Popular Electronics®

WORLD'S LARGEST-SELLING ELECTRONICS MAGAZINE FEBRUARY 1978/$1

HOW TO ADD

- A Cassette Interface to an Elf Computer
- An Audio Blend Control to a Hi-Fi Receiver

The New Micro/Mini Cassette Tape Formats

Now: Transfer PC Guides Without Film or Chemicals!

"Oscar": Communications Satellites for Everyone

TESTED THIS ISSUE

Garrard GT25 Turntable
Sansui AU-717 Stereo Integrated Amplifier
Sherwood Micro/CPU 100 Stereo FM Tuner
Ten-Tec Century/21 CW Transceiver

BUILD A LOW-COST EPROM PROGRAMMER

0 02
14024 14278
Popular Electronics

FEBRUARY 1978

Using Existing House Wiring For Computer Remote Control PART 3

Construction and Software

BY DAN SOKOL, GARY MUHONEN, AND JOEL MILLER

THIS concludes the series of articles on computer remote control.

Data Recovery and Clock Generator. This circuit (Fig. 4) is also similar to its counterpart in the controller except that, in this case, the frequency-adjust potentiometer, *R15*, is a 10-turn potentiometer that is used for accurately synchronizing the frequencies of the controller and the remote.

UART. The data received from the controller is decoded and "unformated" in this circuit (Fig. 5). Data is also put into the proper format to be sent to the controller. The receiver section of the UART accepts the serial input data from the phase-locked loop (*IC12* in Fig. 4), and converts it to parallel data and status information.

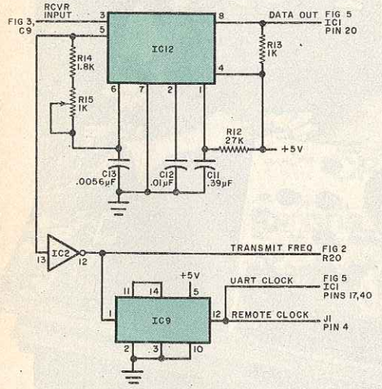


Fig. 4. Data-clock recovery is made by a PLL that delivers data output and a clock signal. The latter is divided by 16 for use in the UART.

The status information is used by the address and decode logic (Fig. 6) to indicate when data is available and if any errors occurred. Data at the receiver output is looped back to provide the first six bits of the transmit data word. The seventh and eighth bits of the transmit data word are originated by the address and command decode logic. The transmit side of the UART responds with data to the controller when the address and command logic gets a poll command.

The data word sent from the computer through the controller has a specific meaning to the remote. The first five bits (Table I) contain the address of the remote to be controlled while the sixth and seventh bits contain the command information. If the seventh bit is a zero, all remotes (up to 32 in the system) ignore the word. However, if the seventh bit is a one, the word is defined as a command to the remote whose address is contained in the first five bits. The sixth bit contains the actual command; and if it is a one, it toggles the remote channel addressed. If the sixth bit is a zero, the remote responds with poll information that informs the computer of its status (on or off). Bit 6 of the transmitted word contains the on or off information about the remote being polled (1 is on, 0 is off). Bit 7 is always a 0 during a poll.

Address, Command Logic. In the circuit shown in Fig. 6, the incoming data word is compared with that formed by the user-selected address jumpers to determine that it, and no other remote, is being addressed. The circuit then decodes one of the four possible commands and executes the decoded information. In *IC3* and *IC7*, the address is decoded and checked for errors, while *IC4* and *IC5* decode the specific command. Flip-flop *IC6* controls the state of outputs A and B, while portions of *IC8* provide the transmit side of the UART with correct poll information on the status of each side of the remote—circuits A and/or B.

Relay drivers *Q6* and *Q7* convert the outputs of the CMOS circuits to a sufficient power level.

Construction. Due to the complexity of the circuit, it is best to use a double-sided pc board as shown in Fig. 7. Note that, on the component layout guide, diodes are designated "CR" instead of "D" and integrated circuits are "U" instead of "IC." Sockets may be used for all IC's. Regulator *VR1* is mounted with a conventional heat sink and *VR2* can be mounted directly on the board with the seven transistors. Observe the polarity of the capacitors and diodes and make sure of the orientation of the IC's before installation. Note also that the conductive pot covering transformer *T1* should be electrically isolated from the foil traces beneath it by means of an insulating mica washer.

External wiring is made in accordance with Fig. 8, which shows the connections to be made to the two manual override pushbutton switches and the two sockets to be controlled. These parts are mounted on the rear apron of the selected chassis.

The pc board can be installed in any convenient chassis. If a metal chassis is used, be sure the pc board and other components are well insulated from the metal structure. Keep in mind that there is 117 volts ac on the pc board.

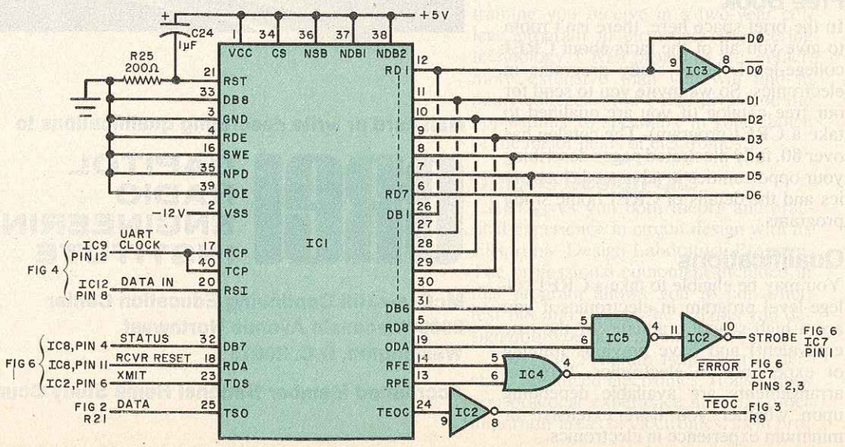


Fig. 5. UART (IC1) decodes data received by remote and formats it to be sent to controller. It also provides interface signals for other parts of circuit.

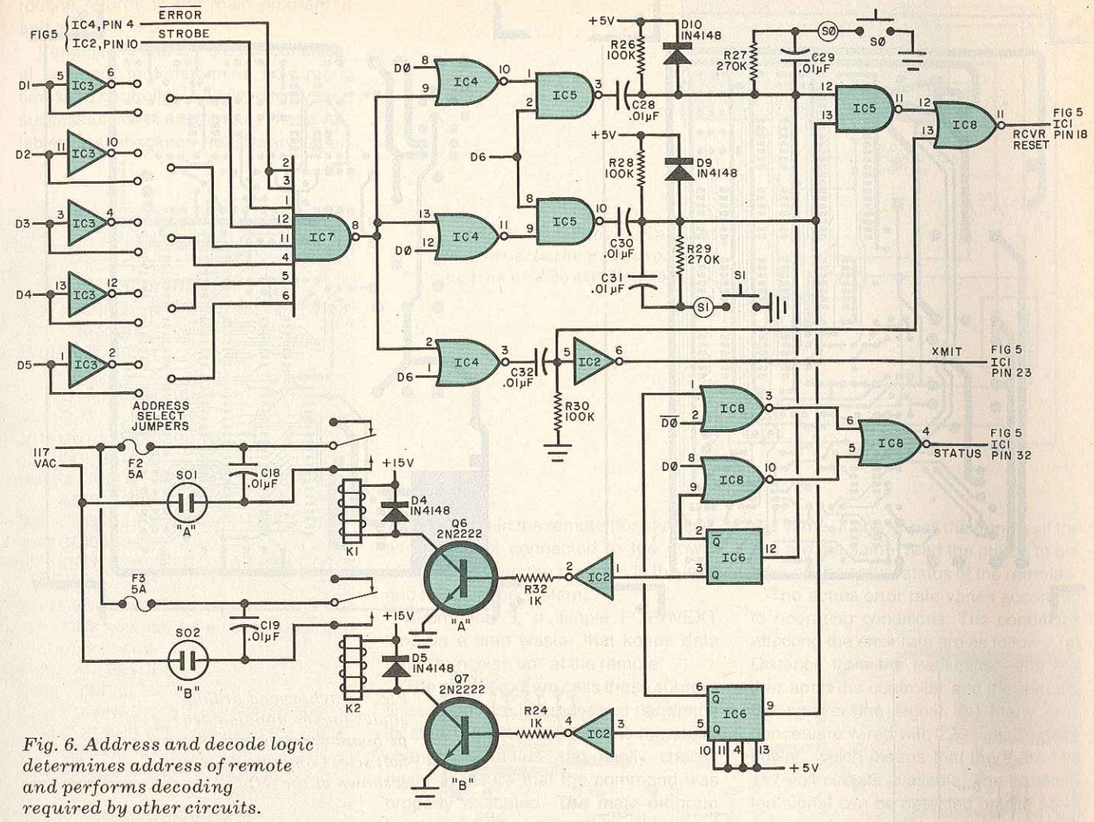


Fig. 6. Address and decode logic determines address of remote and performs decoding required by other circuits.

TABLE I

Information available at status port:

	MSB						LSB
Bit	7	6	5	4	3	2	1 0
	not	T	O	R	R		
Use	used	B	D	O	F	P	
	(always=1)	E	A	R	E	E	
Decimal	224	16	8	4	2	1	
Octal	340	40	10	4	2	1	
Hex	EO	10	8	4	2	1	

RPE = receive parity error. If this bit is a 1, then the character at the input port was received with a parity error. This bit clears when a word is received without error.

RFE = receive framing error. If this bit is a 1, then the character at the input port did not have the correct number of bits when it was received. This bit clears when a word is received without error.

ROR = receiver overrun error. If this bit is a 1, then the character at the input has overwritten the previous word (that is, the previous word was not read out prior to receiving this word).

ODA = output data available. When this bit is a 1, there is a character waiting to be read at the input data port. This bit clears when the input port is addressed.

TBE = transmitter buffer empty. This bit is a 0 during the time that the output port is busy. When it is a 1, data can be presented to the output port.

MSB							LSB
7	6	5	4	3	2	1	0
P	C	address					
O	T	0–63					
L	R						
L	L						

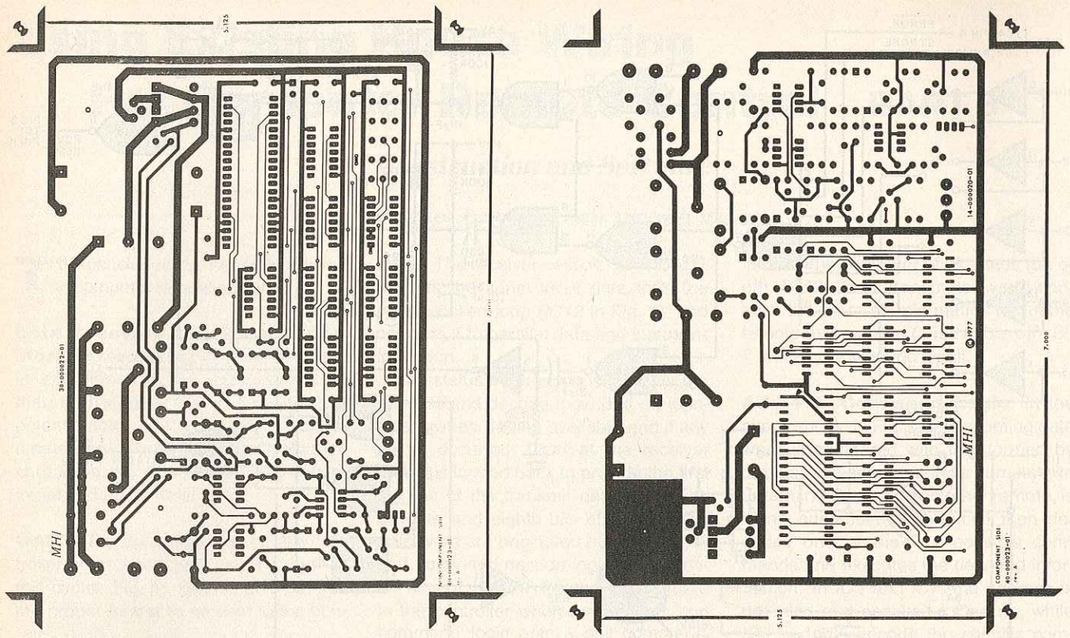
The first six bits contain the address of the remote being contacted. The poll and control bits will determine how the data is interpreted as follows:

Bit	7	6
toggle this remote	1	1
poll this remote	1	0
ignore this data	0	x

(x = don't care)

A toggle command will cause the remote to turn on (or off) depending on its previous state. For example, to toggle remote 41 (decimal) output 233 (decimal) to the controller's output port.

Software. The Intelligent Remote



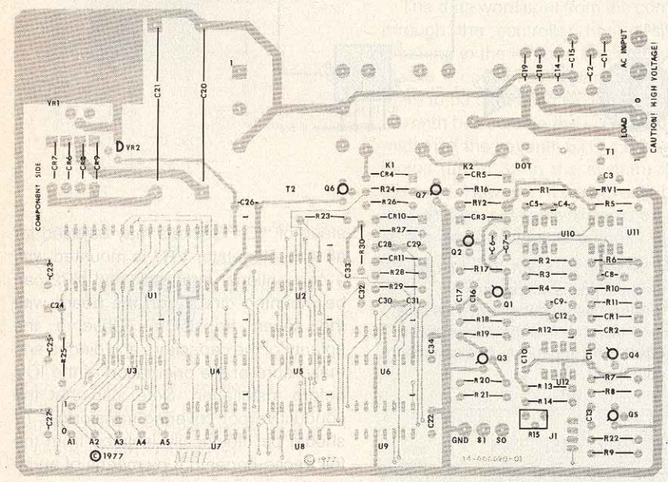


Fig. 7. Etching and drilling guides for the double-sided pc board are shown above half size. Component layout is shown at the left.

Controller is software-oriented to give the user broad flexibility in use. The software set is in two parts: (1) a group of subroutines designed to provide a format for the user to develop software particular to his application, and (2) a program to determine the background error rate and eliminate it. Both programs are written in BASIC for ease of use and are shown in Table II.

Subroutine 1 is a loop that waits for the transmit buffer to clear (TMBT = 1). Several assumptions are made, and the user may have to change these depending on where he has placed his board on the I/O map. These assumptions are: (a) the data input port is jumpered for 5; (b) the data output port is jumpered for 5; (c) the status port is jumpered for 4; (d) the remote is jumpered for addresses 52 and 53.

Subroutine 1 inputs the data at the status port and masks out all but TMBT (which is equal to 16). When TMBT is true, it returns to the main program. If TMBT is false, it remains in the loop to continue the search for a true TMBT.

Subroutine 2 is the polling subroutine that contains a series of conditional loops that wait for a valid response from a specific remote. The following variables in subroutine 2 have these meanings: (a) E is the main error flag and, if this routine returns to the main program with E = 1, then there was an error that could not be corrected within the constraints of this subroutine; (b) P contains the data that is transmitted to the remote plus 128 (bit-7 = 0); (c) X is the data in the status port that is updated during the subroutine; (d) D is the data received from the remote. It is valid when the subroutine returns to the main program, if and only if, E = 0.

Variables C and C1 are the conditional counters that determine how many times the controller is allowed to try for a successful poll of the remote. These variables are absolutely necessary otherwise a failure in the remote (for example, a remote not connected to the power line) would keep the program in the loop and hang up the system.

TABLE II

Subroutine 1

```
1000 X=INP(4) : IF (X AND 16)=16 THEN
     RETURN
1010 GOTO 1000
```

Subroutine 2

```
5000 C=0 : C1=0 : E=0
5010 OUT 5,P : GOSUB 1000
5015 GOSUB 8000
5020 X=INP(4) : IF (X AND 8)=8 THEN
     GOTO 5200
5030 C=C+1 : IF C > 5 THEN GOTO 5100
5040 GOTO 5020
5100 C1=C1+1 : IF C1 > 5 THEN GOTO
     5150
5110 GOTO 5010
5150 REM you can put an error flagging
     routine here
5160 E=1 : RETURN
5200 D=INP(5)
5210 IF(X AND 7) > 0 THEN GOTO 5100
5230 IF (D AND 63) <> (P AND 63) THEN
     GOTO 5010
5240 RETURN
```

Subroutine 3

```
8000 REM time waster
8010 FOR N=1 TO 15
8020 N1=N1+1
8030 NEXT N : RETURN
```

Main Program

```
 10 DIM R(2),A(2)
 20 A(1)=52 : A(2)=53
 30 FOR I=1 TO 2
 40 R(I)=A(I)+128+64
 45 P=R(I)-64
 50 GOSUB 5000 : REM Call the polling
    routine
 55 Z=Z+E
 60 T1=(D AND 64)
 70 IF E=1 THEN GOTO 50
 80 OUT 5,R(I) : GOSUB 1000
 90 GOSUB 8000 : REM time waster
100 GOSUB 5000
105 Z=Z+E
106 IF E=1 THEN GOTO 150
110 T2=(D AND 64)
120 IF T2=T1 THEN GOTO 80
130 CO =CO+1 : REM CO counts the num-
    ber of times through the loop
135 IF CO/25 <> INT(CO/25) THEN
    GOTO 150
140 PRINT "CYCLES =";CO;" ERRORS
    =";Z;" % =";(Z/CO)*100
150 NEXT : GOTO 30
```

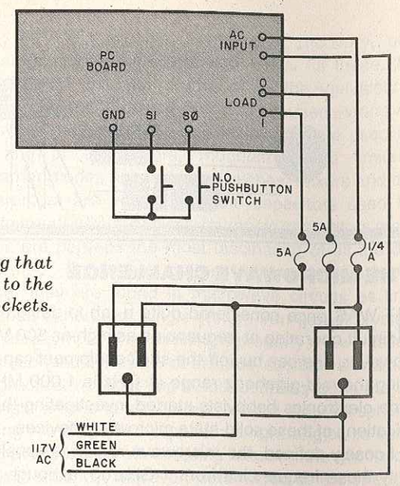


Fig. 8. External wiring that connects the pc board to the ac line and controlled sockets.

Armed with this knowledge, the user can change the conditional loops until the point of zero errors is reached. Typical error rates with only one pass are 5 to 8%. With this as a background error rate, four passes will make the error rate less than 0.01%.

Errors induced by noise from the ac line are a fact of life. Fortunately, the computer can be taught to recognize and correct errors in transmission. If an error is detected by a remote, it ignores the command. If an error rate exists in response to a poll, it is easily detected.

For example, first test bits 1, 2, and 3 of the status port. If any of these three bits is a 1, then an error has been detected. Read the input port to clear the RDA bit, but ignore the data. If all three bits are 0, then compare bits 0 through 5 with the address polled. They must be the same. If not, re-poll the remote. If the bits are the same, read the poll bit to determine the actual status of the remote.

The actual error rate varies according to operating conditions. The conditions affecting the error rate are as follows: (a) Distance from the transmitter—the farther apart the controller and the remote, the weaker the signal. (b) Many residences are wired with 220-volt, 3-phase power, which means that there are two 117-volt circuits available. The transmitted signal can be detected on the other phase, but greatly attenuated. A 0.01-μF, 600-volt capacitor across the 220-volt line will correct this problem. (c) High-amplitude, wideband noise, generated by older brush-type ac motors, can cause problems. If you can't replace the motors, then you will have to live with the problems. (d) Impulse noise caused by high-current inductive devices (refrigerators, air conditioners, etc.) when they turn on and off is a random factor that can produce single-bit errors. Fortunately, this type of noise is just as rapidly attenuated as the useful digital signal. (e) Triac noise, usually produced by poorly designed light dimmers, can raise the error rate.

The variables in the main program are as follows: (a) A(I) is an array that contains the addresses of all remotes; (b) R(I) is the toggle command for the remote channel and is equal to A(I) plus 128 plus 64; (c) P is the poll command for each remote and is equal to R(I) minus 64; (d) T1 and T2 contain the poll status of the remote before and after it has been toggled; (e) D is the data from the remote; (f) Z is the total number of errors that have been detected. ◇