31 VSE update

September 1998

In this issue

© Xephon plc 1998

VSE Update

Published by
Xephon
27-35 London Road
Newbury
Berkshire RG14 1JL
England
Telephone: 01635 38342
From USA: 01144 1635 38342
E-mail: xephon@compuserve.com

North American office
Xephon/QNA
1301 West Highway 407, Suite 201-405
Lewisville, TX 75077-2150
USA
Telephone: 940 455 7050

Subscriptions and back-issues
A year's subscription to *VSE Update*, comprising four quarterly issues, costs £100.00 in the UK, $150.00 in the USA and Canada, £106.00 in Europe, £112.00 in Australasia and Japan, and £110.50 elsewhere. In all cases the price includes postage. Individual issues starting with the March 1991 issue, are available separately to subscribers for £25.00 ($37.50) each including postage.

Editorial panel
Articles published in *VSE Update* are reviewed by our panel of experts. Members of the panel include Stanley Stewart (USA), Robert Botsis (USA), and Jesse Joyner (USA).

Editor
Fiona Hewitt

Disclaimer
Readers are cautioned that, although the information in this journal is presented in good faith, neither Xephon nor the organizations or individuals that supplied information in this journal give any warranty or make any representations as to the accuracy of the material it contains. Neither Xephon nor the contributing organizations or individuals accept any liability of any kind howsoever arising out of the use of such material. Readers should satisfy themselves as to the correctness and relevance to their circumstances of all advice, information, code, JCL, EXECs, and other contents of this journal before making any use of it.

Contributions
Articles published in *VSE Update* are paid for at the rate of £170 ($250) per 1000 words for original material. To find out more about contributing an article, without any obligation, please contact us at any of the addresses above and we will send you a copy of our *Notes for Contributors*.

VSE Update on-line
Code from *VSE Update* can be downloaded from our Web site at http://www.xephon.com; you will need the user-id shown on your address label.

© Xephon plc 1998. All rights reserved. None of the text in this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, without the prior permission of the copyright owner. Subscribers are free to copy any code reproduced in this publication for use in their own installations, but may not sell such code or incorporate it in any commercial product. No part of this publication may be used for any form of advertising, sales promotion, or publicity without the written permission of the publisher. Copying permits are available from Xephon in the form of pressure-sensitive labels, for application to individual copies. A pack of 240 labels costs $36 (£24), giving a cost per copy of 15 cents (10 pence). To order, contact Xephon at any of the addresses above.

Printed in England.

Searching a source string against a target string

This subroutine searches a source string (with a maximum length of 70 bytes) against a target string (with a maximum length of 35,565 bytes). It also allows the target string to be replaced.

Four parameters must be passed, while two additional parameters are optional.

FIRST PARAMETER

The first parameter consists of two one-byte fields.

First field

The first field, the direction indicator, determines in which direction the search is to be performed against the target string:

- If ‘F’, indicating a forward direction, the search begins at the value specified in the first field of the fifth parameter. If the fifth parameter isn’t passed or if its value contains binary zero, the search begins at the default value.
- If ‘B’, indicating a backward direction, the search begins at the value specified in the second field of the fifth parameter. If the fifth parameter isn’t passed or if its value contains binary zero, the search begins at the default value.
- If neither ‘F’ nor ‘B’ is specified, ‘F’ is defaulted.

Second field

The second field, the relational operator, determines how the search is to be performed against the source and target strings. Normally, the subroutine searches for an ‘EQUAL’ condition. This field allows the search to be performed based on a ‘NOT EQUAL’, ‘GREATER THAN’ or ‘LESS THAN’ condition. You should specify as follows:

© 1998. Reproduction prohibited. Please inform Xephon of any infringement.

- ‘N’ for not equal
- ‘G’ for greater than
- ‘L’ for less than.

Specifying any other value defaults to ‘E’, indicating equal. Note that if ‘E’ is not specified or defaulted, you must pass the fifth parameter, setting the begin/end values appropriately.

SECOND PARAMETER

The second parameter, the source string, is the 70-byte string that is to be searched for against the target string. If the string is shorter than 70 bytes, it must be left-justified and padded on the right with blanks. If leading/trailing blanks are to be included in the search, the string must be enclosed within quotes. For example, if searching for the string ‘280’, specify ‘‘280’’. The use of leading/trailing quotes reduces the length of the string by two bytes.

This parameter may contain skip characters, specified by a plus (+) sign, indicating that the character(s) in that position is/are always considered to be matched. For example, as long as the target string contains the other characters in the string, the following string:

```
'ASSGN SYS+++,28Ø'
```

will be considered matched regardless what is contained between ‘SYS’ and ‘,280’.

THIRD PARAMETER

The third parameter, the target string, is the area of storage against which the source string is to be searched. This area can range from a minimum of 1 byte up to a maximum of 65,535 bytes. The default length is 80 bytes, and is determined as follows:

- If the fifth parameter isn’t passed, a length of 80 bytes is used.
- If the fifth parameter is passed and the second field contains zero, a length of 80 bytes is used. Otherwise, the non-zero value specified in the fifth parameter determines the length.

© 1998. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.

The ‘length’ mentioned here doesn’t imply the length of the field; by convention, a ‘CALL’ simply provides a four-byte address to this program, which points to the first byte of the field.

FOURTH PARAMETER

The fourth parameter, the return code, is a one-byte field, and contains one of the following values on return to the calling program:

- ‘0’ – the source string wasn’t found within the target string.
- ‘1’ – the source string was found within the target string. If the sixth parameter, the replacement parameter, was also passed, the target string will contain the replacement string beginning in the position indicated or defaulted, for the number of positions indicated or defaulted.
- ‘2’– the replacement starting position (the first field of the sixth parameter, either specified or defaulted) is greater than the value of the ending position (the second field of the fifth parameter, either specified or defaulted). This return code applies only if the sixth parameter was passed.
- ‘3’ – the replacement starting position (specified in the first field of the sixth parameter) plus the replacement number of positions (specified in the second field of the sixth parameter) extends beyond the ending position specified in the second field of the fourth parameter. This return code applies only if the sixth parameter was passed.
- ‘4’ – the starting position (specified in the first field of the fifth parameter) is greater than the ending position (specified in the second field of the fifth parameter). This return code applies only if the fifth parameter was passed.
- ‘5’ – the total number of bytes to be searched is greater than the ending position minus the starting position. You cannot search a 15-byte string starting in position 70, or a 15-byte string ending in position 10. This return code applies only if the fifth parameter was passed.

© 1998. Reproduction prohibited. Please inform Xephon of any infringement.

- ‘6’ – the starting or ending positions are negative. This return code applies only if the fifth parameter was passed.
- ‘7’ – the number of positions (specified in the second field of the sixth parameter) was greater than 70 (ie. X'0046') and wasn’t low-value or high-value. This return code applies only if the sixth parameter was passed.
- ‘8’ – too few or too many parameters were passed. The minimum number of parameters that can be passed is four. The maximum number is six.
- ‘9’ – either the source string (specified in the second parameter) was blank, or a leading quote was specified without a trailing quote.

The return code need not be cleared before each call.

FIFTH PARAMETER

The fifth parameter is optional, and consists of four fields.

First field

The first field, the starting position, determines where in the target string the search is to begin. It must be specified in binary format (ie PIC 9(4) COMP FOR COBOL). If this parameter isn’t passed, the default is 1 (ie X'0001'). If passed and it is zero, 1 (ie X'0001') is defaulted.

Second field

The second field, the ending position, determines where in the target string the search is to end. It must be specified in binary format (ie PIC 9(4) COMP FOR COBOL). If this parameter isn’t passed, the default is 80 (ie X'0050'). If passed and it is zero, 80 (ie X'0050') is defaulted.

Third field

On return to the calling program, the third field, the source number of positions, contains the number of positions calculated in the source

© 1998. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.

string (the second parameter) in unpacked decimal format (ie PIC 99 FOR COBOL). Leading/trailing quotes, if specified, are not included.

Fourth field

If a match between the source and target strings occurs, the fourth field, the target address, contains the four-byte address of where the source string was found in the target string. The address is shown in binary format (ie PIC 9(8) COMP). The use of this field is up to you.

SIXTH PARAMETER

The sixth parameter is optional, and consists of four fields.

First field

The first field, the replacement starting position, determines the place in the target string to which the replacement string is to be moved if a match between the source string and the target string occurs. This field must be specified in binary format (ie PIC 9(4) COMP FOR COBOL).

- If zero, the replacement string will be moved to the target string, beginning at the position at which the source string was found in the target string, for the number of positions specified in the next field.
- If not zero, the replacement string will be moved to the target string at the position specified, for the number of positions specified in the next field.

Second field

The second field, the replacement number of positions, determines the number of positions the replacement string is to be moved in the target string if a match between the search string and target string occurs. This field must be specified in binary format (ie PIC 9(4) COMP FOR COBOL).

- If zero, the replacement string will be moved to the target string

© 1998. Reproduction prohibited. Please inform Xephon of any infringement.

for the number of positions calculated in the source string, excluding beginning/ending quotes, if any.

- If high-value, the replacement string will be moved to the target string for the number of positions calculated in the replacement string, including beginning/ending quotes, if any.
- If the replacement string contains all spaces, 70 bytes of spaces are moved to the target string.
- If not zero or high-value, the replacement string will be moved to the target string for the number of positions specified.

Third field

On return to the calling program, the third field, the replacement number of positions, contains the number of positions calculated in the replacement string (the next field), in binary format (ie PIC S9(4) COMP FOR COBOL). If leading or trailing quotes were specified in the replacement string, they are included.

Fourth field

The fourth field, the replacement string, beginning at its first position, is moved to the target string, beginning with the value in the first field of this parameter, for the number of positions determined by the value in the second field of this parameter.

NOTES

1 You can't pass the sixth parameter without passing the fifth parameter.

2 You should always pass the fifth parameter and specify the starting/ending positions. This improves performance.

3 There is no particular performance advantage to searching forwards or backwards unless the target string is always found beyond the middle. In this case, you would improve performance by specifying beginning and ending positions.

© 1998. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.

4 The actual data format of the source and target strings (ie character or hexadecimal) must be the same, otherwise no match will occur. In other words, if the target string contains hexadecimal data, the source string must also contain hexadecimal data. This subroutine does not convert the contents of either string from one data format to another.

5 The actual data format of the replacement string, if specified, is moved to the target string. Thus, if the replacement string contains hexadecimal data, the target string will contain hexadecimal data.

6 You can request that the starting/ending positions be returned to the calling program so that you can alter these fields before the next call, by entering high-values (X'FF') into the return code just before each call. If the string you were looking for was found, the starting/ending positions are returned in their respective fields; if the string is not found, these fields are not returned. If you request that the starting/ending positions be returned but don't pass the fifth parameter, these fields will not be returned.

7 When a match occurs between the source string and the target string, you can continue to search the target string, to find all possible matches, by doing the following:

- If searching forwards, use the ending position, plus one, as the new starting postion, and the same ending position used in the original call. You will of course have to save the ending position before each call.
- If searching backwards, use the starting position, minus one, as the new ending position and the same starting position used in the original call. You will of course have to save the starting position before each call.

Note that you should be very careful when manipulating either the starting or ending positions, expecially if the replacement parameter is passed.

8 When the sixth parameter (the replacement parameter) is passed,

© 1998. Reproduction prohibited. Please inform Xephon of any infringement.

and you indicate that the replacement string is to begin at a specific position in the target string by specifying a non-zero value in the first field of the sixth parameter, you must ensure that the ending position (specified in the second field of the second parameter) is accurate. Although protection is provided to ensure that the replacement string doesn't go beyond the target string, as specified by the ending position, there is no way of knowing the actual length of the target string.

For example, suppose that the actual length of the target string is 50 bytes. Suppose that you fail to set this value in the number of positions field (ie it contains binary zero, which defaults to 80 (ie X'0050'), or that you set it to a value greater than 50. You also, inadvertently, indicate (in the first field of the sixth parameter) that the replacement string is to start at a position greater than the actual target string. You then call this subroutine, which finds a match, and the replacement string is moved to the target string. Although this may sound good, what in fact happens is that, because the source string is found in the target string beyond its actual length, you just overlay a portion of storage up to 30 bytes, beyond the actual target string.

Storage overlay can also occur even if you don't inadvertently specify the starting position, but rather set it to zero (ie low-value). In this case, the subroutine will move the replacement string to the target string where it found the match. This would occur if, by chance, the search string was found within the 30 bytes beyond the actual 50 bytes of the target string.

CALLING SEQUENCES

The calling sequences are as follows.

COBOL

```
CALL 'DPSRCH' USING OPTN, SOURCE, TARGET, RTNCDE.
```

or

```
CALL 'DPSRCH' USING OPTN, SOURCE, TARGET, RTNCDE,
                    STSESN.
```


© 1998. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.

or

```
     CALL 'DPSRCH' USING OPTN, SOURCE, TARGET, RTNCDE,
                         STSESN, RSRCRN.
```

ALC

```
     LA    13,SAVEAREA (13 CAN ALSO BE R13 OR RD).
                   CALL  DPSRCH,(OPTN,SOURCE,TARGET,RTNCDE)
```

or

```
                   CALL  DPSRCH,(OPTN,SOURCE,TARGET,RTNCDE,STSESN)
```

or

```
                   CALL  DPSRCH,(OPTN,SOURCE,TARGET,RTNCDE,STSESN,  X
                         RSRCRN,)
                   .
                   .  (MAINLINE PART OF PROGRAM).
                   .
            SAVEAREA DC    18F'0'
```

RPGII

```
     CALL 'DPSRCH'
                   PARM           OPTN
                   PARM           SOURCE
                   PARM           TARGET
                   PARM           RTNCDE
```

or

```
                   PARM           STSESN
```

or

```
                   PARM           RSRCRN
```

An 18-word save area must be passed through register 13 by the user (STD COBOL LINKAGE).

DPSRCH

```
DPSR     TITLE 'DPSRCH - 1.0 - SEARCH STRING SUBROUTINE.'
*
DPSRCH   CSECT
DPSRCH   AMODE 31
```

© 1998. Reproduction prohibited. Please inform Xephon of any infringement.

```
DPSRCH   RMODE ANY
         BALR  15,Ø
         USING *,15
         B     SRCHBEG             BRANCH TO SRCHBEG.
*
         DC    C'DPSRCH STARTS HERE. ' INSERT EYE CATCHER.
         DC    CL8'&SYSDATE'                                          <--
         DC    CL8'&SYSTIME'                                          <--
         DS    ØF
*
SRCHBEG  EQU   *
         SAVE  (14,12)
         DROP  15                  DROP TEMPORARY BASE.
         BALR  2,Ø
         USING *,2
         ST    13,SAVEAREA+4
         LA    13,SAVEAREA
         LM    3,6,Ø(1)            GET ADDRESSES OF PARAMETERS.
         ST    5,SVØ5              SVE THIRD PARAMETER.
         SR    8,8                 CLEAR REG 8.
*
SRCHCNT  EQU   *
         TM    Ø(1),X'8Ø'          IS THIS THE END OF THE PARAMETER LIS
         BO    SRCHLST             YES-BRANCH TO SRCHLST.
         LA    8,4(8)              INCREMENT REG 8 BY FOUR (4).
         LA    1,4(1)              INCREMENT REG 1 TO NEXT PARAMETER.
         B     SRCHCNT             BRANCH TO SRCHCNT.
*
SRCHLST  EQU   *
         SR    1,8                 RESTORE REG 1.
         SRL   8,2                 DIVIDE BY TWO (2).
         LA    8,1(8)              BUMP BY ONE FOR 1ST TIME.
         STC   8,NUMPRM            SVE NUMBER OF PARAMETERS PASSED.
         CLI   NUMPRM,X'Ø4'        WERE AT LEAST FOUR (4) PARAMETERS PA
         BL    SRCHPRM             NO-BRANCH TO SRCHPRM.
         CLI   NUMPRM,X'Ø6'        WERE MORE THAN SIX PARAMETER PASSED.
         BH    SRCHPRM             YES-BRANCH TO SRCHPRM.
         MVC   FORBAK(2),Ø(3)      MVE SEARCH 'DIR' & 'NOT' INDICATORS.
         MVI   SRCHCMP+7,X'8Ø'     SET BRANCH 'EQUAL'.
         CLI   SNOT,C'N'           IS SEARCH FOR 'NOT EQUAL'.
         BE    *+12                YES-SKIP NEXT TWO (2) INST.
         CLI   SNOT,C'Y'           IS SEARCH FOR 'NOT EQUAL'.
         BNE   SRCHLST1            NO-BRANCH TO SRCHLST1.
         MVI   SRCHCMP+7,X'7Ø'     SET BRANCH 'NOT EQUAL'.
         B     SRCHLST3            BRANCH TO SRCHLST3.
*
SRCHLST1 EQU   *
         CLI   SNOT,C'G'           IS SEARCH FOR 'GREATER THAN'.
         BNE   SRCHLST2            NO-BRANCH TO SRCHLST2.
```


© 1998. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.

```
         MVI   SRCHCMP+7,X'4Ø'     SET BRANCH 'LESS THAN'.
         B     SRCHLST3            BRANCH TO SRCHLST3.
*
SRCHLST2 EQU   *
         CLI   SNOT,C'L'           IS SEARCH FOR 'LESS THAN'.
         BNE   SRCHLST3            NO-BRANCH TO SRCHLST3.
         MVI   SRCHCMP+7,X'2Ø'     SET BRANCH 'GREATER THAN'.
         B     SRCHLST3            BRANCH TO SRCHLST3.
*
SRCHLST3 EQU   *
         CLI   FORBAK,C'B'         IS SEARCH DIRECTION BACKWARD.
         BE    SRCHLST5            YES-BRANCH TO SRCHLST5.
         MVI   FORBAK,C'F'         FORCE DIRECTION TO FORWARD.
*
SRCHLST5 EQU   *
         MVC   SOURCE,Ø(4)         MVE SOURCE.
         MVC   RTNCDE,Ø(6)         MVE RETURN CODE.
         MVC   STREND,COØ18Ø       SET STARTING/ENDING POSITION DEFAULT
         LA    7,STREND            LOAD ADDRESS OF DUMMY FIFTH PARAMETE
         ST    7,SVØ7              SVE IT.
         CLI   NUMPRM,X'Ø6'        WAS SIXTH PARAMETER PASSED.
         BNE   SRCHLST6            NO-BRANCH TO SRCHLST6.
         L     8,2Ø(1)             LOAD ADDRESS OF SIXTH PARAMETER.
         ST    8,SVØ8A             SVE IT.
         B     SRCHLST7            BRANCH TO SRCHLST7.
*
SRCHLST6 EQU   *
         CLI   NUMPRM,X'Ø5'        WAS FIFTH PARAMETER PASSED.
         BNE   SRCHSKP             NO-BRANCH TO SRCHSKP.
*
SRCHLST7 EQU   *
         L     7,16(1)             LOAD ADDRESS OF STARTING/ENDING/NUMB
         ST    7,SVØ7              SVE IT.
         XC    ADDR,ADDR           CLEAR ADDRESS OF WHERE TARGET FOUND.
         MVC   STREND(L'STREND+2),Ø(7) MVE STARTING/ENDING/NUMBER OF BY
         CLC   =X'ØØØØ',STREND     IS STARTING POSITION ZERO.
         BNE   SRCHLST9            NO-BRANCH TO SRCHLST9.
         MVC   STREND(2),COØ18Ø    FORCE IT TO X'ØØØ1'.
*
SRCHLST9 EQU   *
         CLC   =X'ØØØØ',STREND+2   IS ENDING POSITION ZERO.
         BNE   SRCHSKP             NO-BRANCH TO SRCHSKP.
         MVC   STREND+2(2),COØ18Ø+2 FORCE IT TO X'ØØ5Ø'.
*
SRCHSKP  EQU   *
         MVC   STRENDO,STREND      SVE STARTING/ENDING POSITIONS.
         SR    11,11               CLEAR REG 11.
         ICM   11,3,Ø(7)           INSERT STARTING POSITION TO REG 11.
         ST    11,SV11             SVE IT.
```

© 1998. Reproduction prohibited. Please inform Xephon of any infringement.

```
         LTR   11,11               IS IT NEGATIVE.
         BP    SRCHSKP3            NO-BRANCH TO SRCHSKP3.
*
SRCHSKP1 EQU   *
         MVI   RTNCDE,C'6'         INDICATE STARTING POSITION NEGATIVE.
         B     SRCHRTN             BRANCH TO SRCHRTN.
*
SRCHSKP3 EQU   *
         BCTR  11,Ø                REDUCE STARTING POSITION BY ONE (1)
         SR    12,12               CLEAR REG 12.
         ICM   12,3,2(7)           INSERT ENDING POSITION TO REG 12.
         ST    12,SV12             SVE IT.
         LTR   12,12               IS IT NEGATIVE.
         BNP   SRCHSKP1            YES-BRANCH TO SRCHSKP1.
         CR    12,11               IS ENDING POSITION LOWER THAN STARTI
         BL    SRCHELS             YES-BRANCH TO SRCHELS.
         LA    8,SOURCE            LOAD ADDRESS OF SOURCE TO REG 8.
         ST    8,SVØ8B             SVE IT.
         MVI   SRCHNXT+1,C' '      SET TO LOOK FOR TRAILING NON-BLANK C
         MVI   SRCHNXT+5,X'7Ø'     SET BRANCH 'NOT EQUAL'.
         LA    9,L'SOURCE          LOAD LENGTH OF SOURCE TO REG 9.
         CLI   SOURCE,C''''        DOES SOURCE START WITH QUOTE (').
         BNE   SRCHSKP5            NO-BRANCH TO SRCHSKP5.
         LA    8,SOURCE+1          LOAD ADDRESS OF SOURCE+1 TO REG 8.
         ST    8,SVØ8B             SVE IT.
         MVI   SRCHNXT+1,C''''     SET TO LOOK FOR TRAILING QUOTE.
         MVI   SRCHNXT+5,X'8Ø'     SET BRANCH 'EQUAL'.
         LA    9,L'SOURCE-2        LOAD LENGTH OF SOURCE-2 TO REG 9.
*
SRCHSKP5 EQU   *
         LA    8,SOURCE+L'SOURCE-1 LOAD BACK END OF SOURCE TO REG 8.
*
* WARNING: DON'T ADD ANY INSTRUCTIONS AFTER THE LABEL WITHOUT
* CHANGING THE MVI INSTRUCTIONS REFERENCING SRCHNXT+1 AND SRCHNXT+5.
*
SRCHNXT  EQU   *
         CLI   Ø(8),C' '           IS THIS POSITION BLANK/QUOTE.
         BNE   SRCHFND             NO/YES-BRANCH TO SRCHFND.
         BCTR  8,Ø                 REDUCE REG 8 BY ONE (1).
         BCTR  9,Ø                 REDUCE REG 9 BY ONE (1).
         C     8,SVØ8B             ARE WE BELOW BEGINNING OF SOURCE.
         BNL   SRCHNXT             NO-BRANCH TO SRCHNXT.
         MVI   RTNCDE,C'9'         INDICATE SOURCE BLANK/NO TRAILING QU
         B     SRCHRTN             BRANCH TO SRCHRTN.
*
SRCHFND  EQU   *
         ST    8,SVØ8C             SVE CONTENTS OF REG 8.
         ST    9,SVØ9A             SVE CONTENTS OF REG 9.
         CVD   9,DOUBWORD          CONVERT NUMBER OF BYTES IN SOURCE.
         MVC   SLNGTHP,DOUBWORD+6  MVE IT TO SLNGTHP.
```


© 1998. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.

```
         OI    SLNGTHP+L'SLNGTHP-1,X'ØF' ENSURE GOOD SIGN.
         UNPK  SLNGTHC,SLNGTHP     UNPACK IT.
         BCTR  9,Ø                 REDUCE REG 9 BY ONE (1) FOR CLC.
         ST    9,SVØ9B             SVE CONTENTS OF REG 9.
         MVC   SRCHCMP+1(1),SVØ9B+3 ALTER LENGTH OF CLC INST AT SRCHCMP
         LR    1Ø,12               LOAD ENDING POSTION TO REG 1Ø.
         SR    1Ø,11               REDUCE IT BY STARTING POSITION.
         ST    1Ø,SV1Ø             SVE IT.
         CR    1Ø,9                IS RESULT GREATER THAN NUMBER OF BYT
         BH    SRCHSET             YES-BRANCH TO SRCHSET.
         MVI   RTNCDE,C'5'         INDICATE NUMBER OF BYTES GREATER THA
         B     SRCHRTN             BRANCH TO SRCHRTN.
*
SRCHSET  EQU   *
         MVC   COUNT,=F'1'         SET COUNT.
         LA    13,Ø(5)             LOAD ADDRESS OF TARGET TO REG 13.
         AR    13,11               ADD STARTING POSITION TO IT.
         ST    13,SV13A            SVE IT.
         LR    4,12                LOAD ENDING POSITION TO REG 4.
         SR    4,11                SUBTRACT STARTING POSITION FROM IT.
         ST    4,SVØ4              SVE IT.
         LA    14,Ø(5)             LOAD ADDRESS OF TARGET TO REG 14.
         AR    14,12               ADD ENDING POSITION TO IT.
         ST    14,SV14             SVE IT.
         BAL   14,SRCHPLS          PERFORM SRCHPLS ROUTINE.
         L     14,SV14             RESTORE REG 14.
         SR    14,9                SUBTRACT NUMBER OF BYTES IN SOURCE F
         ST    14,SV14             SVE IT.
         L     8,SVØ8B             LOAD START OF SOURCE TO REG 8.
         CLI   FORBAK,C'F'         IS DIRECTION FORWARD.
         BE    SRCHCMP             YES-BRANCH TO SRCHCMP.
         LA    13,Ø(5)             LOAD ADDRESS OF TARGET TO REG 13.
         AR    13,12               ADD ENDING POSITION TO IT.
         SR    13,9                SUBTRACT NUMBER OF BYTES IN SOURCE F
         BCTR  13,Ø                REDUCE IT BY ONE (1).
*
* WARNING: DON'T ADD ANY INSTRUCTIONS AFTER THE LABEL WITHOUT
* CHANGING THE MVI INSTRUCTIONS REFERENCING SRCHCMP+1 AND SRCHCMP+7.
*
SRCHCMP  EQU   *
         CLC   Ø(1,8),Ø(13)        IS SOURCE FOUND/NOT FOUND IN TARGET.
         BE    SRCHHIT             EQ/NE/LT/GT-BRANCH TO SRCHHIT.
         L     15,COUNT            LOAD INTERNAL SEARCH COUNTER TO REG
         LA    15,1(15)            INCREMENT IT BY ONE (1).
         ST    15,COUNT            SVE IT.
         CLI   FORBAK,C'F'         IS DIRECTION FORWARD.
         BE    SRCHCMP5            YES-BRANCH TO SRCHCMP5.
         BCTR  13,Ø                REDUCE REG 13 TO NEXT POSITION.
         CR    13,14               ARE WE DONE.
         BNL   SRCHNOH             YES-BRANCH TO SRCHNOH.
```

© 1998. Reproduction prohibited. Please inform Xephon of any infringement.

```
         C     13,SV13A            ARE WE DONE.
         BL    SRCHNOH             YES-BRANCH TO SRCHNOH.
         B     SRCHCMP             BRANCH TO SRCHCMP.
*
SRCHCMP5 EQU   *
         LA    13,1(13)            INCREMENT REG 13 TO NEXT POSITION.
         CR    13,14               ARE WE DONE.
         BL    SRCHCMP             NO-BRANCH TO SRCHCMP.
*
SRCHNOH  EQU   *
         MVI   RTNCDE,C'Ø'         INDICATE TARGET NOT FOUND.
         B     SRCHRTN             BRANCH TO SRCHRTN.
*
SRCHPRM  EQU   *
         MVI   RTNCDE,C'8'         INDICATE TOO FEW/MANY PARAMETERS PAS
         B     SRCHRTN             BRANCH TO SRCHRTN.
*
SRCHHIT  EQU   *
         ST    13,SV13B            SVE ADDRESS OF TARGET FOUND.
         LA    14,Ø(5)             LOAD ADDRESS OF TARGET TO REG 14.
         LR    15,13               LOAD STARTING POSITION OF TARGET FOU
         SR    15,14               REDUCE IT BY START OF TARGET.
         LA    15,1(15)            ADJUST IT BY ONE (1).
         ST    15,SV15             SVE STARTING POSITION OF TARGET FOUN
         AR    9,15                ADD NUMBER OF BYTES TO STARTING POSI
         ST    9,SVØ9C             SVE ENDING POSITION OF TARGET FOUND.
         MVC   STRENDS,STRENDC     SVE STARTING/ENDING POSITIONS.
         CLI   NUMPRM,X'Ø6'        WERE SIX (6) PARAMETERS PASSED.
         BE    SRCHHIT1            YES-BRANCH TO SRCHHIT1.
         CLI   NUMPRM,X'Ø5'        WERE FIVE (5) PARAMETERS PASSED.
         BNE   SRCHHIT2            NO-BRANCH TO SRCHHIT2.
*
SRCHHIT1 EQU   *
         CLI   RTNCDE,X'FF'        ARE WE TO RETURN THE STARTING/ENDING
         BNE   SRCHHIT2            NO-BRANCH TO SRCHHIT2.
         MVC   STREND(2),SV15+2    MVE STARTING POSITION OF TARGET FOUN
         MVC   STREND+2(2),SVØ9C+2 MVE ENDING POSITION OF TARGET FOUND.
         MVC   Ø(L'STREND,7),STREND MVE STARTING/ENDING POSITIONS.
*
SRCHHIT2 EQU   *
         MVI   RTNCDE,C'1'         INDICATE TARGET FOUND.
         CLI   NUMPRM,X'Ø6'        WERE SIX (6) PARAMETERS PASSED.
         BNE   SRCHRTN             NO-BRANCH TO SRCHRTN.
         L     8,SVØ8A             LOAD ADDRESS OF SIXTH PARAMETER.
         MVC   REPPRM,Ø(8)         SVE IT.
         CLC   REPSTR,=X'ØØØØ'     DO WE REPLACE TARGET AT SPECIFIC POS
         BE    SRCHHIT4            NO-BRANCH TO SRCHHIT4.
         CLC   REPSTR,ENDO         IS REPLACEMENT POSITION GREATER THA
         BNH   SRCHHIT3            NO-BRANCH TO SRCHHIT3.
```


© 1998. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.

```
         MVI   RTNCDE,C'2'         INDICATE REPLACEMENT POSITION GREATE
         B     SRCHRTN             BRANCH TO SRCHRTN.
*
SRCHHIT3 EQU   *
         SR    15,15               CLEAR REG 15.
         ICM   15,3,REPSTR         INSERT STARTING POSITION TO REG 15.
         LA    13,Ø(5)             LOAD ADDRESS OF TARGET TO REG 13.
         AR    13,15               ADD EM TOGETHER.
*
SRCHHIT4 EQU   *
         SR    11,11               CLEAR REG 11.
         ICM   11,3,REPNCL         INSERT NUMBER OF POSITIONS TO REG 11
         BNZ   SRCHHIT5            NOT ZERO-BRANCH TO SRCHHIT5.
         L     11,SVØ9A            LOAD NUMBER OF BYTES IN SOURCE STRIN
*
SRCHHIT5 EQU   *
         C     11,=F'65535'        DO WE USE REPLACEMENT LENGTH.
         BNE   SRCHHIT8            NO-BRANCH TO SRCHHIT8.
         LA    8,REPDAT+L'REPDAT-1 LOAD BACK END OF REPLACEMENT TO REG
         LA    11,L'REPDAT         LOAD LENGTH OF REPLACEMENT TO REG 11
*
SRCHHIT6 EQU   *
         CLI   Ø(8),C' '           IS THIS POSITION BLANK.
         BNE   SRCHHIT7            YES-BRANCH TO SRCHHIT7.
         BCTR  11,Ø                REDUCE REG 11 BY ONE (1).
         BCTR  8,Ø                 REDUCE REG 8 BY ONE (1).
         LTR   11,11               ARE WE DONE.
         BNZ   SRCHHIT6            NO-BRANCH TO SRCHHIT6.
         LA    11,L'REPDAT         LOAD LENGTH OF REPLACEMENT TO REG 11
         B     SRCHHIT8            BRANCH TO SRCHHIT8.
*
SRCHHIT7 EQU   *
         LA    1Ø,REPDAT           LOAD START OF REPLACEMENT TO REG 1Ø.
         SR    8,1Ø                CALCULATE REPLACEMENT LENGTH.
         LR    11,8                LOAD IT TO REG 11.
         LA    11,1(11)            ADJUST IT BY ONE (1).
*
SRCHHIT8 EQU   *
         CH    11,=H'7Ø'           IS NUMBER OF POSITIONS GREATER THAN
         BH    SRCHNCE             YES-BRANCH TO SRCHNCE.
         LR    1Ø,11               LOAD NUMBER OF REPLACEMENT POSITIONS
         AR    1Ø,13               ADD REPLACEMENT STARTING POSITION TO
         C     1Ø,SV14             ARE WE ABOVE END OF TARGET.
         BNH   SRCHHIT9            NO-BRANCH TO SRCHHIT9.
         MVI   RTNCDE,C'3'         INDICATE REPLACEMENT POSITION + STAR
         B     SRCHRTN             BRANCH TO SRCHRTN.
*
SRCHHIT9 EQU   *
         STCM  11,3,REPNUM         SVE NUMBER OF POSITIONS IN REPLACEME
         BCTR  11,Ø                MAKE IT MVC LENGTH.
```

© 1998. Reproduction prohibited. Please inform Xephon of any infringement.

```
         EX    11,SRCHMOVE         EXECUTE MVC AT SRCHMOVE.
         B     SRCHRTN             BRANCH TO SRCHRTN.
*
SRCHMOVE EQU   *
         MVC   Ø(Ø,13),REPDAT      MVE REPLACEMENT TO TARGET.
*
SRCHRTN  EQU   *
         CLI   NUMPRM,X'Ø6'        WAS SIXTH PARAMETER PASSED.
         BE    SRCHRTN3            YES-BRANCH TO SRCHRTN3.
         CLI   NUMPRM,X'Ø5'        WAS FIFTH PARAMETER PASSED.
         BNE   SRCHRTN5            NO-BRANCH TO SRCHRTN5.
*
SRCHRTN3 EQU   *
         CLI   NUMPRM,X'Ø6'        WAS SIXTH PARAMETER PASSED.
         BNE   SRCHRTN4            NO-BRANCH TO SRCHRTN4.
         L     8,SVØ8A             LOAD ADDRESS OF SIXTH PARAMETER.
         MVC   4(L'REPNUM,8),REPNUM MVE REPLACEMENT NUMBER.
*
SRCHRTN4 EQU   *
         MVC   4(L'SLNGTHC,7),SLNGTHC MVE NUMBER OF BYTES IN SOURCE.
         MVC   6(L'ADDR,7),SV13B   MVE ADDRESS OF WHERE TARGET FOUND.
*
SRCHRTN5 EQU   *
         MVC   Ø(L'RTNCDE,6),RTNCDE MVE RETURN CODE.
*        PDUMP DPSRCHS,DPSRCHE     FOR DEBUGGING.
         SR    15,15               CLEAR REG 15.
         L     13,SAVEAREA+4
         RETURN (14,12),RC=(15)
*
SRCHELS  EQU   *
         MVI   RTNCDE,C'4'         INDICATE ENDING POSITION LOWER THAN
         B     SRCHRTN             BRANCH TO SRCHRTN.
*
SRCHNCE  EQU   *
         MVI   RTNCDE,C'7'         INDICATE NUMBER OF POSITIONS ERROR.
         B     SRCHRTN             BRANCH TO SRCHRTN.
*
SRCHPLS  EQU   *
         MVI   MTCSW,C'Ø'          INDICATE NO MATCH FOUND.
         MVI   FSTSW,C'Ø'          INDICATE NOT FIRST TIME.
         L     8,SVØ8B             LOAD START OF SOURCE TO REG 8.
*
SRCHPLSØ EQU   *
         CLI   Ø(8),C'+'           DOES SOURCE CONTAIN A PLUS (+) SIGN.
         BE    SRCHPLS1            YES-BRANCH TO SRCHPLS1.
         LA    8,1(8)              INCREMENT REG 8 TO NEXT POSITION.
         C     8,SVØ8C             ARE WE AT THE END OF SOURCE.
         BHR   14                  YES-RETURN TO CALLER.
         B     SRCHPLSØ            BRANCH TO SRCHPLSØ.
*
```


© 1998. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.

```
SRCHPLS1 EQU   *
         L     8,SV08B             LOAD START OF SOURCE TO REG 8.
         L     9,SV09A             LOAD NUMBER OF BYTES IN SOURCE TO RE
         MVI   SRCHPLS3+15,X'80'   SET BRANCH 'EQUAL'.
         CLI   SNOT,C'N'           IS SEARCH FOR 'NOT EQUAL'.
         BE    *+12                YES-SKIP NEXT TWO (2) INST.
         CLI   SNOT,C'Y'           IS SEARCH FOR 'NOT EQUAL'.
         BNE   *+12                NO-SKIP NEXT TWO (2) INST.
         MVI   SRCHPLS3+15,X'70'   SET BRANCH 'NOT EQUAL'.
         B     SRCHPLS2            BRANCH TO SRCHPLS2.
         CLI   SNOT,C'G'           IS SEARCH FOR 'GREATER THAN'.
         BNE   *+12                NO-SKIP NEXT TWO (2) INST.
         MVI   SRCHPLS3+15,X'40'   SET BRANCH 'LESS THAN'.
         B     SRCHPLS2            BRANCH TO SRCHPLS2.
         CLI   SNOT,C'L'           IS SEARCH FOR 'LESS THAN'.
         BNE   SRCHPLS2            NO-SKIP NEXT TWO (2) INST.
         MVI   SRCHPLS3+15,X'20'   SET BRANCH 'GREATER THAN'.
*
SRCHPLS2 EQU   *
         CLI   FORBAK,C'B'         IS SEARCH DIRECTION BACKWARD.
         BNE   SRCHPLS3            NO-BRANCH TO SRCHPLS3.
         L     8,SV08C             LOAD END OF SOURCE TO REG 8.
         LA    13,0(5)             LOAD ADDRESS OF TARGET TO REG 13.
         AR    13,12               ADD ENDING POSITION TO IT.
         BCTR  13,0                REDUCE IT BY ONE (1).
*
* WARNING: DON'T ADD ANY INSTRUCTIONS AFTER THE LABEL WITHOUT
* CHANGING THE MVI INSTRUCTIONS ABOVE REFERENCING SRCHPLS3+15.
*
SRCHPLS3 EQU   *
         CLI   0(8),C'+'           DO WE MATCH THIS POSITION.
         BE    SRCHPLS7            NO-BRANCH TO SRCHPLS7.
         CLC   0(1,8),0(13)        DOES THIS POSITION MATCH.
         BE    SRCHPLS7            YES-BRANCH TO SRCHPLS7.
         MVI   MTCSW,C'0'          INDICATE NO MATCH
         MVI   FSTSW,C'0'          INDICATE NOT FIRST TIME.
         L     8,SV08B             LOAD START OF SOURCE TO REG 8.
         CLI   FORBAK,C'B'         IS SEARCH DIRECTION BACKWARD.
         BNE   *+8                 NO-SKIP NEXT INST.
         L     8,SV08C             LOAD END OF SOURCE TO REG 8.
         L     9,SV09A             LOAD NUMBER OF BYTES IN SOURCE TO RE
         B     SRCHPLS5            BRANCH TO SRCHPLS5.
*
SRCHPLS4 EQU   *
         CLI   FORBAK,C'B'         IS SEARCH DIRECTION BACKWARD.
         BE    *+12                NO-SKIP NEXT TWO (2) INST.
         LA    8,1(8)              INCREMENT REG 8 TO NEXT POSITION.
         B     SRCHPLS5            BRANCH TO SRCHPLS5.
         BCTR  8,0                 DECREMENT REG 8 TO NEXT POSITION.
*
```

© 1998. Reproduction prohibited. Please inform Xephon of any infringement.

```
SRCHPLS5 EQU   *
         CLI   FORBAK,C'B'         IS SEARCH DIRECTION BACKWARD.
         BE    SRCHPLS6            YES-BRANCH TO SRCHPLS6.
         LA    13,1(13)            INCREMENT REG 13 TO NEXT POSITION.
         C     13,SV14             ARE WE DONE.
         BH    SRCHNOH             YES-BRANCH TO SRCHNOH.
         B     SRCHPLS3            BRANCH TO SRCHPLS3.
*
SRCHPLS6 EQU   *
         BCTR  13,Ø                DECREMENT REG 13 TO NEXT POSITION.
         C     13,SV13A            ARE WE DONE.
         BL    SRCHNOH             YES-BRANCH TO SRCHNOH.
         B     SRCHPLS3            BRANCH TO SRCHCMP.
*
SRCHPLS7 EQU   *
         CLI   FSTSW,C'1'          FIRST TIME.
         BE    SRCHPLS9            NO-BRANCH TO SRCHPLS9.
         LR    14,13               SVE START OF POSITION FOUND.
         MVI   FSTSW,C'1'          INDICATE NOT FIRST TIME.
*
SRCHPLS9 EQU   *
         L     15,COUNT            LOAD INTERNAL SEARCH COUNTER TO REG
         LA    15,1(15)            INCREMENT IT BY ONE (1).
         ST    15,COUNT            SVE IT.
         MVI   MTCSW,C'1'          INDICATE MATCH.
         BCTR  9,Ø                 REDUCE NUMBER OF SOURCE BYTES BY ONE
         LTR   9,9                 ARE WE DONE.
         BNZ   SRCHPLS4            NO-BRANCH TO SRCHPLS4.
         CLI   MTCSW,C'1'          DID WE MATCH SOURCE.
         BNE   SRCHNOH             NO-BRANCH TO SRCHNOH.
         ST    13,SV13C
         L     9,SVØ9B
         LR    13,14
         CLI   FORBAK,C'F'         IS SEARCH DIRECTION FORWARD.
         BE    SRCHHIT             YES-BRANCH TO SRCHHIT.
         L     13,SV13C
         B     SRCHHIT             BRANCH TO SRCHHIT.
*
DPSRCHS  DC    C'DPSRCH STORAGE HERE. ' INSERT EYE CATCHER.
OPTN     DS    ØCL2
FORBAK   DS    C
SNOT     DS    C
SOURCE   DS    CL7Ø
         DC    C'PM34'
RTNCDE   DS    C
STREND   DS    ØXL4
STR      DS    XL2
END      DS    XL2
SLNGTHC  DS    CL2
ADDR     DS    XL4
```


© 1998. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.

```
         DC    C'PMØ6'
REPPRM   DS    ØCL76
REPSTR   DS    XL2
REPNCL   DS    XL2
REPNUM   DS    XL2
REPDAT   DS    CL7Ø
*
NUMPRM   DC    X'ØØ'               NUMBER OF PARAMETERS PASSED.
SLNGTHP  DS    XL2
STRENDC  DS    ØXL4
STRPOSC  DC    X'ØØØ1'
ENDPOSC  DC    X'ØØ5Ø'
STRENDS  DS    XL4
STRENDO  DS    ØXL4
STRO     DS    XL2
ENDO     DS    XL2
COØ18Ø   DC    X'ØØØ1ØØ5Ø'
MTCSW    DC    C'Ø'
FSTSW    DC    C'Ø'
*
         DC    C'SVXX HERE. '
COUNT    DC    F'Ø' Ø1
SVØ4     DC    F'Ø' Ø2             ENDING POSITION MINUS STARTING POSIT
SVØ5     DC    F'Ø' Ø3             ADDRESS OF BEGIN OF TARGET.
SVØ7     DC    F'Ø' Ø4             ADDRESS OF FIFTH PARAMETER, IF PASSE
SVØ8A    DC    F'Ø' Ø5             ADDRESS OF SIXTH PARAMETER, IF PASSE
SVØ8B    DC    F'Ø' Ø6             ADDRESS OF START OF SOURCE.
SVØ8C    DC    F'Ø' Ø7             ADDRESS OF END OF SOURCE.
SVØ9A    DC    F'Ø' Ø8             NUMBER OF BYTES IN SOURCE.
SVØ9B    DC    F'Ø' Ø9             NUMBER OF BYTES IN SOURCE - 1.
SVØ9C    DC    F'Ø' 1Ø             ENDING POSITION IN HEX.
SV1Ø     DC    F'Ø' 11             END POSITION - BEG POSITION + 1
SV11     DC    F'Ø' 12             STARTING POSITION IN HEX.
SV12     DC    F'Ø' 13             ENDING POSTION IN HEX.
SV13A    DC    F'Ø' 14             ADDRESS OF TARGET PLUS STARTING POSI
SV13B    DC    F'Ø' 15             ADDRESS OF TARGET FOUND.
SV13C    DC    F'Ø' 16             ADDRESS OF TARGET FOUND. (BACKWARDS)
SV14     DC    F'Ø' 17             ADDRESS OF TARGET PLUS ENDING POSITI
SV15     DC    F'Ø' 18             STARTING POSITION OF SOURCE FOUND.
DOUBWORD DS    ØD
         DS    F'Ø'
FULLWORD DS    F'Ø'
SAVEAREA DC    18F'Ø'
*
DPSRCHE  DC    X'FF'
*
         END
```

Bob Botsis
Senior Systems Programmer (USA)
© Xephon 1998

© 1998. Reproduction prohibited. Please inform Xephon of any infringement.

CDLOADing a phase and passing control to it

This subroutine CDLOADs a phase (ie a program or subroutine) into the GETVIS partition, and passes control to it.

The number of parameters passed must be one more than the number of parameters passed by the subroutine to be invoked (as described in the subroutine documentation). The additional parameter must be eight bytes long, and must contain the name of the subroutine to be invoked (ie CDLOADed).

The first byte of the first parameter indicates whether or not the desired subroutine was successfully CDLOADed. It will contain one of the following values:

- X'00' – this indicates that the subroutine was called with only one parameter passed, whereas a minimum of two is required. If the subroutine is called without any parameters being passed, an abend will occur.
- X'02' – this indicates that the phase name to be CDLOADed, passed in the first parameter, was 'DPCALL '.
- X'03' – this indicates that the subroutine was called from within a CICS program and that the phase to be CDLOADed was not DPEIBC or TXER400.
- X'FF' – this indicates that a CDLOAD error occurred. The next four bytes will contain the CDLOAD return code. For the meaning of CDLOAD return codes, consult the *VSE Message Manual*, under 'CDLOAD'. If the first parameter contains spaces, low-values, or high-values, or if the first byte contains either of these values, the entire eight bytes of the first parameter will be returned with high-values.

NOTES

1 The use of this subroutine has at least two benefits:

  – The first is that a CALL statement autolinks the called

© 1998. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.

subroutine(s) (ie OBJ MEMBERTYPE) into the program calling the subroutine. This means that the size of the program calling the subroutine is increased by the size of the subroutine.

– The second is that, should the subroutine need to be changed because of an error or for any other reason, all programs calling the changed subroutine must be recompiled and re-link-edited.

2 It doesn't matter whether either the program that calls the subroutine or the phase to be CDLOADed runs above or below the 16MB line; code has been provided to correctly handle either possibility. However, you should note that no program will run above the 16MB line if there is no storage defined above the 16MB line in the partition. Moreover, any program that is CDLOADed below the line and the program which CDLOADs another program or table above the line will fail if the first program attempts to branch into the second; incorrect results will occur if the second is a table to which the first expects to gain access. In other words, a 24-bit program cannot be expected to handle 31-bit addressing. Finally, note that certain IBM macros will not function above the line and will cancel if they are link-edited in programs or used in progams that are CDLOADed above the line.

3 A special feature of this subroutine enables you to delete previously CDLOADed phases. This means that you can remove phases from the GETVIS partition if, for example, the GETVIS anchor table becomes full. Note that the deletion of a phase from GETVIS is governed by the rules stated by the 'CDDELETE' macro. To use this feature:

– move the name of the phase to be deleted to the first parameter

– move the character constant 'CDDELETE' to the second parameter

– issue a CALL to the subroutine.

© 1998. Reproduction prohibited. Please inform Xephon of any infringement.

Successful deletion is indicated by the first byte of the first parameter, as described above.

4 It is your responsibility to check the first byte of the first parameter on return to the calling program before executing any other instructions, to ensure that all was successful. Note that the contents of this byte indicate not whether or not the invoked subroutine was successful, but rather whether or not it was CDLOADed.

5 This subroutine is normally used to CDLOAD phases that were originally written as subroutines (ie those that perform normal linkage conventions via save/return macros or equivalent instructions). It can also be used to CDLOAD phases not conforming to these rules. However, note that CDLOADing phases that issue 'STOP RUN' for COBOL programs, or the EOJ macro for Assembler programs, will cause a return to the supervisor rather than to the program that called the subroutine.

Note also that tables should not be CDLOADed, since they contain no executable instructions. However, if you wish to do this, a feature has been provided. To use this feature:

– move the name of the phase to be CDLOADed only to the first parameter

– move the character constant 'CDLOAD ' to the second parameter

– issue a call to the subroutine.

Successful CDLOADing is indicated in the first byte of the first parameter, as described above. If successful, the address to which the phase was CDLOADed is returned in the first four bytes of the second parameter.

6 Since this subroutine requires at least two parameters to be passed, you will need to use a dummy parameter in the call if you CDLOAD a subroutine that doesn't require any parameters.

7 Generally, a CDLOAD error will occur for one of two reasons:

© 1998. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.

– GETVIS problems indicated by one of X'00000004', X'00000008', or X'0000000C'.

– ‘REQUESTED PHASE NOT FOUND’, indicated by X'00000014'.

8 This subroutine must not be called from within a program that was CDLOADed by this subroutine.

9 It’s advisable to at least restrict, if not avoid, using this subroutine in CICS programs. This is because CDLOADed phases are shared by others. This could cause problems in CICS programs, especially if the CDLOAded phase is not reenterable or if a subroutine needs to be invoked multiple times to perform a single task.

10 This subroutine has only marginal benefits in COBOL/VSE programs, as COBOL/VSE supports dynamic calls, which is effectively what this subroutine does.

11 The example below shows DOS/COBOL and COBOL/VSE (using a static call) using this subroutine to invoke (ie CDLOAD) the 'DPMODE' subroutine.

```
77  MODE-VALUE              PIC X.
Ø1  SUB-NAME.
    Ø3  SUB-NAME-RTN        PIC X.
    Ø3  FILLER              PIC X(7).
...
MOVE 'DPMODE' TO SUB-NAME.
CALL 'DPCALL' USING SUB-NAME MODE-VALUE.
IF SUB-NAME-RTN NOT = 'D'  <--- 1ST LETTER OF SUBROUTINE
    ... ERROR CODE HERE ...
```

12 The example below shows COBOL/VSE doing the same thing using a dynamic call.

```
77  DPCALL                  PIC X(8) VALUE 'DPCALL'.
...
CALL DPCALL USING SUB-NAME MODE-VALUE.
MOVE Ø TO RETURN-CODE.
IF SUB-NAME-RTN NOT = 'D'  <--- 1ST LETTER OF SUBROUTINE
    ... ERROR CODE HERE ...
```

© 1998. Reproduction prohibited. Please inform Xephon of any infringement.

CALLING SEQUENCES

The calling sequences are as follows.

COBOL

```
     CALL 'DPCALL' USING SUBNAME ...
```

ALC

```
     LA    13,SAVEAREA (13 CAN ALSO BE R13 OR RD).
                       CALL  DPCALL,(SUBNAME,...)
                         .
                         .  (MAINLINE PART OF PROGRAM).
                         .
     SAVEAREA DC    18F'Ø'
```

RPGII

```
     CALL 'DPCALL'
                  PARM           SUBNME
                  PARM           ......
```

An 18-word save area must be passed through register 13 by the user (STD COBOL LINKAGE).

DPCALL

```
DPCA     TITLE 'DPCALL - 1.Ø - CDLOAD A PHASE AND PASS CONTROL TO IT
SUBROUTINE.'
DPCALL   CSECT Ø
DPCALL   AMODE 31
DPCALL   RMODE ANY
         SAVE  (14,12)
         BALR  R3,RØ
         USING *,R3
         ST    RD,SAVEARE1+4
         LA    RD,SAVEARE1
         B     CALLØØ              BRANCH TO CALLØØ.
*
         IPW$EQU ,                 REG EQUATES.
*
DPCALLS  DC    C'DPCALL STARTS HERE. ' INSERT EYE CATCHER.
*
CALLØØ   EQU   *
         MVI   HERE,C'Ø'           INDICATE WE'VE NOT BEEN HERE.
         AP    COUNTØ,=P'1'        INCREMENT TIMES CALLED COUNTER.
         ST    RE,SVRE             SVE REG 14.
```


© 1998. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.

```
         LR    RB,R1               SVE REG 1.
         LTR   R1,R1               WERE ANY PARAMETERS PASSED.
         BNZ   CALLØ3              YES-BRANCH TO CALLØ3.
*
CALLØ1   EQU   *
         L     R2,Ø(R1)            POINT TO NAME OF PHASE TO LOAD.
         MVI   Ø(R2),X'ØØ'         INDICATE PARAMETER ERROR.
*
CALLØ1C  EQU   *
         XC    SVR1B,SVR1B         CLEAR CURRENT ENTRY POINT.
         XC    SVPHASEC,SVPHASEC   CLEAR CURRENT PHASE NAME.
*
CALLØ2   EQU   *
         SR    RF,RF               CLEAR REG 15.
         L     RD,SAVEARE1+4
         RETURN (14,12),RC=(RF)
*
CALLØ3   EQU   *
         L     R5,4(R1)            LOAD SECOND PARAMETER TO REG 5.
         TM    Ø(R1),X'8Ø'         WAS ONLY ONE (1) PARAMETER PASSED.
         BO    CALLØ1              YES-BRANCH TO CALLØ1.
         SR    RF,RF               CLEAR REG 15.
*
* THE CDDELETE MACRO CANNOT BE USED ON PRE VSE/ESA 1.3 SYSTEMS AS IT
* DIDN'T EXIST. IF YOU AREN'T RUNNING VSE/ESA 1.3 OR HIGHER UNCOMMENT
* THE FOLLOWING INSTRUCTION AND COMMENT THE NEXT SEVEN (7) INSTRUC-
* TIONS. DOING SO WILL CAUSE THIS SUBROUTINE TO RETURN AN ERROR
* SHOULD AN ATTEMPT BE MADE TO USE 'CDDELETE'.
*
*        B     CALLØ4
         CLC   =C'CDDELETE',Ø(R5)  IS 'CDDELETE' SPECIFIED.
         BNE   CALLØ5              NO-BRANCH TO CALLØ5.
         L     R2,Ø(R1)            POINT TO NAME OF PHASE TO DELETE.
         CDDELETE (R2)             DELETE PREVIOUSLY CDLOAD'ED PHASE.
         AP    COUNT5,=P'1'        INCREMENT CDDELETE COUNTER.
         LTR   RF,RF               WAS CDDELETE OK.
         BZ    CALLØ2              YES-BRANCH TO CALLØ2.
*
CALLØ4   EQU   *
         ST    RF,SVRF             SVE REG 15.
         MVI   Ø(R2),X'FF'         INDICATE CDDELETE ERROR.
         MVC   1(L'SVRF,R1),SVRF   MVE IT TO PHASE NAME.
         B     CALLØ1C             BRANCH TO CALLØ1C.
*
CALLØ5   EQU   *
         LA    RA,4(R1)            LOAD ADDRESS OF SECOND PARAMETER TO
         L     R2,Ø(R1)            POINT TO NAME OF PHASE TO CDLOAD.
         CLC   =C'DPCALL  ',Ø(R2)  IS PHASE TO BE CDLOAD'ED 'DPCALL  '.
         BE    CALLØ5B             YES-BRANCH TO CALLØ5B.
         ST    R1,SVR1A            SVE CONTENTS OF REG 1.
```

© 1998. Reproduction prohibited. Please inform Xephon of any infringement.

```
*
* THE FOLLOWING INSTRUCTIONS, UP TO LABEL 'CALLØ5B', PREVENT THIS
* SUBROUTINE FROM BEING CALLED FROM WITHIN A CICS PROGRAM, EXCEPT IF
* IT'S CALLED FROM 'DPCICS' OR 'DPSTUB'. IF YOU DON'T CARE IF THIS
* SUBROUTINE IS CALLED FROM WITHIN CICS PROGRAMS THEN UNCOMMENT THE
* FOLLOWING STATEMENT.
*
*        B     CALLØ5C             SKIP THIS ...
         GETFLD FIELD=ICCFPP       GET INTERACTIVE PARTITION FLAG.
         LTR   R1,R1               ARE WE RUNNING IN ICCF PSEUDO
                                    PARTIT
         BNZ   CALLØ5C             YES-BRANCH TO CALLØ5C.
         COMRG ,                   GET COMMUNICATIONS REGION.
         USING COMREG,R1           INFORM ASSEMBLER.
         ICM   R9,15,IJBAFCB       LOAD ADDRESS OF IJBAFCB. (X'B4').
         BZ    CALLØ5C             ZERO-BRANCH TO CALLØ5C.
         DROP  R1                  (COMREG).
         CLC   =C'DPEIBC  ',Ø(R2)  IS PHASE TO BE CDLOAD'ED 'DPEIBC  '.
         BE    CALLØ5C             YES-BRANCH TO CALLØ5C.
         CLC   =C'TXER4ØØ ',Ø(R2)  IS PHASE TO BE CDLOAD'ED 'TXER4ØØ '.
         BE    CALLØ5C             YES-BRANCH TO CALLØ5C.
         L     R9,8(R9)            LOAD ADDRESS OF CSA TO REG 9.
         ST    R9,ADDRCSA          SVE IT.
         USING DFHCSADS,R9         INFORM ASSEMBLER.
         L     R9,ADDRCSA          ADDRESS CSA.
         L     RC,CSACDTA          ADDRESS CURRENT DISPATCHED USER TCA.
         L     RC,Ø(RC)            LOAD ADDRESS OF SYSTEM TCA TO REG 12
         USING DFHTCADY,RC         INFORM ASSEMBLER.
         SR    R7,R7               CLEAR REG 7.
         ICM   R7,7,TCAPCTA+1      INSERT PCT ADDRESS TO REG 7.
         MVC   PGMNME,Ø(R7)        MVE PPT NAME.
         DROP  R9,RC               (DFHCSADS).(DFHTCADY).
         CLC   =C'DPCICS',PGMNME   CALLED FROM DPCICS.
         BE    CALLØ5C             YES-BRANCH TO CALLØ5C.
         CLC   =C'DPSTUB',PGMNME   CALLED FROM DPSTUB.
         BE    CALLØ5C             YES-BRANCH TO CALLØ5C.
         MVI   Ø(R2),X'Ø3'         INDICATE BEING CALLED FROM A CICS PR
         L     R1,SVR1A            RESTORE REG 1.
         B     CALLØ1C             BRANCH TO CALLØ1C.
*
CALLØ5B  EQU   *
         MVI   Ø(R2),X'Ø2'         INDICATE PHASE TO BE CDLOAD'ED WAS D
         B     CALLØ1C             BRANCH TO CALLØ1C.
*
CALLØ5C  EQU   *
         L     R1,SVR1A            RESTORE REG 1.
         CLC   =C'        ',Ø(R2)  IS PHASE TO BE CDLOAD'ED BLANK.
         BE    CALLØ5M             YES-BRANCH TO CALLØ5M.
         CLC   Ø(L'LV,R2),LV       IS PHASE TO BE CDLOAD'ED LOW-VALUES.
         BE    CALLØ5M             YES-BRANCH TO CALLØ5M.
```


© 1998. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.

```
         CLC   Ø(L'HV,R2),HV       IS PHASE TO BE CDLOAD'ED HIGH-VALUES
         BE    CALLØ5M             YES-BRANCH TO CALLØ5M.
         CLI   Ø(R2),C' '          IS FIRST BYTE OF PHASE TO BE CDLOAD'
         BE    CALLØ5M             YES-BRANCH TO CALLØ5M.
         CLI   Ø(R2),X'ØØ'         IS FIRST BYTE OF PHASE TO BE CDLOAD'
         BE    CALLØ5M             YES-BRANCH TO CALLØ5M.
         CLI   Ø(R2),X'FF'         IS FIRST BYTE OF PHASE TO BE CDLOAD'
         BE    CALLØ5M             YES-BRANCH TO CALLØ5M.
         CLI   Ø(R2),C'A'          IS PHASE TO BE CDLOAD'ED VALID.
         BL    CALLØ5M             NO-BRANCH TO CALLØ5M.
         CLI   Ø(R2),C'Z'          IS PHASE TO BE CDLOAD'ED VALID.
         BH    CALLØ5M             NO-BRANCH TO CALLØ5M.
         CLC   SVPHASEC,Ø(R2)      DID WE CDLOAD THIS PHASE LAST TIME.
         BE    CALLØ6              YES-BRANCH TO CALLØ6.
         CDLOAD (R2),RETPNF=YES    LOAD INTO PARTITION GETVIS.
         LTR   RF,RF               WAS CDLOAD SUCCESSFUL.
         BZ    CALLØ7              YES-BRANCH TO CALLØ7.
         AP    COUNT3,=P'1'        INCREMENT CDLOAD ERROR COUNTER.
         STCM  RF,B'1111',SVRF     SVE REG 15.
         MVI   Ø(R2),X'FF'         INDICATE CDLOAD ERROR.
         MVC   1(L'SVRF,R1),SVRF   MVE IT TO PHASE NAME.
         B     CALLØ1C             BRANCH TO CALLØ1C.
*
CALLØ5M  EQU   *
         MVC   Ø(L'HV,R2),HV       TRASH PHASE TO BE CDLOAD'ED.
         B     CALLØ1C             BRANCH TO CALLØ1C.
*
CALLØ6   EQU   *
         ICM   R1,B'1111',SVR1B    LOAD CDLOAD'ED PHASE LAST TIME ENTRY
         AP    COUNT2,=P'1'        INCREMENT CDLOAD SKIPPED COUNTER.
*
CALLØ7   EQU   *
         AP    COUNT1,=P'1'        INCREMENT CDLOAD COUNTER.
         MVC   SVPHASEP,SVPHASEC   MVE CURRENT PHASE NAME TO PREVIOUS.
         MVC   SVPHASEC,Ø(R2)      SVE CURRENT PHASE NAME.
         MVC   SVR1C,SVR1B         MVE CURRENT PHASE ENTRY POINT TO PRE
         STCM  R1,B'1111',SVR1B    SVE CURRENT PHASE ENTRY POINT.
         CLC   =C'CDLOAD  ',Ø(R5)  ARE WE CDLOAD'ING ONLY.
         BNE   CALLØ8              NO-BRANCH TO CALLØ8.
         MVC   Ø(L'SVR1B,R5),SVR1B MVE ENTRY POINT ADDRESS.
         AP    COUNT4,=P'1'        INCREMENT CDLOAD ONLY COUNTER.
         MVI   HERE,C'3'           INDICATE WE'VE BEEN HERE.
         B     CALLØ2              BRANCH TO CALLØ2.
*
CALLØ8   EQU   *
         LR    RF,R1               LOAD CDLOAD'ED PHASE ENTRY POINT TO
         LA    RD,SAVEARE2         LOAD ADDRESS OF SAVEAREA TO REG 13.
         AMODESW QRY ,             L     R1,=F'1Ø73741824'
*                                  LA    R1,Ø(R1)
*                                  SLL   R1,R1
```

© 1998. Reproduction prohibited. Please inform Xephon of any infringement.

```
         STCM  R1,B'1000',AMODE    SVE HIGH ORDER BYTE.
         LR    R1,RA               LOAD SECOND PARAMETER TO REG 1.
         CLI   AMODE,X'80'         ARE WE RUNNING ABOVE THE LINE.
         BE    CALL08C             YES-BRANCH TO CALL08C.
         TM    SVR1B,X'80'         WAS CURRENT PHASE CDLOAD'ED ABOVE TH
         BNO   CALL08C             NO-BRANCH TO CALL08C.
*
* THE FOLLOWING INSTRUCTIONS DEAL WITH BRANCHING TO A PHASE THAT WAS
* CDLOAD'ED ABOVE THE 16M LINE AND IF THIS SUBROUTINE WAS CALLED FROM
* A PROGRAM RUNNING ABOVE THE 16M LINE. IF YOU DON'T HAVE THE HIGH
* LEVEL ASSEMBLER YOU MUST COMMENT THE BSM, BASSM, AND AMODESW STATE-
* MENTS. NOTE: DOING SO WILL MEAN THAT A PHASE, TO BE CDLOAD'ED, MUST
* NOT BE CATALOGED TO RUN ABOVE THE 16M LINE. AS A CONVENIENCE THE
* ACTUAL MACHINE INSTRUCTIONS THAT THE BSM AND BASSM INSTRUCTIONS
* GENERATE AND THE ACTUAL SOURCE CODE THE AMODESW MACRO GENERATES ARE
* PROVIDED SO THAT IF YOU'D LIKE YOU CAN STILL USE THEM. THIS
* ASSUMES THAT THE OP CODES WILL EXECUTE ON YOUR HARDWARE.
*
         LA    RB,CALL09
         BSM   RB,R0               DC  X'0BB0'
         BASSM RE,RF               DC  X'0CEF'
         AMODESW SET,AMODE=24      TM  577(0),X'02'
*                                  BNO CALL09
*                                  LA  RF,CALL09
*                                  DC  X'0B0F'
         B     CALL09              BRANCH TO CALL09.
*
CALL08C  EQU   *
         MVI   HERE,C'4'           INDICATE WE'VE BEEN HERE.
         BALR  RE,RF               BRANCH TO SUBROUTINE.
*
CALL09   EQU   *
*        SDUMP STORAGE=(DPCALLM,DPCALLE) ???
*        PDUMP DPCALLM,DPCALLE ???
         MVI   HERE,C'5'           INDICATE WE'VE BEEN HERE.
         B     CALL02              BRANCH TO CALL02.
*
DPCALLM  DC    C'DPCALL STORAGE HERE. ' INSERT EYE CATCHER.
PGMNME   DS    CL8                 CICS PROGRAM NAME BEING INVOKED FROM
HERE     DS    C                   WE'VE BEEN HERE INDICATOR.
AMODE    DS    X                   AMODE BEING EXECUTED UNDER.
COUNT0   DC    PL4'0'              NUMBER OF TIMES CALLED COUNTER.
COUNT1   DC    PL4'0'              CDLOAD SUCCESSFUL COUNTER.
COUNT2   DC    PL4'0'              CDLOAD SKIPPED COUNTER.
COUNT3   DC    PL4'0'              CDLOAD NOT SUCCESSFUL COUNTER.
COUNT4   DC    PL4'0'              CDLOAD ONLY COUNTER.
COUNT5   DC    PL4'0'              CDDELETE COUNTER.
SVPHASEC DS    CL8                 CURRENT PHASE CDLOAD'ED.
SVPHASEP DS    CL8                 PREVIOUS PHASE CDLOAD'ED.
ADDRCSA  DS    F                   ADDRESS OF CSA.
```


© 1998. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.

```
SVR1A    DS    F
SVR1B    DS    F                   CURRENT PHASE ENTRY POINT.
SVR1C    DS    F                   PREVIOUS PHASE ENTRY POINT.
SVR2     DS    F                   REG Ø2 SVE AREA.
SVRE     DS    F                   REG 14 SVE AREA.
SVRF     DS    F                   REG 15 SVE AREA.
SVRFB    DS    F                   REG 15 CONTENTS AFTER CDDELETE.
SAVEARE1 DS    18F
SAVEARE2 DS    18F
* VEARE3 DS    18F
LV       DC    X'0000000000000000'
HV       DC    X'FFFFFFFFFFFFFFFF'
DPCALLE  DC    X'FF'
*
         LTORG
*
         DFHCSAD
*        COPY  DFHCSADS
         DFHTCA CICSYST=YES        COPY DFHTCADS.
*
* PARTITION COMMUNICATIONS REGION. (MAPCOMR).
*
         MAPCOMR ,
*
         END
```

Bob Botsis
Senior Systems Programmer (USA)

© Xephon 1998

Approximately 3,500 files containing code from Xephon's technical journals can be viewed and downloaded from our Web site, free of charge. All code published before the end of 1996 is included. (Articles from January 1997 onwards are still controlled by password.)

There are three means of access: a chronological listing by issue date, an alphabetical listing by article title, and a keyword free-text search facility (only article titles are indexed). Our Web site is at http://www.xephon.com

© 1998. Reproduction prohibited. Please inform Xephon of any infringement.

Subroutine for sorting table or other data

This subroutine allows the sorting of table or other such data passed to it by the calling program.

Two parameters must be passed:

- The first parameter is the data to be sorted. It must be terminated by high-values, and the high values must be the same length as the data to be sorted, as specified in the second parameter. There is no limit to the size of the data to be sorted, just as long as you remember to end the data with high-values.
- The second parameter is the length of the data to be sorted, expressed as a half-word binary value, which must be greater than X'0000' and less than X'00FF'. This of course limits the size of each entry to be sorted to 255 bytes. If the value specified isn't within the specified range, low-values will be returned in this field, and no sorting will take place. If multiple calls are to be made, this field must be set to the proper value for each call after the first.

CALLING SEQUENCES

The calling sequences are as follows.

COBOL

```
CALL 'DPSORT' USING TABDAT, TABLEN.
```

ALC

```
LA    13,SAVEAREA (13 CAN ALSO BE R13 OR RD).
                CALL  DPSORT,(TABDAT,TABLEN)
                .
                .  (MAINLINE PART OF PROGRAM).
                .
       SAVEAREA DC    18F'0'
```


© 1998. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.

RPGII

```
     CALL 'DPSORT'
                  PARM           TABDAT
                  PARM           TABLEN
```

An 18-word save area must be passed through register 13 by the user (STD COBOL LINKAGE).

DPSORT

```
DPSO     TITLE 'DPSORT - 1.Ø - BUBBLE SORT SUBROUTINE.'
DPSORT   CSECT
DPSORT   AMODE 31
DPSORT   RMODE ANY
         BALR  15,Ø                LOAD TEMPORARY BASE.
         USING *,15                INFORM ASSEMBLER.
         SAVE  (14,12)
         DROP  15                  DROP TEMPORARY BASE.
         BALR  3,Ø
         USING *,3
         ST    13,SAVEAREA+4
         LA    13,SAVEAREA
         B     SORBEGN
*
         DC    C'DPSORT STARTS HERE. ' INSERT EYE CATCHER.
*
SORBEGN  EQU   *
         LM    5,6,Ø(1)            LOAD ADDRESSES OF PARAMETERS.
         ST    5,SV5
         MVC   LENG,Ø(6)           MVE LENGTH.
         CLC   LENG,=X'ØØFF'       IS DATA LENGTH GREATER THAN 255.
         BNH   SORNX5              NO-BRANCH TO SORNX5.
*
SORNX3   EQU   *
         MVC   Ø(L'LENG,6),=X'ØØØØ' INDICATE LENGTH ERROR.
         B     SORDONE             BRANCH TO SORDONE.
*
SORNX5   EQU   *
         LH    8,LENG              LOAD DATA LENGTH INTO REG 8.
         LTR   8,8                 IS IT ZERO.
         BZ    SORNX3              YES-BRANCH TO SORNX3.
         LR    9,8                 LOAD DATA LENGTH TO REG 9.
         BCTR  9,Ø                 REDUCE IT BY ONE.
         STC   9,SORDATA3+1        MODIFY LENGTH OF CLC.
         STC   8,SORDATA3+5        ...
         STC   9,SORDATA5+1        MODIFY LENGTH OF FIRST XC.
         STC   8,SORDATA5+5        ...
         STC   9,SORDATA6+1        MODIFY LENGTH OF SECOND XC.
```

© 1998. Reproduction prohibited. Please inform Xephon of any infringement.

```
         STC   8,SORDATA6+3        ...
         STC   9,SORDATA7+1        MODIFY LENGTH OF THIRD XC.
         STC   8,SORDATA7+5        ...
         STC   8,SORSKIP+3         MODIFY LENGTH OF LA.
         MVI   SORSSW,C'Y'         INDICATE FIRST PASS.
*
SORSWAP  EQU   *
         CLI   SORSSW,C'Y'         DID ANY SWAPS OCCUR.
         BNE   SORDONE             NO-BRANCH TO SORDONE.
         MVI   SORSSW,C'N'         RESET SWAP FLAG.
         LR    7,5
*
SORDATA  EQU   *
         CLI   Ø(7),X'FF'          IS THIS END OF DATA.
         BE    SORSWAP             YES-BRANCH TO SORSWAP.
SORDATA3 CLC   Ø(L'DATA,7),L'DATA(7) DO WE NEED TO SWAP.
         BNH   SORSKIP             NO-BRANCH TO SORSKIP.
SORDATA5 XC    Ø(L'DATA,7),L'DATA(7)
SORDATA6 XC    L'DATA(L'DATA,7),Ø(7)
SORDATA7 XC    Ø(L'DATA,7),L'DATA(7)
         MVI   SORSSW,C'Y'         INDICATE WE SWAPPED.
*
SORSKIP  EQU   *
         LA    7,L'DATA(,7)        INCREMENT TO NEXT DATA.
         B     SORDATA             BRANCH TO SORDATA.
*
SORDONE  EQU   *
         SR    15,15               CLEAR REG 15.
         L     13,SAVEAREA+4
         RETURN (14,12),RC=(15)
*
DPSORTS  DC    C'DPSORT STORAGE HERE. ' INSERT EYE CATCHER.
*
DATA     DS    CL3
         DC    X'FF'
SORSSW   DS    C
LENG     DS    H
SV5      DS    F
*
SAVEAREA DC    18F'Ø'
*
DPSORTE  DC    X'FF'
*
         END
```

Bob Botsis
Senior Systems Programmer (USA)
© Xephon 1998

© 1998. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.

Subroutine for expanding one byte into eight

This subroutine expands one byte into eight bytes, each of which will be 0 or 1, depending on the bit values of the passed byte.

Two parameters must be passed:

- The first parameter must be the one-byte field to be expanded. It can be any character or hexadecimal value.
- The second parameter is the eight-byte target field into which the bit expansion of the first parameter is to be placed. You need not clear this field before each call.

CALLING SEQUENCES

The calling sequences are as follows.

COBOL

```
CALL 'DPBITX' USING CHAR, XPND.
```

ALC

```
LA    13,SAVEAREA (13 CAN ALSO BE R13 OR RD).
                  CALL  DPBITX,(CHAR,XPND)
                     .
                     .  (MAINLINE PART OF PROGRAM).
                     .
SAVEAREA DC    18F'Ø'
```

RPGII

```
CALL 'DPBITX'
               PARM           CHAR
               PARM           XPND
```

An eighteen-word save area must be passed through register 13 by the user (STD COBOL LINKAGE).

© 1998. Reproduction prohibited. Please inform Xephon of any infringement.

DPBITX

```
DPBI     TITLE 'DPBITX - 1.Ø - EXPANDS 1 BYTE TO 8 BYTES EACH BYTE BEING
Ø OR 1 SUBROUTINE.'
DPBITX   CSECT
DPBITX   AMODE 31
DPBITX   RMODE ANY
         BALR  15,Ø                LOAD TEMPORARY BASE.
         USING *,15                INFORM ASSEMBLER.
         SAVE  (14,12)
         DROP  15                  DROP TEMPORARY BASE.
         BALR  3,Ø
         USING *,3
         ST    13,SAVEAREA+4
         LA    13,SAVEAREA
         B     BITXBEG
*
         DC    C'DPBITX STARTS HERE. ' INSERT EYE CATCHER.
*
BITXBEG  EQU   *
         LM    5,6,Ø(1)            GET ADDRESSES OF PARAMETERS.
         MVC   CHAR,Ø(5)           MVE CHARACTER TO BE EXPANDED.
         MVC   TARG,Ø(6)           MVE TARGET.
         MVC   TARG,=C'ØØØØØØØØ'   CLEAR TARGET.
         TM    CHAR,B'1ØØØØØØØ'    IS BIT ZERO ON.
         BNO   *+8                 NO-SKIP NEXT INST.
         MVI   TARG,C'1'           INDICATE BIT ZERO (Ø) ON.
         TM    CHAR,B'Ø1ØØØØØØ'    IS BIT ONE ON.
         BNO   *+8                 NO-SKIP NEXT INST.
         MVI   TARG+1,C'1'         INDICATE BIT ONE (1) ON.
         TM    CHAR,B'ØØ1ØØØØØ'    IS BIT TWO ON.
         BNO   *+8                 NO-SKIP NEXT INST.
         MVI   TARG+2,C'1'         INDICATE BIT TWO (2) ON.
         TM    CHAR,B'ØØØ1ØØØØ'    IS BIT THREE ON.
         BNO   *+8                 NO-SKIP NEXT INST.
         MVI   TARG+3,C'1'         INDICATE BIT THREE (3) ON.
         TM    CHAR,B'ØØØØ1ØØØ'    IS BIT FOUR ON.
         BNO   *+8                 NO-SKIP NEXT INST.
         MVI   TARG+4,C'1'         INDICATE BIT FOUR (4) ON.
         TM    CHAR,B'ØØØØØ1ØØ'    IS BIT FIVE ON.
         BNO   *+8                 NO-SKIP NEXT INST.
         MVI   TARG+5,C'1'         INDICATE BIT FIVE (5) ON.
         TM    CHAR,B'ØØØØØØ1Ø'    IS BIT SIX ON.
         BNO   *+8                 NO-SKIP NEXT INST.
         MVI   TARG+6,C'1'         INDICATE BIT SIX (6) ON.
         TM    CHAR,B'ØØØØØØØ1'    IS BIT SEVEN ON.
         BNO   *+8                 NO-SKIP NEXT INST.
         MVI   TARG+7,C'1'         INDICATE BIT SEVEN (7) ON.
         MVC   Ø(L'TARG,6),TARG    MVE TARGET.
```


© 1998. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.

```
         SR    15,15               CLEAR REG 15.
         L     13,SAVEAREA+4
         RETURN (14,12),RC=(15)
*
         DC    C'DPBITX STORAGE HERE. ' INSERT EYE CATCHER.
CHAR     DC    X'ØØ'
TARG     DS    CL8
*
SAVEAREA DC    18F'Ø'
*
BITXE    DC    X'FF'
*
         END
```

Bob Botsis
Senior Systems Programmer (USA)
© Xephon 1998

Program-driven output segmentation

The program presented here supports program-driven output segmentation by simple subroutine call. It was developed and tested under VSE/ESA Version 1.3, and is now running under VSE/ESA Version 2.2.

Two parameters must be passed, and two further parameters are optional.

FIRST PARAMETER

The first parameter is an input parameter, and must contain the VSE/POWER JECL statement valid for the new segment created after the subroutine call. If the fourth parameter is omitted, this must be a valid * $$ LST JECL statement. The default length is 71; this can be changed by the optional third parameter. Note that VSE/POWER ignores both the LST= operand of a * $$ LST JECL statement and the PUN= operand of a * $$ PUN JECL statement; use the fourth parameter instead.

There must be one blank delimiter between *, $$, and the desired LST or PUN operation. Statements in positional format are not supported.

© 1998. Reproduction prohibited. Please inform Xephon of any infringement.

The new segment will usually receive the specified attributes, completed by the attributes of the preceding segment. However, if the first parameter starts with the string '* SS ', all omitted attributes will be replaced by their default values.

SECOND PARAMETER

The second parameter is an output parameter. It is a one-byte field. Successful segmentation is indicated by '0', errors by '1'.

THIRD PARAMETER

The optional third parameter is an input parameter, and must be a halfword containing the length of the first parameter. All values between 9 and 1042 are allowed. The usual JECL statement continuation rules do not apply; instead, the first parameter must contain the JECL statement with a series of operands delimited by commas. If the supplied or defaulted length extends beyond the last character of the last operand, there must be at least one trailing blank, optionally followed by a comment.

FOURTH PARAMETER

The optional fourth parameter is an input parameter, and specifies the system logical unit (SYSLST or SYSPCH) or programmer logical unit (from SYS000 to SYS254) assigned to the device on which segmentation is to occur. If this parameter is omitted, SYSLST is assumed.

IPWSEGM MACRO

B220SEG uses the new IPWSEGM macro, and replaces an old version which used the SEGMENT macro. For a description of these macros, see *VSE/POWER Application Programming*.

Because of restrictions on the IPWSEGM macro, the program can execute only below the 16 MB line (RMODE 24). It can accept callers in any AMODE, and uses capping (see *VSE/ESA Extended*

© 1998. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.

Addressability) to execute the IPWSEGM macro in the required AMODE 24.

EXAMPLE USAGE

We have to print a lot of addresses for different newspapers. These newspapers are identified by numbers (keys) from 0000 to 9999. The addresses are produced in ascending order, but our technical staff often need them in another sequence. We decided to create one output segment for each newspaper, with the segment name ADDRnnnn, where *nnnn* is the number identifying the corresponding paper.

This processing is illustrated by the following example, which shows the relevant parts of the corresponding COBOL program calling subroutine B220SEG.

```
...
 Ø1  LST-CARD.
     Ø2 FILLER                PIC X(17) VALUE '* $$ LST JNM=ADDR'.

     Ø2 PAPER-KEY-SEGM        PIC X(4)  VALUE SPACES.
     Ø2 FILLER                PIC X(55) VALUE
        ',CLASS=5,FCB=CFCBHBVA,UCS=CUCBABHH,FNO=Ø94,DISP=K,RBS=Ø'.
 Ø1  LENGTH-LST               PIC S9(4) COMP.
 Ø1  SEGM-RC                  PIC X(1).
     ...

 PROLOGUE.
     ...

     MOVE LENGTH OF LST-CARD TO LENGTH-LST
     ...
     ...

 NEXT-RECORD.
     ...
     Program reads next record with the next address,
     record contains field PAPER-KEY
     ...
     IF PAPER-KEY NOT = PAPER-KEY-SEGM
     THEN
       MOVE PAPER-KEY TO PAPER-KEY-SEGM
       CALL 'B22ØSEG' USING LST-CARD SEGM-RC LENGTH-LST
```

© 1998. Reproduction prohibited. Please inform Xephon of any infringement.

```
      IF SEGM-RC NOT = 'Ø' ....
    END-IF
    ...
    Program prints next address on SYSLST
    ...
```

SOURCE OF B220SEG

```
TITLE 'B22ØSEG - PROGRAM-DRIVEN OUTPUT SEGMENTATION'
B22ØSEG  CSECT
B22ØSEG  AMODE ANY
B22ØSEG  RMODE 24
         EJECT
*********************************************************************
*        PARAMETER LIST
*********************************************************************
PARAM    DSECT
ADRJECL  DS    A             ADDRESS FIRST PARAMETER
ADRRETC  DS    A             ADDRESS SECOND PARAMETER
ADRJECLG DS    A             ADDRESS THIRD PARAMETER (OPTIONAL)
ADRUNIT  DS    A             ADDRESS FOURTH PARAMETER (OPTIONAL)
         EJECT
*********************************************************************
*        PARAMETER AREA USED BY THE IPWSEGM MACRO (DSECT)
*********************************************************************
         IPW$MXD
         EJECT
*********************************************************************
*        REGISTER EQUATES
*********************************************************************
RØ       EQU   Ø
R1       EQU   1
R2       EQU   2
R3       EQU   3
R4       EQU   4
R5       EQU   5
R6       EQU   6
R7       EQU   7
R8       EQU   8
R9       EQU   9
R1Ø      EQU   1Ø
R11      EQU   11
R12      EQU   12
R13      EQU   13
R14      EQU   14
R15      EQU   15
         EJECT
B22ØSEG  CSECT
```


© 1998. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.

```
*********************************************************************
* REGISTER USAGE:
*    R15 PROGRAM ENTRY POINT, RETURN CODE
*    R14 RETURN ADDRESS
*    R13 SAVE AREA ADDRESS
*    R12
*    R11 CALLER'S ADDRESSING MODE, ADDRESS OF EPILOGUE
*    R1Ø WORK REGISTER
*    R9  BASE REGISTER
*    R8
*    R7
*    R6
*    R5  ADDRESS OF LOGICAL UNIT (SYSLST OR OPTIONAL FOURTH
         PARAMETER)
*    R4  LENGTH OF FIRST PARAMETER (JECL STATEMENT)
*    R3  ADDRESS OF SECOND PARAMETER (RETURN CODE)
*    R2  ADDRESS OF FIRST PARAMETER (JECL STATEMENT)
*    R1  ADDRESS OF PARAMETER LIST, USED BY IBM MACROS
*    RØ  USED BY IBM MACROS
*********************************************************************
         EJECT
*********************************************************************
*        PROLOGUE USING CALLER'S ADDRESSING MODE (24 OR 31)
*        CAPPING (SEE VSE/ESA EXTENDED ADDRESSABILITY)
*********************************************************************
B22ØSEG  CSECT
         USING *,R15                ESTABLISH ADDRESSABILITY
         STM   R14,R12,12(R13)      SAVE CALLER'S REGISTERS
         LA    R1Ø,SAVEAREA         ADDRESS OWN SAVE AREA
         ST    R1Ø,8(,R13)          SAVE ADDRESS OWN SAVE AREA
         LA    R9,START24           LOAD NEW BASE REGISTER
         LA    R11,STARTANY         EPILOGUE IN CALLER'S AMODE
         BSM   R11,R9               SAVE OLD AMODE, START AMODE 24
START24  DS    ØH
         DROP  R15
         USING *,R9                 ESTABLISH ADDRESSABILITY
         ST    R11,SAVEAREA+72      SAVE ADDRESS EPILOGUE, OLD
                                    AMODE
         ST    R13,SAVEAREA+4       SAVE ADDRESS CALLER'S SAVE
                                    AREA
         LR    R13,R1Ø              ESTABLISH OWN SAVE AREA
         EJECT
*********************************************************************
*        LOAD ADDRESS OF PARAMETERS
*********************************************************************
         USING PARAM,R1             ADDRESS PARAMETER LIST
         L     R2,ADRJECL           ADDRESS JECL STATEMENT
         L     R3,ADRRETC           ADDRESS RETURN CODES
         LA    R4,71                DEFAULT LENGTH OF JECL
                                    STATEMENT
```

© 1998. Reproduction prohibited. Please inform Xephon of any infringement.

```
         LA    R5,=C'SYSLST'           DEFAULT LOGICAL UNIT
         MVI   Ø(R3),C'Ø'              RETURN CODE (SUCCESS)
         TM    ADRRETC,X'8Ø'           ONLY TWO PARAMETERS
         BO    TESTKEEP                YES, USE DEFAULTS
         L     R4,ADRJECLG             ADDRESS THIRD PARAMETER
         LH    R4,Ø(,R4)               LENGTH OF JECL STATEMENT
         CH    R4,=H'9'                LESS THAN 9
         BL    ERROR                   YES, ERROR
         CH    R4,=H'1Ø24'             GREATER THAN 1Ø42
         BH    ERROR                   YES, ERROR
         TM    ADRJECLG,X'8Ø'          ONLY THREE PARAMETERS
         BO    TESTKEEP                YES, USE DEFAULT SYSLST
         L     R5,ADRUNIT              ADDRESS OF SUPPIED LOGICAL
                                       UNIT
         DROP  R1
         EJECT
*********************************************************************
*        TEST SUPPLIED FOURTH PARAMETER
*********************************************************************
         CLC   Ø(3,R5),=C'SYS'         LOGICAL UNIT SYSXXX
         BNE   ERROR                   NO, ERROR
         CLC   3(3,R5),=C'LST'         IS IT SYSLST
         BE    TESTKEEP                YES, PARAMETER IS OK
         CLC   3(3,R5),=C'PCH'         IS IT SYSPCH
         BE    TESTKEEP                YES, PARAMETER IS OK
         CLI   3(R5),C'Ø'              FOURTH BYTE LESS THAN Ø
         BL    ERROR                   YES, ERROR
         CLI   4(R5),C'Ø'              FIFTH BYTE LESS THAN Ø
         BL    ERROR                   YES, ERROR
         CLI   4(R5),C'9'              FIFTH BYTE GREATER THAN 9
         BH    ERROR                   YES, ERROR
         CLI   5(R5),C'Ø'              SIXTH BYTE LESS THAN Ø
         BL    ERROR                   YES, ERROR
         CLI   5(R5),C'9'              SIXTH BYTE GREATER THAN 9
         BH    ERROR                   YES, ERROR
         CLC   3(3,R5),=C'254'         LOGICAL UNIT GREATER THAN 254
         BH    ERROR                   YES, ERROR
         EJECT
*********************************************************************
*        TEST WHETHER MISSING KEYWORDS ARE TO BE REPLACED BY DEFAULTS
*********************************************************************
TESTKEEP DS    ØH
         CLC   Ø(5,R2),=CL5'*  '     KEEP OLD OUTPUT ATTRIBUTES
         BE    DEFAULTS                NO, USE DEFAULTS
         EJECT
*********************************************************************
*        PERFORM OUTPUT SEGMENTATION, KEEP OLD VALUES
*********************************************************************
```


© 1998. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.

```
         IPWSEGM DEVADDR=(R5),
               JECL=(R2),                                              *
               JECLN=(R4),                                             *
               KEEP=YES
         B     TESTRC                  TEST RETURN CODE
         EJECT
*********************************************************************
*        PERFORM OUTPUT SEGMENTATION, USE DEFAULTS
*********************************************************************
DEFAULTS DS    ØH
         MVC   2(2,R2),=CL2'$$'        MAKE JECL STATEMENT VALID
         IPWSEGM DEVADDR=(R5),                                         *
               JECL=(R2),                                              *
               JECLN=(R4),                                             *
               KEEP=NO
         MVC   Ø(5,R2),=CL5'*  '     RESTORE OLD CONTENT
         EJECT
*********************************************************************
*        TEST RETURN CODE OF IPWSEGM MACRO
*********************************************************************
TESTRC   DS    ØH
         SRL   R15,8                   REMOVE SEGMENT FEEDBACK CODE
         LTR   R15,R15                 TEST RETURN CODE
         BZ    RETURN                  NO ERROR, RETURN TO CALLER
ERROR    DS    ØH
         MVI   Ø(R3),C'1'              ERROR, SET RETURN CODE TO 1
         EJECT
*********************************************************************
*        RETURN CONTROL TO CALLING PROGRAM
*********************************************************************
RETURN   DS    ØH
         L     R13,SAVEAREA+4          ADDRESS OF CALLER'S SAVE AREA
         L     R11,SAVEAREA+72         ADDRESS EPILOGUE, CALLER'S
                                       AMODE
         BSM   Ø,R11                   RESTORE CALLER'S AMODE
STARTANY DS    ØH
         LM    R14,R12,12(R13)         RESTORE CALLER'S REGISTERS
         SR    R15,R15                 SET REGISTER 15 TO Ø
         BR    R14                     RETURN TO CALLER
         EJECT
*********************************************************************
*        OWN SAVE AREA
*********************************************************************
SAVEAREA DS    19F                     OWN SAVE AREA
         END
```

Walter Richters
(Germany)

© Xephon 1998

© 1998. Reproduction prohibited. Please inform Xephon of any infringement.

Physical unit address of a logical unit

The subroutine presented in this article was developed and tested under VSE/ESA Version 2.2.

Two parameters must be passed:

- The first is an input parameter. It is a six-byte character string and must contain the name of a logical unit. The logical unit may be a programmer logical unit from SYS000 to SYS254, or a system logical unit such as SYSLST, SYSPCH, or SYSLOG.
- The second parameter is an output parameter. It is three-bytes long, and contains the physical address to which the logical unit is assigned. The values ‘UA ’ (unassigned) and ‘IGN’ (ignored) are allowed. If the first parameter does not contain a valid logical unit, the second parameter remains unchanged.

CALLING THE SUBROUTINE

The following example shows how to call the B224CUU subroutine from a COBOL program:

```
...
WORKING-STORAGE SECTION.
  ...
Ø1 LUNIT PIC X(6) VALUE 'SYSØØ5'.
Ø1 PUNIT PIC X(3) VALUE SPACES.
  ...
PROCEDURE DIVISION.
  ...
  CALL 'B224CUU' USING LUNIT PUNIT.
  IF PUNIT NOT = SPACES ...
  ...
```

If you link-edit the object module, you can also call it from a REXX/VSE procedure. Two simple procedures, GETCUU and ARCMD, illustrate this possibility:

```
/*********************************/
/* GETCUU              REXX/VSE */
/*********************************/
arg lunit
```


© 1998. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.

```
punit = "   "
address LINKPGM "B224CUU LUNIT PUNIT"
address JCL "// SETPARM" lunit"="punit
exit


/*********************************/
/* ARCMD                 REXX/VSE */
/*********************************/
arg command
call SENDCMD command
exit
```

In the following example, these two procedures are used to save all VSE/POWER queue entries on a labelled tape, and to check the volume serial number:

```
// JOB BACKUP
// SETPARM SYS005=
// ASSGN SYS005,348Ø,Ø8,VOL=volser
// EXEC REXX=GETCUU,PARM='SYS005',SYS005
// ASSGN SYS005,UA
// EXEC REXX=ARCMD,PARM='O BACKUP,ALL,&SYS005,Ø8,TLBL=OFFL,LTAPE=YES'
/&
```

Let’s assume that the tape with the required volume serial number is mounted on physical unit X'580'. The logical unit SYS005 is then assigned to 580, and the GETCUU procedure issues the following job control statement:

```
// SETPARM SYS005=58Ø
```

Finally, the SYS005 logical unit is unassigned and the ARCMD procedure issues the following VSE console command to create the back-up tape:

```
O BACKUP,ALL,58Ø,Ø8,TLBL=OFFL,LTAPE=YES
```

It is assumed that there is a

```
// TLBL OFFL,...
```

statement in a subarea of the system’s label information area used by the VSE/POWER partition.

© 1998. Reproduction prohibited. Please inform Xephon of any infringement.

NOTES

Note that VSE/POWER Version 6.1.2 is the first version that supports the processing of labelled POFFLOAD tapes.

All the information used in this article can be found in the usual VSE/ESA manuals, and no logic manuals are needed. Although the internal representation of the SYSDMP system logical unit is missing in the *VSE/ESA System Macros Reference* manual, it can be retrieved using a CCB SYSDMP,... macro.

B224CUU

```
TITLE 'B224CUU - PHYSICAL UNIT ADDRESS OF A LOGICAL UNIT'
B224CUU  CSECT
B224CUU  AMODE ANY
B224CUU  RMODE ANY
         EJECT
*********************************************************************
*        DSECT, INFORMATION RETRIEVED WITH MACRO EXTRACT ID=PUB
*********************************************************************
         IJBPUB
         EJECT
*********************************************************************
*        REGISTER EQUATES
*********************************************************************
R0       EQU   0
R1       EQU   1
R2       EQU   2
R3       EQU   3
R4       EQU   4
R5       EQU   5
R6       EQU   6
R7       EQU   7
R8       EQU   8
R9       EQU   9
R10      EQU   10
R11      EQU   11
R12      EQU   12
R13      EQU   13
R14      EQU   14
R15      EQU   15
         EJECT
B224CUU  CSECT
```


© 1998. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.

```
*********************************************************************
* REGISTER USAGE:
*    R15 PROGRAM ENTRY POINT, RETURN CODE
*    R14 RETURN ADDRESS
*    R13 ADDRESS OF CALLER'S SAVE AREA

*    R12
*    R11
*    R1Ø
*    R9  BASE REGISTER
*    R8
*    R7  LENGTH OR ADDRESS OF AREA WITH EXTRACTED INFORMATION

*    R6
*    R5
*    R4
*    R3  ADDRESS OF SECOND PARAMETER
*    R2  ADDRESS OF FIRST PARAMETER
*    R1  ADDRESS OF PARAMETER LIST, WORK REGISTER
*    RØ
*********************************************************************
         EJECT
*********************************************************************
*        SET UP LINKAGE REGISTERS
*********************************************************************
         STM   R14,R12,12(R13)         SAVE CALLER'S REGISTERS
         BALR  R9,Ø                    LOAD BASE REGISTER
         USING *,R9                    ESTABLISH ADDRESSABILITY
         EJECT
*********************************************************************
*        TEST PARAMETERS
*********************************************************************
         TM    Ø(R1),X'8Ø'             ONLY ONE PARAMETER
         BO    RETURN                  YES, ERROR
         TM    4(R1),X'8Ø'             TWO PARAMETERS
         BZ    RETURN                  NO, ERROR
         L     R2,Ø(,R1)               ADDRESS OF FIRST PARAMETER
         L     R3,4(,R1)               ADDRESS OF SECOND PARAMETER
         CLC   Ø(3,R2),=C'SYS'         FIRST PARAMETER STARTS WITH
                                        SYS
         BNE   RETURN                  NO, ERROR
         CLI   3(R2),C'Ø'              PROGRAMMER LOGICAL UNIT
         BL    SYSUNIT                 NO, SYSTEM LOGICAL UNIT
         EJECT
*********************************************************************
*        TEST PROGRAMMER LOGICAL UNIT (SYSØØØ, SYSØØ1, ..., SYS254)
*********************************************************************
         CLI   4(R2),C'Ø'              FIFTH CHARACTER LESS THAN Ø
         BL    RETURN                  YES, ERROR
```

© 1998. Reproduction prohibited. Please inform Xephon of any infringement.

```
         CLI   4(R2),C'9'              FIFTH CHARACTER GREATER THAN 9
         BH    RETURN                  YES, ERROR
         CLI   5(R2),C'Ø'              SIXTH CHARACTER LESS THAN Ø
         BL    RETURN                  YES, ERROR
         CLI   5(R2),C'9'              SIXTH CHARACTER GREATER THAN 9
         BH    RETURN                  YES, ERROR
         CLC   3(3,R2),=C'254'         LOGICAL UNIT GREATER THAN
                                        SYS254
         BH    RETURN                  YES, ERROR
         PACK  CONVT,3(3,R2)           PACK NUMBER OF LOGICAL UNIT
         CVB   R1,CONVT                CONVERT NUMBER TO BINARY
         STC   R1,LUNIT+1              STORE NUMBER FOR EXTRACT MACRO
         MVI   LUNIT,X'Ø1'             INDICATE PROGRAMMER LOGICAL
                                        UNIT
         B     DOEXTR                  EXTRACT UNIT INFORMATION

         EJECT
*********************************************************************
*        SYSTEM LOGICAL UNIT
*********************************************************************
SYSUNIT  DS    ØH
         LA    R1,SYSLU                TABLE WITH SYSTEM LOGICAL
                                        UNITS
TSTSYSLU DS    ØH
         CLI   Ø(R1),X'FF'             END OF TABLE
         BE    RETURN                  YES, ERROR
         CLC   3(3,R2),Ø(R1)           TEST NEXT TABLE ENTRY
         BE    SYSLUOK                 EQUAL, SUPPLIED UNIT FOUND
         LA    R1,L'SYSLU+L'SYSHX(,R1) ADDRESS OF NEXT TABLE ENTRY
         B     TSTSYSLU                CONTINUE WITH NEXT TABLE ENTRY
SYSLUOK  DS    ØH
         MVC   LUNIT+1(1),L'SYSLU(R1)  STORE INTERNAL REPRESENTATION

         MVI   LUNIT,X'ØØ'             INDICATE SYSTEM LOGICAL UNIT
         EJECT
*********************************************************************
*        EXTRACT UNIT INFORMATION
*********************************************************************
DOEXTR   DS    ØH
         LA    R7,IJBPLEN              LENGTH OF BUFFER
         EXTRACT ID=PUB,                                          *
               AREA=BUFFER,                                       *
               LEN=(R7),                                          *
               SEL=LUNIT
         LA    R7,BUFFER               ADDRESS OF UNIT INFORMATION
         USING IJBPUB,R7               ESTABLISH ADDRESSABILITY
         EJECT
*********************************************************************
*        TEST RETURN CODE OF EXTRACT MACRO
*********************************************************************
```


© 1998. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.

```
         LTR   R15,R15                 TEST RETURN CODE
         BZ    EXTROK                  NO PROBLEMS
         CH    R15,=H'12'              UNIT UNASSIGNED OR IGNORED
         BNE   RETURN                  NO, ERROR
         CLI   IJBPUB,X'FF'            UNASSIGNED
         BNE   NOUA                    NO, CONTINUE
         MVC   Ø(3,R3),=C'UA '         MOVE UA TO SECOND PARAMETER
         B     RETURN                  RETURN TO CALLER
NOUA     DS    ØH
         CLI   IJBPUB,X'FE'            IGNORED
         BNE   RETURN                  NO, OTHER ERROR
         MVC   Ø(3,R3),=C'IGN'         MOVE IGN TO SECOND PARAMETER
         B     RETURN                  RETURN TO CALLER
         EJECT
*********************************************************************
*        RETURN PHYSICAL ADDRESS TO CALLER
*********************************************************************
EXTROK   DS    ØH
         UNPK  CONVT(3),IJBPCHAN(2)    UNPACK CHANNEL ADDRESS
         UNPK  CONVT+2(3),IJBPDEVN(2)  UNPACK UNIT ADDRESS
         TR    CONVT(4),HEXTAB         MAKE ADDRESS READABLE
         MVC   Ø(3,R3),CONVT+1         MOVE ADDRESS TO SECOND
                                        PARAMETER
         EJECT
*********************************************************************
*        RETURN CONTROL TO CALLING PROGRAM
*********************************************************************
RETURN   DS    ØH
         LM    R14,R12,12(R13)         RESTORE CALLER'S REGISTERS
         SR    R15,R15                 SET R15 TO Ø (SUCCESS)
         BSM   Ø,R14                   RETURN TO CALLING PROGRAM
         EJECT
*********************************************************************
*        WORKAREA
*********************************************************************
CONVT    DS    D                       STORAGE FOR CONVERSION
LUNIT    DS    XL2                     LOGICAL UNIT (CCB FORMAT)
BUFFER   DS    ØC                      EXTRACTED INFORMATION
         ORG   *+IJBPLEN               LENGTH OF EXTRACTED
                                        INFORMATION
         DC    C'Ø123456789ABCDEF'     CONVERSION FROM HEX TO EBCDIC
HEXTAB   EQU   *-256
         EJECT
*********************************************************************
*        TABLE WITH SYSTEM LOGICAL UNITS
*********************************************************************
SYSLU    DC    CL3'RDR'                SYSTEM LOGICAL UNIT SYSXXX

SYSHX    DC    X'ØØ'                   INTERNAL REPRESENTATION
```

© 1998. Reproduction prohibited. Please inform Xephon of any infringement.

```
DC    CL3'IPT'
DC    X'Ø1'
DC    CL3'PCH'
DC    X'Ø2'
DC    CL3'LST'
DC    X'Ø3'
DC    CL3'LOG'
DC    X'Ø4'
DC    CL3'LNK'
DC    X'Ø5'
DC    CL3'RES'
DC    X'Ø6'
DC    CL3'USE'
DC    X'Ø9'
DC    CL3'REC'
DC    X'ØA'
DC    CL3'DMP'
DC    X'ØC'
DC    CL3'CAT'
DC    X'ØD'
DC    X'FF'                    END OF TABLE
END   B224CUU
```

Walter Richters
(Germany)

© Xephon 1998

REXX VSE/ESA function packages

A function package is a group of external functions and subroutines that are packaged together to extend the REXX language with new functions. You can write external functions that supplement or to some extent replace functions provided with REXX. Functions and subroutines can be written in REXX itself, to be stored in a sublibrary of member type PROC. You can also write functions and subroutines in a high-level programming language; these are stored in a sublibrary of member type PHASE.

Function package routines must be written in a language that produces object code that can then be link-edited into a phase (for example, COBOL, PL/I, and Assembler). Routines written in interpreted REXX cannot be part of a function package, although routines written in compiled REXX can.

 © 1998. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.

One of the primary advantages of function packages is speed. Not only is the compiled code faster than an equivalent interpreted REXX function, but, by packaging functions together in one phase, several routines can in effect be preloaded, cutting down the time spent trying to locate and load them. When the REXX language processor encounters a call to a function or subroutine, the language processor searches the function packages before searching sublibraries. The function package phase must be in a sublibrary which is in the active PHASE chain.

There are three types of function package. As defined in the *VSE/REXX Reference Manual*, these are:

- USER packages – function packages that an individual user can write to replace or supplement certain supplied functions.
- LOCAL packages – function packages that a system support or application group can write. Local packages may contain functions and subroutines that are available to specific groups of users or to the entire installation.
- SYSTEM packages – function packages that an installation can write for system-wide use, or for use in a particular language processor environment.

The search order for function packages is USER, LOCAL, and then SYSTEM. This article describes the steps required to create a USER function package.

Function packages consist of two parts: a directory and the individual external functions or subroutines. Both parts must be contained in a phase. Each part may be in a separate phase, or combined in a single phase (see the example below). Better performance is obtained by combining both in the same phase. Because the function package directory is loaded during REXX environment initialization and remains in storage, the functions in the package are loaded once and will be in storage when you need them. If the functions and subroutines are in a separate phase, that phase will be loaded each time a call is made to a function or subroutine and then deleted after that function or subroutine has completed.

© 1998. Reproduction prohibited. Please inform Xephon of any infringement.

The function package directory name is the name of the entry point at the beginning of the directory. The name of the directory is specified on the CSECT. There are two dummy function package names defined in the default parameters module ARXPARMS: these are ARXFLOC for a local function package, and ARXFUSER for a user function package. If you create a function package you can user either of the dummy names; if you do this, the external functions and subroutines in the packages are automatically available to REXX programs. If you need more than one user or local function package, you will need to create a function package table containing the name of your function package directory. The source for the default parameters module is ARXPARMS.Z, which is located in PRD1.BASE.

The function package directory consists of a header and a table of functions or subroutines contained in the function package. The header contains a field which is a count of the number of entries in the function package. If you add a new function to the package, you must modify the count field to include the new routine. There is one table entry in the function package directory for each function contained in the package. A new table entry must be added for each new routine added. Each table entry indicates the name of the routine, the address of the routine's code, and the entry point for the routine.

On entry to your code, register 1 will contain the address of the external function parameter list EFPL. At offset +20 into EFPL is the address of a control block called the evaluation block (EVALBLOCK). EVALBLOCK is the control block used to return the results of the function or subroutine to the calling REXX statement. The result is returned in EVDATA. The length of the result being returned must be set in EVLEN. Mapping macros for EFPL and EVALBLOCK, ARXEFPL and ARXEVALB are available in PRD1.BASE. The same interface is used whether your routine is called as a function or as a subroutine. If the code is called as a subroutine, the result in the evaluation block is placed in special variable RESULT; if the routine is called as a subroutine, it has to return a value in EVDATA. If the code is called as a function, the result in the evaluation block is used in the interpretation of the REXX instruction that contained the function. A function must provide a return value in EVDATA, otherwise a syntax error 44 will result.

© 1998. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.

Parameters passed to your function or subroutine are available via a field in the external function parameter list. EFPLARG is the address of the arguments that are passed to a function or subroutine. The arguments are in the form of a list, two full-words per argument. The first full-word for each argument is the address of the data being passed to the function or subroutine. The second full-word for each is the length of the data being passed. The list of arguments is terminated with 'FFFFFFFF'.

Register 15 is used to provide a return code to the calling REXX program. Return code 0 indicates that the function or subroutine processing was successful. A non-zero return code indicates that the function or subroutine processing was unsuccessful. A non-zero return code will cause the language processor to stop the REXX program that called the function or subroutine with an error code 40.

The function package sample presented below is a USER function package using the dummy package name ARXFUSER. The package provides five new functions. Note that two of the routines, ISBUSY and ISJOB, were included as part of an article in *VSE Update*, Issue 29 (March 1998). The two routines contained several bugs which have been corrected in the versions included below.

- ISBUSY(partid) – This function returns 1 if the partition indicated by partid has a job running in it; otherwise it returns 0.
- ISJOB(jobname) – This function returns the partition id identifying where the jobname is running. If the jobname is not found, a blank partition id is returned.
- CPUID() – This function returns the CPU id.
- GETPIK(partid) – This function returns the PIK of the partition specified.
- GETPID(pik) – This function returns the partition id corresponding to the supplied PIK.

For complete information on REXX function packages, please refer to IBM's *VSE/REXX Reference* manual (SC33-6642).

© 1998. Reproduction prohibited. Please inform Xephon of any infringement.

ARXFUSER FUNCTION PACKAGE

```
// LIBDEF *,CATALOG=USRE.ESA21
// LIBDEF *,SEARCH=(PRD2.PROD,PRD1.BASE,PRD2.GEN1,PRD1.MACLIB)
// OPTION CATAL,XREF,NODECK
 PHASE ARXFUSER,*
// EXEC ASMA9Ø,SIZE=1M
         PRINT NOGEN
*********************************************************************
*  FUNCTION PACKAGE OF USEFUL ROUTINES FOR REXX                     *
*********************************************************************
*        EQUREGS
RØ       EQU   Ø
R1       EQU   1
R2       EQU   2
R3       EQU   3
R4       EQU   4
R5       EQU   5
R6       EQU   6
R7       EQU   7
R8       EQU   8
R9       EQU   9
R1Ø      EQU   1Ø
R11      EQU   11
R12      EQU   12
R13      EQU   13
R14      EQU   14
R15      EQU   15
         MAPPCB
COMREG   MAPCOMR
SYSCOM   SYSCOM
         ARXEFPL
         ARXEVALB
ARXFUSER CSECT
ARXFUSER AMODE 31
ARXFUSER RMODE ANY
         DC    CL8'ARXFPACK'  FUNCTION PACKAGE DIRECTORY NAME
         DC    FL4'24'        LENGTH OF DIRECTORY HEADER
         DC    FL4'5'         NUMBER OF FUNCTIONS IN PACKAGE
         DC    FL4'Ø'         RESERVED
         DC    FL4'32'        LENGTH OF DIRECTORY ENTRY
*
         DC    CL8'ISBUSY   ' FUNCTION NAME
         DC    VL4(ISBUSY)    ADDR OF FUNCTION
         DC    FL4'Ø'         RESERVED
         DC    CL8' '         ENTRY POINT-USES FUNCTION ADDR IF BLANK
         DC    CL8' '         RESERVED
*
         DC    CL8'ISJOB    '  FUNCTION NAME
         DC    VL4(ISJOB)     ADDR OF FUNCTION
```


© 1998. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.

```
         DC    FL4'Ø'          RESERVED
         DC    CL8' '          ENTRY POINT-USES FUNCTION ADDR IF BLANK
         DC    CL8' '          RESERVED
*
         DC    CL8'CPUID   '   FUNCTION NAME
         DC    VL4(CPUID)      ADDR OF FUNCTION
         DC    FL4'Ø'          RESERVED
         DC    CL8' '          ENTRY POINT-USES FUNCTION ADDR IF BLANK
         DC    CL8' '          RESERVED
*
         DC    CL8'GETPIK  '   FUNCTION NAME
         DC    VL4(GETPIK)     ADDR OF FUNCTION
         DC    FL4'Ø'          RESERVED
         DC    CL8' '          ENTRY POINT-USES FUNCTION ADDR IF BLANK
         DC    CL8' '          RESERVED
*
         DC    CL8'GETPID  '   FUNCTION NAME
         DC    VL4(GETPID)     ADDR OF FUNCTION
         DC    FL4'Ø'          RESERVED
         DC    CL8' '          ENTRY POINT-USES FUNCTION ADDR IF BLANK
         DC    CL8' '          RESERVED
*********************************************************************
* FUNCTION WHICH GIVEN A PARTITION ID WILL RETURN 1 IF THE          *
* PARTITION HAS A JOB RUNNING IN IT OR Ø IF NO JOB IS RUNNING.      *
* PARTITION ID CAN BE FOR EITHER A STATIC OR DYNAMIC PARTITION.     *
*********************************************************************
ISBUSY   CSECT
ISBUSY   AMODE 31
ISBUSY   RMODE ANY
         USING ISBUSY,R15
         STM   R14,R12,12(R13)
         LR    R12,R15                  COPY ENTRY ADDRESS
         DROP  R15
         USING ISBUSY,R12               AND USE AS BASE
         LR    R2,R13
         LA    R13,SAVEREGS
         ST    R13,8(R2)
         ST    R2,4(R13)
         B     BEGIN
         DC    CL8'ISBUSY '
SAVEREGS DS    18F
BEGIN    EQU   *
         SR    R15,R15                  CLEAR REG
         LA    R15,1                    RC = 1 (ERRORS)
         USING EFPL,R1
         L     R8,EFPLEVAL              POINTER TO ADDR OF EVALBLOCK
         L     R8,Ø(R8)                 ADDR OF EVALBLOCK
         USING EVALBLOCK,R8
         L     R3,EFPLARG               REXX ARGS
         L     R4,Ø(R3)                 GET ADDR OF 1ST ARG
```

© 1998. Reproduction prohibited. Please inform Xephon of any infringement.

```
         C     R4,=F'-1'                  NO PARMS?
         BE    ERROR
         L     R5,ANCHOR                  APCBATAB (1ST PCB IS AR)
LOOPP    EQU   *
*        MODESET KEY=ZERO                 MAY OR MAY NOT NEED PKØ
         LA    R5,4(,R5)                  BUMP THE PCB POINTER
         CLI   Ø(R5),X'FF'                END OF PCB TABLE?
         BE    RETØ
         L     R6,Ø(R5)                   LOAD A PCB
         LTR   R6,R6                      EMPTY SLOT
         BZ    LOOPP
         L     R6,Ø(R5)                   LOAD A PCB
         USING PCBADR,R6
         L     R2,PCECOMRA                GET COMREG
         USING COMREG,R2
         MVC   PARTID,PCELID              SAVE PARTID
         MVC   FLAG,PCEFLAG               SAVE FLAG
         MODESET KEY=USER
         CLC   PARTID(2),Ø(R4)            IS IT THE RIGHT PARTID?
         BE    FOUND
         B     LOOPP
*********************************************************************
*  FOUND AN ACTIVE PARTITION. IF IT IS A STATIC PARTITION WE NEED   *
*  TO CHECK FURTHER TO SEE IF A PGM IS RUNNING IN IT. IF IT IS A    *
*  DYNAMIC PARTITION WE CAN RETURN SINCE IT WOULD NOT HAVE A PCB IF *
*  IT WAS INACTIVE.                                                 *
*********************************************************************
FOUND    EQU   *
         SR    R15,R15                    CLEAR RC
         TM    FLAG,X'4Ø'                 TEST FOR DYNAMIC PARTITION
         BO    RET1                       MUST BE DYNAMIC
         CLC   COMNAME(8),=CL8'NO NAME '  INACTIVE  STATIC PARTITION?
         BNE   RET1
RETØ     EQU   *
         SR    R15,R15                    CLEAR REG
         LA    R15,1                      RETURN FIELD LENGTH
         ST    R15,EVALBLOCK_EVLEN        SAVE IT FOR REXX
         MVC   EVALBLOCK_EVDATA(1),=C'Ø'  Ø = ISBUSY
         B     RETURN
RET1     EQU   *
         SR    R15,R15                    CLEAR REG
         LA    R15,1                      RETURN FIELD LENGTH
         ST    R15,EVALBLOCK_EVLEN        SAVE IT FOR REXX
         MVC   EVALBLOCK_EVDATA(1),=C'1'  1 = INACTIVE
RETURN   EQU   *
         SR    R15,R15                    CLEAR REG
ERROR    EQU   *
         L     R13,4(R13)                 RETURN TO CALLER
         LM    R2,R12,28(R13)
         L     R14,12(R13)
```


© 1998. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.

```
         MVI   12(R13),X'FF'
         BR    R14                       R15 CONTAINS RC FOR REXX
PARTID   DC    CL2' '
FLAG     DC    X'ØØ'
PHOLD    DC    CL2' '
PADDR    DC    F'Ø'
         LTORG
*        END   ISBUSY
*********************************************************************
* FUNCTION WHICH GIVEN A JOBNAME WILL RETURN THE PARTITION ID       *
* THAT THE JOB IS RUNNING IN. IF THE JOBNAME IS NOT FOUND A BLANK   *
* PARTID WILL BE RETURNED. JOBNAME MAY BE TERMINATED WITH AN        *
* ASTERISK WHICH WILL CAUSE A GENERIC MATCH ON CHARACTERS UP TO     *
* THE ASTERISK.                                                     *
*********************************************************************
ISJOB    CSECT
ISJOB    AMODE 31
ISJOB    RMODE ANY
         USING ISJOB,R15
         STM   R14,R12,12(R13)
         LR    R12,R15                   COPY ENTRY ADDRESS
         DROP  R15
         USING ISJOB,R12                 AND USE AS BASE
         LR    R2,R13
         LA    R13,SAVEREG
         ST    R13,8(R2)
         ST    R2,4(R13)
         B     BEGIN1
         DC    CL8'ISJOB'
SAVEREG  DS    18F
BEGIN1   EQU   *
         SR    R15,R15                   CLEAR REG
         LA    R15,1                     RC = 1 (ERRORS)
         USING EFPL,R1
         L     R8,EFPLEVAL               POINTER TO ADDR OF EVALBLOCK
         L     R8,Ø(R8)                  ADDR OF EVALBLOCK
         USING EVALBLOCK,R8
         L     R3,EFPLARG                REXX ARGS
         L     R4,Ø(R3)                  GET ADDR OF 1ST ARG
         C     R4,=F'-1'
         BE    ERROR1
         L     R5,ANCHOR                 APCBATAB (1ST PCB IS AR)
LOOPP1   EQU   *
*        MODESET KEY=ZERO                MAY OR MAY NOT NEED PKØ
         LA    R5,4(,R5)                 BUMP THE PCB POINTER
         CLI   Ø(R5),X'FF'               END OF PCB TABLE?
         BE    ENDPCB1
         L     R6,Ø(R5)                  LOAD A PCB
         LTR   R6,R6                     EMPTY SLOT
         BZ    LOOPP1
```

© 1998. Reproduction prohibited. Please inform Xephon of any infringement.

```
         L     R6,Ø(R5)                  LOAD A PCB
         USING PCBADR,R6
         L     R2,PCECOMRA               GET COMREG
         USING COMREG,R2
         MVC   PARTID1,PCELID            SAVE PARTID
         MVC   JNAME,COMNAME             SAVE JOB NAME
         MODESET KEY=USER                RESET PROTECT KEY
         MVI   ANAME,C' '                CLEAR ANAME
         MVC   ANAME+1(L'ANAME-1),ANAME
         L     R4,Ø(R3)                  1ST ARG
         L     R7,4(R3)                  LENGTH OF 1ST ARG
         CH    R7,=H'8'                  LENGTH > 8?
         BNH   DOWN1                     NO, CONTINUE
         LA    R7,8                      SET TO MAX LEN
DOWN1    EQU   *
         BCTR  R7,Ø                      LEN - 1
         EX    R7,SAVEARG                SAVE 1ST ARG
         MVI   TRTAB+C'*',X'FF'          SET CHAR FOR SEARCH
         LA    R7,ANAME                  POINT TO JOBNAME
         LA    R1,8(,R7)                 POINT TO MAX POSITION
         TRT   Ø(8,R7),TRTAB             LOOK FOR *
         CLI   Ø(R1),C'*'                FOUND *?
         BNE   MAXLEN                    NO, SET TO MAX LEN
         SR    R1,R7                     CALC LENGTH
         LR    R7,R1                     SAVE IT
         B     CHKNAME
MAXLEN   EQU   *
         LA    R7,8                      SET TO MAXLEN
CHKNAME  EQU   *
         BCTR  R7,Ø                      SUBTRACT 1 FOR THE EX
         EX    R7,COMPARE
         BE    FOUND1                    YES, FOUND IT
         B     LOOPP1                    GO CHECK PCB FIELDS
ENDPCB1  EQU   *
         SR    R15,R15                   NOT FOUND
         LA    R15,Ø                     RETURN NULL LENGTH
         ST    R15,EVALBLOCK_EVLEN       SAVE IT FOR REXX
         MVC   EVALBLOCK_EVDATA(2),=C'  ' SPACES=NOT FOUND
         B     RETURN1                   GO EXIT FUNCTION
FOUND1   EQU   *
         SR    R15,R15                   CLEAR RC
         LA    R15,2                     RETURN LENGTH
         ST    R15,EVALBLOCK_EVLEN       SAVE IT FOR REXX
         MVC   EVALBLOCK_EVDATA(2),PARTID1   RETURN PARTID
RETURN1  EQU   *
         SR    R15,R15
ERROR1   EQU   *
         L     R13,4(R13)                RESTORE REGS
         LM    R2,R12,28(R13)
         L     R14,12(R13)
```


© 1998. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.

```
         BR    R14                        RETURN
ANCHOR   EQU   X'2C4'                     PCB TABLE ANCHOR
PARTID1  DC    CL2' '
ANAME    DC    CL16' '                    SAVE AREA
         DC    C'JNAME'
JNAME    DC    CL8' '
COMPARE  CLC   ANAME(Ø),JNAME            CHK JOB NAME
SAVEARG  MVC   ANAME(Ø),Ø(R4)            SAVE ARG IN ANAME
TRTAB    DC    256X'ØØ'
         LTORG
*        END   ISJOB
*********************************************************************
*                                                                   *
* THIS FUNCTION EXTRACTS THE CPU ID AND RETURNS IT                  *
*                                                                   *
*********************************************************************
CPUID    CSECT
CPUID    AMODE 31
CPUID    RMODE ANY
         USING CPUID,R15
         STM   R14,R12,12(R13)
         LR    R12,R15                    COPY ENTRY ADDRESS
         DROP  R15
         USING CPUID,R12                  AND USE AS BASE
         LR    R2,R13
         LA    R13,SAVEREG2
         ST    R13,8(R2)
         ST    R2,4(R13)
         B     BEGIN2
         DC    CL8'CPUID '
SAVEREG2 DS    18F
BEGIN2   EQU   *
         SR    R15,R15                    CLEAR REG
         LA    R15,1                      RC = 1 (ERRORS)
         L     R8,EFPLEVAL                POINTER TO ADDR OF EVALBLOCK
         L     R8,Ø(R8)                   ADDR OF EVALBLOCK
         USING EVALBLOCK,R8
         L     R3,EFPLARG                 REXX ARGS
         L     R4,Ø(R3)                   GET ADDR OF 1ST ARG
         LA    R1Ø,1Ø
         EXTRACT ID=CPUID,AREA=CPUSTOR,LEN=(1Ø)
         MVC   DWORD(3),CPUSTOR+1
         UNPK  CPUSTOR(8),DWORD(4)
         SR    R15,R15                    CLEAR RC
         LA    R15,5                      RETURN LENGTH
         ST    R15,EVALBLOCK_EVLEN        SAVE IT FOR REXX
         MVC   EVALBLOCK_EVDATA(5),CPUSTOR+1 RETURN CPUID
RETURN2  EQU   *
         SR    R15,R15                    CLEAR REG
         L     R13,4(R13)                 RETURN TO CALLER
```

© 1998. Reproduction prohibited. Please inform Xephon of any infringement.

```
         LM    R2,R12,28(R13)
         L     R14,12(R13)
         MVI   12(R13),X'FF'
         BR    R14                       R15 CONTAINS RC FOR REXX
         LTORG
DWORD    DC    D'Ø'
CPUSTOR  DC    CL1Ø' '
*        END   CPUID
*********************************************************************
*                                                                   *
* THIS FUNCTION RETURNS THE PIK OF A PARTITION                      *
*                                                                   *
*********************************************************************
GETPIK   CSECT
GETPIK   AMODE 31
GETPIK   RMODE ANY
         USING GETPIK,R15
         STM   R14,R12,12(R13)
         LR    R12,R15                   COPY ENTRY ADDRESS
         DROP  R15
         USING GETPIK,R12                AND USE AS BASE
         LR    R2,R13
         LA    R13,SAVEREG3
         ST    R13,8(R2)
         ST    R2,4(R13)
         B     BEGIN3
         DC    CL8'GETPIK  '
SAVEREG3 DS    18F
BEGIN3   EQU   *
         SR    R15,R15                   CLEAR REG
         LA    R15,1                     RC = 1 (ERRORS)
         USING EFPL,R1
         L     R8,EFPLEVAL               POINTER TO ADDR OF EVALBLOCK
         L     R8,Ø(R8)                  ADDR OF EVALBLOCK
         USING EVALBLOCK,R8
         L     R3,EFPLARG                REXX ARGS
         L     R4,Ø(R3)                  GET ADDR OF 1ST ARG
         C     R4,=F'-1'                 NO PARMS?
         BE    ERROR3
         MVC   LOGID,Ø(R4)               SAVE ARG
         GETFLD FIELD=PIK,LOGID=LOGID    GET PIK
         LTR   R15,R15                   Ø=PARTID FOUND
         BZ    OK3
         SR    R1,R1                     CLEAR R1 TO INDICATE ERR
OK3      EQU   *
         ST    R1,DWORD2+4               CONVERT TO HEX STRING
         UNPK  WORK2+1(1Ø),DWORD2+4(5)
         TR    WORK2+1(8),HEXTBL-X'FØ'
         MVC   PIK(4),WORK2+6            RETURN 4 CHARS
         SR    R15,R15                   CLEAR RC
```


© 1998. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.

```
         LA    R15,4                       RETURN LENGTH
         ST    R15,EVALBLOCK_EVLEN         SAVE IT FOR REXX
         MVC   EVALBLOCK_EVDATA(4),PIK     RETURN PIK
RETURN3  EQU   *
         SR    R15,R15                     CLEAR REG
ERROR3   EQU   *
         L     R13,4(R13)                  RETURN TO CALLER
         LM    R2,R12,28(R13)
         L     R14,12(R13)
         MVI   12(R13),X'FF'
         BR    R14                         R15 CONTAINS RC FOR REXX
         LTORG
DWORD2   DC    D'0'
PIK      DC    F'0'
         DC    C'LOGID'
LOGID    DC    H'0'
         DC    XL240'00'
HEXTBL   DC    C'0123456789ABCDEF'
WORK2    DC    CL10' '
*        END   GETPIK
*********************************************************************
*                                                                   *
* THIS FUNCTION RETURNS THE 2 CHAR PARTITION ID FOR A PIK           *
*   NOTE: WILL CAUSE ABEND IF SUPPLIED WITH AN INVALID PIK          *
*                                                                   *
*********************************************************************
GETPID   CSECT
GETPID   AMODE 31
GETPID   RMODE ANY
         USING GETPID,R15
         STM   R14,R12,12(R13)
         LR    R12,R15                     COPY ENTRY ADDRESS
         DROP  R15
         USING GETPID,R12                  AND USE AS BASE
         LR    R2,R13
         LA    R13,SAVEREG4
         ST    R13,8(R2)
         ST    R2,4(R13)
         B     BEGIN4
         DC    CL8'GETPID  '
SAVEREG4 DS    18F
BEGIN4   EQU   *
         SR    R15,R15                     CLEAR REG
         LA    R15,1                       RC = 1 (ERRORS)
         USING EFPL,R1
         L     R8,EFPLEVAL                 POINTER TO ADDR OF EVALBLOCK
         L     R8,0(R8)                    ADDR OF EVALBLOCK
         USING EVALBLOCK,R8
         L     R3,EFPLARG                  REXX ARGS
         L     R4,0(R3)                    GET ADDR OF 1ST ARG
```

© 1998. Reproduction prohibited. Please inform Xephon of any infringement.

```
         C     R4,=F'-1'                  NO PARMS?
         BE    ERROR4
         MVC   WORK5(4),Ø(R4)             MOVE IT TO WORK AREA
         TR    WORK5(4),HEXTAB            CONVERT STRING
         PACK  WORK6(4),WORK5(5)          TO HEX
         MVC   PIK2(4),=F'Ø'              INIT WORK AREA
         MVC   PIK2+2(2),WORK6+1          MOVE IT TO PIK
         L     R1,PIK2                    PUT PIK IN REG1
         GETFLD FIELD=LOGID,PART=(1)      GET PART ID
         LTR   R15,R15                    Ø=PIK FOUND
         BZ    OK4
         SR    R1,R1                      CLEAR R1 TO INDICATE ERR
OK4      EQU   *
         STH   R1,LOGID2                  SAVE PART ID
         SR    R15,R15                    CLEAR RC
         LA    R15,2                      RETURN LENGTH
         ST    R15,EVALBLOCK_EVLEN        SAVE IT FOR REXX
         MVC   EVALBLOCK_EVDATA(2),LOGID2    RETURN PARTID
RETURN4  EQU   *
         SR    R15,R15                    CLEAR REG
ERROR4   EQU   *
         L     R13,4(R13)                 RETURN TO CALLER
         LM    R2,R12,28(R13)
         L     R14,12(R13)
         MVI   12(R13),X'FF'
         BR    R14                        R15 CONTAINS RC FOR REXX
         LTORG
PIK2     DC    F'Ø'
LOGID2   DC    H'Ø'
         DC    C'WORK5'
WORK5    DC    C'ØØØØØ'
WORK6    DC    F'Ø'
HEXTAB   DC    193X'ØØ'
         DC    X'ØAØBØCØDØEØF'             ABCDEF
         DC    41X'ØØ'
         DC    X'FØF1F2F3F4F5F6F7F8F9'       Ø123456789
         END   ARXFUSER
St
/*
// IF $RC>8 THEN
// GOTO SKP
// EXEC LNKEDT
/. SKP
/*
/&
```

Stanley Stewart
(USA)

© Xephon 1998

© 1998. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.

Contributing to *VSE Update*

In addition to *VSE Update*, the Xephon family of *Update* publications now includes *CICS Update*, *VM Update*, *MVS Update*, *TCP/SNA Update*, *VSAM Update*, *DB2 Update*, *RACF Update*, *AIX Update*, *Domino Update*, *NT Update*, *Oracle Update*, and *Web Update*. Although the articles published are of a very high standard, the vast majority are not written by professional writers, and we rely heavily on our readers themselves taking the time and trouble to share their experiences with others. Many have discovered that writing an article is not the daunting task that it might appear to be at first glance.

They have found that the effort needed to pass on valuable information to others is more than offset by our generous terms and conditions and the recognition they gain from their fellow professionals. Often, just a few hundred words are sufficient to describe a problem and the steps taken to solve it.

If you have ever experienced any difficulties with VSE, or made an interesting discovery, you could receive a cash payment, a free subscription to any of our *Updates*, or a credit against any of Xephon's wide range of products and services, simply by telling us all about it. For a copy of our *Notes for Contributors*, which explains the terms and conditions under which we publish articles, please write to the editor, Fiona Hewitt, at any of the addresses shown on page 2, or e-mail her on 100336.1412@compuserve.com.

© 1998. Reproduction prohibited. Please inform Xephon of any infringement.

Macro 4 has announced Version 3.3 of its VSAMTUNE VSAM data management package, which supports the latest version of VSE and is year 2000-compliant.

Improved facilities let users write tailored reports and run them as either a batch job or on an *ad hoc* basis. The Alternate Index Builder is claimed to reduce file search times and improve file access dramatically. There's also claimed better dynamic buffering and performance tuning plus a file tuning early-warning system.

It's out now for VSE/ESA and VSE, as well as OS/390 and MVS, on a rental or perpetual lease basis.

For further information contact:
Macro 4, The Orangery, Turners Hill Road, Worth, Crawley, West Sussex, RH10 4SS, UK.
Tel: (01293) 886060.
Macro 4, 35 Waterview Blvd, PO Box 292, Parsippany, NJ 07054-0292, USA.
Tel: (201) 402 8000.
URL: http://www.macro4.com.

* * *

Data 21 is shipping the VSE version of its IpServer for CICS. It's a Web Server that runs natively within CICS, allowing it to directly take advantage of the multitasking, security, reliability, scalability, and transaction serving facilities within the System/390 environment. A CICS Web Server is complemented by a native CICS CGI interface to simplify the Web-enabling of CICS applications.

The CGI interface lets VSE users include existing hardware, software, and programming skills in applications. It's multi-threaded, and allows programmers to write CGIs in CICS command-level languages. SSI directives are supported for helping to separate business logic from HTML-based Web presentation.

Web authors can create and upload standard Web pages that include SSI directives for incorporating dynamic CGI data. CGI programs themselves return raw host data and the filename of the Web page that the data is to be included in. Then, IpServer loads the Web page, replaces the SSI directives with the CGI's data, and sends the page on to the browser.

As a result, the Web pages, which are simply presenting the data, can be changed at anytime without requiring additional programming.

For further information contact:
Data 21, 18093-H South Prairie Avenue, Torrance, CA 90504-3700, USA.
Tel: (702) 832 2191.
URL: http://www.data21.com.

* * *

IBM has announced enhancements to VSE. VSE/ESA TCP/IP performance gets a boost with the addition of NFS Version 2 protocol support, which is available on OS/390, AIX, and OS/2, and enables a VSE/ESA system to act as a file server. Systems with the NFS Version 2 client function can access VSE/VSAM files, VSE/ESA Library files, and VSE/POWER queues as if those files were residing locally. Other enhancements include a DFSORT/VSE feature, designed to decrease sorting times through greater data-in-memory exploitation of the System/390 architecture.

For further information, contact your local IBM representative.