a 4702A containing the CHOMP program plugged into the Chip Zero socket, we can now move on and put the system to work for its living.

This month we will be covering the operation of the system, and showing how it can be used for the development of software and hardware, which can later be used in a separate "SON OF CHAMP" dedicated system, or be used as an extension of CHAMP itself.

LOADING A PROGRAM

When CHAMP is turned on, with the keyboard connected, a display of 000 200 should appear. The 200 tells us that the CHOMP address counter points to the first available RAM location, Chip 2 Location 00, and that program loading can now begin. Program instructions or data are entered one byte at a time, by pressing the two appropriate hexadecimal keys in succession. As each key is pressed the digits appear in the proper position on the left of the display, replacing the zeros which existed to start with. If an error is made on entry, the incorrect digits can be replaced with zeros or overwritten simply by pressing extra keys until the display is satisfactory.

If all is well, pressing the ENTER DATA key will enter the single byte into RAM location 200 and the CHOMP counter will be incremented to show 201, the next location in sequence. A complete program can be entered in this step by step fashion quite rapidly, each pair of hexadecimal digits being entered into the next available location by means of the ENTER DATA key.

If you wish to enter subroutines or data tables starting at some address *other* than 200H, the ENTER ADDRESS key can be used to reset the CHOMP address counter. A CHAMP address is twelve bits long, and so three hexadecimal keys are pressed before using the ENTER ADDRESS key. Any address in the range 200H to 3FFH can be entered in this way, in preparation for program entry, and in fact any address from 000H to 3FFH can be entered ready for an examination of its contents using the DUMP key. This means that a PROM based user program in the Chip One socket, or even CHOMP itself, can be examined if required.

Operation of the DUMP key will display on the two left-hand digits the hexadecimal content of the memory location indicated by the three right-hand display digits. Note that it displays the byte whose address was indicated *before* depression of the DUMP key, and that DUMP, like ENTER DATA, automatically increments the CHOMP address counter so that whole programs can be quickly examined by rapid operation of this key.

After using the DUMP key once, the two left-hand digits display the byte resident at the next lowest address to that indicated on the three right-hand digits.

RUNNING A PROGRAM

When a program has been entered and is considered satisfactory, it may be allowed to run by changing the MODE switch to RUN MODE. Operation of this switch causes CHOMP to carry out a JUN to location 200H where the first instruction of any user program should be situated.

CHOMP assumes that *every* program starts at 200H and this may be inconvenient, particularly if a number of small programs are co-resident in the CHAMP RAM area. To ensure that entering RUN MODE starts the required program, regardless of its position in memory, a JUN can be entered into locations 200H and 201H. To leave this option open it is best to begin any program which

starts at 200H with a couple of NOP instructions, thereby leaving room for the JUN should it ever be required.

Once a user program has been started, CHOMP takes no further part in the proceedings until PROGRAM MODE is re-selected and the RESET key is depressed. User programs can of course use any of the CHOMP subroutines such as DDRV or CLRF, without prejudice.

PROGRAM DE-BUGGING

When a program is tried out for the first time, it is normal for it to contain "bugs" which prevent it from operating correctly. To help in the de-bugging process CHAMP can be set to STOP and instructions carried out one at a time using the SINGLE SHOT key. For simple programs which control lamps or relays, for example, the use of the SINGLE SHOT can quickly point to the problem area, but for more complicated programs which contain several JCN, JMS, or JUN instructions, the SINGLE SHOT capability alone is not enough.

In these circumstances it is an advantage to have a knowledge of the 4040 data bus contents, or the 4289 data and address bus contents so that the operation of the program can be closely studied, and the changes after each instruction execution, monitored.

Monitoring the data and address buses cannot be achieved with software of course, and requires the use of a hardware device normally called a "Bus Analyser" which samples the buses at an appropriate moment and latches their content for display on l.e.d.s or lamps.

A very simple bus analyser which we have called "BUS BOX" has been designed for use with CHAMP, and the circuit for this unit is shown in Fig. 7.1. The "BUS BOX" can be plugged into sockets 2, 4 or 6 on CHAMP to observe the 4040 main data bus, the program memory data bus, or the program memory demultiplexed address bus, respectively. BUS BOX will display up to eight bits in binary form, although a hexadecimal format could easily be achieved with the use of appropriate decoders and seven bar displays if required.

The principle of operation is quite simple: The 4040 SYNC pulse is used to start a variable delay formed by a 74121 monostable circuit. When the delay expires, a second 74121 monostable generates a strobe pulse to load up to eight bits from the CHAMP buses into 7475 quad latches whose $\overline{Q}$ outputs drive l.e.d. lamps. When the BUS BOX is connected to the four bit 4040 bus (SK2) any of the eight time periods from A1 to X3 (see 4040 manual, Fig. 1-2) can be monitored and the bus contents displayed on four of the eight l.e.d.s. By this means all twelve bits of the current address, eight bits of the fetched instruction, four bits of the current accumulator value, and the eight bit SRC address can all be monitored in sequence by selecting an appropriate delay with the first 74121 monostable.

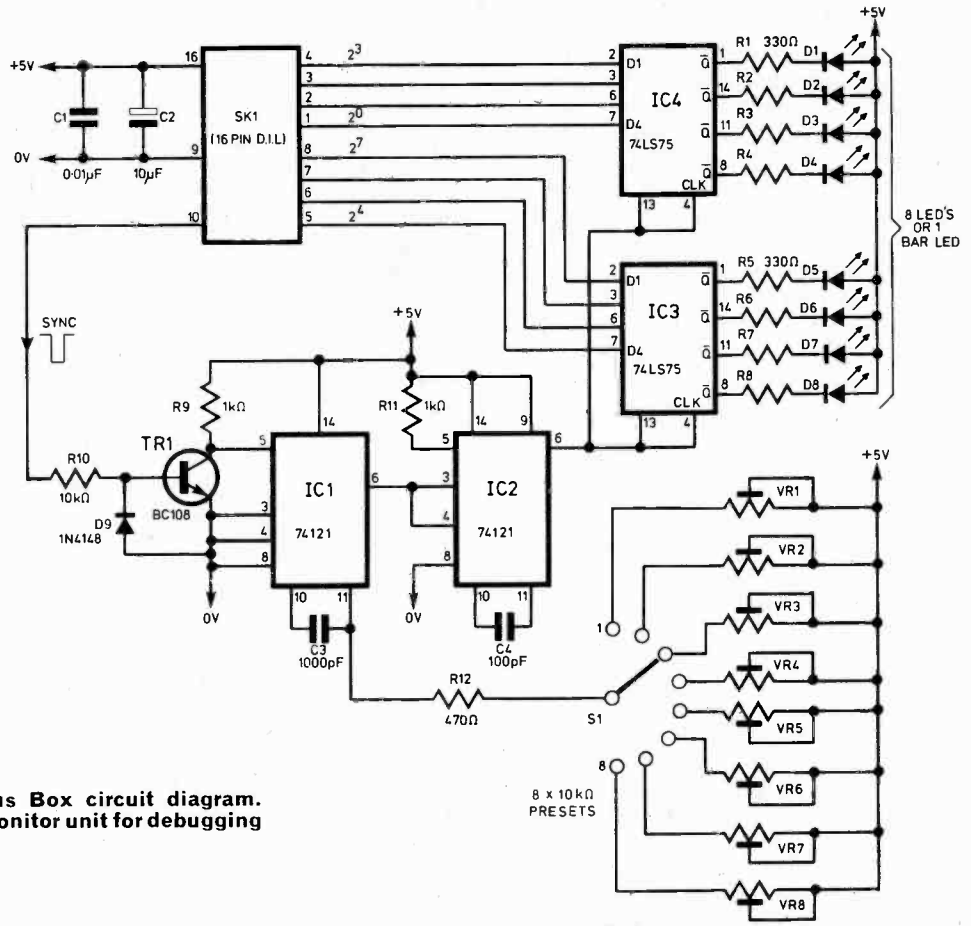

Fig. 7.1. Bus Box circuit diagram. A simple monitor unit for debugging CHAMP

Notice that the data on the 4040 bus is inverted by TR2 to TR5 on the CHAMP main board so that the required positive logic 1 = l.e.d. ON is realised. Being able to monitor the main data bus is very useful when sorting out elusive bugs, but looking at the data four bits at a time can be tedious if all you need to do is to follow the address flow of a program, or to monitor each instruction byte as it is fetched. To simplify this task the BUS BOX can also be connected directly to the 4289 data and address buses (SK4, SK6) where all eight l.e.d.s are used simultaneously, and in this case the delay is set so that it monitors the M2 bus time slot when both address and data information are available on their respective buses.

SIMPLE

The Bus Box circuit presented here is of the most basic type possible, and suffers from several disadvantages as a result. This was a deliberate policy so that construction costs could be kept to an absolute minimum; but anyone who feels that the rather crude method of time slot selection (which really requires an oscilloscope for initial set-up) or the primitive binary display, are not good enough can of course design something better. A counter, reset by the SYNC pulse can be used with a decoder for time slot selection, and combined hexadecimal latch, decoder, display chips can be used to present bus content.

All kinds of other embellishments can be added to produce a very powerful de-bugging tool, but if low cost is high on your list of priorities the Bus Box makes a good starting point.

WRITING PROGRAMS FOR CHAMP

Program writing for a simple four-bit chip like the 4040 can be carried out quite successfully at the machine code level, and there is no need for an extensive knowledge of computer science or of any high level languages such as Fortran. If you are already knowledgeable about such things, the simplicity of the 4040 might strike you as a disadvantage, but if you are basically a "hardware person" you will soon feel at home thanks to an intimate contact with the registers, gates and flip-flops which you will control via your programs. The creation of programs is of course a skill which must be learned gradually, by trial and error really, and the most important tip that we can give is to start with something simple, so that the almost inevitable "bugs" can easily be unravelled.

To start you on your way we have put together a simple program which involves both hardware and software design, and which serves as a useful springboard for a greater range of more sophisticated projects. As we describe the creation of this program we will introduce a number of useful programming tips and aids.

SAMPLE PROGRAM

The program we have written is called "TONE", and its sole purpose is to generate an audio tone of about 1kHz in an external speaker whenever the CHAMP TEST button is depressed.

The first step in program writing is of course the flow chart, and Fig. 7.2 illustrates this first stage in the design of TONE.

Box 1 indicates that we want to do nothing but idle while waiting for TEST to be pressed, and here we have a simple wait loop which will use the JCN conditional branch instruction to monitor the state of the TEST input.

Box 2 is entered when TEST is pressed and here we want to generate a delay of about one millisecond to set the frequency of the generated tone. The best way to generate delays of this sort is to set up a counter chain using ISZ instructions with 4040 registers acting as the counters; it also seems reasonable (though not essential) to make this a separate subroutine. After the delay, we can activate the output pulse which will drive the speaker, and one of the 4265 output lines which is subject to the bit set/reset command, WRM, seems a good choice to receive this output signal.

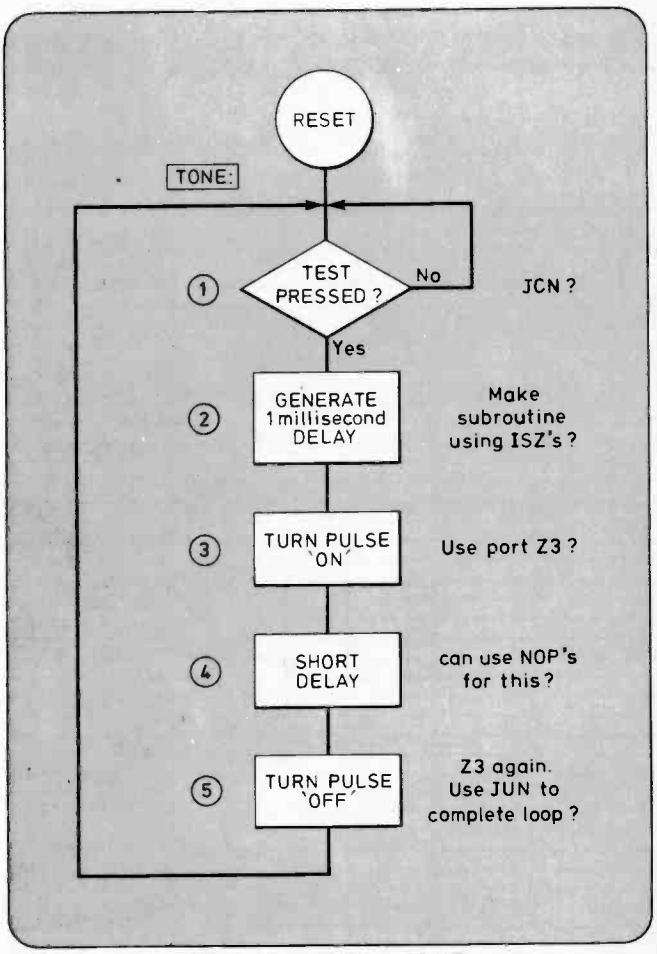

Fig. 7.2. Flow chart for the TONE program

In this simple program we are not after a 1:1 mark space ratio for the tone signal and so we can set the output pulse width by means of a very simple delay formed from NOP instructions (Box 4) before turning the pulse off again (Box 5). Having generated a single pulse we must of course loop back to see whether TEST is still depressed, and we can achieve this by means of a JUN.

Note that after drawing the basic boxes and lines required, we have added notes on the way we may want to code the program when we eventually reach that stage.

HARDWARE

During the flow-charting stage, port Z3 was proposed as a suitable interface to the external hardware, in this case, the speaker. A glance at the 4265 data sheet (page 5–41, 4040 Handbook) shows that any of the Z outputs can sink 1·6mA in the low state, whereas their high

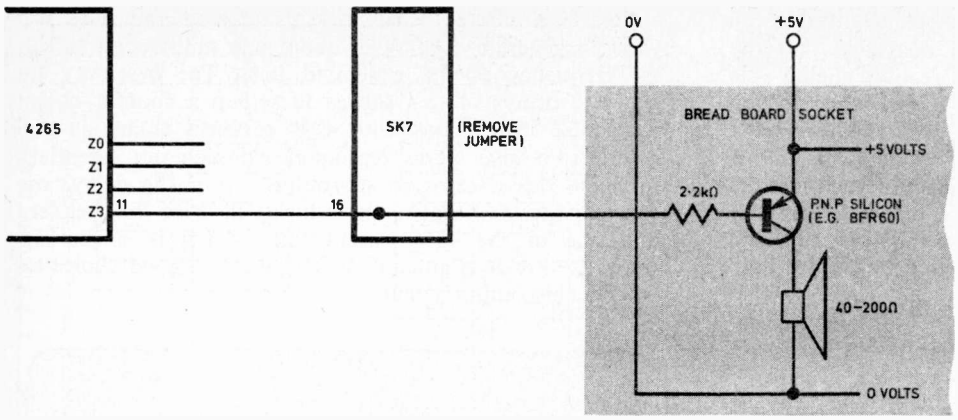

Fig. 7.3. Hardware required by TONE program. The breadboard mounted on the CHAMP facia is provided for such supporting hardware.

MCS 40 PROGRAM SHEET

TITLE: TONE generates 1k3 note when TEST is pressed — DATE 26-11-77 — PAGE No. 1 OF 1

HEX / BIN — MNEMONIC

PAGE	LINE	ROM	CODING	LABEL	OPERATION	OPERAND	COMMENTS
2	0 0	00		TONE:	NOP		
	1	00			NOP		
	2	11			JNT		(Wait for TEST)
	3	00			TONE		
	4	52			JMS		(Delay subroutine)
	5	18			ONEK		
	6	28			FIM	8	(Address 4265
	7	80			8	0	I/O chip)
	8	29			SRC	9	
	9	DE			LDM	E	
	A	E0			WRM		(Pulse "ON")
	B	00			NOP		
	C	00			NOP		
	D	00			NOP		
	E	00			NOP		
	F	00			NOP		
2	1 0	DF			LDM	F	
	1	E0			WRM		(Pulse "OFF")
	2	42			JUN		(Continue while
	3	00			TONE		TEST pressed)
	4	XX					
	5	XX					
	6	XX					
	7	XX					
2	1 8	22		ONEK:	FIM	2	(Preset counters)
	9	DD			D	D	
2	1 A	72		LOOP:	ISZ	2	(821 microsecond
	B	1A			LOOP		delay)
	C	73			ISZ	3	
	D	1A			LOOP		
	E	C0			BBL	0	(Branch back)
	F	XX					

NOTES (1) Port Z3 used for TONE output
(2) Keyboard disconnected at jumper
(3) XX = don't care

Fig. 7.4. TONE program instructions laid out on a standard program sheet

state sourcing ability is probably poor. This fact suggests the use of a p.n.p transistor stage to drive the speaker, and so the final hardware circuit is as shown in Fig. 7.3. Notice that these few external components can be assembled on the breadboard socket, and that the SK7–SK8 jumper is removed to gain access to Z3 on pin 16.

CODING THE PROGRAM

With the flow-charting and hardware design out of the way it is now possible to turn the program outline into a set of ready-to-load 4040 instructions, and our attempt at this is shown in Fig. 7.4. To make life a little easier we have designed our own 4040 program sheets which we duplicate and make up into pads, each sheet having room for 32 separate instructions. If you can get sheets like this duplicated then it is a good idea to copy our design, although an exercise book with a few lines ruled on it would serve just as well.

Each line on the sheet corresponds to a single address in program memory, hexadecimal address information being entered in the first two columns as required. The second two columns are for entry of hexadecimal instruction codes (and the binary equivalent if required) but these columns are filled out last of all. Column 5 is used to hold any address label or name that may be applied to any particular location, and columns 6 and 7 are used to write out the mnemonic form of the instructions as the program is developed.

Column 8 allows the insertion of plain-English comments to explain the action of the program; a necessary addition as you will soon appreciate when trying to unravel programs which you may have written some weeks previously, without useful comments!

The first address of TONE is 200H, the start of program RAM, and the first four lines of the program represent Box 1 of the flow chart. The two NOPs are not essential but we inserted them to allow a JUN to be entered when running programs elsewhere in the RAM address range, as discussed earlier.

SUBROUTINE

The one millisecond delay is coded as a subroutine which we have called ONEK. The actual location of the subroutine is unimportant but we chose address 218H, to allow some room between it and the end of the main TONE program, in case TONE "grew" after de-bugging. Reference to page 2–18 of the 4040 manual shows that a one millisecond delay can be achieved with two four bit counters, given the standard 5·185MHZ clock frequency normal with 4040 systems, including CHAMP.

Since the total period of the pulse stream is determined not only by ONEK but also the time the pulse is ON and the time taken to execute other instructions in the loop, a value of 821 microseconds is actually used, and the "fine tuning" of this delay is achieved by loading the two

count registers with hexadecimal data before counting begins. This register, preset to DDH, is performed by means of the FIM instruction at the start of ONEK.

After the JMS ONEK instruction in the main program, comes another FIM which loads register-pair 8 with the SRC address value of the 4265 (actually 80H). This is then sent out by the SRC 9 in line 208H to select the 4265 ready for output.

The pulse is turned on by setting Z3 to the low output state using LDME WRM, a sequence which can be best understood by reference to page 5–39 of the 4040 manual. After the 54 microsecond delay produced by the five NOPs, LDMF, WRM is used to return Z3 to its high state, followed by a JUN back to TONE. When all the mnemonics and comments had been entered, we coded the program by looking up the hexadecimal equivalents in the manual and entered these along with hexadecimal addresses (in place of the labels) into column 3 of the program sheet.

REGISTER MAP

The subroutine ONEK required Index registers for use as counters, and the main program used an Index register pair as a source for SRC addresses, but you may be wondering just *why* we used the registers that we did use.

TONE is a very simple program which uses only two register pairs out of the total of twelve available, and so we *could* have used any of the Index registers with equal success. When writing larger programs this is often not the case; CHOMP and PROMPT use every available

MCS 40 . INDEX REGISTER MAP

TITLE: TONE

BANK 0			
0		1	
2	ONEK COUNTER	3	ONEK COUNTER
4		5	
6		7	
8	SRC	9	USE
A		B	
C		D	
E		F	

BANK 1			
0		1	
2		3	
4		5	
6		7	

NOTES

1	FIN USES PAIR 0 AS ADDRESS BUT JIN CAN USE ANY PAIR				
2	'OR' LOGIC USES 4 AND 5 'AND' LOGIC USES 6 AND 7				
3	SRC RAM ADDRESS	EVEN X2 CHIP	EVEN X2 REGISTER	ODD X3 CHARACTER	
	SRC ROM 1/0	CHIP		DONT CARE	
	SRC 4265	CHIP / DONT		CARE	
	SRC PROGRAM MEMORY	HIGH ORDER		LOW ORDER	
	SRC RAM O/P	CHIP	DONT	CARE	
	SRC RAM STATUS	CHIP	REGISTER	DONT CARE	
4	BANK 1 CAN BE USED DURING INTERRUPTS TO SAVE BANK 0				

Fig. 7.5. Index register map. A standard sheet such as the one shown would serve as a method of recording the deployment of each register, and this one has been entered up for TONE

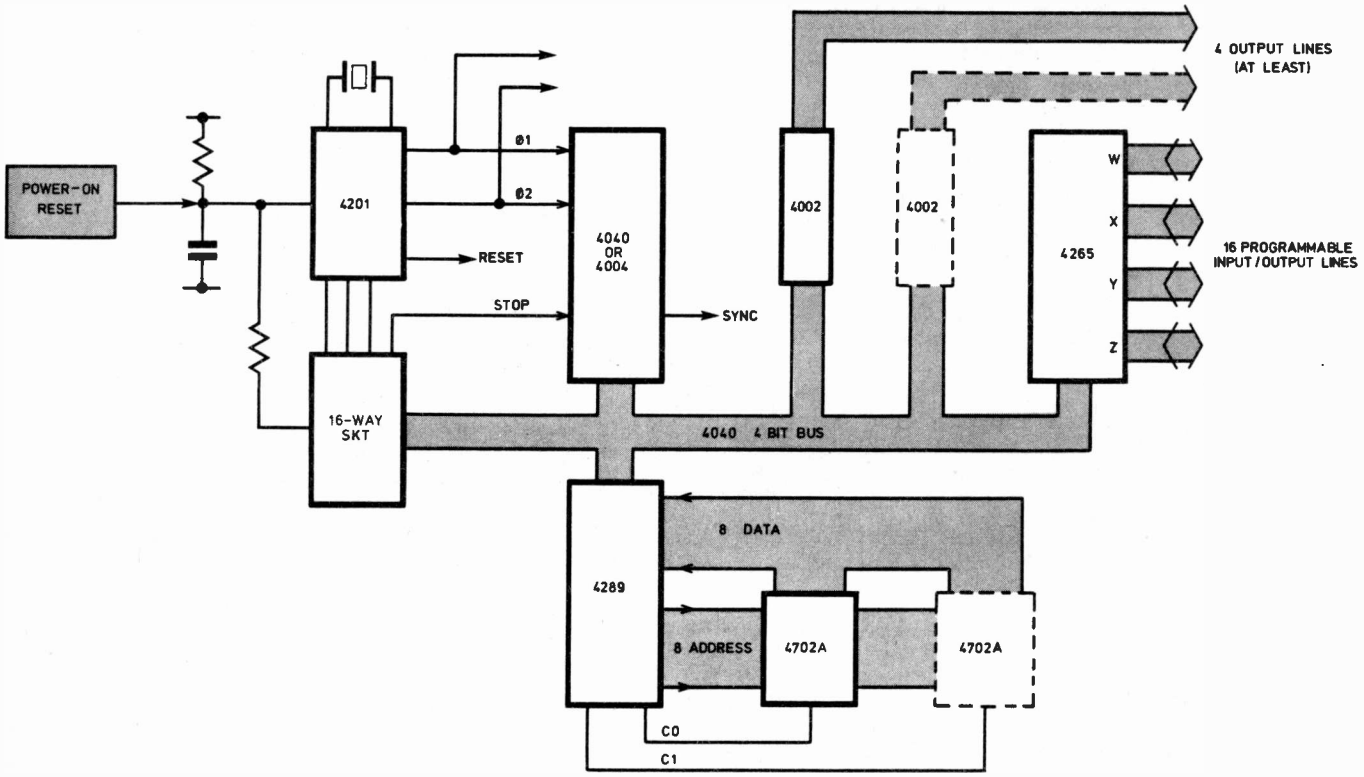

Fig. 7.6. Possible "Son of CHAMP" layout for a 4040 or 4004 based system for dedicated application. The 16-way socket is for bus analyser testing and a manual control unit. The 4002s and 4702As should be socket mounted so that only the required chips are used

COMPONENTS . . .

BUS BOX

Resistors

8 off 330Ω	R1–R8
2 off 1kΩ	R9, R11
1 off 470Ω	R12
1 off 10kΩ	R10

All ¼W 5% carbon

Potentiometers

8 off 10kΩ presets	VR1–VR8

Capacitors

1 off 0·01μF	C1
1 off 10μF elect	C2
1 off 1000pF	C3
1 off 100pF	C4

Semiconductors

8 off discrete l.e.d.s, or one bar l.e.d. array	D1–D8
1 off 1N4148	D9
1 off BC108	TR1

Integrated circuits

2 off 74121	IC1, IC2
2 off 74LS75	IC3, IC4

Miscellaneous

1 off 16-way d.i.l. socket	SK1
1 off single pole 8-way rotary switch	S1

Stripboard, cabinet e.t.c.

register for example, and use some of them for several different jobs. This means that keeping track of Index register usage is very important. To help with this aspect of programming we have put together another duplicated sheet which we call an Index Register Map, and Fig. 7.5 shows how this looks for TONE.

Of course TONE is a very simple program and not much use as it stands, but we feel that its basic principles can be incorporated in such projects as Stylophone type instruments, musical doorbells and a host of others.

CHAMP PROGRAMMING SERVICE

Readers who have no PROM programming facilities may have their own 4702A or 1702A PROM programmed by post with the following:

(a) CHOMP	£5·35
(b) Reader's own software	£10·35
(c) Reader's own software re-programmed with up to 16 corrections to original program	£3·35

All prices include postage and packing.

Programs, or corrections to programs, *must* be supplied as a *clear* list of two-digit hexadecimal code with hex' address information alongside. Also, PROMS should be sent well packed, and protected with conductive foam.

CHOMP software will be tested on a CHAMP system, otherwise programs are committed to PROM at reader's own risk.

This service is provided by, and payment should be made to:

C. C. CONSULTANTS,
Dept P.E., 3 Gainsborough Drive, Worle, Weston-super-Mare, Avon.

Do *not* send PROMS to P.E.

DEDICATED SYSTEM

After developing hardware and software for, say, a musical doorbell, you will need to produce a small dedicated hardware system into which you can plug the PROMs programmed by CHAMP–PROG.

Figure 7.6 shows a possible layout for a small 4040 or 4004 based system which could be put to a multitude of different uses, and which would fit onto a six inch square circuit board.

NEXT MONTH: CHAMP–PROG.

NEWS BRIEFS

SOUND 78 INTERNATIONAL

This exhibition has been organised by the Association of Sound and Communications Engineers (formerly the Association of Public Address Engineers), and is to be held at the Cunard International Hotel, Shortlands (near Hammersmith flyover), London W6.

The Sound 78 International Exhibition will be held over the period from March 14–16 inclusive, and on display will be some of the most sophisticated and up to date sound and communications equipment in the world. It will be possible to view amplifiers, microphones, automatic announcement equipment, alarm systems, background music systems, sports event timing equipment, loudspeakers, hotel and hospital communication systems, discotheque equipment, intercoms and paging equipment, mixers, studio recording and audio visual equipment.

There will also be experts present who can discuss the design, installation and function of most of the equipment on display.

The exhibition is to be open each day from 10.00 to 18.00 (17.00 on the last day), and admission will be absolutely free to anyone having a professional or business interest.