```
%-------------------------------------------------------------------------------
% CALCULATOR-1
% VER 1.0
%
% RPN CALCULATOR PROGRAM FOR PERSEUS-7 (CPU:R6502) AND NUMPAD-1
%
% THE FOUR ARITHMETIC OPERATIONS BY 128 BITS FIXED-POINT BCD
% HAND ASSEMBLED
%
% MITSURU YAMADA 20/OCT/2020
%
%-------------------------------------------------------------------------------
% COPYRIGHT (C) 2020 MITSURU YAMADA. ALL RIGHTS RESERVED.
%
%-------------------------------------------------------------------------------
% ADDRESS MAPPING: RPN STACK REGISTER 128BIT BCD (R0, R1), RESULT REGISTER (R2)
%
%     SIGN TOP                                                              BOTTOM
% R0:  $0F $0E $0D $0C $0B $0A $09 $08 DECIMAL POINT $07 $06 $05 $04 $03 $02 $01 $00
%
% R1:  $1F $1E $1D $1C $1B $1A $19 $18 DECIMAL POINT $17 $16 $15 $14 $13 $12 $11 $10
%
% R2:  $2F $2E $2D $2C $2B $2A $29 $28 DECIMAL POINT $27 $26 $25 $24 $23 $22 $21 $20
%
%                                                    DISPLAY REGISTER (R3)
%                                             TOP(LEFT)                    BOTTOM(RIGHT)
%                                           R3:  $37 $36 $35 $34 $33 $32 $31 $30
%
% ------------------------------------------------------------------------------
% VARIABLES
             SYMBOL                    DATA
             TABLE_AD                  $38          KEY SCAN TABLE ADDRESS LOWER
             TABLE_AD+1                $39          KEY SCAN TABLE ADDRESS UPPER
             SCAN_PAT                  $3A          KEY SCAN PATTERN EVACUATION
             FLAG_1                    $3B          KEY PRESSED FLAG
             FLAG_2                    $3C          KEY PRESSED LAST TIME FLAG
             START_FLAG                $3D          NUMERIC DETECTION(NUMDET) FLAG
             COL_COUNI_1               $3E          DIGIT COUNTER #1
             COL_COUNT_2               $3F          DIGIT COUNTER #2
             REG_0                     $40          LED FONT PATTERN EVACUATION
             P0                        $40          16BIT ADDRESS POINTER LOWER
             P0+1                      $41          16BIT ADDRESS POINTER UPPER
             REG_2                     $42          FONT OUTPUT LOOP COUNTER
             SFT_COUNT                 $42          SHIFT WIDTH COUNTER
             TEMP_1                    $43          TEMPORARY EVACUATION REGISTER
             COL_COUNT                 $43          DIGIT COUNTER
             REG_3                     $43          DISPLAY BUFFER POINTER
             B_COUNT                   $43          BCD DIGIT COUNTER
             REG_4                     $44          FONT TABLE ADDRESS MUL. EVAC.
             MUL_COUNT                 $44          MULTIPLIER COUNTER
             QUO_COUNT                 $44          QUOTIENT COUNTER
             TEMP_COUNT                $45          TEMPORARY COUNTER (MOVE_REG_1)
             SIGN                      $46          SIGN EVACUATION REGISTER
             SIGN_FLAG                 $46          SIGN FLAG
             REG_1                     $47          LED CONTROL PATTERN EVACUATION
             DISP_P0                   $A1          DISPLAY POSITION #0
             DISP_P1                   $A2          DISPLAY POSITION #1
%
% ------------------------------------------------------------------------------
             BUF_START                 $30          DISPLAY BUFFER R3 HEAD ADDRESS
             PORT                      $4000        PARALLEL INTERFACE ADDRESS (R/W)
% ------------------------------------------------------------------------------
%
%                          ADDRESS(HEX) DATA(HEX)
%-------------------------------------------------------------------------------
% PROGRAM START ADDRESS    CALC_MAIN_1  $0EF
%
% RESET VECTOR
%             .ORIGIN  $3FC
                                  3FC    EF 00
%-------------------------------------------------------------------------------
%
```

```
%-------------------------------------------------------------------------------
% OUTOUT FONT TO LED DISPLAY MODULE BY SERIAL
            .ORIGIN  $0A8
%
OUT_FNT_SERL LDY  #$08           0A8    A0 08         SET LOOP COUNTER (8BIT)
             STA  REG_0          0AA    85 40         EVACUATE FONT DATA
             LDA  REG_1          0AC    A5 47         RETURN CONTROL DATA
             AND  #$3F           0AE    29 3F         (DATA=L,CLK=L,RS=,*CE=)
             STA  REG_1          0B0    85 47
L11          LDA  REG_0          0B2    A5 40         RETURN FONT DATA
             AND  #$80           0B4    29 80         EXTRACTION D7
             ORA  REG_1          0B6    05 47         SYNTHESIZE CONTROL DATA
             STA  PORT           0B8    8D 00 40      OUTPUT TO LED MODULE
             ORA  #$40           0BB    09 40         (DATA=,CLK=H,RS=,*CE=)
             STA  PORT           0BD    8D 00 40      OUTPUT TO LED MODULE
             ASL  REG_0          0C0    06 40         SHIFT LEFT FONT DATA
             DEY                 0C2    88            LOOP COUNTER -1
             BNE  L11            0C3    D0 ED         8BIT FINISH?
             RTS                 0C5    60
%
%-------------------------------------------------------------------------------
% INITIALIZE LED DISPLAY MODULE
INIT_DISPLAY LDA  #$00           0C6    A9 00
             STA  PORT           0C8    8D 00 40      RESET LED MODULE (*RE=L)
             LDA  #$08           0CB    A9 08
             STA  PORT           0CD    8D 00 40      RELEASE RESET LED MODULE (*RE=H)
             LDA  #$38           0D0    A9 38
             STA  PORT           0D2    8D 00 40      (CLK=L,RS=H,*CE=H)
             LDA  #$28           0D5    A9 28
             STA  PORT           0D7    8D 00 40      (CLK=L,RS=H,*CE=L)
             STA  REG_1          0DA    85 47         EVACUATE CONTROL PATTERN
             LDX  #$02           0DC    A2 02         LOOP COUNER = 2
L15          STX  REG_3          0DE    86 43         EVACUATE LOOP COUNTER
             LDA  #$4A           0E0    A9 4A         SET CONTROL WORD 0
             JSR  OUT_FNT_SERL   0E2    20 A8 00      SEND SERIAL DATA
             LDX  REG_3          0E5    A6 43         RETURN LOOP COUNTER
             DEX                 0E7    CA            LOOP COUNTER -1
             BNE  L15            0E8    D0 F4
             LDA  #$38           0EA    A9 38
             STA  REG_1          0EC    85 47         (CLK=L,RS=H,*CE=H)
             RTS                 0EE    60
%
%-------------------------------------------------------------------------------
% CALCULATOR MAIN PROGRAM
CALC_MAIN_1  CLD                 0EF    D8            SET BINARY MODE
CALC_MAIN    LDX  #$FF           0F0    A2 FF
             TXS                 0F2    9A            INIT. STACK POINTER
             INX                 0F3    E8
             STX  START_FLAG     0F4    86 3D         INIT. NUMERIC DETECTION(NUMDET) FLAG
             STX  TABLE_AD+1     0F6    86 39         KEY SCAN TABLE ADDRESS UPPER
             STX  FLAG_2         0F8    86 3C         INIT. PRESSED KEY LAST TIME FLAG
             LDX  #$12           0FA    A2 12
             STX  DISP_P0        0FC    86 A1
             LDX  #$04           0FE    A2 04
             STX  DISP_P1        100    86 A2
             JSR  INIT_DISPLAY   102    20 C6 00      INIT. LED DISPLAY MODULE
L06          JSR  SCAN_KEY       105    20 0E 01      KEY SCANNING
             JSR  BCD_TO_DISP    108    20 82 03      CONVERT BCD TO DISPLAY CODE & OUT
             CLC                 10B    18
             BCC  L06            10C    90 F7         REPEAT
%
%-------------------------------------------------------------------------------
% KEY SCANNING
%
SCAN KEY     LDX  #$48           10E    A2 48
             STX  TABLE_AD       110    86 38         INIT. TABLE ADDRESS
             LDX  #$00           112    A2 00
             STX  FLAG_1         114    86 3B         RESET KEY PRESSED FLAG
             INX                 116    E8            INIT. SCAN PATTERN ($01)
             STX  SCAN_PAT       117    86 3A

L31          LDA  SCAN_PAT       119    A5 3A
```

```
             EOR  #$FF           11B    49 FF         INVERT SCAN PATTERN
             STA  PORT           11D    8D 00 40      OUTPUT SCAN PATTERN
             LDA  PORT           120    AD 00 40      INPUT SCAN RESULT
             EOR  #$FF           123    49 FF         INVERT SCAN RESULT
             BEQ  L32            125    F0 18         $00? (NOT PRESSED)
%
             LDY  #$00           127    A0 00         INIT. LOOP COUNTER
L34          LSR                 129    4A            SHIFT RIGHT 1BIT
             BCS  L33            12A    B0 03
             INY                 12C    C8            LOOP COUNTER +1
             BCC  L34            12D    90 FA
%
L33          INC  FLAG_1         12F    E6 3B         SET PRESSED FLAG TO '1'
             LDA  FLAG_2         131    A5 3C         LAST PRESSED FLAG = '1' ?
             BNE  L32            133    D0 0A         IGNORE HOLDING DOWN
             LDA  (TABLE_AD),Y   135    B1 38         GET NUMBER BY REFERRING TO TABLE
             JSR  STORE_REG_1    137    20 51 01      PROCESSING OF ACQUIRED NUMBER
%
L35          LDA  FLAG_1         13A    A5 3B
             STA  FLAG_2         13C    85 3C         UPDATE KEY PRESSED LAST TIME FLAG
             RTS                 13E    60
%
L32          LDA  TABLE_AD       13F    A5 38
             CLC                 141    18
             ADC  #$08           142    69 08         UPDATE TABLE ADDRESS (INCREMENT D3)
             STA  TABLE_AD       144    85 38
             LDA  SCAN_PAT       146    A5 3A         UPDATE SCAN PATTERN
             ASL                 148    0A
             STA  SCAN_PAT       149    85 3A
             CMP  #$08           14B    C9 08         SCAN PATTRN '100' FINISH?
             BNE  L31            14D    D0 CA         IF NOT FINISH RETURN TO OUT SCAN PATTERN
             BEQ  L35            14F    F0 E9         IF FINISH GO TO UPDATE FLAG
%
%-------------------------------------------------------------------------
% ARROW KEY FUNCTION
STORE_REG_1  CMP  #$12           151    C9 12         ARROW KEY CODE?
             BNE  STORE_REG_2    153    D0 15
             LDX  DISP_P0        155    A6 A1
             LDY  DISP_P1        157    A4 A2
             CPY  #$07           159    C0 07
             BNE  L21            15B    D0 09         IF DISP_P1 IS NOT $07 THEN P0,P1 +1
             LDX  #$10           15D    A2 10         IF DISP_P1 IS $07 THEN P0=$10,P1=$02
             LDY  #$02           15F    A0 02
L22          STX  DISP_P0        161    86 A1         STORE DISP_P0,DISP_P1
             STY  DISP_P1        163    84 A2
             RTS                 165    60
L21          INX                 166    E8
             INY                 167    C8
             BNE  L22            168    D0 F7
%
%-------------------------------------------------------------------------------
% STORE TO RPN STACK REGISTER
STORE_REG_2  CMP  #$10           16A    C9 10         INVERSION (NEGATIVE) KEY CODE?
             BNE  L01            16C    D0 07
             LDA  $0F            16E    A5 0F         LOAD SIGN BYTE OF R0 RPN REGISTER
             EOR  #$80           170    49 80         INVERT SIGN
             STA  $0F            172    85 0F         STORE SIGN BYTE
             RTS                 174    60
%
L01          CMP  #$0D           175    C9 0D         + KEY CODE?
             BNE  L02            177    D0 03
             JMP  ADD_SABS_30    179    4C 1B 03      GO TO ADDITION
L02          CMP  #$0C           17C    C9 0C         - KEY CODE?
             BNE  L03            17E    D0 03
             JMP  SUB_SABS_30    180    4C 30 03      GO TO SUBTRACTION
L03          CMP  #$0E           183    C9 0E         * KEY CODE?
             BNE  L04            185    D0 03
             JMP  MUL_BCD_16_1   187    4C 00 02      GO TO MULTIPLICATION
L04          CMP  #$0F           18A    C9 0F         / KEY CODE?
             BNE  L05            18C    D0 03
             JMP  DIV_BCD_16_1   18E    4C 54 02      GO TO DIVISION
%
```

```
L05           CMP  #$11            191    C9 11        ENTER CODE?
              BNE  L06             193    D0 0A
              LDX  #$00            195    A2 00
              STX  START_FLAG      197    86 3D        RESET NUMDET FLAG
              LDY  #$10            199    A0 10
              JSR  MOV_REG_1       19B    20 00 03     PUSH RPN REGISTER R0 TO R1
              RTS                  19E    60
%
L06           CMP  #$0A            19F    C9 0A        DECIMAL POINT CODE?
              BNE  L07             1A1    D0 05
              LDX  #$02            1A3    A2 02
              STX  START_FLAG      1A5    86 3D        SET START_FLAG
              RTS                  1A7    60
%
L07           STA  TEMP_1          1A8    85 43        EVACUATE ACQUIRED VALUE
              LDX  START_FLAG      1AA    A6 3D        FIRST NUMERIC DETECTED?
              BNE  L08             1AC    D0 0D
              INC  START_FLAG      1AE    E6 3D        NUMDET FLAG CHANGE 0 TO 1
              LDX  #$04            1B0    A2 04
              STX  COL_COUNT_1     1B2    86 3E        INITIALIZE COL_COUNT_1
              LDX  #$00            1B4    A2 00
              STX  COL_COUNT_2     1B6    86 3F        INITIALIZE COL_COUNT_2
              JSR  CLR_REG_1       1B8    20 10 03     CLEAR RPN REGISTER R0
%
L08           CPX  #$02            1BB    E0 02        IF NUMDET FLAG IS 2 THEN
              BEQ  GET_FRACTION    1BD    F0 18            GO TO GET FRACTION
%
GET_INT       LDA  $0F             1BF    A5 0F        GET INTEGER PART
              STA  SIGN            1C1    85 46        EVACUATE SIGN
              LDX  #$04            1C3    A2 04        SET SHIFT WIDTH (4BIT)
              LDY  #$10            1C5    A0 10        SET POINTER LOWER 8BIT
              LDA  #$FF            1C7    A9 FF        SET POINTER UPPER 8BIT
              JSR  SHIFT_L_NBIT    1C9    20 ED 02     SHIFT LEFT 4BIT RPN REGISTER R0
              LDA  TEMP_1          1CC    A5 43        PUT ACQUIRED VALUE TO R0_8 LOWER 4BIT
              ORA  $08             1CE    05 08
              STA  $08             1D0    85 08
              LDA  SIGN            1D2    A5 46
              STA  $0F             1D4    85 0F        SIGN RETURN
              RTS                  1D6    60
%
GET_FRACTION  LDX  COL_COUNT_1     1D7    A6 3E
              BEQ  L09             1D9    F0 0C
              LDY  COL_COUNT_2     1DB    A4 3F        DIGIT POSITION $0:ODD, $1:EVEN
              BNE  L08             1DD    D0 09
              ASL                  1DF    0A           IF DIGIT POSITION IS ODD THEN
              ASL                  1E0    0A                  SHIFT LEFT 4BIT
              ASL                  1E1    0A
              ASL                  1E2    0A
              STA  $03,X           1E3    95 03        PUT VALUE TO R0 UPPER 4BIT
              INC  COL_COUNT_2     1E5    E6 3F        NEXT DIGIT POSITION IS EVEN
L09           RTS                  1E7    60
%
L08           ORA  $03,X           1E8    15 03
              STA  $03,X           1EA    95 03        PUT VALUE TO R0 LOWER 4BIT
              DEC  COL_COUNT_2     1EC    C6 3F        NEXT DIGIT POSITION IS ODD
              DEC  COL_COUNT_1     1EE    C6 3E        DIGIT COUNT -1
              RTS                  1F0    60
%
%-------------------------------------------------------------------------------
% 128BIT BCD FIXED POINT MULTIPLICATION & DIVISION
%
% SIGNED MULTIPLICATION
              .ORIGIN  $200
%
MUL_BCD_16_3  JSR  PRE_MUL_1       200    20 BE 02
              LDX  #$20            203    A2 20        SET SHIFT WIDTH (32BIT=4BYTE)
              LDY  #$00            205    A0 00        SET POINTER LOWER 8BIT
              TYA                  207    98           SET POINTER UPPER 8BIT
              JSR  SHIFT_R_NBIT    208    20 DA 02     SHIFT RIGHT 4BYTE RPN REGISTER R0
%
% MULTIPLICATION CORE
% R2 = R1 X R0
```

```
MUL_BCD_16_4 LDA  #$10           20B   A9 10
             STA  COL_COUNT      20D   85 43       INIT. DIGIT COUNTER
L48          LDA  $00            20F   A5 00
             AND  #$0F           211   29 0F       EXTRACT. MULTIPLIER LEAST SIGNIFICANT 4BIT
             STA  MUL_COUNT      213   85 44
             BEQ  L41            215   F0 10       IF MUL_COUNT=0 THEN DO NOT ACCUMULATE
L43          LDX  #$F0           217   A2 F0       INIT. DIGIT COUNTER
             CLC                 219   18
L42          LDA  $30,X          21A   B5 30       LOAD RESULT REGISTER  R2
             ADC  $20,X          21C   75 20       ACCUMULATE MULTIPLICAND R1
             STA  $30,X          21E   95 30       STORE TO R2
             INX                 220   E8          DIGIT COUNTER +1
             BNE  L42            221   D0 F7       REPEAD UNTIL END OF TOP
             DEC  MUL_COUNT      223   C6 44       MULTIPLIER COUNTER -1
             BNE  L43            225   D0 F0       REPEAD UNTIL MULTIPLIER COUNTER=0
L41          LDX  #$04           227   A2 04
             LDY  #$00           229   A0 00
             TYA                 22B   98
             JSR  SHIFT_R_NBIT   22C   20 DA 02     SHIFT RIGHT 4BIT MULTIPLIER R0
             LDX  #$04           22F   A2 04
             LDY  #$20           231   A0 20
             LDA  #$FF           233   A9 FF
             JSR  SHIFT_L_NBIT   235   20 ED 02     SHIFT LEFT 4BIT MULTIPLICAND R1
             DEC  COL_COUNT      238   C6 43
             BNE  L48            23A   D0 D3
%
AFTER_OPE_1  LDX  #$20           23C   A2 20
             LDY  #$00           23E   A0 00
             JSR  MOV_REG_1      240   20 00 03     MOVE FROM R2 TO R0
             LDA  SIGN_FLAG      243   A5 46
             STA  $0F            245   85 0F       JOIN RESULT R0 AND SIGN
AFTER_OPE_2  LDX  #$00           247   A2 00
             STX  START_FLAG     249   86 3D       RESET NUMDET FLAG
             LDX  #$00           24B   A2 00
             LDY  #$10           24D   A0 10
             JSR  MOV_REG_1      24F   20 00 03     MOVE FROM R0 TO R1
             CLD                 252   D8          SET BINALY MODE
             RTS                 253   60
%
%
% SIGNED DIVISION
DIV_BCD_16_3 JSR  PRE_MUL_1      254   20 BE 02
             LDX  #$20           257   A2 20
             LDY  #$10           259   A0 10
             LDA  #$FF           25B   A9 FF
             JSR  SHIFT_L_NBIT   25D   20 ED 02     SHIFT LEFT 4BYTE DIVISOR R0
%
% DIVISION CORE
% R2 = R1 / R0
DIV_BCD_16_4 LDA  #$00           260   A9 00
             STA  COL_COUNT      262   85 43       INIT. DIGIT COUNTER
L57          LDX  #$04           264   A2 04
             LDY  #$00           266   A0 00
             TYA                 268   98
             JSR  SHIFT_R_NBIT   269   20 DA 02     SHIFT RIGHT 4BIT DIVISOR R0
             BEQ  DET_ZERO_R0    26C   F0 39
L59          LDA  #$FF           26E   A9 FF
             STA  QUO_COUNT      270   85 44       RESET QUOTIENT COUNTER
L54          INC  QUO_COUNT      272   E6 44       QUOTIENT COUNTER +1
             LDX  #$F0           274   A2 F0       BCD 32DIGIT SUBTRACTION
             SEC                 276   38          CLEAR BOLLOW
L53          LDA  $20,X          277   B5 20       ROAD DIVIDEND R1
             SBC  $10,X          279   F5 10       SUBTRACT DIVISOR R0
             STA  $20,X          27B   95 20       STORE TO R1
             INX                 27D   E8          DIGIT POINTER +1
             BNE  L53            27E   D0 F7
             BCS  L54            280   B0 F0       IF RESULT IS PLUS, REPEAT SUB.
%
L58          LDX  #$04           282   A2 04       IF RESULT IS MINUS, SHIFT QUOTIENT
             LDY  #$30           284   A0 30
             LDA  #$FF           286   A9 FF
             JSR  SHIFT_L_NBIT   288   20 ED 02     SHIFT LEFT 4BIT QUOTIENT R2
```

```
             LDA  R2_0           28D    A5 20
             ORA  QUO_COUNT      28D    05 44        PUT QUOTIENT
             STA  R2_0           28F    85 20            TO R2 LEAST SIGNIFICANT BYTE
%
             LDX  #$F0           291    A2 F0        RESTORING
             CLC                 293    18
L56          LDA  $20,X          294    B5 20
             ADC  $10,X          296    75 10
             STA  $20,X          298    95 20        R1 + R0 = R1
             INX                 29A    E8
             BNE  L56            29B    D0 F7
             INC  COL_COUNT      29D    E6 43        DIGIT COUNTER +1
             LDA  COL_COUNT      29F    A5 43
             CMP  #$20           2A1    C9 20        SHIFT & SUBTRACTION FINISH 32DIGIT?
             BNE  L57            2A3    D0 BF
             BEQ  AFTER_OPE_1    2A5    F0 95
%
% DETERMINE IF R0 IS ZERO
DET_ZERO_R0  LDX  #$10           2A7    A2 10        ADDRESS POINTER
             LDY  #$10           2A9    A0 10        NUMBER OF BYTE ZERO COUNTER
L52          LDA  $FF,X          2AB    B5 FF
             BNE  L51            2AD    D0 01
             DEY                 2AF    88
L51          DEX                 2B0    CA
             BNE  L52            2B1    D0 F8
             TYA                 2B3    98
             BEQ  L50            2B4    F0 02
             BNE  L59            2B6    D0 B6        IF R0 IS NOT ZERO GO TO
L50          LDA  #$00           2B8    A9 00        IF R0 IS ZERO THEN
             STA  QUO_COUNT      2BA    85 44               SET ZERO QUOTIENT COUNTER
             BEQ  L58            2BC    F0 C4
%
% COMMON PROCESSING OF MULTIPLICATION & DIVISION
PRE_MUL_1    SED                 2BE    F8           SET DECIMAL MODE
             LDA  $1F            2BF    A5 1F        IF R0 & R1 SIGN IS DIFFERENT
             EOR  $0F            2C1    45 0F               SET NEGATIVE TO SIGN FLAG
             STA  SIGN_FLAG      2C3    85 46
             LDA  #$00           2C5    A9 00
             STA  $1F            2C7    85 1F        CLEAR R1 SIGN BYTE
             STA  $0F            2C9    85 0F        CLEAR R0 SIGN BYTE
             LDX  #$20           2CB    A2 20
             LDY  #$10           2CD    A0 10
             LDA  #$00           2CF    A9 00
             JSR  SHIFT_R_NBIT   2D1    20 DA 02     SHIFT RIGHT 4BYTE R1
             LDX  #$20           2D4    A2 20
             JSR  CLR_REG_1      2D6    20 10 03     CLEAR R2
             RTS                 2D9    60
%
%-------------------------------------------------------------------------------
% 128BIT REGISTER SHIFT RIGHT N BIT
% PARAMETER:  IX:SHIFT WIDTH (BIT), IY:P0, ACC:P0+1
SHIFT_R_NBIT STY  P0             2DA    84 40
             STA  P0+1           2DC    85 41
L67          CLC                 2DE    18
             LDY  #$0F           2DF    A0 0F        INIT. ADDRESS OFFSET
L66          LDA  (P0),Y         2E1    B1 40
             ROR                 2E3    6A
             STA  (P0),Y         2E4    91 40
             DEY                 2E6    88           ADDRESS -1
             BPL  L66            2E7    10 F8        MOST LOWER BIT FINISH?
             DEX                 2E9    CA           SHIFT WIDTH LOOP COUNTER -1
             BNE  L77            2EA    D0 F2
             RTS                 2EC    60
%
%-------------------------------------------------------------------------------
% 128BIT REGISTER SHIFT LEFT N BIT
% PARAMETER:  IX:SHIFT WIDTH (BIT), IY:P0, ACC:P0+1
SHIFT_R_NBIT STA  P0+1           2ED    85 41
             STY  P0             2EF    84 40
L69          CLC                 2F1    18
             LDY  #$F0           2F2    A0 F0        INIT. ADDRESS OFFSET
L68          LDA  (P0),Y         2F4    B1 40
```

```
             ROL                 2F6    2A
             STA  (P0),Y         2F7    91 40
             INY                 2F9    C8           ADDRESS +1
             BNE  L68            2FA    D0 F8        MOST UPPER BIT FINISH?
             DEX                 2FC    CA           SHIFT WIDTH LOOP COUNTER -1
             BNE  L69            2FD    D0 F2
             RTS                 2FF    60
%
%-------------------------------------------------------------------------------
% 128BIT REGISTER MOVE
% PARAMETER:  IX:SOURCE HEAD ADDRESS, IY:DESTINATION HEAD ADDRESS
MOV_REG_1    LDA  #$10           300    A9 10
             STA  TEMP_COUNT     302    85 45
L62          LDA  $00,X          304    B5 00
             STA  $0000,X        306    99 00 00
             INX                 309    E8
             INY                 30A    C8
             DEC  TEMP_COUNT     30B    C6 45
             BNE  L62            30D    D0 F5
             RTS                 30F    60
%
%-------------------------------------------------------------------------------
% 128BIT REGISTER CLEAR
% PARAMETER:  IX:HEAD ADDRESS,
CLR_REG_1    LDY  #$10           310    A0 10
             LDA  #$00           312    A9 00
L61          STA  $00,X          314    95 00
             INX                 316    E8
             DEY                 317    88
             BNE  L61            318    D0 FA
             RTS                 31A    60
%
%-------------------------------------------------------------------------------
% 128BIT BCD FIXED POINT ADDITION & SUBTRACTION
% SIGNED ABSOLUTE ADDITION
% R0 = R1 + R0
ADD_SABS_128 JSR  PRE_ADD        31B    20 41 03     CONVERT ABSOLUTE TO COMPLEMENTARY
             LDX  #$F0           31E    A2 F0        BCD COMPLEMENTARY ADDITION
             CLC                 320    18           CLEAR CARRY
L71          LDA  $20,X          321    B5 20
             ADC  $10,X          323    75 10        8BIT ADDITION
             STA  $10,X          325    95 10
             INX                 327    E8
             BNE  L71            328    D0 F7
L73          JSR  COMP_2_ABS     32A    20 63 03     CONVERT COMPLEMENTARY TO ABSOLUTE
             JMP  AFTER_OPE_2    32D    4C 47 02
%
% SIGNED ABSOLUTE SUBTRACTION
% R0 = R1 - R0
SUB_SABS_128 JSR  PRE_ADD        330    20 41 03     CONVERT ABSOLUTE TO COMPLEMENTARY
             LDX  #$F0           333    A2 F0        BCD COMPLEMENTARY SUBTRACTION
             SEC                 335    38           CLEAR BORROW
L72          LDA  $20,X          336    B5 20
             SBC  $10,X          338    F5 10        8BIT SUBTRACTION
             STA  $10,X          33A    95 10
             INX                 33C    E8
             BNE  L72            33D    D0 F7
             BEQ  L73            33F    F0 E9        GO TO COMMON PROCESSING
%
% COMMON PROCESSING OF ADDITION & SUBTRACTION
PRE_ADD      SED                 341    F8
             LDA  #$00           342    A9 00
             STA  P0+1           344    85 41
             LDA  #$1F           346    A9 1F
             STA  P0             348    85 40
             JSR  ABS_2_COMP     34A    20 55 03     CONVERT R1 SIGNED ABS TO COMPLEMENTARY
             LDA  #$0F           34D    A9 0F
             STA  P0             34F    85 40
             JSR  ABS_2_COMP     351    20 55 03     CONVERT R0 SIGNED ABS TO COMPLEMENTARY
             RTS                 354    60
%
% CONVERT SIGNED ABS TO COMPLEMENTARY
```

```
ABS_2_COMP    LDY  #$00           355    A0 00
              LDA  (P0),Y         357    B1 40        EXTRACTION SIGN BYTE
              BEQ  L81            359    F0 07        IF SIGN = POSITIVE THE DO NOT CONVERT
              JSR  COMP_1         35B    20 71 03     CONVERT TO COMPLEMENT
              LDA  #$99           35E    A9 99        SET UPPER BYTE = $99
              STA  (P0),Y         360    91 40
L81           RTS                 362    60
%
% CONVERT COMPLEMENTARY TO SIGNED ABS
COMP_2_ABS    LDY  #$00           363    A0 00
              LDA  (P0),Y         365    B1 40        EXTRACTION SIGN BYTE
              BPL  L82            367    10 07        IF POSITIVE DO NOT CONVERT
              JSR  COMP_1         369    20 71 03     CONVERT TO COMPLEMENTARY
              LDA  #$80           36C    A9 80
              STA  (P0),Y         36E    91 40        SET $80 TO TOP BYTE
L82           RTS                 370    60
%
% COMPLEMENTARY CONVERSION CORE
COMP_1        DEC  P0+1           371    C6 41         REASON: P0+1,P0($FF0F) + $F1(IY) = $00
              SEC                 373    38
              LDY  #$F1           374    A0 F1
L83           LDA  #$00           376    A9 00
              SBC  (P0),Y         378    F1 40        ACC = 0 - M - BORROW
              STA  (P0),Y         37A    91 40
              INY                 37C    C8
              BNE  L83            37D    D0 F7
              INC  P0+1           37F    E6 41
              RTS                 381    60
%
%-------------------------------------------------------------------------------
% TRANSFER BCD RPN REGISTER R0 TO DISPLAY BUFFER R2
% CONVERT NUMERIC CODE TO CHARACTER FONT AND SEND TO LED DISPLAT BY SERIAL
BCD_TO_DISP   LDX  #$00           382    A2 00
              LDY  #$20           384    A0 20
              JSR  MOV_REG_1      386    20 00 03     MOVE FROM R0 TO R2
%
              LDA  $2F            389    A5 2F        EXTRACTION R2 TOP BYTE
              BNE  L91            38B    D0 04        IF SIGN = POSITIVE
              LDA  #$0B           38D    A9 0B              SET BLANC TO R3 LEFT END
              BNE  L92            38F    D0 02        IF SIGN = NEGATIVE
L91           LDA  #$0D           391    A9 0C              SET '-' CODE TO R3 LEFT END
L92           STA  $37            393    85 37
%
              LDA  DISP_P0        395    A5 A1
              STA  B_COUNT        397    85 43        INIT. BCD DIGIT COUNTER
L93           LDX  #$04           399    A2 04
              LDY  #$30           39B    A0 30
              LDA  #$FF           39D    A9 FF
              JSR  SHIFT_L_NBIT   39F    20 ED 02     SHIFT LEFT 4BIT R2
              LDX  B_COUNT        3A2    A6 43
              DEX                 3A4    CA
              STX  B_COUNT        3A5    86 43        DO NOT DISPLAY UNTIL 5TH DIGIT
              CPX  #$08           3A7    E0 08
              BCS  L93            3A9    B0 EE
              CPX  #$00           3AB    E0 00        DISPLAY UP TO 2ND DECIMAL PLACE?
              BEQ  DISP_BUFFER_1  3AD    F0 13        DISPLAY R3
              LDA  $2F            3AF    A5 2F        EXTRACTION LEFT END R2
              AND  #$0F           3B1    29 0F
              STA  $2F,X          3B3    95 2F        PUT DISPLAY BUFFER R3
              CPX  DISP_P1        3B5    E4 A2        1'ST INTEGER POINT?
              BNE  L93            3B7    D0 E0
              DEX                 3B9    CA
              LDA  #$0A           3BA    A9 0A        PUT '.'CODE TO R3 3RD BYTE
              STA  $2F,X          3BC    95 2F
              STX  B_COUNT        3CE    86 43        EVACUATE POINTER
              BNE  L93            3C0    D0 D7
%
% DISPLAY R3
% SEND CHARACTER FONT TO LED DISPLAY MODULE
DISP_BUFFER_1 LDA  REG_1          3C2    A5 47
              AND  #$D8           3C4    29 D8
              STA  PORT           3C6    8D 00 40     (DATA=,CLK=,RS=L,*CE=)
```

```
             AND  #$C8           3C9    29 C8
             STA  PORT           3CB    8D 00 40     (DATA=,CLK=,RS=L,*CE=L)
             STA  REG_1          3CE    85 47        EVACUATE CONTROL PATTERN
             LDX  #$07           3D0    A2 07        INIT. BUFFER OFFSET
L94          LDA  BUF_START,X    3D2    B5 30        DISPLAY BUFFER R3 HEAD ADDRESS
             STX  REG_3          3D4    86 43        EVACUATE BUFFER OFFSET
%
% OUTPUT FONT OF ONE CHARACTER
OUT_FONT_1   STA  REG_4          3D6    85 44
             ASL                 3D8    0A           SHIFT LEFT 2BIT (CODE * 4)
             ASL                 3D9    0A
             CLC                 3DA    18
             ADC  REG_4          3DB    65 44        CODE * 5 = CODE(CODE * 4 + CODE)
             TAX                 3DD    AA
             LDY  #$05           3DE    A0 05        INIT. LOOP COUNTER
L95          LDA  FONT_TABLE,X   3E0    B5 60        EXTRACTION FONT DATA
             STY  REG_2          3E2    84 42        EVACUATE LOOP COUNTER
             JSR  OUT_FNT_SERL   3E4    20 A8 00     SERIAL OUTPUT
             INX                 3E7    E8           FONT POINTER +1
             LDY  REG_2          3E8    A4 42        RETURN LOOP POINTER
             DEY                 3EA    88
             BNE  L95            3EB    D0 F3
%
             LDX  REG_3          3ED    A6 43        RETURN BUFFER OFFSET
             DEX                 3EF    CA           OFFSET -1
             BPL  L94            3F0    10 E0        8 CHARACTERS FINISH?
             LDA  #$38           3F2    A9 38
             STA  REG_1          3F4    85 47
             RTS                 3F6    60
%
%-------------------------------------------------------------------------------
% KEY CODE TABLE
% KEY NAME ‘0’ ‘1’ ‘2’ ‘3’ ‘4’ ‘5’ ‘6’ ‘7’ ‘8’ ‘9’ ‘.’ ‘ ‘ ‘-‘ ‘+’ ‘*’ ‘/’ ‘NE’ ‘EN’ ‘AR’
%     CODE $00 $01 $02 $03 $04 $05 $06 $07 $08 $09 $0A $0B $0C $0D $0E $0F $10  $11  $12
%
%      NE: NEGATIVE
%      EN: ENTER
%      AR: ARROW
%-------------------------------------------------------------------------------
% KEY SCAN TABLE
             .ORIGIN  $048
%                          ADDRESS(HEX)  DATA(HEX)

                                 048     00 01 04 07 11 0A 02 00
                                 050     05 08 10 00 03 06 09 00
                                 058     12 0D 0C 0E 0F 00 00 00
%-------------------------------------------------------------------------------
% FONT DATA TABLE
% 5 X 7 DOT MATRIX CHARACTER FONT
             .ORIGIN  $060
%                          ADDRESS(HEX)  DATA(HEX)           CHARACTER

                                 060     3E 51 49 45 3E      ‘0’
                                 065     00 42 7F 40 00      ‘1’
                                 06A     62 51 49 49 46      ‘2’
                                 06F     22 41 49 49 36      ‘3’
                                 074     18 14 12 7F 10      ‘4’
                                 079     27 45 45 45 39      ‘5’
                                 07E     3C 4A 49 49 30      ‘6’
                                 083     01 71 09 05 03      ‘7’
                                 088     36 49 49 49 36      ‘8’
                                 08D     06 49 49 29 1E      ‘9’
                                 092     00 30 30 00 00      ‘.’
                                 097     00 00 00 00 00      ‘ ‘
                                 09C     08 08 08 08 08      ‘-‘

%-------------------------------------------------------------------------------
% END OF PROGRAM
```