

SWTPC 6800 Short Memory Address Convergence MEMCON-3 MODIFIED FOR MIKBUG® OR SWTBUG®

This Memory Convergence diagnostic is one designed to check for and locate address convergence problems in SWTPC 6800 Computer System memory boards. The program itself uses 56₁₀ words and is meant to be loaded within the 128 word RAM used by the SWTBUG® operating system on the MPA Microprocessor System board making the program independent of the MPM RAM memory. The diagnostic may be loaded from either tape or from the terminal instruction by instruction using SWTBUG® starting from address A014₁₆ thru A033₁₆ and then from A048₁₆ thru A051₁₆. The program must be loaded in two parts to avoid interfering with the system's pushdown stack. The section of memory to be tested is set by loading the most significant byte of the lower memory address into A002₁₆ the least significant byte of the lower memory address into A003₁₆ the most significant byte of the upper memory address into A004₁₆ and the least significant byte of the upper memory address into A005₁₆ using SWTBUG® just as is done for SWTBUG® punch routine. The lower and upper addresses are inclusive and may be any addresses between 0000₁₆ and FFFF₁₆ with the only requirement that the lower address be less than or equal to the upper address. Since addresses A05F₁₆ thru A07F₁₆ of the SWTBUG® RAM are still available for program use, the diagnostic may run on these locations just to make sure the diagnostic itself is functioning correctly. Since the program counter is set when the program is initially loaded, the routine is initiated by typing **G** for "Go To User Program". Once initiated, the program can be stopped only by depressing the "RESET" button. The program may then be restarted after setting the program counter to A014₁₆ at A048 and A049.

The test sequence starts by loading a continuous stream of 256 sequential binary numbers from the low memory address to the high memory address, inclusive. It then goes back and sequentially reads the data in each of the locations and compares it to what actually should be there. If it finds any discrepancies within the memory cycle, one X is printed and the cycle is restarted, otherwise a # is printed to indicate successful cycle completion. Since the actual location of any detected errors does not point to the source of the problem, no provision is made for indicating the addresses of detected errors. It must also be noted that the program is not 100% effective. It would be possible to set bits in multiple locations that coincidentally would have been set anyway. However, each cycle puts different data in each memory location, so the chances of a missed problem are reduced. The program loops forever and may be exited when desired by depressing the "RESET" switch which loads the SWTBUG® control program.

If you wish to eliminate the cyclic printout of the "# " sign you can do so by changing the data in address locations A059, A05A and A05B to NOP instructions (01₁₆) using SWTBUG®. This way you only get a printout of the error cycles, if any.

MIKBUG® is a registered trademark of Motorola Inc.

SWTBUG® is a registered trademark of Southwest Technical Products Corporation.

NAM MEMCON3
 *SHORT MEMORY ADDRESS CONVERGENCE TEST
 *FOR SWTPC 6800 COMPUTER SYSTEMS
 *MIKBUG AND SWTBUG COMPATIBLE

A002		LOMEM	EQU	\$A002		STARTING ADDRESS
A004		HIMEM	EQU	\$A004		ENDING ADDRESS
E1D1		OUTEEE	EQU	\$E1D1		CHARACTER OUTPUT
A014			ORG	\$A014		
A014	F7 A0 5F	START	STA B	BSTORE		STORE ACCCB
A017	FE A0 02		LDX	LOMEM		LOAD LOW MEMORY ADDRESS
A01A	E7 00	LOOP1	STA B	0,X		
A01C	BC A0 04		CPX	HIMEM		END OF MEMORY?
A01F	27 04		BEQ	CHECK		CHECK IF FINISHED
A021	08		INX			
A022	5C		INC B			
A023	20 F5		BRA	LOOP1		
A025	F6 A0 5D	CHECK	LDA B	BSTORE		CHECKS ALL LOCATIONS FOR CORR
A028	FE A0 02		LDX	LOMEM		
A02B	E1 00	LOOP2	CMP B	0,X		
A02D	26 21		BNE	ERROR		
A02F	BC A0 04		CPX	HIMEM		
A032	20 16		BRA	JUMP		
A048			ORG	\$A048		
A048	A0 14		FDB	START		
A04A	27 0B	JUMP	BEQ	CYCLE		
A04C	08		INX			
A04D	5C		INC B			
A04E	20 DB		BRA	LOOP2		
A050	86 58	ERROR	LDA A	'X		PRINT X IF ERROR FOUND
A052	BD E1 D1		JSR	OUTEEE		
A055	20 BD		BRA	START		START OVER
A057	86 23	CYCLE	LDA A	'#		PRINT # FOR CORRECT CYCLE
A059	BD E1 D1		JSR	OUTEEE		
A05C	5A		DEC B			
A05D	20 B5		BRA	START		
A05F	00	BSTORE	FCB	0		
			END			

Dual Address Memory Test CDAT By John Christensen

The CDAT memory diagnostic can be used to help locate memory problems in a SWTPC 6800 computer system that MEMCON and ROBIT may miss. The program itself resides entirely within the 128 byte SWTBUG® RAM. The program must be loaded in two parts to avoid interfering with the systems push down stack. The contiguous section of memory to be tested is set by loading the most significant byte of the lower memory address into A002, the least significant byte into A003, the most significant byte of the upper memory address in A004 and its least significant byte in A005. The low address must be less than or equal to the upper address.

The test starts from the low address and writes a 00 into all memory up to the high address. An FF is then written into the first address and all other locations are checked to be sure they contain 00. If all are OK the FF is replaced with a 00 and an FF is written in the next memory location. This pattern continues until all memory is checked or an error is found. If the computer returns to SWTBUG®, then no errors were found.

If the program displays a register dump then a problem was discovered on the board. The register dump should look similar to the following:

ADDRESS	ERROR MSG.
V	V
F3 00 FF 0400	A079 A042.

The important parts of the dump are the ADDRESS and the ERROR MSG. areas, as denoted above. The error messages are interpreted as follows:

A077	Error on initial test pattern (can't write 0's into mem.)
A078	Error on second test pattern (can't write FF's into mem.)
A079	Dual address error low
A07A	Dual address error high

If a dual address error is found then writing into one memory location affects another. For example, if ADDRESS = 0400 and A014 contains 0410 then writing into 0400 will change the contents of 0410 or vice-versa. (A014 is a temporary index register storage location within the program that you can compare with ADDRESS in the register dump to see which two memory locations caused the error). The IC assignments table included with the memory board instructions can then be used to help locate the problem by comparing the bit pattern of the locations in error.

The CDAT program takes some time to run, so run the diagnostic over only one complete board at a time.

MEM. SIZE	APPROX. RUN TIME
1K	29 sec.
2K	1 min. 53 sec.
3K	4 min. 13 sec.
4K	7 min. 29 sec.
8K	more than 30 min.

MIKBUG® is a registered trademark of Motorola Inc.

SWTBUG® is a registered trademark of Southwest Technical Products Corporation.

```

          NAM      CDAT-2
*MEM DIAGNOSTIC (JOHN CHRISTENSEN'S)
*MODIFIED FOR MIKBUG AND SWTBUG OPERATION
E0E3      CONTRL  EQU      $E0E3
A002      A002      ORG      $A002
A002      LOTE MP  RMB      2          STARTING ADDRESS
A004      HITE MP  RMB      2          ENDING ADDRESS
A014      A014      ORG      $A014
A014      IXRT MP  RMB      2          IXR TEMPORARY STORAGE
A016 FE A0 02  START  LDX      LOTE MP
A019 B6 A0 7E          LDA  A  INIPAT
A01C A7 00      LOOP1  STA  A  0,X
A01E A1 00          CMP  A  0,X
A020 26 55          BNE  ERPNT1
A022 BC A0 04          CPX  HITE MP
A025 27 03          BEQ  LOAPAT
A027 08          INX
A028 20 F2          BRA  LOOP1
A02A FE A0 02  LOAPAT  LDX      LOTE MP
A02D F6 A0 7F          LDA  B  TESPAT
A030 E7 00      LOOP4  STA  B  0,X
A032 20 16          BRA  CHECK
A048      A048      ORG      $A048
A048 A0 16          FDB  START
A04A E1 00      CHECK  CMP  B  0,X
A04C 26 2A          BNE  ERPNT2
A04E FF A0 14  CHKLOW  STX  IXRT MP
A051 BC A0 02  LOOP2  CPX  LOTE MP
A054 27 07          BEQ  CHCKHI
A056 09          DEX
A057 A1 00          CMP  A  0,X
A059 26 1E          BNE  ERPNT3
A05B 20 F4          BRA  LOOP2
A05D FE A0 14  CHCKHI  LDX  IXRT MP
A060 BC A0 04          CPX  HITE MP
A063 27 16          BEQ  END
A065 08      LOOP3  INX
A066 A1 00          CMP  A  0,X
A068 26 10          BNE  ERPNT4
A06A BC A0 04          CPX  HITE MP
A06D 26 F6          BNE  LOOP3
A06F FE A0 14  RESTRE  LDX  IXRT MP
A072 A7 00          STA  A  0,X
A074 08          INX
A075 20 B9          BRA  LOOP4
A077 3F      ERPNT1  SWI          ERROR ON INITIAL PATTERN
A078 3F      ERPNT2  SWI          ERROR ON TEST PATTERN
A079 3F      ERPNT3  SWI          DUAL ADDRESS ERROR LOW
A07A 3F      ERPNT4  SWI          DUAL ADDRESS ERROR LO
A07B 7E E0 E3  END      JMP      CONTRL

A07E 00      INIPAT  FCB      0
A07F 00      TESPAT  FCB      0
          END

```

SWTPC 6800 Rotating Bit RAM Memory Diagnostic ROBIT-2 Modified for MIKBUG® or SWTBUG® use

This rotating bit memory diagnostic is designed to check for and locate memory retaining problems in SWTPC 6800 Computer System memory boards. The program itself uses 8510 words and is meant to be loaded within the 128 word RAM used by the SWTBUG® operating system on the MP-A Microprocessor System board. This makes the program independent of external RAM memory. The diagnostic may be loaded from either tape or from the terminal instruction by instruction using SWTBUG® starting from address A014₁₆ thru A067E₁₆. The program must be loaded in two parts to avoid interfering with the system's pushdown stack. The contiguous section of memory to be tested is set by loading the most significant byte of the lower memory address into A002₁₆, the least significant byte of the lower memory address into A003₁₆, the most significant byte of the upper memory address into A004₁₆, and the least significant byte of the upper memory address into A005₁₆ using SWTBUG® just as is done for the SWTBUG® punch routine. The lower and upper addresses are inclusive and may be any addresses between 0000₁₆ and FFFF₁₆ with the only requirement being that the lower address be less than or equal to the upper address. Since the program counter is set when the program is initially loaded, the routine is initiated after loading by typing G for "Go To User's Program". Once initiated, the program may then be restarted after setting the program counter to A014₁₆ at A048 and A049.

The test sequence starts from the lower address and loads that address with a binary 0000 0001 or 01₁₆. The data in this location is then read and verified. If accurate the "one" bit is shifted left to form a binary 0000 0010 or 02₁₆ and is then again tested. This shift left sequence continues until a binary 1000 0000 or 80₁₆ has been loaded and verified, at which time the entire sequence is repeated at the next sequential memory address. This sequence continues until the selected upper memory address is reached. The program then prints a "+" on the control terminal to indicate cycle completion and proceeds to repeat itself. The program loops forever and may be exited when desired by depressing the "RESET" switch which loads the SWTBUG® control program. When an error is detected, the memory address followed by what data should have been followed by what the memory data was, are printed out on the control terminal in hexadecimal (base 16) form. Example:

\$0110 02 00

When converted to binary this means that when address 0110, which is located in the first 1,024 words of RAM memory, was loaded with a binary 0000 0010 it was read back as containing a binary 0000 0000 which indicates a possible problem in the 21 bit memory chip in the lower 1,024 words of memory or a possible problem in the 21 bit of the memory board data transceiver or a variety of other possibilities. The best way to tell for sure is to look for a pattern in the indicated errors. Take note that once one bit error has been located at a specific memory address, the one error is printed in the form shown above and the program increments to the next address without searching for more errors in the already defective address.

If you wish to eliminate the cyclic printout of the "+" sign you can do so by changing the data in address locations A076, A077 and A078 to NOP instructions (01₁₆) using SWTBUG. This way you only get a printout of the error locations; that is if there are any. The running time of this program is very fast. It will cycle thru 2,048 words of memory in less than one second.

MIKBUG® is a registered trademark of Motorola Inc.

SWTBUG® is a registered trademark of Southwest Technical Products Corporation.

NAM ROBIT-2
 *ROTATING BIT MEMORY TEST FOR MIKBUG
 *OR SWTBUG 6800 COMPUTER SYSTEM

```

A002          LOTEMP EQU    $A002
A004          HITEMP EQU    $A004
E07E          PDATA1 EQU    $E07E
E0C8          OUT4HS EQU    $E0C8
E0CA          OUT2HS EQU    $E0CA
E19D          MCL      EQU    $E19D
E1D1          OUTEEEE EQU    $E1D1

A014          ORG      $A014
A014 FE A0 02  START    LDX    LOTEMP
A017 86 01      LODREG  LDA  A  #1          STORE 1 IN MEMORY
A019 A7 00          STA  A  0,X
A01B A1 00          CMP  A  0,X          WAS 1 WRITTEN
A01D 26 0D          BNE          ERRPNT
A01F 48          LOOP1  ASL  A
A020 68 00          ASL    0,X
A022 A1 00          CMP  A  0,X
A024 26 06          BNE          ERRPNT
A026 81 80          CMP  A  #$80          SHIFT UNTIL 80
A028 26 F5          BNE          LOOP1
A02A 20 3F          BRA          INCR1
A02C FF A0 7B  ERRPNT  STX    INXMSB
A02F CE E1 9D          LDX    #MCL          PRINT C/R L/F
A032 20 16          BRA          SKIP1

A048          ORG      $A048
A048 A0 14          FDB    START
A04A B7 A0 7D  SKIP1  STA  A  ACCA
A04D BD E0 7E          JSR    PDATA1
A050 CE A0 7B          LDX    #INXMSB          LOAD ERROR ADDRESS
A053 BD E0 C8          JSR    OUT4HS
A056 CE A0 7D          LDX    #ACCA
A059 BD E0 CA          JSR    OUT2HS          OUTPUT WHAT SHOULD BE STORED
A05C FE A0 7B          LDX    INXMSB
A05F BD E0 CA          JSR    OUT2HS
A062 CE E1 9D          LDX    #MCL
A065 BD E0 7E          JSR    PDATA1
A068 FE A0 7B          LDX    INXMSB
A06B BC A0 04  INCR1  CPX    HITEMP          COMPARE TO END ADDRESS
A06E 27 03          BEQ          FINISH
A070 08          INX
A071 20 A4          BRA          LODREG
A073 B6 A0 7E  FINISH  LDA  A  FLAG
A076 BD E1 D1          JSR    OUTEEEE
A079 20 99          BRA          START
A07B          INXMSB RMB    1
A07C          INXLSB RMB    1
A07D          ACCA   RMB    1
A07E 2B          FLAG   FCB    '+'
          END

```

SUMTEST - 2 -- Address Variable Memory Test

By Chris Courtney, RESPCO Inc.

The SUMTEST memory diagnostic can be used to supplement the ROBIT, MEMCON and CDAT tests. The program itself resides entirely within the 128 byte MIKBUG®/SWTBUG® RAM and must be loaded in two parts to avoid interfering with the system's push down stack. The contiguous section of memory to be tested is set by loading the most significant byte of the lower memory address into A002, the least significant byte into A003, the most significant byte of the upper memory address in A004 and its least significant byte in A005. When inputting the upper address use 1+ (upper location you want to check). TO check the entire lowest 4K board, for example, you would use an upper memory address of 1000, not OFFF. When loading the diagnostic the program counter is automatically set to the starting address of the program. Typing G will initiate the program.

If no errors are found in the memory being checked a + will be displayed on the screen. To completely test an area of memory the diagnostic must be allowed to run until 256 +'s have been displayed on the screen. Each time a + is displayed on the screen SUMTEST has successfully cycled through memory storing and reading a different pattern. If an error is detected the output will be similar to the following:

```
$G +++++
$06 20 16A0
(PATTERN #) (ERRANT BITS) (ADDRESS)
```

An error message such as this says that SUMTEST cycled thru memory five times without error, but on the sixth try a pattern was used that detected an error. The 06 tells what pattern number SUMTEST was working on when the error was detected. The 20 tells which bits were in error. 20_{16} converted to binary is 00100000 - the location of the 1 is the bit that is in error, in this case bit 5. Bit numbers start from 0 as shown:

```
2016 = 0 0 1 0 0 0 0 0
        7 6 5 4 3 2 1 0 BIT#
```

The 16A0 is the address where the error was detected. This address may not store a particular number or possibly writing into another address, such as 16B0, changed the contents of 16A0.

The IC assignment table supplied with the memory board should be used to help locate the problem. In the above case on an MP-8M 8K memory board the bit # 5 IC in the upper 4K of memory should be suspected.

Be sure to re-load the program before testing another area of memory. SUMTEST runs fast enough that it can be used on an entire 32K system in an acceptable amount of time.

MIKBUG® is a registered trademark of Motorola, Inc.

SWTBUG® a registered trademark of Southwest Technical Products Corp.

NAM SUMTEST2

*IMPROVED MEMORY TEST FOR SWTPC 6800
 *BYTES STORED IN MEMORY ARE THE SUM OF THE
 *MSB AND LSB OF THE MEMORY POINTER, THEREFORE
 *ADJACENT MEMORY LOCATIONS AND ADJACENT
 *PAGES CONTAIN UNIQUE CONTENTS
 *INITIALIZE LOWEST MEMORY ADDRESS IN LOTEMP
 *AND HIGHEST MEMORY ADDRESS+1 IN HITEMP
 *MODIFIED FOR MIKBUG OR SWTBUG

```

A002          ORG      $A002
A002          LOTEMP  RMB      2
A004          HITEMP  RMB      2
E07E          PDATA1  EQU     $E07E
E1D1          OUTEEEE EQU     $E1D1
E0CA          OUT2HS  EQU     $E0CA
E0C8          OUT4HS  EQU     $E0C8
E19D          MCL     EQU     $E19D

A014          ORG      $A014
A014 00      CTR     FCB      0          PASS COUNTER
A015 00      STORE  FCB      0          BIT MISMATCHED
A016 00      INXMSB FCB      0
A017 00      INXLSB FCB      0
A018 FE A0 02  START  LDX     LOTEMP
A01B 8D 39    LOOP1  BSR     INCRX      INCREMENT INDEX
A01D A7 00    STA   A  0,X
A01F 08      INX
A020 BC A0 04    CPX     HITEMP      END OF MEMORY?
A023 26 F6      BNE     LOOP1
A025 FE A0 02    LDX     LOTEMP
A028 8D 2C      LOOP2  BSR     INCRX
A02A A8 00      EOR   A  0,X
A02C 26 35      BNE     ERROR
A02E 08      RETURN  INX
A02F BC A0 04    CPX     HITEMP
A032 20 16      BRA     SKIP1      BRANCH AROUND HOLE

A048          ORG      $A048
A048 A0 18      FDB     START
A04A 26 DC      SKIP1  BNE     LOOP2
A04C 86 2B      LDA   A  #$2B
A04E BD E1 D1    JSR     OUTEEEE
A051 7C A0 14    INC     CTR
A054 20 C2      BRA     START
A056 FF A0 16    INCRX  STX     INXMSB
A059 B6 A0 16    LDA   A  INXMSB
A05C BB A0 17    ADD   A  INXLSB      ADD IN ADDR LSB
A05F BB A0 14    ADD   A  CTR         ADD IN COUNTER
A062 39      RTS
A063 B7 A0 15    ERROR  STA   A  STORE      STORE ERRANT BIT
A066 CE E1 9D    LDX     #MCL
A069 BD E0 7E    JSR     PDATA1      DO C/R L/F
A06C CE A0 14    LDX     #CTR
A06F BD E0 CA    JSR     OUT2HS      COUNTER, IN HEX
A072 BD E0 CA    JSR     OUT2HS      ERRANT BITS, IN HEX
A075 BD E0 C8    JSR     OUT4HS      ADDRESS, IN HEX
A078 FE A0 16    LDX     INXMSB
A07B 20 B1      BRA     RETURN
  
```