PHILIPS

Electronic components and materials

Technical note 092

2650 sorting routines

signetics

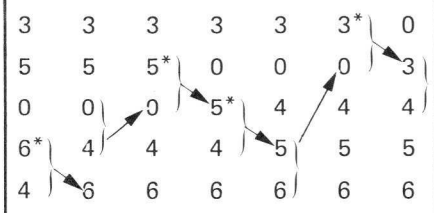
Sorting routines are often required as part of the software design for microprocessor-based systems. This Technical Note provides several examples for implementing sort routines on the 2650 microprocessor. These examples include routines for sorting single byte and multiple byte numbers, both signed and unsigned, in fixed or variable length lists. The techniques demonstrated are the 'bubble' sort, the 'search' sort, and the 'linear' sort.

BUBBLE SORT

An easy way of sorting is to compare two of the listed numbers at a time and exchange them when they are not in the right sequence. One such sorting technique is known as the "bubble" sort. Bubble sorts normally take the longest time to execute.

In a bubble sort, the last two numbers in the list are compared and exchanged if they are not in the right sequence. Then the next to the last number is compared with the number above it and exchanged if they are not in the right sequence. This process continues as each number in turn is compared with those above it. Each time a pair of numbers is exchanged, the order of searching is reversed (now going towards the end of the list), and the part of the list which has already been sorted is examined pair by pair until the larger number is in its proper position. The sort then resumes (in the original direction, towards the top of the list) at the point in the list where the larger number was found. This bottom-to-top (and reverse order) comparison cycle is repeated until all the numbers in the list are in the right sequence. The execution time is proportional to the number of exchanges required. An example of a bubble sort is shown in Figure 1.

EXAMPLE OF A BUBBLE SORT



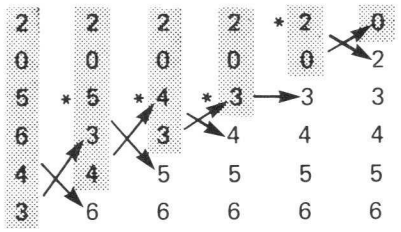
* Indicates places where exchange of numbers occurs.

Figure 1

SEARCH SORT

Another way of sorting is to search the list each time for the number with the largest value and then insert this number in the right place in the list. This sorting technique is known as the "search" sort. First, all the numbers are tested, and the largest-valued number and the last number in the list are exchanged. The list length is then decremented by one, and the list is searched again for the next largest-valued number. This process continues until all the numbers in the list are in the right sequence, as shown in Figure 2.

EXAMPLE OF A SEARCH SORT



NOTE: Shaded areas denote part of list searched during each pass.

* Indicates the largest number in the pass.

Figure 2

The execution time of a search sort is proportional to the list length. On very unsorted lists, the search sort requires less time to execute than the bubble sort.

LINEAR SORT

A "linear" sort consists of several passes. In each pass, the list is examined from the bottom upwards. Two numbers are compared at a time, and the number with the larger value is placed at the bottom of the pair. Any exchange of numbers sets a flag. When a pass is finished, the program tests the flag and, if the flag is set, it clears the flag and begins a new pass. The sort is completed when the flag is not set at the end of a pass. In some cases, linear sorts can be executed faster than bubble or search sorts. An example of a linear sort is given in Figure 3.

EXAMPLE OF A LINEAR SORT

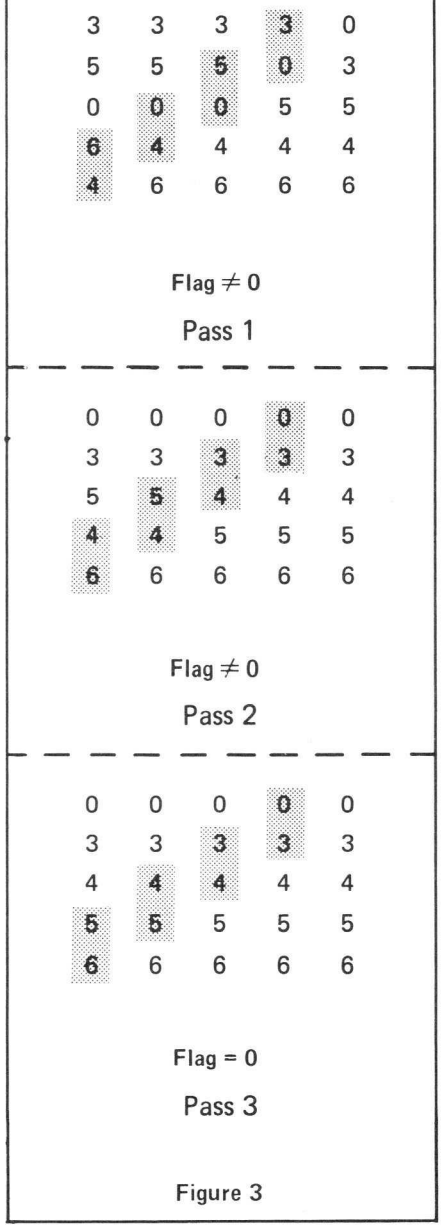


Figure 3

REMARKS FOR SAMPLE PROGRAMS

The sample programs below illustrate the use of the techniques described previously to sort various types of lists. The programs sort the numbers into ascending order by changing the test instructions marked with '#' in the comment column from the Greater Than (GT) to Less Than (LT) and vice versa.

Figure 4 defines the symbols used in all of the example programs. Where multiple-byte numbers are to be sorted, the number of bytes, N, in a number must conform to: $N = 2^n$, where n is an integer.

DEFINITION OF SYMBOLS

```
0001                ********************
0002                *  DEFINITIONS OF SYMBOLS
0003                *   REGISTER EQUATES
0004 0000           R0      EQU     0       REGISTER 0
0005 0001           R1      EQU     1       REGISTER 1
0006 0002           R2      EQU     2       REGISTER 2
0007 0003           R3      EQU     3       REGISTER 3
0008                *  CONDITION CODES
0009 0001           P       EQU     1       POSITIVE RESULT
0010 0000           Z       EQU     0       ZERO RESULT
0011 0002           N       EQU     2       NEGATIVE RESULT
0012 0002           LT      EQU     2       LESS THAN
0013 0000           EQ      EQU     0       EQUAL TO
0014 0001           GT      EQU     1       GREATER THAN
0015 0003           UN      EQU     3       UNCONDITIONAL
0016                *  PSW LOWER EQUATES
0017 0000           CC      EQU     H'00'   CONDITIONAL CODES
0018 0020           IDC     EQU     H'20'   INTERDIGIT CARRY
0019 0010           RS      EQU     H'10'   REGISTER BANK
0020 0008           WC      EQU     H'08'   1=WITH 0=WITHOUT CARRY
0021 0004           OVF     EQU     H'04'   OVERFLOW
0022 0002           COM     EQU     H'02'   1=LOGIC 0=ARITHMETIC COMPARE
0023 0001           C       EQU     H'01'   CARRY/BORROW
0024                *  PSW UPPER EQUATES
0025 0080           SENS    EQU     H'80'   SENSE BIT
0026 0040           FLAG    EQU     H'40'   FLAG BIT
0027 0020           II      EQU     H'20'   INTERRUPT INHIBIT
0028 0007           SP      EQU     H'07'   STACK POINTER
0029                *  END OF EQUATES
```

FIGURE 4

FLOW CHART FOR FIXED-LIST BUBBLE SORT

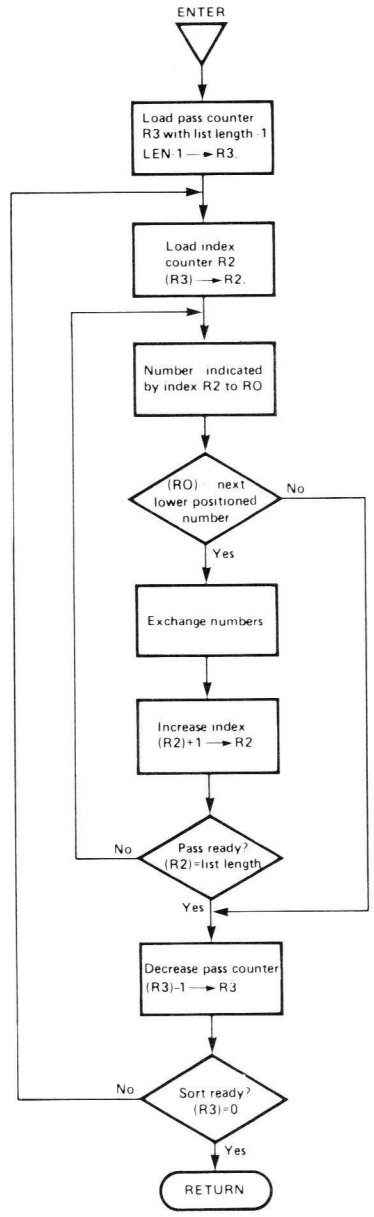


Figure 5

Program Title

BUBBLE SORT FOR A FIXED LIST

Function

This program sorts single-byte numbers (signed or unsigned) into their incrementing order. The bytes are held in a list with a fixed address and a fixed length. The maximum list length is 256 bytes.

Parameters

Input:

Unsorted list.
The compare flag indicates if the numbers are signed or unsigned.
COM=1 means unsigned numbers.
COM=0 means signed numbers.

Output:

Sorted list.

Refer to Figures 5 and 6 for flowchart and program listing.

HARDWARE AFFECTED							
REGISTERS	R0 X	R1 X	R2 X	R3 X	R1'	R2'	R3'
PSU	F	II	SP				
PSL	CC X	IDC	RS	WC	OVF	COM	C

RAM REQUIRED (BYTES):	NONE
ROM REQUIRED (BYTES):	32
EXECUTION TIME:	VARIABLE
MAXIMUM SUBROUTINE NESTING LEVELS:	NONE
ASSEMBLER/COMPILER USED:	TWIN VER 1.0

PROGRAM LISTING FOR FIXED-LIST BUBBLE SORT

```
TWIN ASSEMBLER VER 1.0                                          PAGE 0002

LINE ADDR  OBJECT  E SOURCE

0021                *
0022                ****************************************
0023                *     P0760060                         *
0024                ****************************************
0025                *     BUBBLE SORT FOR FIXED LIST       *
0026                ****************************************
0027                * THIS PROGRAM SORTS A LIST OF SINGLE-BYTE NUMBERS
0028                * INTO THEIR INCREMENTING ORDER.
0029                * THE LIST HAS A FIXED LENGTH AND A FIXED ADDRESS.
0040                * THE MAXIMUM LIST LENGTH IS 256 BYTES.
0041                * UPON ENTRY TO THIS SUBROUTINE, THE COMPARE FLAG
0042                * INDICATES IF THE NUMBERS TO BE SORTED
0043                * ARE SIGNED OR UNSIGNED:
0044                *    COM=1 MEANS UNSIGNED NUMBERS.
0045                *    COM=0 MEANS SIGNED NUMBERS.
0046                *
0047 0000                ORG     H'500'  SORTING SUBROUTINE
0048 0500 0707      SORT  LODI,R3 LEN-1   LOAD PASS COUNTER R3
0049 0502 02        PASS  LODZ    R3      LOAD INDEX R2
0050 0503 C2              STRZ    R2
0051 0504 0E6600    LOOP  LODA,R0 LIST,R2 LOAD FIRST NUMBER IN R0
0052 0507 EE65FF          COMA,R0 LIST-1,R2       COMPARE WITH SECOND NUMBER
0053 050A 9A11            BCFR,LT LOC     # BRANCH IF THE NUMBERS ARE IN
0054                *                       THE RIGHT SEQUENCE
0055 050C C1              STRZ    R1      EXCHANGE THE TWO NUMBERS
0056 050D 0E65FF          LODA,R0 LIST-1,R2
0057 0510 CE6600          STRA,R0 LIST,R2
0058 0512 01              LODZ    R1
0059 0514 CE65FF          STRA,R0 LIST-1,R2
0060 0517 D900            BIRR,R2 $+2     INCREMENT INDEX
0061 0519 E6C8            COMI,R2 LEN     COMPARE INDEX WITH LENGTH
0062 051B 9867            BCFR,EQ LOOP    BRANCH IF PASS NOT READY
0063 051D FB63      LOC   BDRR,R3 PASS    BRANCH IF SORT NOT READY
0064 051F 17              RETC,UN         RETURN TO MAIN PROGRAM
0065                *
0066                ********************
0067                *  SORTING LIST    *
0068                ********************
0069 0520                ORG     H'600'  LIST
0070 00C8           LEN   EQU     200     LENGTH OF THE LIST
0071 0600           LIST  RES     LEN     ADDRESS OF THE LIST
0072                *
0073 0500                END     SORT

TOTAL ASSEMBLY ERRORS = 0000
```

Figure 6

Program Title

SEARCH SORT FOR A FIXED LIST

Function

This program sorts single-byte numbers (signed or unsigned) into their incrementing order. The bytes are held in the list with a fixed address and fixed length. The maximum list length is 256 bytes.

Parameters

Input:

Unsorted list.
The compare flag indicates if the numbers are signed or unsigned.
COM=1 means unsigned numbers.
COM=0 means signed numbers.

Output:

Sorted list.

Refer to Figures 7 and 8 for flowchart and program listing.

FLOW CHART FOR A FIXED-LIST SEARCH SORT

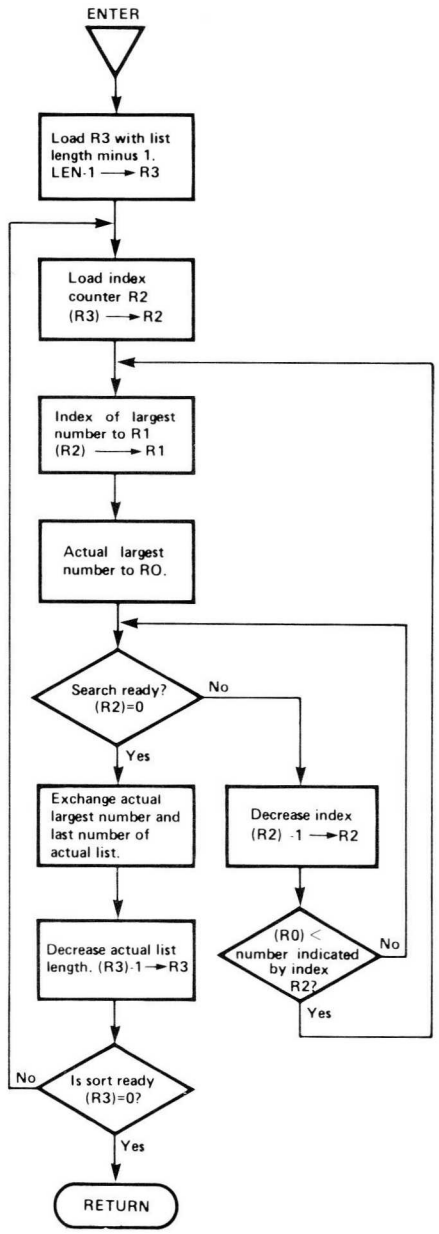


Figure 7

HARDWARE AFFECTED							
REGISTERS	R0 X	R1 X	R2 X	R3 X	R1'	R2'	R3'
PSU	F	II	SP				
PSL	CC X	IDC	RS	WC	OVF	COM	C

RAM REQUIRED (BYTES): NONE
ROM REQUIRED (BYTES): 32
EXECUTION TIME: VARIABLE
MAXIMUM SUBROUTINE NESTING LEVELS: NONE
ASSEMBLER/COMPILER USED: TWIN VER 1.0

PROGRAM LISTING FOR FIXED-LIST SEARCH SORT

```
TWIN ASSEMBLER VER 1.0                                        PAGE 0002

LINE ADDR  OBJECT  E SOURCE

0031                *
0032                ************************************
0033                *  PD760061                        *
0034                ************************************
0035                *  SEARCH SORT FOR A FIXED LIST    *
0036                ************************************
0037                * THIS PROGRAM SORTS A LIST OF SINGLE-BYTE NUMBERS
0038                * INTO THEIR INCREMENTING ORDER.
0039                * THE LIST HAS A FIXED LENGTH AND A FIXED ADDRESS.
0040                * THE MAXIMUM LIST LENGTH IS 256 BYTES.
0041                * UPON ENTRY TO THIS SUBROUTINE, THE COMPARE FLAG
0042                * INDICATES IF THE NUMBERS TO BE SORTED
0043                * ARE SIGNED OR UNSIGNED:
0044                *   COM=1 MEANS UNSIGNED NUMBERS
0045                *   COM=0 MEANS SIGNED NUMBERS
0046                *
0047 0000               ORG     H'500'    SORTING SUBROUTINE
0048 0500 07C7     SORT LODI,R3 LEN-1     LOAD ACTUAL LIST LENGTH IN R3
0049 0502 03       PASS LODZ    R3        LOAD INDEX R2
0050 0503 C2            STRZ    R2
0051 0504 02       SLEC LODZ    R2        INDEX OF LARGEST NUMBER TO R1
0052 0505 C1            STRZ    R1
0053 0506 0D6600        LODA,R0 LIST,R1   LOAD PRESENT LARGEST NUMBER IN R0
0054 0509 5A0E     LOOP BRNR,R2 COMP      BRANCH IF PASS NOT READY
0055 050B C2            STRZ    R2        EXCHANGE LARGEST NUMBER WITH
0056 050C 0F6600        LODA,R0 LIST,R3     LAST NUMBER IN ACTUAL LIST
0057 050F CD6600        STRA,R0 LIST,R1
0058 0512 02            LODZ    R2
0059 0513 CF6600        STRA,R0 LIST,R3
0060 0516 FB6A          BDRR,R3 PASS      DECREASE ACTUAL LIST LENGTH,
0061                *                       BRANCH TO NEXT PASS IF
0062                *                         LENGTH NOT ZERO
0063 0518 17            RETC,UN           RETURN TO MAIN PROGRAM
0064 0519 EE4600   COMP COMA,R0 LIST,R2,- COMPARE NUMBER WITH PRESENT
0065                *                       LARGEST NUMBER OF LIST
0066 051C 9A6B          BCFR,LT LOOP      # BRANCH FOR NEXT NUMBER
0067 051E 1B64          BCTR,UN SLEC      BRANCH IF NEW NUMBER IS LARGER
0068                *
0069                *****************
0070                *  SORTING LIST *
0071                *****************
0072 0520               ORG     H'600'    LIST
0073 00C8          LEN  EQU     200       LENGTH OF THE LIST
0074 0600          LIST RES     LEN       ADDRESS OF THE LIST
0075 0500               END     SORT

 TOTAL ASSEMBLY ERRORS = 0000
```

Figure 8

Program Title

BUBBLE SORT FOR A VARIABLE-LENGTH LIST

Function

This program sorts a list of single-byte numbers into their incrementing order. The maximum list length is 256 bytes.

Parameters

Input:

Unsorted list.
R1 contains the high-order address of the list.
R2 contains the low-order address.
R3 contains the list length minus 1.
The compare flag indicates if the numbers are signed or unsigned.
COM=1 means unsigned numbers.
COM=0 means signed numbers.

Output:

Sorted list.

Refer to Figures 9 and 10 for flowchart and program listing.

FLOWCHART FOR A VARIABLE LENGTH LIST BUBBLE SORT

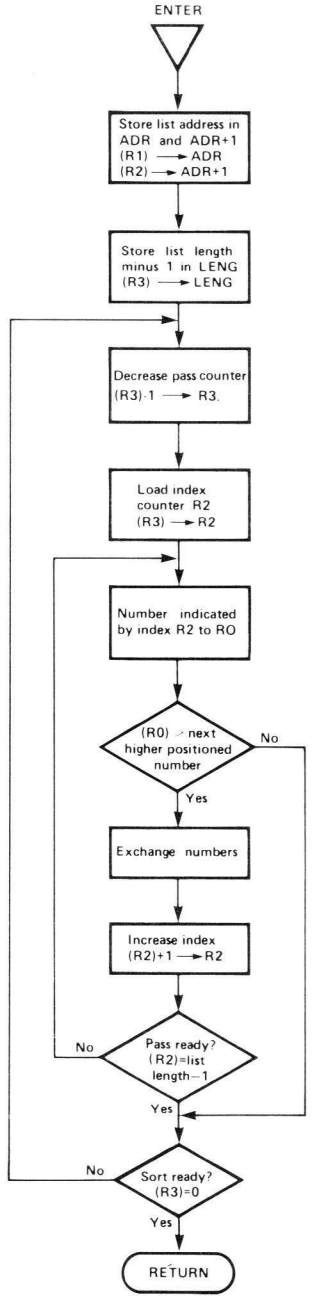


Figure 9

HARDWARE AFFECTED							
REGISTERS	R0 X	R1 X	R2 X	R3 X	R1'	R2'	R3'
PSU	F	II	SP				
PSL	CC X	IDC	RS	WC	OVF	COM	C

RAM REQUIRED (BYTES):	3
ROM REQUIRED (BYTES):	40
EXECUTION TIME:	VARIABLE
MAXIMUM SUBROUTINE NESTING LEVELS:	NONE
ASSEMBLER/COMPILER USED:	TWIN VER 1.0

PROGRAM LISTING FOR A VARIABLE LIST BUBBLE SORT

```
TWIN ASSEMBLER VER 1.0                                        PAGE 0002
LINE ADDR  OBJECT  E SOURCE

0031                *
0032                **********************************
0033                *    PD76005Z                    *
0034                **********************************
0035                * BUBBLE SORT FOR VARIABLE LIST   *
0036                **********************************
0037                * THIS PROGRAM SORTS A LIST OF SINGLE-BYTE NUMBERS
0038                * INTO THEIR INCREMENTING ORDER.
0039                * THE ADDRESS AND THE LENGTH OF THE LIST MUST BE
0040                * DEFINED IN THE MAIN PROGRAM.  THE MAXIMUM LIST LENGTH
0041                * IS 256 BYTES.
0042                * UPON ENTRY TO THIS SUBROUTINE, THE COMPARE FLAG
0043                * INDICATES IF THE NUMBERS TO BE SORTED
0044                * ARE SIGNED OR UNSIGNED:
0045                *   COM=1 MEANS UNSIGNED NUMBERS.
0046                *   COM=0 MEANS SIGNED NUMBERS.
0047                *
0048 0000                ORG     H'4F0'
0049 04F0          ADR   RES     2          ADDRESS OF LIST
0050 04F2          LENG  RES     1          LIST LENGTH MINUS 1
0051                *
0052 04F3                ORG     H'500'     SORTING SUBROUTINE
0053 0500 CD04F0   SORT  STRA,R1 ADR        STORE HIGH ORDER ADDRESS OF
0054                *                          THE LIST
0055 0503 CE04F1         STRA,R2 ADR+1      STORE LOW ORDER ADDRESS
0056 0506 CF04F2         STRA,R3 LENG       STORE LIST LENGTH MINUS 1
0057 0509 FB00     PASS  BDRR,R3 $+2        DECREMENT PASS COUNTER
0058 050B 03             LODZ    R3         LOAD INDEX
0059 050C C2             STRZ    R2
0060 050D 0EE4F0   LOOP  LODA,R0 *ADR,R2    FETCH FIRST NUMBER
0061 0510 EEA4F0         COMA,R0 *ADR,R2,+  COMPARE WITH SECOND NUMBER
0062 0513 9910           BCFR,GT LOC        # BRANCH IF THE NUMBERS ARE IN
0063                *                          THE RIGHT SEQUENCE
0064 0515 C1       EXCH  STRZ    R1         EXCHANGE THE TWO NUMBERS
0065 0516 0EE4F0         LODA,R0 *ADR,R2
0066 0519 CEC4F0         STRA,R0 *ADR,R2,-
0067 051C 01             LODZ    R1
0068 051D CEA4F0         STRA,R0 *ADR,R2,+
0069 0520 EE04F2         COMA,R2 LENG       COMPARE (R2) WITH LENGTH
0070 0523 9968           BCFR,EQ LOOP       BRANCH IF PASS NOT READY
0071 0525 5B62     LOC   BRNR,R3 PASS       BRANCH IF SORT NOT READY
0072 0527 17             RETC,UN            RETURN TO MAIN PROGRAM
0073                *
0074 0500                END     SORT

 TOTAL ASSEMBLY ERRORS = 0000
```

Figure 10

Program Title

SEARCH SORT FOR A VARIABLE-LENGTH LIST

Function

This program sorts a list of single-byte numbers into their incrementing order. The maximum list length is 256 bytes.

Parameters

Input:

Unsorted list.
R1 contains the high-order address of the list.
R2 contains the low-order address.
R3 contains the list length minus 1.
The compare flag indicates if the numbers are signed or unsigned.
COM=1 means unsigned numbers.
COM=2 means signed numbers.

Output:

Sorted list.

Refer to Figures 11 and 12 for flowchart and program listing.

FLOW CHART FOR VARIABLE LENGTH LIST SEARCH SORT

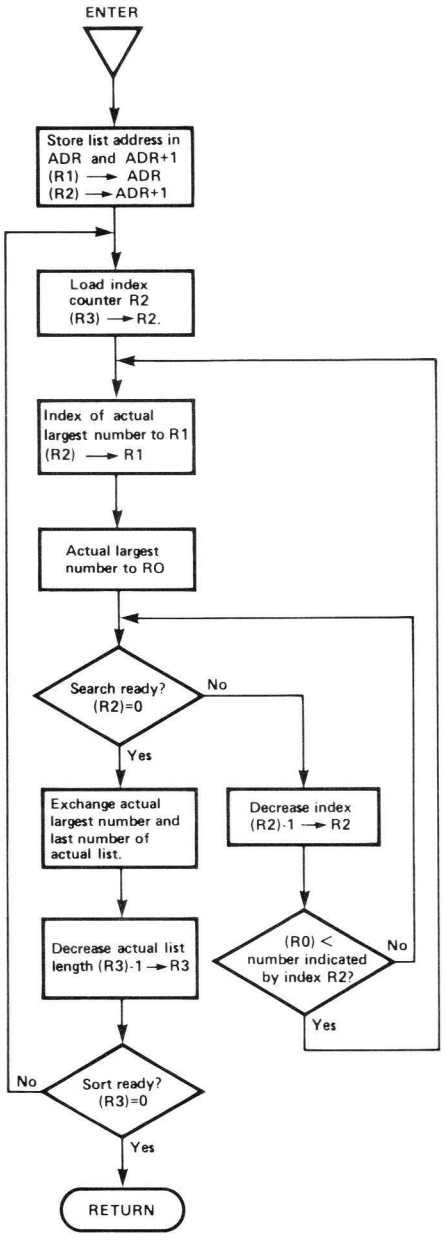


Figure 11

HARDWARE AFFECTED							
REGISTERS	R0 X	R1 X	R2 X	R3 X	R1'	R2'	R3'
PSU	F	II	SP				
PSL	CC X	IDC	RS	WC	OVF	COM	C

RAM REQUIRED (BYTES):	2
ROM REQUIRED (BYTES):	36
EXECUTION TIME:	VARIABLE
MAXIMUM SUBROUTINE NESTING LEVELS:	NONE
ASSEMBLER/COMPILER USED:	TWIN VER 1.0

PROGRAM LISTING FOR A VARIABLE LIST SEARCH SORT

```
TWIN ASSEMBLER VER 1.0                                    PAGE 0002

LINE ADDR  OBJECT  E SOURCE

0031               *
0032               ************************************
0033               *   PD760063                       *
0034               ************************************
0035               * SEARCH SORT FOR VARIABLE LIST    *
0036               ************************************
0037               * THIS PROGRAM SORTS A LIST OF SINGLE-BYTE NUMBERS
0038               * INTO THEIR INCREMENTING ORDER.
0039               * THE ADDRESS AND THE LENGTH OF THE LIST MUST BE
0040               * DEFINED IN THE MAIN PROGRAM.  THE MAXIMUM LIST LENGTH
0041               * IS 256 BYTES.
0042               * UPON ENTRY TO THIS SUBROUTINE, THE COMPARE FLAG
0043               * INDICATES IF THE NUMBERS TO BE SORTED
0044               * ARE SIGNED OR UNSIGNED:
0045               *   COM=1 MEANS UNSIGNED NUMBERS.
0046               *   COM=0 MEANS SIGNED NUMBERS.
0047               *
0048 0000                 ORG     H'4F0'
0049 04F0          ADR    RES     2          ADDRESS OF LIST
0050               *
0051 04F2                 ORG     H'500'     SORTING SUBROUTINE
0052 0500 CD04F0   SORT   STRA,R1 ADR        STORE HIGH ORDER ADDRESS OF
0053               *                           THE LIST
0054 0503 CE04F1          STRA,R2 ADR+1      STORE LOW ORDER ADDRESS
0055 0506 03       PASS   LODZ    R3         LOAD INDEX R2
0056 0507 C2              STRZ    R2
0057 0508 02       MAXN   LODZ    R2         INDEX OF LARGEST NUMBER TO R1
0058 0509 C1              STRZ    R1
0059 050A 0DE4F0          LODA,R0 *ADR,R1    LOAD PRESENT LARGEST NUMBER IN R0
0060 050D 5A0E     SRCH   BRNR,R2 COMP       BRANCH IF PASS NOT READY
0061 050F C2              STRZ    R2         EXCHANGE LARGEST NUMBER WITH
0062 0510 0FE4F0          LODA,R0 *ADR,R3      LAST NUMBER IN ACTUAL LIST
0063 0513 CDE4F0          STRA,R0 *ADR,R1
0064 0516 02              LODZ    R2
0065 0517 CFE4F0          STRA,R0 *ADR,R3
0066 051A FB6A            BDRR,R3 PASS       DECREASE ACTUAL LIST LENGTH,
0067               *                           BRANCH TO NEXT PASS IF
0068               *                             LENGTH NOT ZERO
0069 051C 17              RETC,UN            RETURN TO MAIN PROGRAM
0070 051D EEC4F0   COMP   COMA,R0 *ADR,R2,- COMPARE NUMBER WITH PRESENT
0071               *                             LARGEST NUMBER OF LIST
0072 0520 9A6B            BCFR,LT SRCH       # BRANCH FOR NEXT NUMBER
0073 0522 1B64            BCTR,UN MAXN       BRANCH IF NEW NUMBER IS LARGER
0074               *
0075 0500                 END     SORT

 TOTAL ASSEMBLY ERRORS = 0000
```

Figure 12

Program Title

LINEAR SORT SUBROUTINE

Function

This program sorts multiple-byte numbers (signed or unsigned) into their incrementing order. In this example, the list contains 64 four-byte numbers. The list has a fixed starting address and a fixed length. The maximum list length is 256 bytes.

Parameters

Input:

Unsorted list.
The SIGN flag indicates if the numbers are signed or unsigned.
SIGN = 0 means signed numbers.
SIGN = 1 means unsigned numbers.

Output:

Sorted list.

Refer to Figures 13 and 14 for flowchart and program listing.

FLOW CHART FOR LINEAR SORT FOR MULTIPLE-BYTE NUMBERS

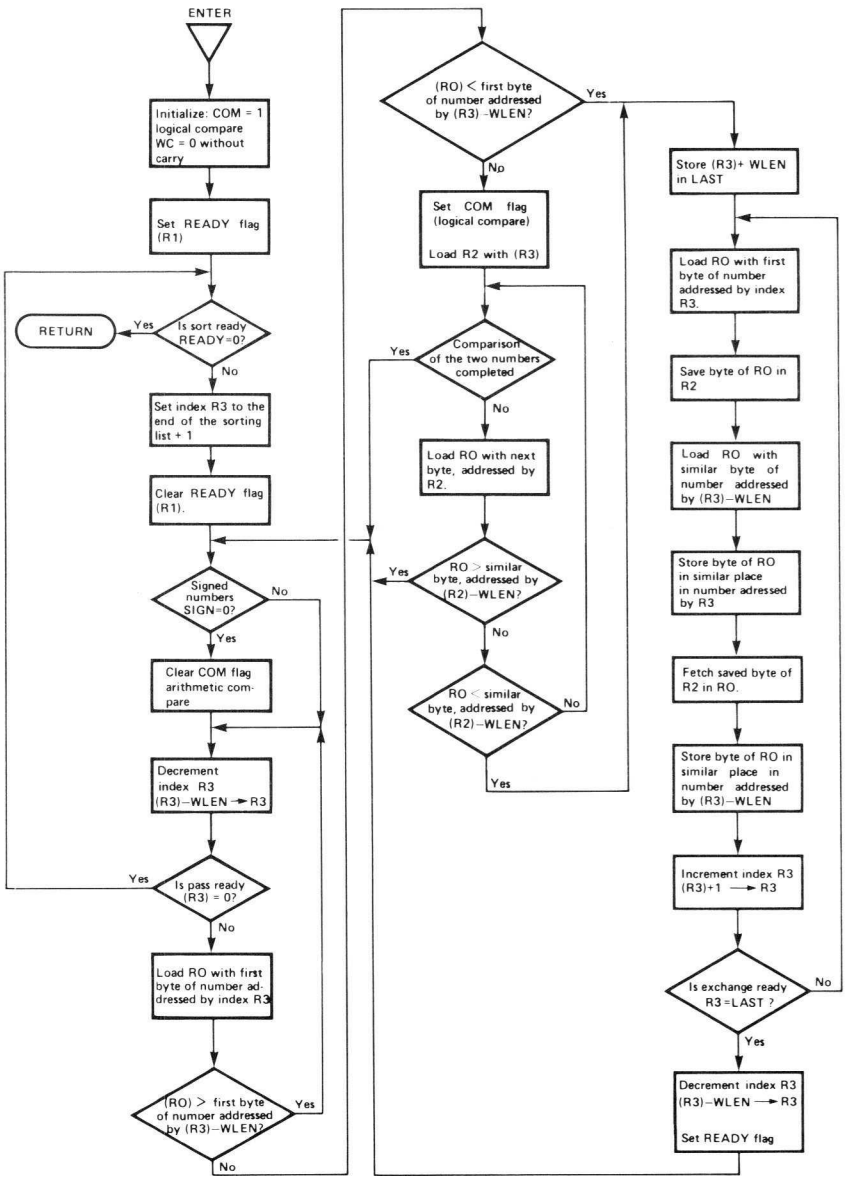


Figure 13

HARDWARE AFFECTED							
REGISTERS	R0 X	R1 X	R2 X	R3 X	R1'	R2'	R3'
PSU	F	II	SP				
PSL	CC X	IDC X	RS	WC X	OVF X	COM X	C X

RAM REQUIRED (BYTES): 2
ROM REQUIRED (BYTES): 89
EXECUTION TIME: VARIABLE
MAXIMUM SUBROUTINE NESTING LEVELS: NONE
ASSEMBLER/COMPILER USED: TWIN VER 1.0

PROGRAM LISTING FOR LINEAR SORT FOR MULTIPLE-BYTE NUMBERS

```
TWIN ASSEMBLER VER 1.0                                    PAGE 0002

LINE ADDR  OBJECT  E SOURCE

0031                *
0032                ******************************
0033                *  PD760064                  *
0034                ******************************
0035                * LINEAR SORTING SUBROUTINE  *
0036                ******************************
0037                * THIS PROGRAM SORTS A LIST OF MULTIPLE-BYTE NUMBERS
0038                * INTO INCREMENTING ORDER.  THE MAXIMUM
0039                * NUMBER OF BYTES OF THE LIST TO BE SORTED IS
0040                * 256.  THE START ADDRESS OF THE LIST IS H'600'.
0041                * THE NUMBER OF BYTES IN EACH NUMBER IS VARIABLE,
0042                * BUT IT MUST BE A POWER OF TWO.  IN THIS CASE
0043                * THE LIST CONSISTS OF 64 NUMBERS OF 4 BYTES EACH.
0044                * UPON ENTRY TO THIS ROUTINE, THE SIGN FLAG INDI-
0045                * CATES IF THE NUMBERS TO BE SORTED ARE
0046                * SIGNED OR UNSIGNED: SIGN=0 MEANS SIGNED NUMBERS
0047                *             SIGN=NOT 0 MEANS UNSIGNED NUMBERS.
0048                * THE ORDER OF SORTING CAN BE CHANGED FROM AN
0049                * INCREMENTING ORDER TO A DECREMENTING ORDER BY
0050                * CHANGING THE INSTRUCTIONS MARKED WITH
0051                * A #.  THE GREATER THAN (GT) TESTS  MUST BE
0052                * CHANGED TO LESS THAN (LT) TESTS  AND VICE VERSA.
0053                *
0054 0000                ORG     H'4FE'
0055 04FE           LAST RES     1          MEMORY LOCATION WHICH SAVES INDEX
0056                *                        OF NUMBER WHICH FOLLOWS NUMBER
0057                *                        ADDRESSED BY R3
0058 04FF           SIGN RES     1          SIGN FLAG: SIGN=0 SIGNED NUMBER
0059                *                          SIGN= NOT 0 UNSIGNED NUMBER
0060                *
0061 0500                ORG     H'600'
0062 0100           LEN  EQU     H'100'     LENGTH OF SORTING LIST
0063 0600           LIST RES     LEN        SORTING LIST
0064 0004           WLEN EQU     4          WORD LENGTH IN BYTES
0065                *
0066                *
0067 0700                ORG     H'500'
0068 0500 7702      SORT PPSL    COM        LOGICAL COMPARE
0069 0502 7508           CPSL    WC         WITHOUT CARRY
0070 0504 05FF           LODI,R1 H'FF'      SET READY FLAG
0071 0506 01        PASS LODZ    R1         TEST AND RETURN IF READY FLAG
0072 0507 14             RETC,Z             IS NOT SET
0073 0508 0700           LODI,R3 >LEN       LOAD INDEX R3
0074 050A 0500           LODI,R1 00         CLEAR READY FLAG
0075 050C 0C04FF    COMP LODA,R0 SIGN       TEST SIGN:
0076 050F 9802           BCFR,Z  COM1        SIGN= NOT 0, UNSIGNED NUMBER
0077 0511 7502           CPSL    COM        SIGN=0,SIGNED NUMBER,CLEAR COM
0078 0513 A704      COM1 SUBI,R3 WLEN       DECREMENT INDEX R3
0079 0515 186F           BCTR,Z  PASS       TEST AND BRANCH IF PASS READY
0080 0517 0F6600         LODA,R0 LIST,R3    LOAD R0 WITH FIRST BYTE OF
0081                *                        NUMBER ADDRESSED BY R3
0082 051A EF65FC         COMA,R0 LIST-WLEN,R3 COMPARE WITH FIRST BYTE OF
0083                *                        NEXT NUMBER, ADDRESSED BY
0084                *                        R3-WLEN
0085 051D 1974           BCTR,GT COM1       # IF GT, CONTINUE COMPARING
```

```
TWIN ASSEMBLER VER 1.0                                    PAGE 0003

LINE ADDR  OBJECT  E SOURCE

0086 051F 1A16           BCTR,LT EXCH       # IF LT, EXCHANGE NUMBERS
0087 0521 7702           PPSL    COM        LOGICAL COMPARE
0088 0523 03             LODZ    R3         STORE INDEX R3 IN R2
0089 0524 C2             STRZ    R2
0090 0525 8404      NEXT ADDI,R0 WLEN       TEST AND BRANCH IF COMPARE
0091 0527 E2             COMZ    R2          OF BOTH NUMBERS IS READY
0092 0528 1862           BCTR,EQ COMP
0093 052A 0E2600         LODA,R0 LIST,R2,+  LOAD R0 WITH NEXT BYTE OF
0094                *                        NUMBER, ADDRESSED BY R3
0095 052D EE65FC         COMA,R0 LIST-WLEN,R2 COMPARE WITH SIMILAR BYTE
0096                *                        OF NEXT NUMBER, ADDRESSED BY
0097                *                        R3-WLEN
0098 0530 195A           BCTR,GT COMP       # IF GT, CONTINUE COMPARING
0099                *                        NEXT TWO NUMBERS
0100 0532 1A03           BCTR,LT EXCH       # IF LT, EXCHANGE NUMBERS
0101 0534 03             LODZ    R3
0102 0535 1B6E           BCTR,UN NEXT       CONTINUE COMPARING NEXT
0103                *                        SIMILAR BYTES OF NUMBERS
0104 0537 03        EXCH LODZ    R3         STORE INDEX OF NEXT NUMBER
0105 0538 8404           ADDI,R0 WLEN        IN LAST
0106 053A CC04FE         STRA,R0 LAST
0107 053D 0F6600    EXC1 LODA,R0 LIST,R3    EXCHANGE SIMILAR BYTE OF
0108 0540 C2             STRZ    R2          BOTH NUMBERS
0109 0541 0F65FC         LODA,R0 LIST-WLEN,R3
0110 0544 CF6600         STRA,R0 LIST,R3
0111 0547 02             LODZ    R2
0112 0548 CF65FC         STRA,R0 LIST-WLEN,R3
0113 054B DB00           BIRR,R3 $+2        INCREMENT INDEX R3
0114 054D EF04FE         COMA,R3 LAST       TEST AND BRANCH IF
0115 0550 986B           BCFR,EQ EXC1        EXCHANGE IS NOT READY
0116 0552 A704           SUBI,R3 WLEN       RESET INDEX R3
0117 0554 05FF           LODI,R1 H'FF'      SET READY FLAG
0118 0556 1F050C         BCTA,UN COMP       CONTINUE COMPARING NEXT
0119                *                        TWO NUMBERS
0120                *
0121 0500                END     SORT

 TOTAL ASSEMBLY ERRORS = 0000
```

Figure 14

Program Title

SEARCH SORT SUBROUTINE FOR A FIXED LIST

Function

This program sorts multiple-byte numbers (signed or unsigned) into their incrementing order. In this example, the list contains 64 four-byte numbers. The list has a fixed starting address and a fixed length. The maximum list length is 256 bytes.

Parameters

Input:

Unsorted list.
The SIGN flag indicates if the numbers are signed or unsigned.
SIGN = 0 means signed numbers.
SIGN = 1 means unsigned numbers.

Output:

Sorted list.
Refer to Figures 15 and 16 for flowchart and program listing.

FLOWCHART FOR SEARCH SORT FOR MULTIPLE-BYTE NUMBERS

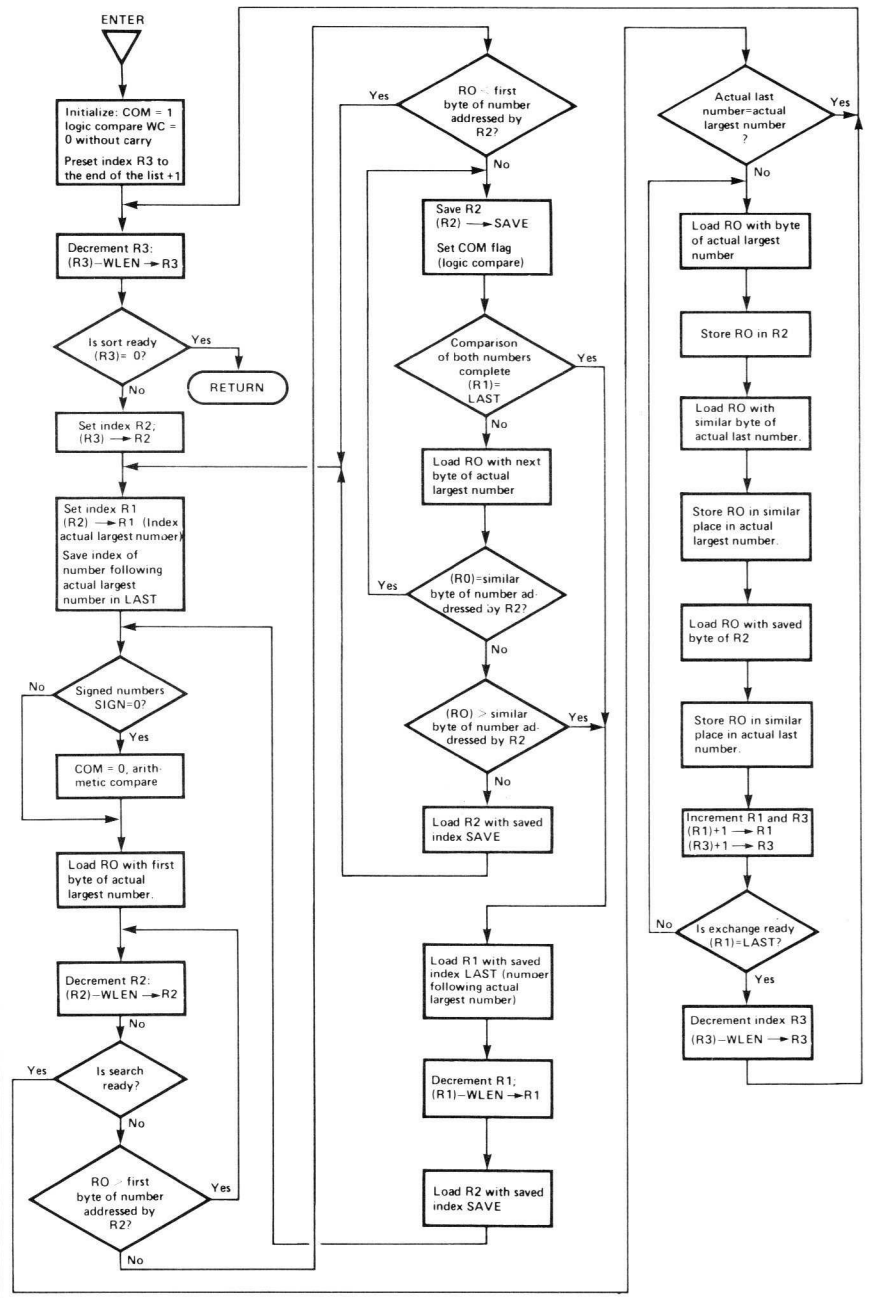


Figure 15

HARDWARE AFFECTED							
REGISTERS	R0 X	R1 X	R2 X	R3 X	R1'	R2'	R3'
PSU	F	II	SP				
PSL	CC X	IDC X	RS	WC X	OVF X	COM X	C X

RAM REQUIRED (BYTES): 3
ROM REQUIRED (BYTES): 106
EXECUTION TIME: VARIABLE
MAXIMUM SUBROUTINE NESTING LEVELS: NONE
ASSEMBLER/COMPILER USED: TWIN VER 1.0

PROGRAM LISTING FOR SEARCH SORT FOR MULTIPLE-BYTE NUMBERS

```
TWIN ASSEMBLER VER 1.0                                    PAGE 0002

LINE ADDR  OBJECT  E SOURCE

0021                *
0022                ******************************
0023                *  PD760065                  *
0024                ******************************
0025                *SEARCH SORTING SUBROUTINE  *
0026                ******************************
0027                *  THIS PROGRAM SORTS A LIST OF MULTIPLE-BYTES NUMBERS
0028                *  INTO THEIR INCREMENTING ORDER.  THE MAXIMUM NUMBER
0029                *  OF BYTES IN THE LIST TO BE SORTED IS 256.
0030                *  THE START ADDRESS OF THE SORTING LIST IS 600.
0031                *  THE NUMBER OF BYTES IN EACH NUMBER IS VARIABLE,
0032                *  BUT IT MUST BE A POWER OF TWO.  IN THIS CASE THE
0033                *  LIST CONSISTS OF 64 NUMBERS OF 4 BYTES EACH.
0034                *  UPON ENTRY TO THIS SUBROUTINE, THE SIGN FLAG
0035                *  INDICATES IF THE NUMBERS, TO BE SORTED
0036                *  ARE SIGNED OR UNSIGNED:
0037                *    SIGN= NOT 0 MEANS UNSIGNED NUMBERS.
0038                *    SIGN = 0 MEANS SIGNED NUMBERS.
0039                *  THE ORDER OF SORTING CAN BE CHANGED  FROM AN
0040                *  INCREMENTING TO A DECREMENTING ORDER BY CHANGING
0041                *  THE INSTRUCTIONS  MARKED WITH A #.
0042                *  THE GREATER THAN (GT) TESTS MUST BE
0043                *  CHANGED TO LESS THAN (LT) TESTS AND VICE VERSA.
0044                *
0045 0000                ORG     H'4F0'
0046 04F0           SIGN PES     1          SIGN FLAG: SIGN=0 SIGNED NUMBER
0047                *                           SIGN= NOT 0 UNSIGNED NUMBER
0048 04F1           SAVE RES     1          MEMORY LOCATION TO SAVE INDEX R2
0049 04F2           LAST RES     1          INDEX OF NUMBER WHICH FOLLOWS
0050                *                         LAST NUMBER OF ACTUAL LIST
0051 0004           WLEN EQU     4          WORD LENGTH IN BYTES
0052                *
0053 04F2                ORG     H'600'     LIST ADDRESS
0054 0100           LEN  EQU     256        LIST LENGTH
0055 0600           LIST RES     LEN
0056                *
0057                * SORTING SUBROUTINE
0058 0700                ORG     H'500'
0059 0500 7702      SORT PPSL    COM        LOGICAL COMPARE
0060 0502 7508           CPSL    WC         WITHOUT CARRY
0061 0504 0700           LODI,R3 >LEN       LOAD INDEX R3
0062 0506 A704      PASS SUBI,R3 WLEN       DECREMENT INDEX R2 TO LAST NUMBER
0063                *                        OF ACTUAL LIST
0064 0508 14             RETC,Z             RETURN IF SORT READY
0065 0509 03        SKIP LODZ    R3         LOAD INDEX COUNTER R2
0066 050A C2             STRZ    R2
0067 050B 02        MAXM LODZ    R2         SET INDEX R1 AT ACTUAL
0068 050C C1             STRZ    R1          LARGEST NUMBER OF ACTUAL LIST
0069 050D 8404           ADDI,R0 WLEN
0070 050F CC04F2         STRA,R0 LAST       SAVE INDEX OF NUMBER WHICH
0071                *                        FOLLOWS LARGEST NUMBER OF
0072                *                        ACTUAL LIST
0073 0512 0C04F0    LOAD LODA,R0 SIGN       IF SIGN IS 0, CLEAR COMPARE
0074 0515 9802           BCFR,Z  LOA1        SIGNED NUMBERS
0075 0517 7502           CPSL    COM        ELSE BRANCH, UNSIGNED NUMBER
```

```
TWIN ASSEMBLER VER 1.0                                    PAGE 0003

LINE ADDR  OBJECT  E SOURCE

0076 0519 0D6600    LOA1 LODA,R0 LIST,R1    FETCH FIRST BYTE OF ACTUAL
0077                *                        LARGEST NUMBER
0078 051C A604      COMP SUBI,R2 WLEN       DECREMENT INDEX R2
0079 051E EEFC           COMI,R2 >LIST-WLEN TEST AND BRANCH IF SEARCH
0080 0520 182A           BCTR,EQ EXCH        IS READY
0081 0522 EE6600         COMA,R0 LIST,R2    COMPARE R0 WITH FIRST BYTE
0082                *                        OF NUMBER ADDRESSED BY R2
0083 0525 1975           BCTR,GT COMP       # IF GT, THEN NUMBER ADDRESSED
0084                *                        BY R1 IS STILL LARGEST NUMBER
0085 0527 1A62           BCTR,LT MAXM       # IF LT, THEN NEW ACTUAL LAR-
0086                *                        GEST NUMBER IS FOUND.
0087                *                       ELSE COMPARE NEXT BYTES
0088                *                        OF BOTH NUMBERS
0089 0529 CE04F1         STRA,R2 SAVE       SAVE INDEX R2
0090 052C 7702           PPSL    COM        LOGICAL COMPARE
0091 052E ED04F2    NEXT COMA,R1 LAST       TEST AND BRANCH IF COMPARE OF
0092 0531 180F           BCTR,EQ RSET        FOLLOWING BYTES IS READY
0093 0533 0D2600         LODA,R0 LIST,R1,+ COMPARE FOLLOWING BYTES
0094 0536 EE2600         COMA,R0 LIST,R2,+  OF BOTH NUMBERS
0095 0539 1872           BCTR,EQ NEXT       BYTES EQUAL, THEN CONTINUE
0096 053B 1905           BCTR,GT RSET       # IF GT NUMBER ADDRESSED BY R1
0097                *                        IS STILL ACTUAL LARGEST NUMBER
0098                *                       ELSE NEW ACTUAL LARGEST NUMBER
0099                *                        IS FOUND
0100 053D 0E04F1         LODA,R2 SAVE       FETCH SAVED INDEX
0101 0540 1B49           BCTR,UN MAXM       NEW ACTUAL LARGEST NUMBER,BRANCH
0102 0542 0D04F2    RSET LODA,R1 LAST       RESET INDEX LARGEST NUMBER
0103 0545 A504           SUBI,R1 WLEN
0104 0547 0E04F1         LODA,R2 SAVE       FETCH SAVED INDEX R2
0105 054A 1B46           BCTR,UN LOAD
0106 054C 02        EXCH LODZ    R3         TEST IF LARGEST NUMBER IS THE
0107 054D E1             COMZ    R1         SAME AS THE LAST NUMB OF THE
0108 054E 1817           BCTR,EQ BRCH       ACTUAL LIST
0109 0550 0D25FF    EXC2 LODA,R0 LIST-1,R1,+ EXCHANGE THE LAST NUMBER
0110 0553 C2             STRZ    R2         OF THE ACTUAL LIST AND THE
0111 0554 0F6600         LODA,R0 LIST,R3    ACTUAL LARGEST NUMBER OF
0112 0557 CD5FF          STRA,R0 LIST-1,R1  THE LIST
0113 055A 02             LODZ    R2
0114 055B CF6600         STRA,R0 LIST,R3
0115 055E D800           BIRR,R3 $+2
0116 0560 ED04F2         COMA,R1 LAST       TEST AND BRANCH IF EXCHANGE
0117 0562 986B           BCFR,EQ EXC2        IS NOT READY
0118 0565 A704           SUBI,R3 WLEN      RESET INDEX R2
0119 0567 1F0506    BRCH BCTA,UN PASS       NEXT PASS
0120                *
0121 0500                END     SORT

 TOTAL ASSEMBLY ERRORS = 0000
```

Figure 16

Program Title

SEARCH SORT SUBROUTINE FOR A FIXED LIST

Function

This program sorts multiple-byte numbers (signed or unsigned) into their incrementing order. The list to be sorted may contain more than 256 bytes. In this case, the list contains 256 eight-byte numbers. The list has a fixed starting address and length.

Parameters

Input:

Unsorted list.
The SIGN flag indicates if the numbers are signed or unsigned.
SIGN = 0 means signed numbers.
SIGN = 1 means unsigned numbers.

Output:

Sorted list.

Refer to Figures 17 and 18 for flowchart and program listing.

FLOW CHART FOR SEARCH SORT SUBROUTINE (256 EIGHT-BYTE NUMBERS)

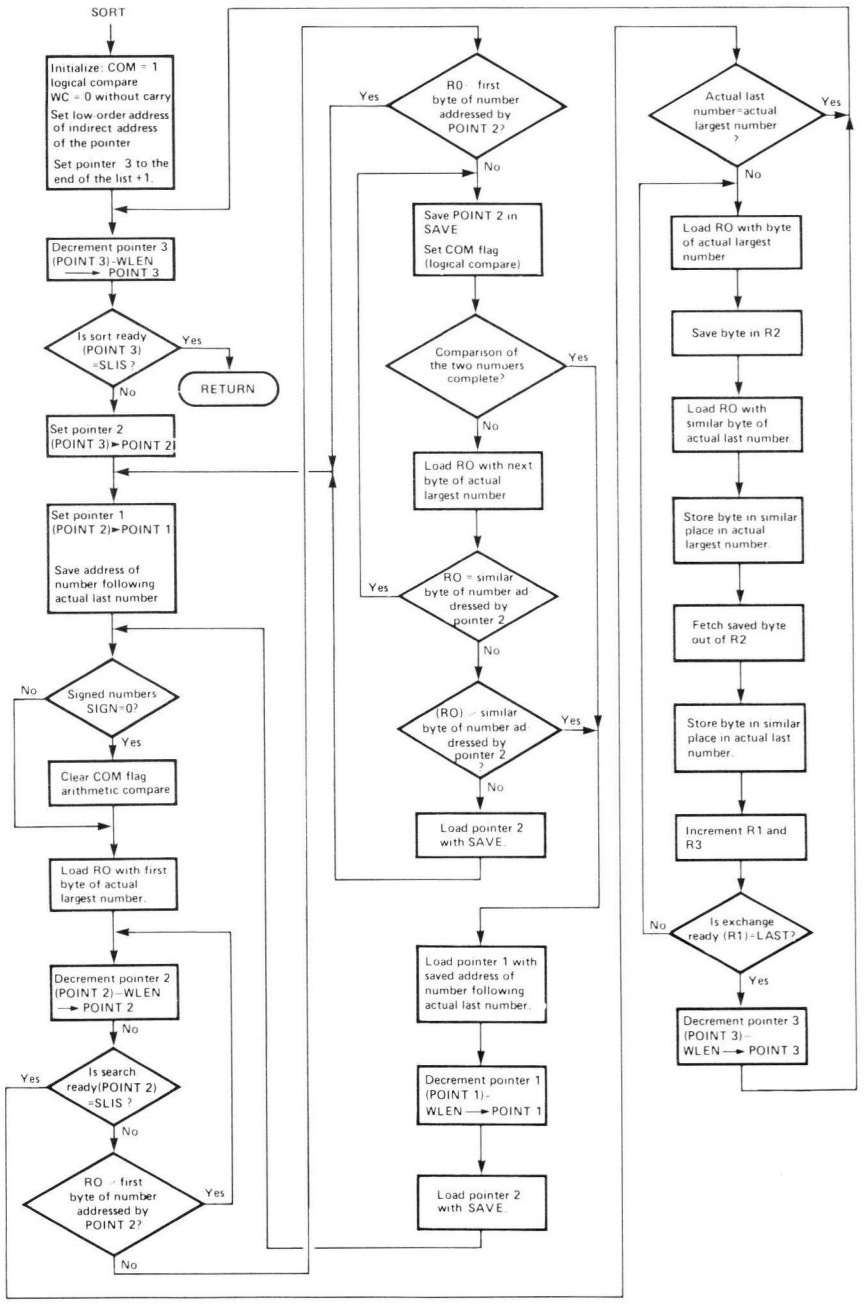


Figure 17

HARDWARE AFFECTED							
REGISTERS	R0 X	R1 X	R2 X	R3 X	R1'	R2'	R3'
PSU	F	II	SP				
PSL	CC X	IDC X	RS	WC X	OVF X	COM X	C X

RAM REQUIRED (BYTES): 9
ROM REQUIRED (BYTES): 167
EXECUTION TIME: VARIABLE
MAXIMUM SUBROUTINE NESTING LEVELS: NONE
ASSEMBLER/COMPILER USED: TWIN VER 1.0

PROGRAM LISTING FOR SEARCH SORT SUBROUTINE (256 EIGHT-BYTE NUMBERS)

```
TWIN ASSEMBLER VER 1.0                                    PAGE 0002

LINE ADDR  OBJECT  E SOURCE

0031                ******************************
0032                *   PD760066                 *
0033                ******************************
0034                * SEARCH SORTING SUBROUTINE  *
0035                ******************************
0036                * THIS PROGRAM SORTS A LIST OF MULTIPLE-BYTE NUMBERS
0037                * INTO THEIR INCREMENTING ORDER.  THE
0038                * START ADDRESS OF THE LIST IS H'500'.  THE NUMBER OF
0039                * BYTES IN EACH NUMBER IS VARIABLE, BUT IT MUST BE A
0040                * POWER OF TWO.  IN THIS CASE THE LIST CONSISTS OF
0041                * 256 EIGHT-BYTE NUMBERS.  UPON ENTRY TO THIS
0042                * SUBROUTINE, THE SIGN FLAG INDICATES IF THE NUMBERS
0043                * ARE SIGNED OR UNSIGNED:
0044                *           SIGN = NOT 0 MEANS UNSIGNED NUMBERS.
0045                *           SIGN = 0 MEANS SIGNED NUMBERS.
0046                * THE ORDER OF SORTING CAN BE CHANGED FROM AN INCRE-
0047                * MENTING TO A DECREMENTING ORDER BY CHANGING
0048                * THE INSTRUCTIONS  MARKED WITH A #.
0049                *  THE GREATER THAN (GT) TESTS MUST BE
0050                * CHANGED TO LESS THAN (LT) TESTS AND VICE VERSA
0051                *
0052 0000                ORG    H'4F7'
0053 04F7          SIGN  RES    1          SIGN FLAG: SIGN=0 SIGNED NUMBER
0054                *                            SIGN= NOT 0 UNSIGNED NUMBER
0055 04F8          SAVE  RES    1          SAVED LOW-ORDER ADDRESS POINTER 2
0056 04F9          LAST  RES    1          SAVED INDEX OF NUMBER FOLLOWING
0057                *                       ACTUAL LARGEST NUMBER
0058 04FA          AD1   RES    2          HIGH ORDER ADDRESS POINTER 1
0059 04FC          AD2   RES    2          HIGH ORDER ADDRESS POINTER 2
0060 04FE          AD3   RES    2          HIGH ORDER ADDRESS POINTER 3
0061                *
0062 0500                ORG    H'500'
0063 0500          SLIS  RES    H'800'     START ADDRESS OF SORTING LIST
0064 0D00          ELIS  RES    1          END ADDRESS OF SORTING LIST
0065 0008          WLEN  EQU    8          WORD LENGTH (BYTES)
0066                *
0067 0D01                ORG    H'440'
0068 0440 7702     SORT  PPSL   COM        LOGICAL COMPARE
0069 0442 7508           CPSL   WC         WITHOUT CARRY
0070 0444 0400           LODI,R0 >SLIS     SET LOW-ORDER ADDRESS OF INDIRECT
0071 0446 CC04FB         STRA,R0 AD1+1      ADDRESS
0072 0449 CC04FD         STRA,R0 AD2+1
0073 044C CC04FF         STRA,R0 AD3+1
0074 044F 070D           LODI,R3 <ELIS     SET POINTER AT THE END OF THE
0075 0451 CF04FE         STRA,R3 AD3        SORTING LIST
0076 0454 0700           LODI,R3 >ELIS
0077 0456 AC04FE   PASS  LODA,R0 AD3       TEST AND RETURN IF SORT IS
0078 0459 5808           BRNR,R2 PAS1       READY
0079 045B E405           COMI,R0 <SLIS
0080 045D 14             RETC:EQ
0081 045E F800           BDRR,R0 $+2       DECREMENT POINTER 3 TO LAST NUMBER
0082 0460 CC04FE         STRA,R0 AD3        OF ACTUAL LIST
0083 0462 A708     PAS1  SUBI,R3 WLEN
0084 0465 CC04FC         STRA,R0 AD2       SET POINTER 2 AT LAST NUMBER OF
0085 0468 03             LODZ   R3          THE ACTUAL LIST
0086 0469 C2             STRZ   R2
0087 046A AC04FC   MAXM  LODA,R0 AD2       SET POINTER 1 AT THE ACTUAL
0088 046D CC04FA         STRA,R0 AD1        LARGEST NUMBER OF THE ACTUAL
0089 0470 02             LODZ   R2          LIST
0090 0471 C1             STRZ   R1
```

Figure 18

PROGRAM LISTING FOR SEARCH SORT SUBROUTINE (256 EIGHT-BYTE NUMBERS) (Cont.)

```
TWIN ASSEMBLER VER 1.0                                          PAGE 0003

LINE ADDR  OBJECT  E SOURCE

0091 0472 8408           ADDI,R0 WLEN     SAVE INDEX OF NUMBER FOLLOWING
0092 0474 CC04F9         STRA,R0 LAST      THE LAST NUMBER OF THE ACTUAL
0093                 *                     LIST
0094 0477 0C04F7     LOAD LODA,R0 SIGN    IF SIGN IS 0, SET COMPARE,SIGNED
0095 047A 9802           BCFR,Z  LOA1      NUMBER.  ELSE CLEAR COMPARE,
0096 047C 7502           CPSL    COM       UNSIGNED NUMBERS
0097 047E 0DE4FA     LOA1 LODA,R0 *AD1,R1  LOAD R0 WITH FIRST BYTE OF
0098                 *                     ACTUAL LARGEST NUMBER
0099 0481 5A0E       COMP BRNR,R2 COM1     TEST AND BRANCH TO EXCH IF
0100 0483 0E04FC         LODA,R2 AD2       SEARCH IS READY.
0101 0486 E605           COMI,R2 <SLIS
0102 0488 1835           BCTR,EQ EXCH
0103 048A FA00           BDRR,R2 $+2       DECREMENT POINTER 2
0104 048C CE04FC         STRA,R2 AD2
0105 048F 0600           LODI,R2 0
0106 0491 A608       COM1 SUBI,R2 WLEN
0107 0493 EEE4FC         COMA,R0 *AD2,R2   COMPARE ACTUAL LARGEST WORD
0108                 *                     WITH WORD ADDRESSED BY POINT 2
0109 0496 1969           BCTR,GT COMP      #IF GT, THEN IT IS STILL
0110                 *                     ACTUAL LARGEST NUMBER
0111 0498 1A50           BCTR,LT MAXM      #IF LT, THEN NEW ACTUAL
0112                 *                     LARGEST NUMBER IS FOUND
0113                 *                    ELSE COMPARE NEXT BYTES OF
0114                 *                     BOTH NUMBERS.
0115 049A CE04F8         STRA,R2 SAVE      SAVE POINTER 2
0116 049D 7702           PPSL    COM       LOGICAL COMPARE
0117 049F ED04F9     NEXT COMA,R1  LAST     TEST AND BRANCH IF COMPARE OF
0118 04A2 1810           BCTR,EQ RSET       THE FOLLOWING BYTES OF THE
0119                 *                     TWO NUMBERS IS READY.
0120 04A4 0DA4FA         LODA,R0 *AD1,R1,+ COMPARE FOLLOWING BYTES
0121 04A7 EEA4FC         COMA,R0 *AD2,R2,+  OF BOTH NUMBERS.
0122 04AA 1873           BCTR,EQ NEXT      IF EQUAL, CONTINUE
0123 04AC 1906           BCTR,GT RSET      # IF GT, NUMBER ADDRESSED BY
0124                 *                     POINTER 1 IS STILL ACTUAL
0125                 *                     LARGEST NUMBER
0126                 *                    ELSE NEW ACTUAL LARGEST
0127                 *                     NUMBER FOUND
0128 04AE 0E04F8         LODA,R2 SAVE      FETCH SAVED POINTER 2
0129 04B1 1F046A         BCTA,UN MAXM
0130 04B4 0D04F9     RSET LODA,R1 LAST     RESET POINTER 1 AND
0131 04B7 A508           SUBI,R1 WLEN       POINTER 2
0132 04B9 0E04F8         LODA,R2 SAVE
0133 04BC 1F0477         BCTA,UN LOAD
0134 04BF 03         EXCH LODZ    R3       TEST AND BRANCH TO EXC2 IF
0135 04C0 E1             COMZ    R1        ACTUAL LARGEST NUMBER IS
0136 04C1 9808           BCFR,EQ EXC1      SAME AS THE LAST NUMBER OF
0137 04C3 0C04FE         LODA,R0 AD3       THE ACTUAL LIST.
0138 04C6 EC04FA         COMA,R0 AD1
0139 04C9 1819           BCTR,EQ EXC2
0140 04CB 0DE4FA     EXC1 LODA,R0 *AD1,R1  EXCHANGE ACTUAL LAST NUMBER
0141 04CE C2             STRZ    R2        AND LAST NUMBER OF ACTUAL
0142 04CF 0FE4FE         LODA,R0 *AD3,R3    LIST.
0143 04D2 CDE4FA         STRA,R0 *AD1,R1
0144 04D5 02             LODZ    R2
0145 04D6 CFE4FE         STRA,R0 *AD3,R3
0146 04D9 DB00           BIRR,R3 $+2       INCREMENT BOTH POINTERS.
0147 04DB D900           BIRR,R1 $+2
0148 04DD ED04F9         COMA,R1 LAST      TEST AND BRANCH TO EXC1 IF
0149 04E0 9869           BCFR,EQ EXC1       EXCHANGE IS NOT READY
0150 04E2 A708           SUBI,R3 WLEN      RESET POINTER 3
0151 04E4 1F0456     EXC2 BCTA,UN PASS     CONTINUE NEW PASS
0152                 *
0153 0440                END     SORT

 TOTAL ASSEMBLY ERRORS = 0000
```

Figure 18

Related 2650 publications

no.	title	summary
AS50	Serial Input/Output	Using the Sense/Flag capability of the 2650 for serial I/O interfaces.
AS51	Bit & Byte Testing Procedures	Several methods of testing the contents of the internal registers in the 2650.
AS52	General Delay Routines	Several time delay routines for the 2650, including formulas for calculating the delay time.
AS52	Binary Arithmetic Routines	Examples for processing binary arithmetic addition, subtraction, multiplication, and division with the 2650.
AS54	Conversion Routines	• Eight-bit unsigned binary to BCD • Sixteen-bit signed binary to BCD • Signed BCD to binary • Signed BCD to ASCH • ASCII to BCD • Hexadecimal to ASCII • ASCII to Hexadecimal
AS55	Fixed Point Decimal Arithmetic	Methods of performing addition, subtraction, multiplication and division of BCD numbers with the 2650.
SP50	2650 Evaluation Printed Circuit Board (PC1001)	Detailed description of the PC1001, an evaluation and design tool for the 2650.
SP51	2650 Demo System	Detailed description of the Demo System, a hardware base for use with the 2650 CPU prototyping board (PC1001 or PC1500).
SP52	Support Software for use with the NCSS Timesharing System	Step-by-step procedures for generating, editing, assembling, punching, and simulating Signetics 2650 programs using the NCSS timesharing service.
SP53	Simulator, Version 1.2	Features and characteristics of version 1.2 of the 2650 simulator.
SP54	Support Software for use with the General Electric Mark III Timesharing System	Step-by-step procedures for generating, editing, assembling, simulating, and punching Signetics 2650 programs using General Electric's Mark III timesharing system.
SP55	The ABC 1500 Adaptable Board Computer	Describes the components and applications of the ABC 1500 system development card.
SS50	PIPBUG	Detailed description of PIPBUG, a monitor program designed for use with the 2650.
SS51	Absolute Object Format	Describes the absolute object code format for the 2650.
MP51	Initialization	Procedures for initializing the 2650 microprocessor, memory, and I/O devices to their required initial states.
MP52	Low-Cost Clock Generator Circuits	Several clock generator circuits, based on 7400 series TTL, that may be used with the 2650. They include RC, LC and crystal oscillator types.
MP53	Address and Data Bus Interfacing Techniques	Examples of interfacing the 2650 address and data busses with ROMs and RAMs, such as the 2608, 2606 and 2602.
MP54	2650 Input/Output Structures and Interfaces	Examines the use of the 2650's versatile set of I/O instructions and the interface between the 2650 and I/O ports. A number of application examples for both serial and parallel I/O are given.
TN 064	Digital cassette interface for a 2650 microprocessor system	Interface hardware and software for the Philips DCR digital cassette drive.
TN 069	2650 Microprocessor keyboard interfaces	Simple interfaces for low-cost keyboard systems.
TN 072	Introducing the Signetics 2651 PCI Terminology and operation modes	Description of the 2651 Programmable Communications Interface IC.
TN 083	Using the Signetics 2651 PCI with popular microprocessors	Simple hardware interfaces to use the 2651 Programmable Communications Interface with various microprocessors.
TN 084	Using seven-segment LED display with the 2650 microprocessor	Interfaces for single and multi-digit LED displays.
TN 085	Cyclic redundancy check by software	A short routine to encode and decode CRC check characters for the 2650.
TN 086	Introducing the Signetics 2655 PPI	Description of the 2655 Programmable Peripheral Interface.
TN 087	Audio cassette recorder interface for the 2650 microprocessor	Economical alternatives to the digital cassette recorder.
TN 089	CRT display using a standard TV monitor for 2650-based microcomputers	Economical solution for a visual display unit

Electronic components and materials

for professional, industrial and consumer uses

from the world-wide Philips Group of Companies

Argentina: FAPESA I.y.C., Av. Crovara 2550, Tablada, Prov. de BUENOS AIRES, Tel. 652-7438/7478.
Australia: PHILIPS INDUSTRIES HOLDINGS LTD., Elcoma Division, 67 Mars Road, LANE COVE, 2066, N.S.W., Tel. 427 08 88.
Austria: ÖSTERREICHISCHE PHILIPS BAUELEMENTE Industrie G.m.b.H., Triester Str. 64, A-1101 WIEN, Tel. 62 91 11.
Belgium: M.B.L.E., 80, rue des Deux Gares, B-1070 BRUXELLES, Tel 523 00 00.
Brazil: IBRAPE, Caixa Postal 7383, Av. Paulista 2073-S/Loja, SAO PAULO, SP, Tel. 284-4511.
Canada: PHILIPS ELECTRONICS LTD., Electron Devices Div., 601 Milner Ave., SCARBOROUGH, Ontario, M1B 1M8, Tel. 292-5161.
Chile: PHILIPS CHILENA S.A., Av. Santa Maria 0760, SANTIAGO, Tel. 39-40 01.
Colombia: SADAPE S.A., P.O. Box 9805, Calle 13, No. 51 + 39, BOGOTA D.E. 1., Tel. 600 600.
Denmark: MINIWATT A/S, Emdrupvej 115A, DK-2400 KØBENHAVN NV., Tel. (01) 69 16 22.
Finland: OY PHILIPS AB, Elcoma Division, Kaivokatu 8, SF-00100 HELSINKI 10, Tel. 1 72 71.
France: R.T.C. LA RADIOTECHNIQUE-COMPELEC, 130 Avenue Ledru Rollin, F-75540 PARIS 11, Tel. 355-44-99.
Germany: VALVO, UB Bauelemente der Philips G.m.b.H., Valvo Haus, Burchardstrasse 19, D-2 HAMBURG 1, Tel. (040) 3296-1.
Greece: PHILIPS S.A. HELLENIQUE, Elcoma Division, 52, Av. Syngrou, ATHENS, Tel. 915 311.
Hong Kong: PHILIPS HONG KONG LTD., Comp. Dept., Philips Ind. Bldg., Kung Yip St., K.C.T.L. 289, KWAI CHUNG, N.T. Tel. 12-24 51 21.
India: PHILIPS INDIA LTD., Elcoma Div., Band Box House, 254-D, Dr. Annie Besant Rd., Prabhadevi, BOMBAY-25-DD, Tel. 457 311-5.
Indonesia: P.T. PHILIPS-RALIN ELECTRONICS, Elcoma Division, 'Timah' Building, Jl. Jen. Gatot Subroto, JAKARTA, Tel. 44 163.
Ireland: PHILIPS ELECTRICAL (IRELAND) LTD., Newstead, Clonskeagh, DUBLIN 14, Tel. 69 33 55.
Italy: PHILIPS S.p.A., Sezione Elcoma, Piazza IV Novembre 3, I-20124 MILANO, Tel. 2-6994.
Japan: NIHON PHILIPS CORP., Shuwa Shinagawa Bldg., 26-33 Takanawa 3-chome, Minato-ku, TOKYO (108), Tel. 448-5611.
(IC Products) SIGNETICS JAPAN, LTD., TOKYO, Tel. (03) 230-1521.
Korea: PHILIPS ELECTRONICS (KOREA) LTD., Elcoma Division, Philips House, 260-199 Itaewon-dong, Yongsan-ku, SEOUL, Tel. 794-4202.
Mexico: ELECTRONICA S.A. de C.V., Varsovia No. 36, MEXICO 6, D.F., Tel. 533-11-80.
Netherlands: PHILIPS NEDERLAND B.V., Afd. Elonco, Boschdijk 525, NL 5600 PD EINDHOVEN, Tel. (040) 79 33 33.
New Zealand: PHILIPS Electrical Ind. Ltd., Elcoma Division, 2 Wagener Place, St. Lukes, AUCKLAND, Tel. 867 119.
Norway: NORSK A/S PHILIPS, Electronica Dept., Vitaminveien 11, Grefsen, OSLO 4, Tel. (02) 15 05 90.
Peru: CADESA, Rocca de Vergallo 247, LIMA 17, Tel. 62 85 99.
Philippines: ELDAC, Philips Industrial Dev. Inc., 2246 Pasong Tamo, MAKATI-RIZAL, Tel. 86-89-51 to 59.
Portugal PHILIPS PORTUGESA S.A.R.L., Av. Eng. Duharte Pacheco 6, LISBOA 1, Tel. 68 31 21.
Singapore: PHILIPS SINGAPORE PTE LTD., Elcoma Div., P.O.B. 340, Toa Payoh CPO, Lorong 1, Toa Payoh, SINGAPORE 12, Tel. 53 88 11.
South Africa: EDAC (Pty.) Ltd., South Park Lane, New Doornfontein, JOHANNESBURG 2001, Tel. 24/6701.
Spain: COPRESA S.A., Balmes 22, BARCELONA 7, Tel. 301 63 12.
Sweden: A.B. ELCOMA, Lidingövägen 50, S-10 250 STOCKHOLM 27, Tel. 08/67 97 80.
Switzerland: PHILIPS A.G., Elcoma Dept., Edenstrasse 20, CH-8027 ZÜRICH, Tel. 01/44 22 11.
Taiwan: PHILIPS TAIWAN LTD., 3rd Fl., San Min Building, 57-1, Chung Shan N. Rd, Section 2, P.O. Box 22978, TAIPEI, Tel. 5513101-5.
Turkey: TÜRK PHILIPS TICARET A.S., EMET Department, Inonu Cad. No. 78-80, ISTANBUL, Tel. 43 59 10.
United Kingdom: MULLARD LTD., Mullard House, Torrington Place, LONDON WC1E 7HD, Tel. 01-580 6633.
United States: (Active devices & Materials) AMPEREX SALES CORP., Providence Pike, SLATERSVILLE, R.I. 02876, Tel. (401) 762-9000.
(Passive devices) MEPCO/ELECTRA INC., Columbia Rd., MORRISTOWN, N.J. 07960, Tel. (201) 539-2000.
(IC Products) SIGNETICS CORPORATION, 811 East Arques Avenue, SUNNYVALE, California 94086, Tel. (408) 739-7700.
Uruguay: LUZILECTRON S.A., Rondeau 1567, piso 5, MONTEVIDEO, Tel. 9 43 21.
Venezuela: IND. VENEZOLANAS PHILIPS S.A., Elcoma Dept., A. Ppal de los Ruices, Edif. Centro Colgate, CARACAS, Tel. 36 05 11.

Printed in The Netherlands Date of release: 31 August 1978 T.D. 9398 009 20011