Joe

A CHEAP GRAPHICS COMPUTER - CONVERT THE RCA STUDIO II

The RCA Studio II is a little known programmable TV game which contains a video-oriented microcomputer based on the RCA 1802 microprocessor. RCA made over 150,000 Studio IIs but then decided to get out of the TV-game business, and so they sold their remaining inventory to Radio Shack. For about $59.95, you can buy one of these sophisticated TV games at Radio Shack, and for an additional $15 - $20 or so, can build a plug-in card (similar to the extra plug-in game cards sold with the Studio II) which will allow you to enter your own machine language programs into the Studio II RAM, edit them, and run them. So for less than $80, you will have a small CMOS microcomputer which has sound output, video output with its own RF modulator, and two keyboards for input. This can be done without any internal modifications to the Studio II; all you need is the plug-in card to be described below, which contains a PROM and a few support ICs. If you are involved in teaching microcomputer fundamentals to a group of students, this is a cheap way to give everybody their own computer. With a copy of the 1802 User's Manual ("User Manual for the CDP1802 COSMAC Microprocessor", MPM-201A, RCA, $5.00) in front of you, it is an excellent and fun way to learn about microprocessor systems and machine language. When you order the 1802 Manual, also ask for the data sheet for the CDP1861 TV Chip. For further articles on 1802 systems, see W. Haberhern's article in the Feb., 1977 Kilobaud, or Joe Weisbecker"s "ELF" articles in Popular Electronics, August and September, 1976; July, 1977.

The Basic Studio II

The Studio II home video game unit was designed to be programmable for an almost unlimited variety of games via the insertion of new ROM cartridges. The basic circuitry is shown in Figure 1. The entire circuit operates from a 5 Vdc supply at a processor clock frequency of 1.76 MHZ. Pushing "CLEAR" resets the 1802 and starts execution of the stored program in the ROM, IC11. This ROM and IC12 contain an interpreter which fetches instructions from the other 2 ROMs, IC13 and IC14. When a card is plugged into the 22 pin socket, J1, IC13 and 14 are disabled, thus causing the interpreter to now fetch instructions from the ROMs on the plug-in card. The PROM card you will build works the same way, except that its instructions tell the Studio II to accept and run your program, rather than telling it to play a specific game. The heart of the Studio II TV display system is made up of the 1802 microprocessor, the 1861 TV chip, and the 1822 RAMs. This combination of ICs bit-maps half of the RAM (hexadecimal addresses 0900-09FF) onto the TV screen, as shown in Figure 2. Thus, the game program puts objects in the display and moves them around by writing 1s to this RAM (1s = white, 0s = black). This is done very easily by a write-to-memory machine language instruction (see the 1802 User's Manual). The 1802 shifts this RAM data out to the TV display 60 times per second by using its DMA-OUT and INTERRUPT capabilities, and special features of the 1861 TV Chip. The following sequence of events is repeated every 1/60th of a second: The TV Chip sends an interrupt request signal to the 1802 which causes the interrupt program to be executed. The purpose of this interrupt routine is to reset the 16 bit register R0 in the 1802 to point to the first address in the RAM to be displayed

(this address is 0900). This is done a few horizontal TV lines before the display is to start. To start the display, the TV Chip now sends a DMA-OUT request to the 1802 which lasts just long enough to cause the 1802 to read 8 sequential bytes in memory, starting at the address pointed to by R0(0900). (The 1802 automatically increments R0 by one every time a byte is read out in the DMA mode). These 8 bytes are read from memory, along the data bus, and into the TV Chip, where they are immediately converted into serial video representing one horizontal TV line. This video is combined with horizontal and vertical sync signals also generated in the TV Chip and is fed into an RF modulator and from there goes directly to the TV. The next 127 video lines which make up the display are generated in the same way. You will notice that each horizontal TV line is repeated four times in the display. This is done by the interrupt routine, which executes a few instructions at the end of each horizontal TV line to reset the R0 pointer three times for each set of 8 bytes. The display RAM has 256 bytes, which is 32 sets of 8 bytes (8 x 32 = 256) or 32 different lines. Repeating each line four times (4 x 32) gives 128 TV lines in the display (See Figure 2). The CDP1861 data sheet explains some of this in more detail.

The keyboard is scanned by latching four-bit numbers into the CD4515 (4 to 16 decoder) for the keys we want to check. As each number is latched in, this causes that particular decoder output to go low and either EF3 (A Keyboard) or EF4 (B Keyboard) is checked by the 1802 at the same time to see if it is low (If EF3 or EF4 is low, it means that key has been pressed).

To summarize, the technique for writing your own game or graphics program is to have your program first write 1s to the appropriate addresses in the display RAM to form your display. Then make it modify the display based either on the keyboard inputs or on your own internal algorithm.

The interrupt routine and the hardware will take care of the display refresh. The plug-in card to be described below will allow you to store your game or graphics programs in the Studio II scratchpad RAM (0800-08FF) and then execute them.

Plug in the card

When the PROM card is inserted in the Studio II, any of four modes are selectable through Keyboard A, as described below (all addresses are in hexadecimal):

A. Load Mode (Key 1) - When Key 1 is pressed, the Studio II waits for the starting address of your program to be entered. You enter this address in hex notation, using the A keyboard for digits 0-9 and the B keyboard for digits A-F (A=1, B=2, C=3, D=4, E=5, F=6). An overlay for the B keyboard will make this easier. The address will be displayed on the TV. The normal starting address is 0800 and your program can normally extend from there to address 08FB. There is also RAM from address 0900 to 09FF; this RAM is normally bit-mapped into the display as described earlier, but may be used for program storage if your program does not need a full display. An example of what can be done with this limited memory is given in a sample program in Figure 5. Programs as complex as a "Lunar Lander" program can be written in this small memory by making use of subroutines that are in the plug-in card PROM and which can be called by your program (See the section, "Using the Card").

After you enter the four-digit starting address, you next enter the first byte of your program. You then continue keying-in your program through the keyboard one byte after another until you are done. You will notice that the address display automatically increments itself as you enter each byte, and that each data byte is displayed as it is entered.

B. Memory-Read Mode (Key 2) - When Key 2 is pressed, enter the address of the memory location you wish to examine. Pressing any key on Keyboard A will then display the byte stored at that location. Continuing to press any A key will sequentially step

through memory, displaying its contents and incrementing the address display at the same time.

C. Run Mode (Key 3) - The program that has just been loaded can be run by pushing Key 3 and then entering the 4 digit starting address of the program. The program will start running as soon as the last digit of the starting address has been entered, and will use R9 as the program counter.

D. Shift Mode (Key 4) - Suppose you have loaded a program and then want to add several bytes in the middle. Pressing Key 4 and then entering the 4 digit starting address for the shift will allow you to shift the entire program after this starting address. The shifting is done by pressing either Key 8 or Key 9, A Keyboard. Press Key 8 if you want to shift page 8 (0800-08FB), and press Key 9 if you want to shift page 9 (0900-09FF). Each key press will shift the entire program after the shift starting address one location higher in memory.

When you have completed your entries in one mode and wish to switch to another, press Key 0 on Keyboard B. DO NOT PUSH "CLEAR"! Pushing "CLEAR" will destroy part of the program that you have entered. After Key 0 has been released, you may select the next mode as before. Pushing Key 0 on Keyboard B also causes the control program in the PROM to store most of the 1802 internal 16-bit registers in memory so that they can be examined by using the Memory-Read Mode. This is very useful when trying to debug a program. All registers except R0, R1, R2, and R8 are stored in the RAM at addresses 09B3 to 09CF. The last digit of the address is the register number; if the third address digit is B, it's the low order register address; if it is C, it's the high order register address. For example: 09B5 contains the contents of 1802 register R5.0 (the low order 8 bits of register R5) while 09C5 contains the contents of register R5.1 (the high order 8 bits of register R5).

Constructing the card

A schematic for the plug-in PROM card is shown in Figure 3. It uses a Harris 7641 fusible-link PROM (tri-state outputs). The circled letters are the edge-connector pins. The program for the PROM is shown in Figure 4. Since the Studio II uses primarily CMOS logic, the 4050s were included on the card to buffer the address lines. The 4042 and 4001 latch and decode the high order address each machine cycle so that the PROM responds to hex addresses 0400 through 05FF. The diode to CE1 disables the Studio II RAMs whenever the PROM is addressed. Before starting construction, make sure the card you are using will fit into the Studio II socket, clearing the aluminum ground pins on each side. When using the card, the PROM will draw an additional 100 mA or so from the Studio II power supply, putting the total drain close to the 250 mA maximum of the 9 Vdc power supply. On most Studio IIs, this will cause no problem unless you have a marginal power supply. If you have problems and suspect the supply, measure the 9 V with the PROM card plugged in. If it has dropped to 7 V or less, you will need to try another supply.

Using the card

Although you have only a small amount of RAM for your programs, you can still write fairly complex programs by making use of some of the subroutines resident on the PROM card. These subroutines are used mainly to simplify keyboard input and numerical output to the TV. Before using them, you must set the program counter of your main program to R5 (1802 register 5) and remember that these subroutines use R3 as their program counter. Calling subroutines with the 1802 is very easy once you get used to it. You simply load the register to be used as the subroutine program counter (R3 in this case) with the entry point of the routine, and then do a DN instruction, which sets the program counter register to RN (R3 in our case, using a D3 instruction). These subroutines end with a D5 instruction, which resets the program counter back to R5, which causes a return back to the main program at the point just after the D3 instruction which originally called the subroutine. The PROM card subroutines are as follows:

1). KYSC2

Scans keyboard looking for a 1 - byte input (2 digits). Returns with MSD (Most Significant Digit) in R6.1 and LSD (Least Significant Digit) in R4.1. These registers contain the digit in the form OX (X = digit). Returns with R3 pointing to entry for the DISP1 subroutine. Adds 09 to the B Keyboard entries so that 1, 2, 3, 4, 5, 6 are converted to A, B, C, D, E, F.

ENTRY: 0570

2). DISP1

Displays two hex digits (1 byte). The MSD is taken from R6.1 and the LSD from R4.1. The digits are displayed at the RAM location pointed to by RD (RD points to top left corner byte). Each digit is 6 bytes high in the display. Returns with R3 pointing to CONV2 subroutine.

ENTRY: 0598

3). KYSC1

Scans keyboard for 1 digit and puts it in R6.0 in the form OX (X=digit). Returns with R3 pointing to KYSC2. Adds 09 to B Keyboard entries so that 1, 2, 3, 4, 5, 6 are converted to A, B, C, D, E, F.

ENTRY: 05BE

4). CONV1

Takes byte in R6.0 (X_1X_2) and converts it to a form suitable for display by DISP1: OX_1 in R6.1, OX_2 in R4.1. Returns with R3 pointing to DISP1.

ENTRY: 05D6

5). CONV2

Takes byte in R6.1 (OX_1) and byte in R4.1 (OX_2) and merges them into R6.0 (X_1X_2). Returns with R3 pointing to KYSC2.

ENTRY: 05E4

Fig. 4 STUDIO II "MICROCOMPUTER" PROM LISTING

ADDRESS	DATA (HEX)							
0400	0402	F804	B1B5	F8AA	A1F8	0DA5	D5F8	09BB;
0410	BDB2	F800	ABA2	22F8	05B3	BC7A	E2E2	F8BE;
0420	A3D3	86B7	D3F8	D0AD	D3D3	86B9	D3F8	D2AD;
0430	D3D3	86A9	97FB	0132	7497	FB03	3245	97FB;
0440	0432	8A30	46D9	F8BE	A3D3	09A6	F8D6	A3D3;
0450	F8D6	ADD3	F8BE	A3D3	1999	A6F8	D6A3	D3F8;
0460	D0AD	D389	A6F8	D6A3	D3F8	D2AD	D397	FB01;
0470	327C	304A	D3F8	D6AD	D3D3	8659	19F8	70A3;
0480	D3F8	D6AD	D3D3	8659	3059	F8BE	A3D3	86B6;
0490	F8FC	A646	5626	E686	7316	89F3	263A	93E2;
04A0	308A	0F0F	0F0B	030F	7270	C422	7822	528B;
04B0	A09B	B028	E2E2	80E2	20A0	E220	A0E2	20A0;
04C0	E23C	B622	F800	5262	37D0	2284	5262	30A8;
04D0	F805	B8F8	43A8	37D6	3CD8	34DA	72F8	2852;
04E0	30A9	0D0F	0B0B	0B0F	0F0F	0F0F	0F0F	0F0F;
04F0	0F0F	0F0F	0F0F	0F0F	0F0F	0B0F	0F0F	0F0F;
0500	3039	222A	3E20	2434	2628	2E18	141C	1012;
0510	F080	F080	F080	8080	F050	7050	F050	5050;
0520	F080	F010	F080	F090	F090	F010	F010	F090;
0530	F090	9090	F010	1010	1060	2020	2070	A0A0;
0540	F020	20F8	D1A2	F809	B2BD	F8D8	73F8	9F52;
0550	92B1	82A1	22D1	7321	2141	21FF	0151	FB82;
0560	3A55	F804	B1B5	F8AA	A1F8	CFAD	1D30	F2D5;
0570	E27A	F805	A888	3A75	1484	5262	2236	833F;
0580	78FC	09FA	0F31	8FB6	3688	3788	7B30	72B4;
0590	7A36	9137	9130	97D5	96AC	0CAC	F805	A7F8;
05A0	005D	8DFC	08AD	4C5D	8DFC	08AD	2787	3AA6;
05B0	31BA	7B8D	FF2F	AD94	3099	7A30	E3D5	E214;
05C0	8452	6222	36CA	3FBF	FC09	FA0F	A67B	36CE;
05D0	37CE	7A30	6FD5	86F6	F6F6	F6B6	86FA	0FB4;
05E0	3097	FFD5	96FE	FEFE	FE52	94F1	A630	6F00;
05F0	0000	F800	5D8D	FBFF	3A6C	F80D	A5D5	081D

Figure 4

SIMPLE DISPLAY PROGRAM

ADDRESS	DATA (HEX)	COMMENTS
08C8	F8 08 B5 B3	R3.1 = 08
08CC	F8 D0 A5 D5	Set R5 = Program Counter = 08D0
08D0	F8 F0 A3	R3.0 = F0
08D3	87 AA A6 B6	R7.0 -> RA.0, R6.0, R6.1 (See Note)
08D7	F8 09 BA BC	RA.1 = RC.1 = 09
08DB	F8 FF AC	RC.0 = FF
08DE	06 5C	M(R6) -> M(RC)
08E0	D3 FF	Call Delay Subroutine at 08F0 with delay = FF
08E2	08 5A 1A	M(R8) -> D, D -> M(RA), RA + 1
08E5	2C 8A	RC-1, RA.0 -> D
08E7	3A DE	Go To 08DE if D ≠ 0
08E9	16 18 D3 FF	R6=R6 + 1; R8=R8 + 1; Call Delay Subroutine
08ED	30 D7	Go to 08D7
08EF	D5	
08F0	45	DELAY
08F1	FF 01	SUBROUTINE
08F3	3A F1	(PROGRAM COUNTER = R3)
08F5	30 EF	

Note: R7.0 is set to zero by PROM program
R8 is incremented once every 60th of a second.

Figure 5 - Sample Program