JASON-RANHEIM COMPANY

580 Parrott Street
(408) 287-0259

San Jose, California 95112
(408) 287-0264

PCC-8 BANK SWITCHING EPROM CARTRIDGE BOARD FOR THE C64

DIRECTIONS FOR USE

The PCC-8 is designed for use with 2764's, 27128's, 27256's (with minor board mod. as described below) and pin compatible types such as the 5133, 5143, X2864AD, etc. From one to eight EPROM's may be used in any combination of types, giving from 8k to 256k of ROM capacity in the cartridge in 8k increments. Any 8k segment of ROM can be switched into the $8000-$9FFF (32768-40959) address space of the computer by storing the appropriate number into the bank select register in the cartridge. In addition, ROM can be switched out completely and 8k of computer RAM is available instead.

Bank Select Register. This register (BSR for short) is a 6 bit write only register located at $DFFF (57343). The address of the BSR may be changed to $DEFF (57087) as described below, to avoid conflict with other cartridges, interfaces, etc. which may be plugged into the cartridge port at the same time.

There are eight sockets on the board, numbered from 0 to 7 as shown below. The active socket is selected by bits 2,3, and 4 of the BSR data. If the selected socket contains a 2764, then data bits 0 and 1 of the BSR have no effect. If the selected socket contains a 27128, then bit 0 of the BSR data controls whether the lower or upper half of the 27128 is selected. (Bit zero acts like A_{13}). If the selected socket contains a 27256, then bit 1 of the BSR data selects the lower or upper half of the 27256 (bit one acts like A_{14}), while bit zero selects the lower or upper quarter within the selected half.

Bit five of the BSR data controls the $\overline{\text{EXROM}}$ line. A one here deselects ROM and restores the underlying 8k of RAM.

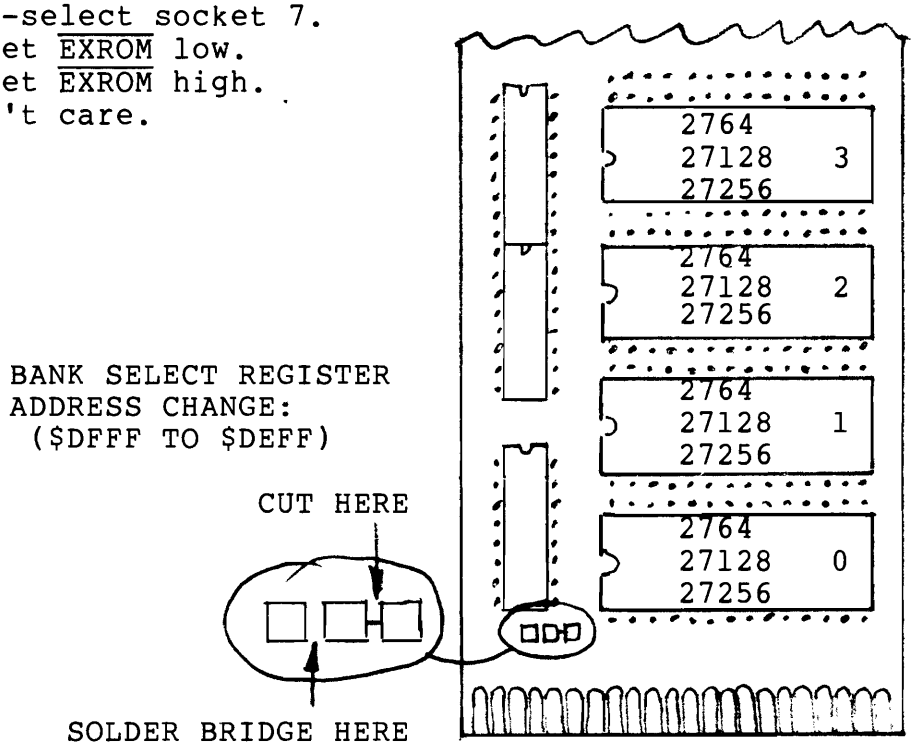
On power up, the BSR is reset to ZERO. Therefore, if the cartridge is to auto-start, bank zero must contain the initial ROM. "Peeking" the BSR resets it to ZERO.

27256 Modification. At each socket where a 27256 is to be used, make the modification shown on the next page. Carefully cut through the trace connecting pins 27 and 28 of the socket. Scrape the solder mask material from the surface of the split trace and bridge with a 'blob' of solder. To use the socket again with a 2764 or 27128, undo the modification desoldering and resoldering as necessary.

BSR CONTENTS

BIT 7 6 5 4 3 2 1 0

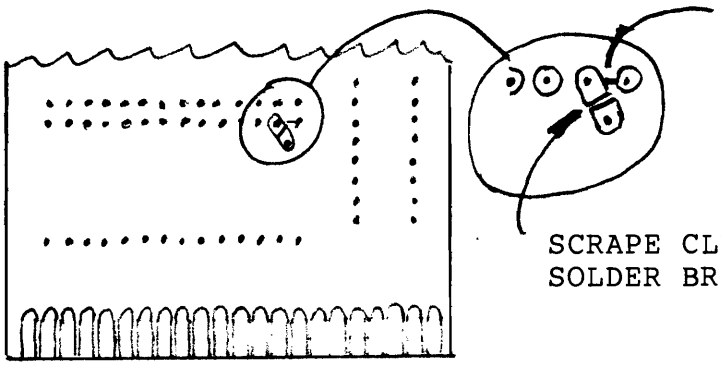
- bit 0: 0-select lower half of 27128 or lower quarter of 27256.
  1-select upper half of 27128 or upper quarter of 27256.
- bit 1: 0-select lower half of 27256.
  1-select upper half of 27256.
- bits 4 3 2: 000-select socket 0.
  001-select socket 1.
  010-select socket 2.
  011-select socket 3.
  100-select socket 4.
  101-select socket 5.
  110-select socket 6.
  111-select socket 7.
- bit 5: 0-set $\overline{\text{EXROM}}$ low.
  1-set $\overline{\text{EXROM}}$ high.
- bits 7 6: don't care.



BANK SELECT REGISTER
ADDRESS CHANGE:
($DFFF TO $DEFF)

CUT HERE

SOLDER BRIDGE HERE



CUT TRACE HERE

MODIFICATION FOR 27256

SCRAPE CLEAN, AND
SOLDER BRIDGE HERE

INSTRUCTIONS FOR PROGRAMMING EPROMS FOR BANK SWITCHING BOARD

This procedure results in up to four programmed EPROMs containing a basic program of up to 38k bytes, with the necessary download and run routine at the beginning of the first EPROM.

STEP 1- LOAD"DOWNLOAD/RUN",8,1 <CR> (or ,1,1 if from cassette).

This loads the auto-boot routine into RAM at 49152.

NEW <CR>

STEP 2- LOAD"PROMOS 1.0 C64",8 <CR> (or LOAD""if PROMOS is on cassette).

Loads 'PROMOS' into the basic area.

STEP 3- POKE56,208 <CR>
RUN <CR>

PROMOS relocates into $C000 block (behind the auto-boot)

POKE55,0 <CR>
POKE56,160 <CR>

Reset the top of basic pointer.

STEP 4- LOAD"OBJECT PROGRAM",8 <CR>

Load the object program into the basic area.

STEP 5- POKE49161,PEEK(45) <CR>
POKE49162,PEEK(46) <CR>

Stores the start of basic variables pointer in the auto-boot routine for transfer to the first EPROM.

Z <CR>

Zero the PROMENADE C1 programming socket.

STEP 6- Insert the first EPROM. (2764 is assumed.)

π 49152,49327,0,5,7 <CR> (176 bytes)

Programs auto-boot routine into EPROM.

π 2049,10064,176,5,7 <CR> (8016 bytes)

Fills balance of first EPROM with 8016 basic bytes.

STEP 7- Insert the second EPROM into the programmer.

π10065,18256,0,5,7 <CR> (8192 bytes)

STEP 8- Insert the third EPROM into the programmer.

π18257,26448,0,5,7 <CR> (8192 bytes)

STEP 9- Insert the fourth EPROM into the programmer.

π26449,<MEM END>,0,5,7 <CR> (up to 8192 bytes with 2764, or 16384 bytes with 27128).

Here, <MEM END> = PEEK(45)+256*PEEK(46) -1

STEP 10- Install the first, second, third and fourth EPROMS into sockets 0, 1, 2, and 3 respectively. The cartridge is ready for use.

NOTES- The RUN/STOP (keyboard break) is disabled by the auto-boot routine.

The auto-boot routine checks for EPROM type (2764, 27128, or 27256) and automatically down-loads the proper data. EPROM types can be mixed and can be used in any socket. The "π" commands must of course contain the proper memory parameters.

DISASSEMBLY OF DOWNLOAD/RUN ROUTINE

```
.C000 0B       ???
.C001 C3       ???
.C002 BC FE C3 LDY $C3FE,X
.C005 C2       ???
.C006 CD 38 30 CMP $3038
.C009 00       BRK
.C00A 22       BRK
.C00B 8E 16 D0 STX $D016
.C00E BD 00 80 LDA $8000,X
.C011 9D 00 C2 STA $C200,X
.C014 E8       INX
.C015 D0 F7    BNE $C00E
.C017 4C 1A C0 JMP $C01A
.C01A 20 A3 FD JSR $FDA3
.C01D 20 50 FD JSR $FD50
.C020 A9 A1    LDA #$A2
.C022 8D 84 02 STA $0284
.C025 20 15 FD JSR $FD15
.C028 20 18 E5 JSR $E518
.C02B 20 53 E4 JSR $E453
.C02E 20 BF E3 JSR $E3BF
.C031 A9 27    LDA #$27
.C033 8D 26 03 STA $0326
.C036 A9 FE    LDA #$FE
.C038 8D 23 03 STA $0323
.C03B A2 00    LDX #$00
.C03D 86 C1    STX $C1
.C03F 86 C3    STX $C3
.C041 A9 51    LDA #$51
.C043 85 A3    STA $A3
.C045 A9 07    LDA #$07
.C047 85 A4    STA $A4
.C049 A0 B0    LDY #$B0
.C04B 2C A0 00 BIT $00A0
.C04E A9 80    LDA #$80
.C050 85 C2    STA $C2
.C052 85 C4    STA $C4
.C054 B1 C1    LDA ($C1),Y
.C056 91 A3    STA ($A3),Y
.C058 C8       INY
.C059 D0 F9    BNE $C054
.C05B E6 A4    INC $A4
.C05D E6 C2    INC $C2
.C05F A5 C2    LDA $C2
.C061 C9 A0    CMP #$A0
.C063 D0 EF    BNE $C054
.C065 AD 0A C0 LDA $C00A
.C068 C5 A4    CMP $A4
.C06A 90 31    BCC $C09D
.C06C D0 07    BNE $C075
.C06E A4 A3    LDY $A3
.C070 CC 29 C0 CPY $C029
.C073 B0 28    BCS $C09D
```

8000, 8001- Pointer to start of auto-boot routine.
8002, 8003- Pointer to NMI handler.
cbm80- Cartridge identifier.

Will contain pointer to start of basic variables.

For b enefit of 'VIC' chip.

Transfer auto-boot routine out of ROM.

Continue in new location.

System initialization.

Set top of memory pointer.

System initialization.

Disable RUN/STOP (keyboard break).

Get ready to down-load basic from ROM.

Down-load 8k segment.

Finished?

```
.C075 A0 00     LDY #$00
.C077 8A        TXA
.C078 29 03     AND #$03
.C07A C9 03     CMP #$03
.C07C F0 13     BEQ $C091
.C07E E8        INX
.C07F 8A        TXA
.C080 29 FC     AND #$FC
.C082 8D FF DF  STA $DFFF
.C085 B1 C3     LDA ($C3),Y
.C087 8E FF DF  STX $DFFF
.C08A D1 C3     CMP ($C3),Y
.C08C D0 BE     BNE $C04C
.C08E C8        INY
.C08F D0 EE     BNE $C07F
.C091 8A        TXA
.C092 29 3C     AND #$3C
.C094 18        CLC
.C095 69 04     ADC #$04
.C097 AA        TAX
.C098 8E FF DF  STX $DFFF
.C09B D0 AF     BNE $C04C
.C09D 85 2E     STA $2E
.C09F AD 09 C0  LDA $C009
.C0A2 85 2D     STA $2D
.C0A4 A9 20     LDA #$20
.C0A6 8D FF DF  STA $DFFF
.C0A9 20 59 A6  JSR $A659
.C0AC 58        CLI
.C0AD 4C AE A7  JMP $A7AE
.C0B0 00        BRK
.C0B1 00        BRK
.C0B2 00        BRK
.C0B3 00        BRK
.C0B4 00        BRK
.C0B5 00        BRK
.C0B6 00        BRK
.C0B7 00        BRK
.C0B8 00        BRK
.C0B9 00        BRK
.C0BA 00        BRK
.C0BB 00        BRK
.C0BC 00        BRK
.C0BD 00        BRK
.C0BE 00        BRK
.C0BF 00        BRK
.C0C0 00        BRK
.C0C1 00        BRK
.C0C2 00        BRK
.C0C3 00        BRK
```

Check for EPROM type and set Bank Select Register to next appropriate value. Then move next 8k segment.

Set start of variables pointer.

Switch out ~~ROM,~~ switch ~~in RAM~~.

Do 'CLR' and reset stack pointer.

Permit normal interrupts.

'RUN' the now down-loaded basic.