Calling REXX from Cobol, C, and PLI - Examples

These examples demonstrate how to call a REXX procedure from one of the higher level languages Cobol, C, and PLI. REXX offers two routines to call a REXX program from a non-REXX program:

- ARXJCL: easier to use, but fewer functionality
- ARXEXEC: more difficult to use, but offers more flexibility

Language	ARXJCL	ARXEXEC
COBOL	COBOL-call to ARXJCL	COBOL-call to ARXEXEC
C	C-call to ARXJCL	C-call to ARXEXEC
PLI	PLI-call to ARXJCL	PLI-call to ARXEXEC

Calling ARXJCL from COBOL

```
* $$ JOB JNM=MINICOBC,DISP=D,CLASS=A,NTFY=YES
* $$ LST DISP=D,CLASS=Q,PRI=3
// JOB MINICOBC COMPILE PROGRAM MINICOB
// EXEC LIBR
  ACC S=PRIMARY.USCH
  CAT HELLO.PROC R=Y
/* simple rexx exec */
Arg n
If n='' Then n=1
Do i=1 to n
  fc = Assgn('STDOUT','SYSLST')
  Say 'hello'
  fc = Assgn('STDOUT','SYSLOG')
  Say 'hello'
End
Return 99
/+
/*
// LIBDEF *,SEARCH=(PRIMARY.USCH,PRD2.SCEEBASE,PRD2.PROD)
// LIBDEF PHASE,CATALOG=PRIMARY.USCH
// OPTION ERRS,SXREF,SYM,NODECK,CATAL
   PHASE USCHI,*
/. ENDCAT
// EXEC IGYCRCTL,SIZE=IGYCRCTL
CBL NOADV,NODYN,NONAME,NONUMBER,QUOTE,SEQ,XREF,VBREF,DUMP,LIST
       TITLE "USCHI COBOL TEST PROGRAM  ".
       Identification Division.

       Program-id.    USCHI.
IA0040 Author.        Ursula Braun-Krahl.
      *Language.      IBM COBOL for VSE/ESA Version 1 Release 1.
      ***                                                            **
      ***  Invoke REXX procedure HELLO with parameter 3 using ARXJCL **
      ***                                                            **
      ***  Expected display messages:                                **
      ***    "PROGRAM USCHI    - BEGINNING"                          **
      ***    "PROGRAM USCHI    - NORMAL END"                         **
      ***                                                            **
      ****************************************************************
```



```
       Environment division.
IA0970 Configuration section.
         Special-names.
       Input-output section.
         File-control.
           Select PRINT-FILE
               assign to SYS014-S-UPDPRNT
               file status is UPDPRINT-FILE-STATUS.
       Data division.
       File section.
IA1570 FD  PRINT-FILE
           recording mode F
           block 0 records
           record 121 characters
           label record standard.
IA1620   01 print-record                  pic x(121).
       Working-storage section.
         01 Working-storage-for-USCHI     pic x.

         77 PGM-NAME                      pic X(8).

         01 ARGUMENT.
            03 ARG-SIZE                   pic 9(2) comp.
            03 ARG-CHAR                   pic x(8).

         77 UPDPRINT-file-status          pic xx.

      /*****************************************************************
      ***                 D O   M A I N   L O G I C                   **
      *****************************************************************
       procedure division.
         000-do-main-logic.
           display "PROGRAM USCHI    - Beginning".
           display RETURN-CODE.
           move "ARXJCL  " to PGM-NAME.
           move "HELLO  3" to ARG-CHAR.
           move 8 to arg-size.
           CALL PGM-NAME USING ARGUMENT.
           CANCEL PGM-NAME.
           display RETURN-CODE.
           display "PROGRAM USCHI    - Normal end".
           stop run.

IA9990 end program USCHI.
/*
// EXEC LNKEDT,SIZE=256K
// IF $MRC >= 4 THEN
// GOTO JOBEND
* run the program
// OPTION DUMP
// EXEC USCHI
/. JOBEND
/&
* $$ EOJ
```


Calling ARXEXEC from COBOL

```
* $$ JOB JNM=MINICOBC,DISP=D,CLASS=A,NTFY=YES
* $$ LST DISP=D,CLASS=Q,PRI=3
// JOB MINICOBC COMPILE PROGRAM MINICOB
// EXEC LIBR
  ACC S=PRIMARY.USCH
  CAT HELLO.PROC R=Y
/* simple rexx exec */
Arg n
If n='' Then n=1
Do i=1 to n
  fc = Assgn('STDOUT','SYSLST')
  Say 'hello'
  fc = Assgn('STDOUT','SYSLOG')
  Say 'hello'
End
Return date('W')
/+
/*
// LIBDEF *,SEARCH=(PRIMARY.USCH,PRD2.SCEEBASE,PRD2.PROD)
// LIBDEF PHASE,CATALOG=PRIMARY.USCH
// OPTION ERRS,SXREF,SYM,NODECK,CATAL
   PHASE USCHI,*
/. ENDCAT
// EXEC IGYCRCTL,SIZE=IGYCRCTL
CBL NOADV,NODYN,NONAME,NONUMBER,QUOTE,SEQ,XREF,VBREF,DUMP,LIST
CBL TRUNC(OPT)
       TITLE "USCHI COBOL TEST PROGRAM  ".
       Identification Division.

       Program-id.    USCHI.
IA0040 Author.        Ursula Braun-Krahl.
      *Language.      IBM COBOL for VSE/ESA Version 1 Release 1.
      ***                                                            **
      ***  Invoke REXX procedure HELLO with parameter 3 using ARXEXEC**
      ***                                                            **
      ***  Expected display messages:                                **
      ***    "PROGRAM USCHI    - BEGINNING"                          **
      ***    "PROGRAM USCHI    - NORMAL END"                         **
      ***                                                            **
      ****************************************************************

       Environment division.
IA0970 Configuration section.
         Special-names.
       Input-output section.
         File-control.
           Select PRINT-FILE
               assign to SYS014-S-UPDPRNT
               file status is UPDPRINT-FILE-STATUS.
       Data division.
       File section.
IA1570 FD  PRINT-FILE
           recording mode F
           block 0 records
           record 121 characters
           label record standard.
IA1620   01 print-record                  pic x(121).
       Working-storage section.
```



```
   01 Working-storage-for-USCHI      pic x.

   77 PGM-NAME                       pic X(8).

   01 EXECBLK.
      03 EXECBLK-ACRYN                pic X(8).
      03 EXECBLK-LENGTH               pic S9(8) binary.
      03 EXECBLK-reserved             pic S9(8) binary.
      03 EXECBLK-MEMBER               pic X(8).
      03 EXECBLK-DDNAME               pic X(8).
      03 EXECBLK-SUBCOM               pic X(8).
      03 EXECBLK-DSNPTR               POINTER.
      03 EXECBLK-DSNLEN               pic 9(4) comp.

   01 EVALBLK.
      03 EVALBLK-EVPAD1               pic S9(8) binary.
      03 EVALBLK-EVSIZE               pic S9(8) binary.
      03 EVALBLK-EVLEN                pic S9(8) binary.
      03 EVALBLK-EVPAD2               pic S9(8) binary.
      03 EVALBLK-EVDATA               pic x(256).

   77 flags                           pic S9(8) binary.
   77 REXX-return-code                pic S9(8) binary.
   77 dummy-zero                      pic S9(8) binary.
   01 ARGUMENT.
     02 ARGUMENT-1 OCCURS 1 TIMES.
      05 ARGSTRING-PTR                POINTER.
      05 ARGSTRING-LENGTH             pic S9(8) binary.
     02 ARGSTRING-LAST1               pic S9(8) binary.
     02 ARGSTRING-LAST2               pic S9(8) binary.
   77 arg1                            pic x(1).
   77 execblk-ptr                     POINTER.
   77 argtable-ptr                    POINTER.
   77 evalblk-ptr                     POINTER.

   77 UPDPRINT-file-status            pic xx.

/*****************************************************************
***                 D O   M A I N   L O G I C                  **
*****************************************************************
 procedure division.
   000-do-main-logic.
     display "PROGRAM USCHI    - Beginning".

     move "3" to arg1.
     call "GET-ARG1-PTR" using arg1 ARGSTRING-PTR(1).
     move 1 to ARGSTRING-LENGTH(1).
     move -1 to ARGSTRING-LAST1.
     move -1 to ARGSTRING-LAST2.
     call "GET-ARGUMENT-PTR" using argument argtable-ptr.
     move "ARXEXECB" to EXECBLK-ACRYN.
     move 48 to EXECBLK-LENGTH.
     move 0 to EXECBLK-reserved.
     move "HELLO   " to EXECBLK-MEMBER.
     move "        " to EXECBLK-DDNAME.
     move "        " to EXECBLK-SUBCOM.
     set EXECBLK-DSNPTR to NULL.
     move 0 to EXECBLK-DSNLEN.
     call "GET-EXECBLK-PTR" using EXECBLK execblk-ptr.
     move 0 to EVALBLK-EVPAD1.
     move 34 to EVALBLK-EVSIZE.
     move 0 to EVALBLK-EVLEN.
     move 0 to EVALBLK-EVPAD2.
```



```
           call "GET-EVALBLK-PTR" using EVALBLK evalblk-ptr.
           move 0 to dummy-zero.
           move 536870912 to flags.
           move 0 to REXX-return-code.

           move "ARXEXEC " to PGM-NAME.
           CALL PGM-NAME USING execblk-ptr
                               argtable-ptr
                               flags
                               dummy-zero
                               dummy-zero
                               evalblk-ptr
                               dummy-zero
                               dummy-zero
                               dummy-zero
                               REXX-return-code.
           CANCEL PGM-NAME.

           display "REXX return code is: " REXX-return-code.
           display "REXX result is: " EVALBLK-EVDATA.
           display "PROGRAM USCHI    - Normal end".
           stop run.

       Identification Division.
       Program-id.    GET-ARG1-PTR.
       Environment division.
       Data division.
       Working-storage section.
       Linkage section.
         77 arg1                          pic x(1).
         77 arg-ptr                       POINTER.

       procedure division using arg1 arg-ptr.
           set arg-ptr to address of arg1.
           goback.
       end program GET-ARG1-PTR.

       Identification Division.
       Program-id.    GET-ARGUMENT-PTR.
       Environment division.
       Data division.
       Working-storage section.
       Linkage section.
         01 ARGUMENT.
           02 ARGUMENT-1 OCCURS 1 TIMES.
            05 ARGSTRING-PTR              POINTER.
            05 ARGSTRING-LENGTH           pic S9(8) binary.
           02 ARGSTRING-LAST1             pic S9(8) binary.
           02 ARGSTRING-LAST2             pic S9(8) binary.
         77 argtable-ptr                  POINTER.

       procedure division using ARGUMENT argtable-ptr.
           set argtable-ptr to address of ARGUMENT.
           goback.
       end program GET-ARGUMENT-PTR.

       Identification Division.
       Program-id.    GET-EXECBLK-PTR.
       Environment division.
       Data division.
       Working-storage section.
       Linkage section.
         01 EXECBLK.
```



```
            03 EXECBLK-ACRYN               pic X(8).
            03 EXECBLK-LENGTH              pic 9(4) comp.
            03 EXECBLK-reserved            pic 9(4) comp.
            03 EXECBLK-MEMBER              pic X(8).
            03 EXECBLK-DDNAME              pic X(8).
            03 EXECBLK-SUBCOM              pic X(8).
            03 EXECBLK-DSNPTR              POINTER.
            03 EXECBLK-DSNLEN              pic 9(4) comp.
          77 execblk-ptr                   POINTER.

        procedure division using EXECBLK execblk-ptr.
            set execblk-ptr to address of EXECBLK.
            goback.
        end program GET-EXECBLK-PTR.

        Identification Division.
        Program-id.    GET-EVALBLK-PTR.
        Environment division.
        Data division.
        Working-storage section.
        Linkage section.
          01 EVALBLK.
             03 EVALBLK-EVPAD1             pic 9(4) comp.
             03 EVALBLK-EVSIZE             pic 9(4) comp.
             03 EVALBLK-EVLEN              pic 9(4) comp.
             03 EVALBLK-EVPAD2             pic 9(4) comp.
             03 EVALBLK-EVDATA             pic x(256).
          77 evalblk-ptr                   POINTER.

        procedure division using EVALBLK evalblk-ptr.
            set evalblk-ptr to address of EVALBLK.
            goback.
        end program GET-EVALBLK-PTR.

IA9990 end program USCHI.
/*
// EXEC LNKEDT,SIZE=256K
// IF $MRC >= 4 THEN
// GOTO JOBEND
* run the program
// OPTION DUMP
// EXEC USCHI
/. JOBEND
/&
* $$ EOJ
```


Calling ARXJCL from C

```
* $$ JOB JNM=MINIC,CLASS=0,DISP=D
// JOB MINIC
// LIBDEF *,SEARCH=(PRIMARY.USCH,PRD2.DBASE,PRD2.SCEEBASE)
// EXEC LIBR
  ACC S=PRIMARY.USCH
  DELETE USCHI.PHASE
  CAT HELLO.PROC R=Y
/* simple rexx exec */
Arg n
If n='' Then n=1
Do i=1 to n
  fc = Assgn('STDOUT','SYSLST')
  Say 'hello'
  fc = Assgn('STDOUT','SYSLOG')
  Say 'hello'
End
Return 99
/+
/*
// LIBDEF *,SEARCH=(PRIMARY.USCH,PRD2.DBASE,PRD2.SCEEBASE)
// LIBDEF  OBJ,SEARCH=(PRIMARY.USCH,PRD2.SCEEBASE)
// LIBDEF  PHASE,SEARCH=(PRD2.SCEEBASE),CATALOG=PRIMARY.USCH
// OPTION LINK,CATAL,LIST,LISTX
// EXEC EDCCOMP,SIZE=EDCCOMP,PARM='SOURCE,XREF,NAME(USCHI)'
 /* invoke REXX procedure HELLO with parameter 3 using ARXIJCL */
#include
#pragma map(InvREXX,"ARXIJCL ")
#pragma linkage(InvREXX,OS)
typedef struct ARXJCL_type
 {
  short int  arg_length;
  char argument[9];
 } ARXJCL_type;
int InvREXX(ARXJCL_type* param_ptr); /* Function invokes REXX */
ARXJCL_type this_param;
ARXJCL_type* param_ptr;
int return_code;
main()
{ printf("start of USCHI\n");
  this_param.arg_length = 8;
  strcpy(this_param.argument,"HELLO  3");
  param_ptr = &this_param;
  return_code = InvREXX(param_ptr);
  printf("REXX return code: %d\n", return_code);
  printf("end of USCHI\n");
}
/*
// EXEC    LNKEDT
/*
// IF $MRC > 4 THEN
// GOTO JOBEND
// LIBDEF *,SEARCH=(PRIMARY.USCH,PRD2.DBASE,PRD2.SCEEBASE)
// EXEC USCHI
/. JOBEND
/&
* $$ EOJ
```


Calling ARXEXEC from C

```
* $$ JOB JNM=MINIC,CLASS=0,DISP=D
// JOB MINIC
// LIBDEF *,SEARCH=(PRIMARY.USCH,PRD2.DBASE,PRD2.SCEEBASE)
// EXEC LIBR
  ACC S=PRIMARY.USCH
  DELETE USCHI.PHASE
  CATALOG HELLO.PROC R=Y
/* simple rexx exec */
trace o
Arg n
If n='' Then n=1
Do i=1 to n
  fc = Assgn('STDOUT','SYSLST')
  Say 'hello'
  fc = Assgn('STDOUT','SYSLOG')
  Say 'hello'
End
Return date('W')
/+
/*
// LIBDEF *,SEARCH=(PRIMARY.USCH,PRD2.DBASE,PRD2.SCEEBASE)
// LIBDEF  OBJ,SEARCH=(PRIMARY.USCH,PRD2.SCEEBASE)
// LIBDEF  PHASE,SEARCH=(PRD2.SCEEBASE),CATALOG=PRIMARY.USCH
// OPTION LINK,CATAL,LIST,LISTX
// EXEC EDCCOMP,SIZE=EDCCOMP,PARM='SOURCE,XREF,NAME(USCHI)'
 /* invoke REXX procedure HELLO with parameter 3 using ARXCEXEC */
#include
#pragma map(InvREXX,"ARXCEXEC")
#pragma linkage(InvREXX,OS)
typedef struct EXECBLK_type
 {
  char EXECBLK_ACRYN[8];
  int  EXECBLK_LENGTH;
  int  EXECBLK_reserved;
  char EXECBLK_MEMBER[8];
  char EXECBLK_DDNAME[8];
  char EXECBLK_SUBCOM[8];
  void * EXECBLK_DSNPTR;
  int  EXECBLK_DSNLEN;
 } EXECBLK_type;
typedef struct one_parameter_type
 {
  void * ARGSTRING_PTR;
  int    ARGSTRING_LENGTH;
 } one_parameter_type;
typedef struct EVALBLK_type
 {
  int  EVALBLK_EVPAD1;
  int  EVALBLK_EVSIZE;
  int  EVALBLK_EVLEN;
  int  EVALBLK_EVPAD2;
  char EVALBLK_EVDATA[256];
 } EVALBLK_type;
typedef struct ARXEXEC_type
 {
  EXECBLK_type ** execblk_ptr;
  one_parameter_type ** argtable_ptr;
  int * flags_ptr;
  int * instblk_ptr;
  int * reserved_parm5;
```



```
  EVALBLK_type ** evalblk_ptr;
  int * reserved_workarea_ptr;
  int * reserved_userfield_ptr;
  int * reserved_envblock_ptr;
  int * REXX_return_code_ptr;
 } ARXEXEC_type;
int InvREXX(ARXEXEC_type this_param); /* Function invokes REXX */
ARXEXEC_type this_param;
EXECBLK_type this_EXECBLK;
EXECBLK_type * an_EXECBLK_ptr;
EVALBLK_type this_EVALBLK;
EVALBLK_type * an_EVALBLK_ptr;
one_parameter_type this_argument[2];
one_parameter_type * an_argtable_ptr;
char arg1;
int flags;
int REXX_return_code;
int dummy_zero;

main()
{ printf("start of USCHI\n");
  arg1 = '3';
  this_argument[0].ARGSTRING_PTR = &arg1;
  this_argument[0].ARGSTRING_LENGTH = 1;
  this_argument[1].ARGSTRING_PTR = (void *)0xFFFFFFFF;
  this_argument[1].ARGSTRING_LENGTH = 0xFFFFFFFF;
  an_argtable_ptr = &this_argument[0];
  an_EXECBLK_ptr = &this_EXECBLK;
  strcpy(this_EXECBLK.EXECBLK_ACRYN,"ARXEXECB");
  this_EXECBLK.EXECBLK_LENGTH = 48;
  this_EXECBLK.EXECBLK_reserved = 0;
  strcpy(this_EXECBLK.EXECBLK_MEMBER,"HELLO   ");
  strcpy(this_EXECBLK.EXECBLK_DDNAME,"        ");
  strcpy(this_EXECBLK.EXECBLK_SUBCOM,"        ");
  this_EXECBLK.EXECBLK_DSNPTR = 0;
  this_EXECBLK.EXECBLK_DSNLEN = 0;
  an_EVALBLK_ptr = &this_EVALBLK;
  this_EVALBLK.EVALBLK_EVPAD1 = 0;
  this_EVALBLK.EVALBLK_EVSIZE = 34;
  this_EVALBLK.EVALBLK_EVLEN = 0;
  this_EVALBLK.EVALBLK_EVPAD2 = 0;
  this_param.execblk_ptr = &an_EXECBLK_ptr;
  this_param.argtable_ptr = &an_argtable_ptr;
  this_param.flags_ptr = &flags;
  this_param.instblk_ptr = &dummy_zero;
  this_param.reserved_parm5 = &dummy_zero;
  this_param.evalblk_ptr = &an_EVALBLK_ptr;
  this_param.reserved_workarea_ptr = &dummy_zero;
  this_param.reserved_userfield_ptr = &dummy_zero;
  this_param.reserved_envblock_ptr = &dummy_zero;
  this_param.REXX_return_code_ptr = &REXX_return_code;
  this_param.REXX_return_code_ptr =
    (int *)((int)this_param.REXX_return_code_ptr | 0x80000000);
  dummy_zero = 0;
  flags = 0x20000000;
  REXX_return_code = 0;
  InvREXX(this_param);
  printf("REXX return code is: %d\n", REXX_return_code);
  printf("REXX result is: %s\n", this_EVALBLK.EVALBLK_EVDATA);
  printf("end of USCHI\n");
}
/*
// EXEC    LNKEDT
```



```
/*
// IF $MRC > 4 THEN
// GOTO JOBEND
// LIBDEF *,SEARCH=(PRIMARY.USCH,PRD2.DBASE,PRD2.SCEEBASE)
// EXEC USCHI
/. JOBEND
/&
* $$ EOJ
```


Calling ARXJCL from PLI

```
* $$ JOB JNM=MINIPLI,CLASS=0,DISP=D
// JOB MINIPLI
// EXEC LIBR
  ACC S=PRIMARY.USCH
  DELETE USCHI.PHASE
  CAT HELLO.PROC R=Y
/* simple rexx exec */
Arg n
If n='' Then n=1
Do i=1 to n
  fc = Assgn('STDOUT','SYSLST')
  Say 'hello'
  fc = Assgn('STDOUT','SYSLOG')
  Say 'hello'
End
Return 99
/+
/*
// LIBDEF *,SEARCH=(PRD2.SCEEBASE,PRD2.PROD)
// LIBDEF PHASE,CATALOG=PRIMARY.USCH
// OPTION CATAL
   PHASE USCHI,*
/. ENDCAT
// EXEC IEL1AA,SIZE=256K
*PROCESS INCLUDE,SYSTEM(VSE),FLAG(I),XREF(SHORT),MAP,LIST;
*PROCESS LINECOUNT(100);
 USCHI: PROC OPTIONS(MAIN);
 /* invoke REXX procedure HELLO with parameter 3 using ARXIJCL */
 DCL ARXIJCL ENTRY EXTERNAL OPTIONS(ASSEMBLER RETCODE);
 DCL 1 ARXJCL_PARM,
       3 ARG_LENGTH     FIXED BINARY(15),
       3 ARGUMENT       CHAR(9);
 DCL PLIRETV BUILTIN;
 DCL RETURN_CODE        PICTURE'9999';
 DISPLAY ('START OF USCHI');
 ARG_LENGTH = 8;
 ARGUMENT = 'HELLO  3';
 CALL ARXIJCL(ARXJCL_PARM);
 RETURN_CODE = PLIRETV;
 DISPLAY ('REXX RETURN CODE: ' || RETURN_CODE);
 DISPLAY ('END OF USCHI');
 RETURN;
 END USCHI;
/*
// EXEC LNKEDT,SIZE=256K
/*
// IF $MRC > 4 THEN
// GOTO JOBEND
// LIBDEF *,SEARCH=(PRIMARY.USCH,PRD2.DBASE,PRD2.SCEEBASE)
// EXEC USCHI
/. JOBEND
/&
* $$ EOJ
```


Calling ARXEXEC from PLI

```
* $$ JOB JNM=MINIPLI,CLASS=0,DISP=D
// JOB MINIPLI
// EXEC LIBR
  ACC S=PRIMARY.USCH
  DELETE USCHI.PHASE
  CAT HELLO.PROC R=Y
/* simple rexx exec */
Arg n
If n='' Then n=1
Do i=1 to n
  fc = Assgn('STDOUT','SYSLST')
  Say 'hello'
  fc = Assgn('STDOUT','SYSLOG')
  Say 'hello'
End
Return date('W')
/+
/*
// LIBDEF *,SEARCH=(PRD2.SCEEBASE,PRD2.PROD)
// LIBDEF PHASE,CATALOG=PRIMARY.USCH
// OPTION CATAL
   PHASE USCHI,*
/. ENDCAT
// EXEC IEL1AA,SIZE=256K
*PROCESS INCLUDE,SYSTEM(VSE),FLAG(I),XREF(SHORT),MAP,LIST;
*PROCESS LINECOUNT(100);
 USCHI: PROC OPTIONS(MAIN);
 /* invoke REXX procedure HELLO with parameter 3 using ARXIEXEC */
 DCL ARXIEXEC ENTRY EXTERNAL OPTIONS(ASSEMBLER RETCODE);
 DCL 1 EXECBLK,
       3 EXECBLK_ACRYN     CHAR(8),
       3 EXECBLK_LENGTH    FIXED BINARY(31),
       3 EXECBLK_reserved  FIXED BINARY(31),
       3 EXECBLK_MEMBER    CHAR(8),
       3 EXECBLK_DDNAME    CHAR(8),
       3 EXECBLK_SUBCOM    CHAR(8),
       3 EXECBLK_DSNPTR    PTR,
       3 EXECBLK_DSNLEN    FIXED BINARY(31);
 DCL 1 EVALBLK,
       3 EVALBLK_EVPAD1    FIXED BINARY(31),
       3 EVALBLK_EVSIZE    FIXED BINARY(31),
       3 EVALBLK_EVLEN     FIXED BINARY(31),
       3 EVALBLK_EVPAD2    FIXED BINARY(31),
       3 EVALBLK_EVDATA    CHAR(256);
 DCL 1 ARGTABLE,
       3 ARGUMENTS(1),
         5 ARGSTRING_PTR     PTR,
         5 ARGSTRING_LENGTH  FIXED BINARY(31),
       3 ARGTABLE_LAST     CHAR(8);
 DCL EXECBLK_PTR           PTR;
 DCL ARGTABLE_PTR          PTR;
 DCL INSTBLK_PTR           PTR;
 DCL reserved_parm5        PTR;
 DCL EVALBLK_PTR           PTR;
 DCL reserved_workarea_ptr PTR;
 DCL reserved_userfield_ptr PTR;
 DCL reserved_envblock_ptr PTR;
 DCL REXX_return_code_ptr  PTR;
 DCL ARG1                  CHAR;
```



```
 DCL flags                CHAR(4);
 DCL REXX_return_code     FIXED BINARY(31);
 DCL PLIRETV BUILTIN;
 DCL SYSNULL BUILTIN;
 DCL ADDR BUILTIN;
 DCL SUBSTR BUILTIN;
 DCL RETURN_CODE        PICTURE'9999';
 DISPLAY ('START OF USCHI');
 ARG1 = '3';
 ARGSTRING_PTR(1) = ADDR(ARG1);
 ARGSTRING_LENGTH(1) = 1;
 ARGTABLE_LAST = 'FFFFFFFFFFFFFFFF'X;
 ARGTABLE_PTR = ADDR(ARGSTRING_PTR(1));
 EXECBLK_PTR = ADDR(EXECBLK);
 EXECBLK_ACRYN = 'ARXEXECB';
 EXECBLK_LENGTH = 48;
 EXECBLK_reserved = 0;
 EXECBLK_MEMBER = 'HELLO   ';
 EXECBLK_DDNAME = '        ';
 EXECBLK_SUBCOM = '        ';
 EXECBLK_DSNPTR = SYSNULL;
 EXECBLK_DSNLEN = 0;
 EVALBLK_PTR = ADDR(EVALBLK);
 EVALBLK_EVPAD1 = 0;
 EVALBLK_EVSIZE = 34;
 EVALBLK_EVLEN = 0;
 EVALBLK_EVPAD2 = 0;
 flags = '40000000'x;
 REXX_return_code_ptr = ADDR(REXX_return_code);
 REXX_return_code = 0;
 INSTBLK_PTR = SYSNULL;
 reserved_parm5 = SYSNULL;
 reserved_workarea_ptr = SYSNULL;
 reserved_userfield_ptr = SYSNULL;
 reserved_envblock_ptr = SYSNULL;
 CALL ARXIEXEC(EXECBLK_PTR,
               ARGTABLE_PTR,
               flags,
               INSTBLK_PTR,
               reserved_parm5,
               EVALBLK_PTR,
               reserved_workarea_ptr,
               reserved_userfield_ptr,
               reserved_envblock_ptr,
               REXX_return_code_ptr);
 RETURN_CODE = PLIRETV;
 DISPLAY ('REXX RETURN CODE: ' || RETURN_CODE);
 DISPLAY ('REXX RETURN CODE: ' || REXX_return_code);
 DISPLAY ('REXX result is: ' || SUBSTR(EVALBLK_EVDATA,1,EVALBLK_EVLEN));
 DISPLAY ('END OF USCHI');
 RETURN;
 END USCHI;
/*
// EXEC LNKEDT,SIZE=256K
/*
// IF $MRC > 4 THEN
// GOTO JOBEND
// LIBDEF *,SEARCH=(PRIMARY.USCH,PRD2.DBASE,PRD2.SCEEBASE)
// EXEC USCHI
/. JOBEND
/&
* $$ EOJ
```


Remarks

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Comments and Questions

Comments or questions on this documentation are welcome. Please send your comments to:

vseesa@de.ibm.com

