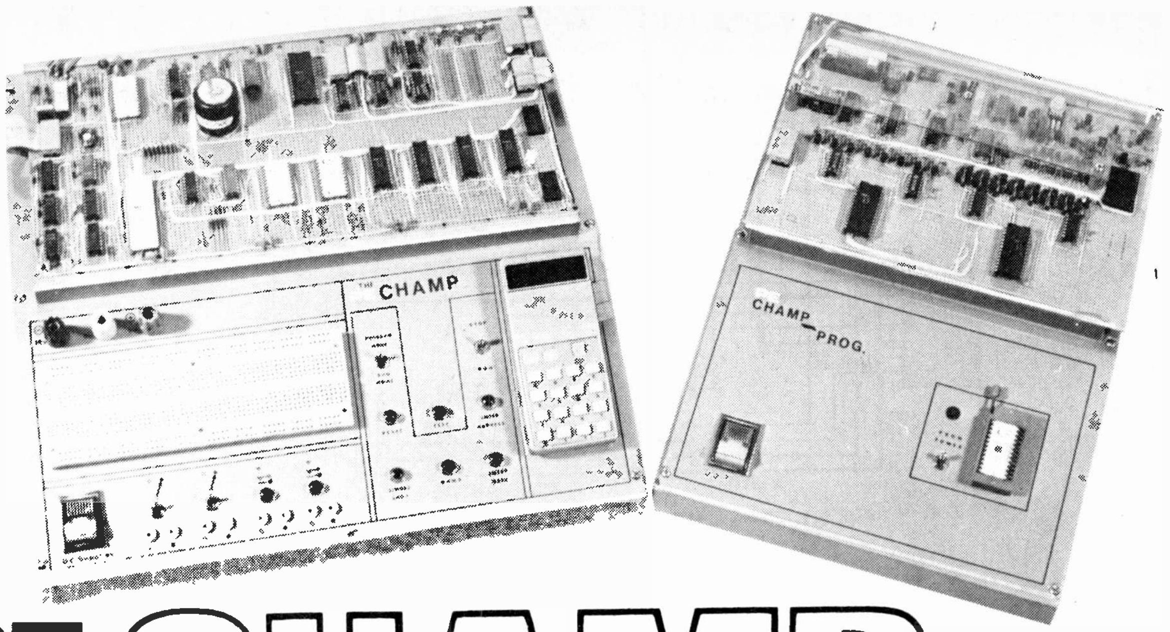


# PE CHAMP

R. W. COLES
B. CULLEN

## PART THREE

LAST month we looked at the operation of the main system components used on the CHAMP board, including the 4040 MPU chip itself, and we are now about ready to look at the operation of the circuit in more detail. Before we start to discuss the hardware at the "gates and wire" level though, a word about the system operation as defined by the CHOMP software would be helpful.

### SYSTEM OPERATION

When considering CHAMP as a development system, i.e. with the CHOMP program running, there are a number of specific tasks to be performed which can be listed as follows:—

(a) Refresh 8-character 7-segment l.e.d. display at a rate which eliminates flicker.

(b) Accept and store hexadecimal keyboard entries of up to three characters.

(c) Scan the control panel to detect any of the following switch closures:—
ENTER DATA, ENTER ADDRESS, DUMP, RUN-MODE, TEST.

(d) In response to *ENTER DATA*, take data from temporary keyboard storage and load into the RAM location pointed to by the "current address pointer" register, then increment the pointer.

(e) In response to *ENTER ADDRESS*, take data from temporary keyboard storage and load it into the "current address pointer" register.

(f) In response to *DUMP*, read data in the program location pointed to by the "current address pointer" register and load it into the display buffer, then increment the pointer.

(g) In response to *RUN MODE*, leave the CHOMP program by jumping to the start of a user program in the first program RAM location (Address 200 Hex).

(h) In response to *TEST*, leave the CHOMP program by jumping to the start of a program in the second PROM chip (Address 100 Hex). This would normally be the PROMPT programmer firmware if fitted.

The important thing to remember about the operations listed above is that they are controlled by *software*, or to be more correct, *firmware* and are *not* purely hardware operations like *RESET*, *RUN/STOP*, or *SINGLE STEP*.

This means that although I shall be discussing the circuitry as it relates to these operations, you should bear in mind that *you* can use the circuitry for other purposes, providing you produce the software to do it.

This means, for example, that when you switch to *RUN*, *your* program can redefine *ENTER DATA* as "change points", *ENTER ADDRESS* as "sound horn", and *DUMP* as "pull the flush", without you having to change a single wire!

### CIRCUIT DETAIL

Referring to Fig. 2.3 (last month), let's start with IC1, the 4201 clock generator. This device is fully described starting on page 5-77 of the users manual, but in outline it is a CMOS chip in a 16-pin package which contains the oscillator and dividers necessary to produce the 4040 two-phase clock signals and the logic for the *SINGLE STEP* and *RESET* operations.

The important point about this chip is that it provides high current clock outputs capable of driving the phase 1 and phase 2 inputs of a *complete* 4040 system, and this leads to a requirement for special decoupling circuitry.

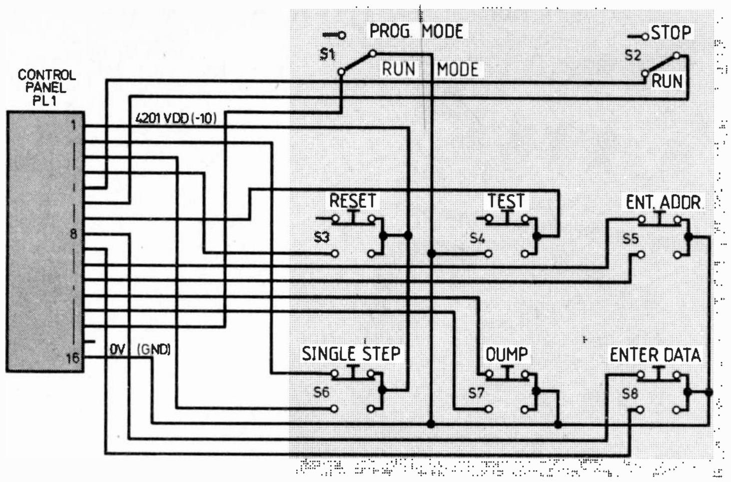


**Fig. 3.1. The CHAMP control panel connections**

R1, and C1, C2 isolate the drive current pulses from the supply line and R2, R3 help to reduce the rise time of the clock waveforms when a complete set of 4040 system components is not used, as is the case with CHAMP. The insertion of R1 produces a separate 4201 $V_{DD}$ node and since the RESET switch and the SINGLE STEP switch require a $V_{DD}$ connection, it is to this node that they must be connected.

Pin 5 on the 4201 is a mode control pin which changes the division ratio of the internal counter to slow down the resultant clock output. Since there are tangible advantages in sticking to a 10·8 microsecond clock cycle this pin is permanently connected to +5 volts in CHAMP. Pins 2 and 16 are clock outputs at TTL rather than MOS levels, and are unused in the CHAMP system, R5 and C3 provide the "power-on-reset" time constant and can be altered as necessary to set an appropriate delay which ensures that the complete system is reliably "cleared" whenever power is first applied.

## 4040 CHIP

The MPU chip, the centre of the CHAMP system, is of course IC2. Note that pins 1, 2, 3, and 4 carry the four-bit multiplexed bus which is the key to 4040 operation and which of course was covered in detail in Part 2 last month. This bus provides communication with the 4002 data RAMS, the 4265 I/O chip, the 4289 program memory interface chip, and can also be accessed via the sockets for system expansion when required. Note the SYNC output, pin 16, and its interconnection to the other system components, and also the STOP input, the STOP ACKNOWLEDGE output, and the RESET input which link to the 4201 clock generator. TEST, pin 13, is an input which can be tested directly with software (e.g. "*JUMP IF TESTS EQUALS LOGIC ONE*") and is a unique 4004/4040 feature.

The COMMAND RAM lines, pins 17, 18, 19 and 20 can each control a data RAM bank which in turn may consist of four 4002s, or three 4002s and one 4265. Ondy $CM_0$ is used on the CHAMP board, but CHAMP PROG uses banks 1 and 2 for the two extra 4265s. These lines are activated using the *DCL* instruction and are used to increase the address range over that possible with only an 8-bit *SRC* operation. The 4040 also has two COMMAND ROM lines so that two separate ROM banks can be used to allow a total of 8K of program if needed. In CHAMP only CM $ROM_0$ is used to control a single 4289, and it is considered unlikely that CM $ROM_1$ would ever be used in a CHAMP derived system.

## NEGATIVE LOGIC

When first introduced the 4004 and 4040 were defined with respect to a negative logic convention, because this is more "natural" in a PMOS system where a transistor turned "on" produces a positive output level and a transistor turned "off" allows its output to be pulled down to a negative level. *Inside* a 4040 system this convention still holds, so that for example, a logic 1 on the DATA BUS is actually represented by a *negative* level, but on the inputs and outputs from the 4265 the more familiar *positive* logic convention is employed.

This means that a logic inversion takes place inside the 4265, so that if for example your program writes binary 1111 (F in Hex) to 4265 port Z you can expect to see four TTL-compatible positive logic levels on the output pins even though they passed over the bus as *negative* levels. The 4289 PROM address and data, and the I/O and CS pins are also defined in positive logic to make life easier, and so usually you don't have to worry about which convention applies for interfacing operations, you can assume good old TTL-type positive logic.

The main exception as far as external interfacing is concerned is the 4002 RAM output port which *is* defined in negative logic, although this port is really only a secondary facility anyway, and only becomes available when a second 4002 is added to the system. It is of course always advisable to check in the MCS-40 User's Manual what the logic convention is on individual pins like INT or INT ACK before connecting these to external circuitry.

## CONTROL PANEL INTERFACE

In Fig. 3.1 we show the interconnection of the CHAMP control panel switches, which of course are mounted on the plinth and hooked up to the CHAMP main board via a 16-way flat strip cable. The RESET, RUN/STOP and SINGLE STEP switches connect directly to the 4201 chip IC1, but the other four switches are wired into the system via the ROM I/O lines and some TTL conditioning circuitry which forms, collectively, a special kind of four-bit input port.

The ENTER DATA, ENTER ADDRESS and DUMP push switches directly control the PRESET and CLEAR inputs of 7474 D-type flip-flops (IC11–IC13) which are used as latches to "debounce" the switch operations and provide a clean positive-going edge for each press. The outputs from these latches are used to "clock" further D-type flip flops which have logic 1s hard-wired to their D inputs, and the $\overline{Q}$ outputs of these pass via a 74125 tri-state buffer, IC14 to the 4289 I/O bus. This second set of three latches can be cleared via a *WRR* instruction since they are controlled by what is, in effect, a ROM output port (part of IC24 and IC25).

Suppose the ENTER DATA switch is pressed, this sets the Q output of its associated latch to a 1 and this in turn clocks a 1 into the second flip-flop whose $\overline{Q}$ output is then available at the input to the 74125. This sequence of events in itself initiates no further action, since the 4040 will not realise that anything has happened until it carries out an *RDR* instruction which strobes the 74125 and allows all four switch data bits to be transferred via the 4289 and the DATA BUS to the accumulator.

## WAIT LOOP

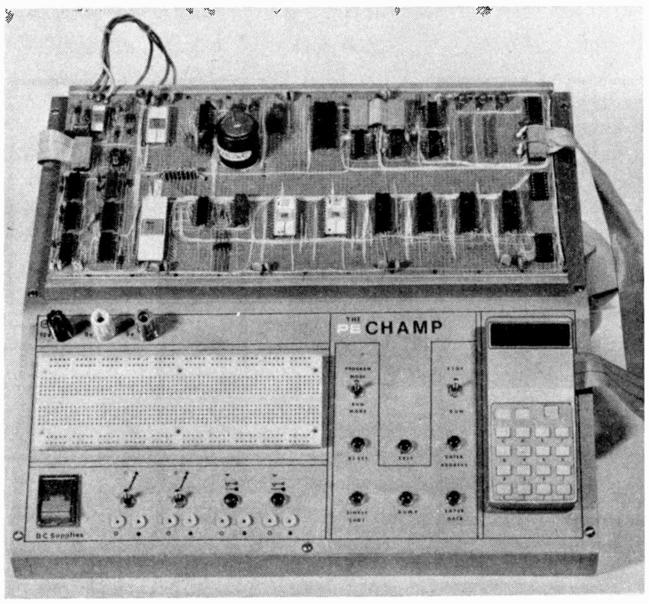
There won't be long to wait of course, and normally the 4040 sits in a "wait loop" which is embodied in the CHOMP software, continuously carrying out a read and check operation on these very control switches. When the ENTER DATA closure *is* recognised the 4040 jumps to a part of CHOMP which deals with the entry of data, and one of the first things this section of the program does is to clear all the switch flip-flops via the 4289 and part of IC24 and IC25. This is necessary to prevent multiple recognitions of the same switch closure, and points to the reason for the second D-type, since with this arrangement no matter how long you keep the switch pressed it can only be recognised once.

Note that the PROG MODE/RUN MODE switch is not provided with TTL latch conditioning circuitry since it is a toggle switch and is not used repetitively like the others; its contacts are connected directly to the 74125. We have termed the switch conditioning circuitry, just described, the switch FLAGS and in future we will use this shorthand name, and refer to the 74125 as the FLAG PORT.

## KEYBOARD INTERFACE

The on-board keyboard interface, comprising IC7–IC10, interposed is between the 4265 ports and the keyboard sockets SK3, and also between the 4040 interrupt lines and SK3. The 4265 is connected to this interface circuitry via a 16-way flat strip jumper which connects SK7 to SK8 when the keyboard is in use. As mentioned in Part 1, this jumper can be removed for direct access to the 4265 when "custom" interfacing is required for user programs.

The keyboard produces a ready encoded hexadecimal output on four lines together with a common strobe, and the display section requires eight-segment anode drives (a–g & d.p.) and a clock and data input to the internal digit-strobe shift register (see Fig. 3.2). The internal circuitry of the keyboard will be covered in detail later on.



The four hexadecimal keyboard outputs connect directly to the 4265 port W which is defined as a mode 9 input port during 4265 initialisation under CHOMP, but the common strobe is fed to the 74123 dual monostable to produce a de-bounced strobe which sets the interrupt latch (half of IC9) aid is also used to enter the hex code into the part W input latches via the port Z1 asychronous strobe line (See MCS-40 User's Manual, pages 5–36 for further details.)

## DISPLAY REFRESH

The display refresh drive is achieved by loading the next eight segment bits into output ports X and Y and then clocking the shift-register produced digit strobe along to the next common-cathode digit line. This operation has to be repeated eight times for the complete eight-character display, and has to be carried out rapidly enough to prevent display "flicker". The digit strobe is in effect a logic 1 shifting through a field of 0s, a new logic 1 being presented to the shift-register via 4265 output line Z3 under software control at the start of a new display sequence.

The shift-register clock pulses are provided by output Z2 which is a synchronous strobe produced when port Y is loaded with segment data during a *WR2* instruction. IC7 and IC8 are special l.e.d. anode driver arrays (75491) which provide the high-current segment drive needed by the multiplexed display, the cathode drives (75492) are contained within the keyboard case and are of course driven by the shift-register outputs. R51–R58 perform the usual l.e.d. current-limiting function and therefore control the display brightness.

## PROGRAM RAM

The original 4004 microprocessor expected its program in a ROM and its data in a RAM and never the twain shall meet, but CHAMP is a development system which requires programs to be easily modified and kept in RAM and so

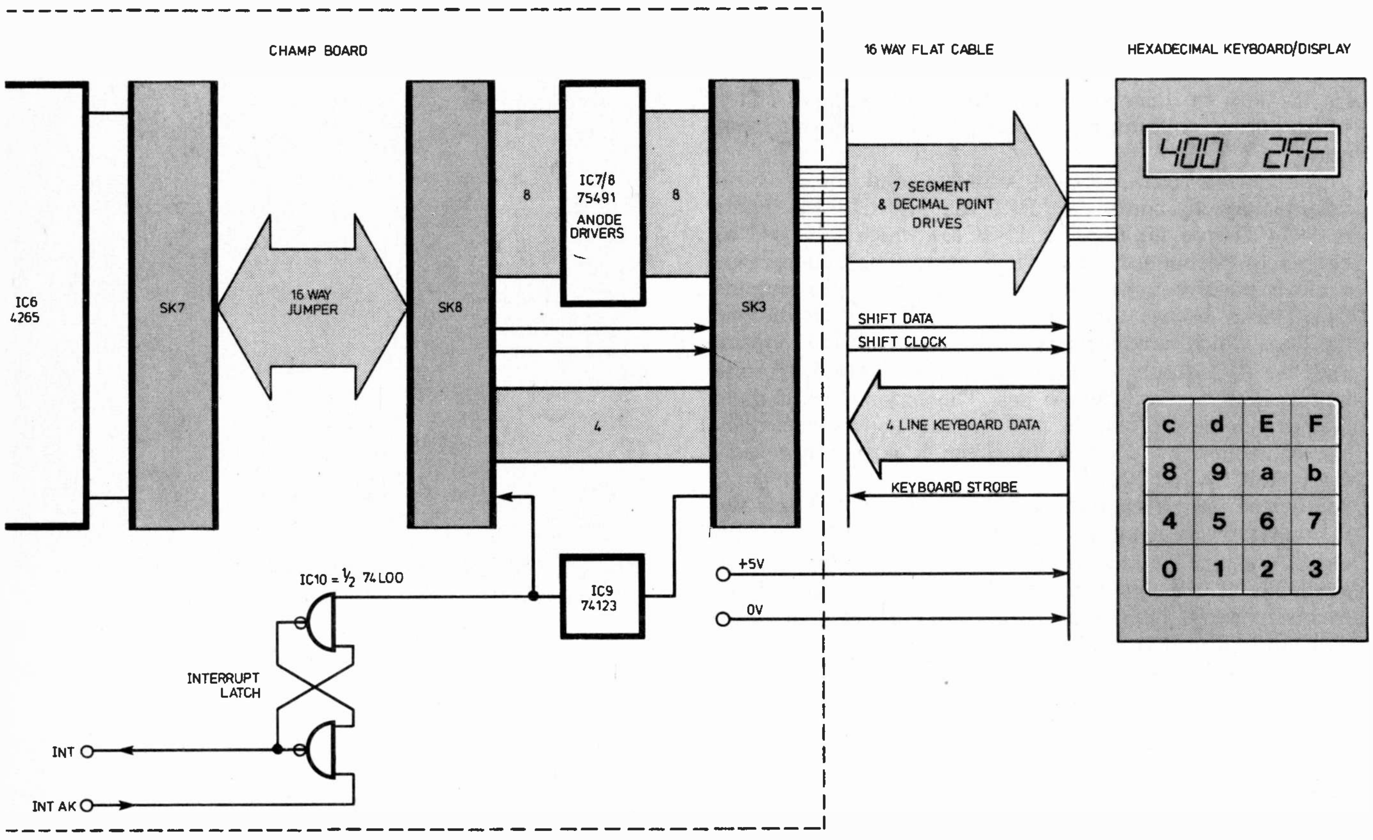


**Fig. 3.2. The CHAMP board/keyboard interface**

some special arrangements have to be made to provide this facility. Fortunately the 4040 does have instructions for writing to and reading from RAM program memory, namely *WPM* and *RPM* respectively, but since the 4040 deals with four-bit nibbles while its program comes in 8-bit bytes, some jiggery-pokery is still required to allow painless operation of the *ENTER DATA* and *DUMP* commands which of course are used to modify and examine program RAM when required.

When a program in RAM is actually *running* the program RAM is addressed via the 4289 just as if it were PROM, and eight-bit instructions are fetched from RAM via the 4289 without the MPU ever knowing the difference. The need for "special treatment" arises when the so-called transitive read or write operations using the *RPM* and *WPM* instructions are undertaken because of the nibble/byte conflict.

To achieve proper operation of the transitive instructions the 4289 contains a FIRST/LAST flip-flop which is toggled by each use of *WPM* or *RPM*. The output of this flip-flop is used externally to steer a nibble to either the FIRST half-byte or the LAST half-byte of a program RAM location during transitive *write* operations, or used internally to send the FIRST half-byte or LAST half-byte of program RAM data back to 4040 over the data bus during transitive *read* operations. To accommodate this mode of operation CHAMP program RAM is organised so that it may be *read* as a byte-orientated array of 512 × 8 bits but *loaded* as a nibble-orientated array of 1024 × 4 bits.

The program RAM write operation is achieved using the 4289 I/O bus to transfer the data a nibble at a time, the correct half-byte of RAM being selected using a logical combination of the 4289 outputs F/L, PM, and OUT to produce individual write strobes for each of the two 256 × 4 RAM chips which together form the equivalent of a single 4702A PROM chip. This gating logic is performed by the remaining parts of IC24 and IC25.

## ADDRESSING PROGRAM MEMORY

As mentioned last month, the 4289 is used to demultiplex the 4040 bus to produce a 12-bit wide address output to program memory. The lower 8 bits of this address are wired directly to each program memory device via what we shall call the 4289 address bus (pins 23 to 30 from 4289). The upper four bits are decoded by a 3205 TTL decoder to produce a unique *CHIP SELECT* strobe for each of the two 4702A PROM chips and each of the two pairs of 5101 RAM chips so that only one "memory chip" (one 4702A or two 5101s) can be enabled at one time.

The 12-bit address is provided by the 4040 program counter during normal operations, but when a transitive read or write is carried out the eight low order address bits must be provided by an *SRC* operation, and the four chip-select bits must be provided via an output port. In CHAMP the port employed for this purpose is the 4002 output port from IC4, buffered by a 74L00 gate IC3, which also provides the necessary logic level inversion.

We now have two possible sources for the four chip-select bits, either pins 31 to 34 of the 4289 (normal operation) or pins 13 to 16 of the 4002 (transitive operation) and so the 74157 quad two-line to one-line data selector (IC16) is interposed between the two sources and the 3205 decoder. The 74157 SELECT input is controlled by the 4289 PM output which is active only during transitive operations, so that proper selection of the source of chip-select data is maintained.

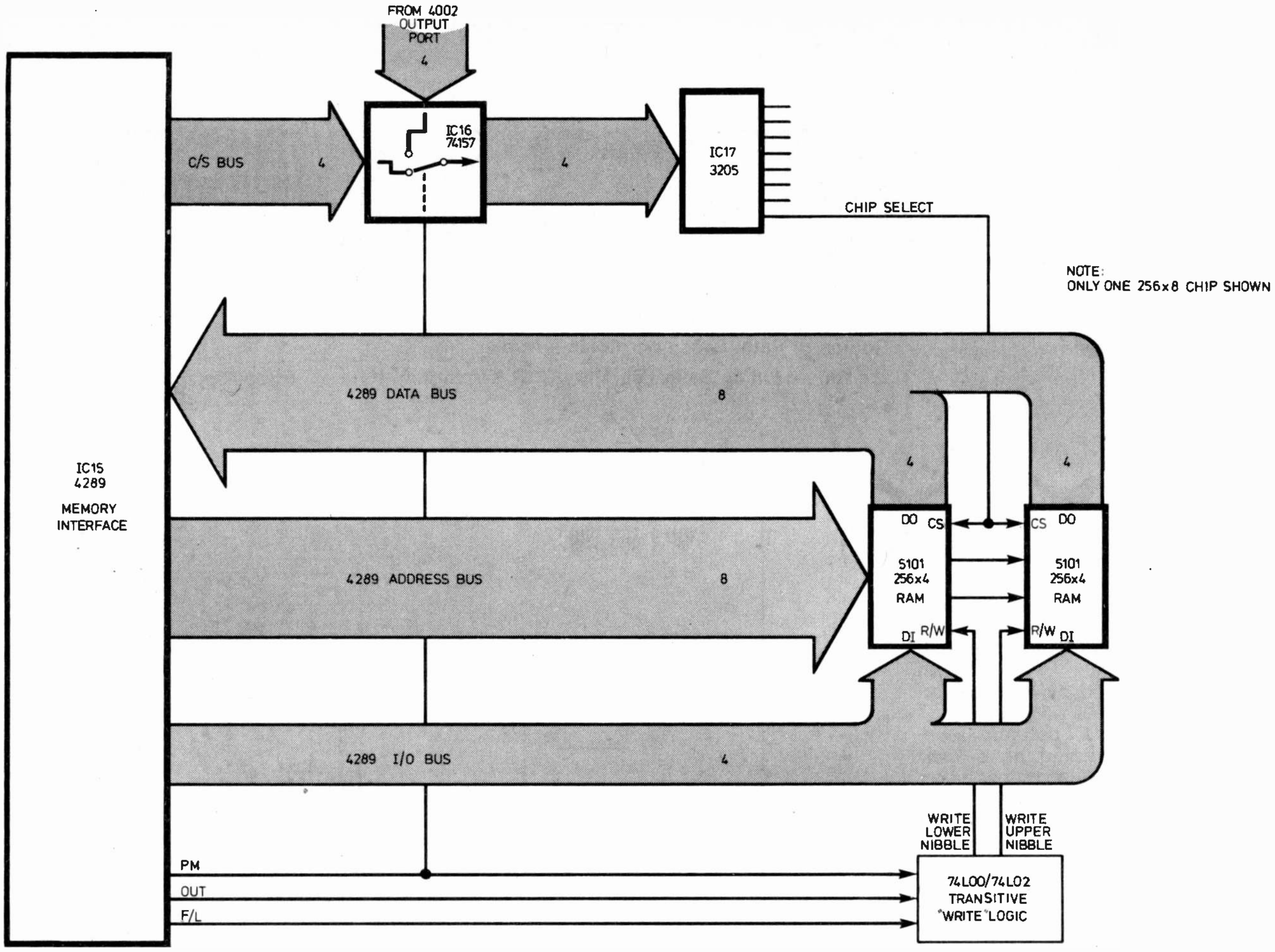


**Fig. 3.3. Simplified schematic diagram of addressing CHAMP RAM program memory**

To carry out a transitive write then, as required by an *ENTER DATA* command, the following sequence is necessary.

1. Select 4002 port
2. Write chip code to 4002 port
3. *SRC* to select location within chip
4. *WPM* to write first half-byte
5. *WPM* to write second half-byte.

A similar procedure is necessary to achieve a transitive read, as required by the *DUMP* command. Further details of the intricacies of addressing program memory can be gained from the 4289 data sheet, although of course to *use* CHAMP as a development system it is not *essential* to be familiar with these. See Fig. 3.3 for simplified circuit operation.

## BATTERY BACK-UP

The 5101 program RAM chips are CMOS devices which have extremely low standby current drain. Components B1, D19, D20 and R40 form a battery supply circuit which will power the 5101s with the CHAMP main supplies turned *off*. B1 is a three-cell DEAC nickel cadmium battery which provides about 4 volts and is recharged via D20 and R40 when the power is on. When power is removed D20 becomes reverse biased, isolating the 5101s from the +5V line, and D19 becomes forward biased to supply the memory standby current. Note that a dry cell battery of 4·5 volts could be used instead of the DEAC if R40 is left out, although you could end up losing data when the battery eventually runs flat. It is difficult to say just how long this would take.

## POWER FAIL DETECT

To ensure that the memory is not corrupted by write transients during power failure or recovery, it is necessary to raise the $CE_2$ input to the 5101s only when the main 5 volt supply is available, and to achieve this control a "power-fail-detection" circuit formed by D13, R32, R33, TR1, R34 and part of IC10 is provided. The transistor is held on by the conduction of D13 until the 5 volt line starts to drop. When it drops below about 4·5 volts D13 and TR1 turn off and $CE_2$ is grounded via the 74L00 gate.

## USING OTHER MEMORIES

If you can do without the non-volatile feature made possible by the 5101 devices for all or part of your program RAM, then you can leave out the battery circuit and the power fail detect circuit and plug in 2101 devices which are available at very low cost. The 2101s are completely compatible with the rest of the CHAMP circuit and have been tried on the prototype.

One final note, the 5101 CMOS devices must *not* have their inputs taken negative more than a few hundred millivolts, and this is the reason for clamp diodes D1 to D12. The use of good quality *germanium* devices in these locations is essential.

**NEXT MONTH: CHAMP Keyboard, power supplies, construction**