ACB Examples

This chapter describes examples of direct calls using the ACB interface.

- ACB Assembler Examples
- ACB COBOL Examples
- ACB PL/I Examples
- ACB Fortran Example

ACB Assembler Examples

This section contains examples of using direct Adabas calls in Assembler. The previously defined Adabas files defined are used in each example.

```
*** CONTROL BLOCK
      DS   0F
CB    DS   0CL80        USER CONTROL BLOCK
      DC   CL2' '       RESERVED FOR ADABAS USE
CCODE DC   CL2' '       COMMAND CODE
CID   DC   CL4' '       COMMAND ID
FNR   DC   H'0'         FILE NUMBER
RC    DC   H'0'         RESPONSE CODE
ISN   DC   F'0'         ISN
ISNLL DC   F'0'         ISN LOWER LIMIT
ISNQ  DC   F'0'         ISN QUANTITY
FBL   DC   H'100'       FORMAT BUFFER LENGTH
RBL   DC   H'250'       RECORD BUFFER LENGTH
SBL   DC   H'50'        SEARCH BUFFER LENGTH
VBL   DC   H'100'       VALUE BUFFER LENGTH
IBL   DC   H'20'        ISN BUFFER LENGTH
COPT1 DC   CL1' '       COMMAND OPTION 1
COPT2 DC   CL1' '       COMMAND OPTION 2
ADD1  DC   CL8' '       ADDITIONS 1
ADD2  DC   CL4' '       ADDITIONS 2
ADD3  DC   CL8' '       ADDITIONS 3
ADD4  DC   CL8' '       ADDITIONS 4
ADD5  DC   CL8' '       ADDITIONS 5
CTIME DC   F'0'         COMMAND TIME
UAREA DC   CL4' '       USER AREA
*
*
*** USER BUFFER AREAS
FB    DC   CL100' '     FORMAT BUFFER
RB    DC   CL250' '     RECORD BUFFER
SB    DC   CL50' '      SEARCH BUFFER
VB    DC   CL100' '     VALUE BUFFER
IB    DC   CL20' '      ISN BUFFER
* * *
```

This section provides the following examples:

- Example 1
- Example 2
- Example 3 : User Session with ET Logic

Example 1

- Find the set of records in file 2 with XB = 99.
- Read each record selected using the GET NEXT option.

Issue Open Command

```
EXMP1 MVC  CCODE,=C'OP'             OP COMMAND
      MVC  RB(4),=C'ACC.'           ACCESS ONLY REQUESTED
      CALL ADABAS,(CB,FB,RB)        CALL ADABAS
      CLC  RC,=H'0'                 CHECK RESPONSE CODE
      BNE  EX1ERR                   BRANCH IF NOT 0
```

Issue Find Command

```
      MVC  CCODE,=C'S1'             FIND COMMAND
      MVC  CID,=C'S101'             NONBLANK CID REQUIRED FOR
*                                   IDENTIFICATION OF THE LIST
      MVC  FNR,=H'2'                FILE 2
      MVC  ISNLL,=F'0'              ALL QUALIFYING ISNS DESIRED
      MVC  IBL,=H'0'                ISN BUFFER NOT REQUIRED
      MVI  FB,C'.'                  NO READ OF DATA STORAGE
      MVC  SB(7),=C'XB,3,U.'        SEARCH CRITERION
      MVC  VB(3),=C'099'            SEARCH VALUE
      CALL ADABAS,(CB,FB,RB,SB,VB)  CALL ADABAS
      CLC  RC,=H'0'                 CHECK RESPONSE CODE
      BNE  EX1ERR                   BRANCH IF NOT 0
      CLC  ISNQ,=F'0'               CHECK NUMBER OF ISNS FOUND
      BE   EX1EXIT                  BRANCH TO EXIT IF NO ISNS FOUND
```

Read Each Qualifying Record

```
EX1B  MVC  CCODE,=C'L1'             READ COMMAND
      MVC  ISN,=F'0'                BEGIN WITH 1ST ISN IN LIST
      MVI  COPT2,C'N'               GET NEXT OPTION TO BE USED
      MVC  FB(3),=C'RG.'            ALL FIELDS TO BE RETURNED
EX1C  CALL ADABAS,(CB,FB,RB)        CALL ADABAS
      CLC  RC,=H'0'                 CHECK RESPONSE CODE
      BE   EX1D                     BRANCH IF RESPONSE CODE 0
      CLC  RC,=H'3'                 CHECK IF ALL RECORDS READ
      BE   EX1EXIT                  BRANCH IF YES
      B    EX1ERR                   BRANCH TO ERROR ROUTINE
EX1D  . . .                         PROCESS RECORD . . .
      B    EX1C                     BRANCH TO READ NEXT RECORD
```

Error Routine

```
EX1ERR EQU *
*                                   DISPLAY ERROR MESSAGE
*      .                            TERMINATE USER PROGRAM
```

Issue Close Command

```
EX1EXIT MVC  CCODE,=C'CL'          CLOSE COMMAND
        CALL ADABAS,(CB)           CALL ADABAS
        CLC  RC,=H'0'              CHECK RESPONSE CODE
        BNE  EX1ERR                BRANCH IF NOT 0
```

Example 2

- All records in file 1 are to be read in physical sequential order.
- Each record read is to be updated with the following values:
  - Field AA = ABCDEFGH
  - Field AB = 500
- User is to have exclusive control of file 1.

Issue Open Command

```
EXMP2 MVC  CCODE,=C'OP'            OPEN COMMAND
      MVC  RB(6),=C'EXU=1.'        EXCLUSIVE CONTROL REQUESTED
      CALL ADABAS,(CB,FB,RB)       CALL ADABAS
      CLC  RC,=H'0'                CHECK RESPONSE CODE
      BE   EX2A                    BRANCH IF RESPONSE CODE 0
      B    EX2ERR                  BRANCH IF NOT 0
```

Issue Read Physical Sequential Command

```
EX2A MVC  CID,=C'L201'             NONBLANK CID REQUIRED
     MVC  FNR,=H'1'                FILE 1 TO BE READ
     MVC  ISN,=F'0'                ALL RECORDS TO BE READ
     MVC  FB(3),=C'GA.'            VALUES FOR FIELDS AA AND AB
*                                 (GROUP GA) TO BE RETURNED
EX2B MVC  CCODE,=C'L2'             READ PHYS. SEQ.
     CALL ADABAS,(CB,FB,RB)        CALL ADABAS
     CLC  RC,=H'0'                 CHECK RESPONSE CODE
     BE   EX2C                     BRANCH IF RESPONSE CODE 0
     CLC  RC,=H'3'                 CHECK FOR END-OF-FILE
     BE   EX2EXIT                  BRANCH TO EXIT IF END-OF-FILE
     B    EX2ERR                   BRANCH TO ERROR ROUTINE
```

Update Record

- The same fields are to be updated as were read.
- The same CID and format buffer can be used for the update command.
- The ISN of the record to be updated is already in the ISN field as a result of the L2 command.

```
EX2C MVC  CCODE,=C'A1'             UPDATE COMMAND
     MVC  RB(8),=C'ABCDEFGH'       VALUE FOR FIELD AA
     MVC  RB+8(2),=PL2'500'        VALUE FOR FIELD AB
     CALL ADABAS,(CB,FB,RB)        CALL ADABAS
     CLC  RC,=H'0'                 CHECK RESPONSE CODE
     BE   EX2B                     BRANCH TO READ NEXT RECORD
```

Error Routine

```
EX2ERR EQU *
*      .                          DISPLAY ERROR MESSAGE
*      .                          TERMINATE USER PROGRAM
```

Close User Session

```
EX2EXIT MVC  CCODE,=C'CL'         CLOSE COMMAND
        CALL ADABAS,(CB)          CALL ADABAS
        CLC  RC,=H'0'             CHECK RESPONSE CODE
        BNE  EX2ERR               BRANCH IF NOT 0
        . . .
```

Example 3 : User Session with ET Logic

During user session initialization, the user program is to display information indicating the last successfully processed transaction of the previous user session.

For each user transaction, the user program is to

- accept from a terminal 8 characters of input to be used as the key for updating files 1 and 2; and
- issue the Find command for file 1 to determine if a record exists with field AA = input key.

If no record is found, the user program is to issue a message. If a record is found, the user program is to

- delete the record from file 1; and
- add a new record to file 2: Field RA = input key entered.

Other fields are to contain a null value.

If the record cannot be successfully added, the user program is to issue a BT command and display an error message.

If both updates are successful, the user program is to issue an ET command.

- Session Initialization
- Transaction Processing

Session Initialization

The section of the program illustrated is only executed during user session initialization:

Issue Open Command

The OP command is issued with ET data of the previous session being read:

```
EXMP3 EQU  *
      MVC  CCODE,=C'OP'           OPEN COMMAND
      MVI  COPT2,C'E'             ET DATA TO BE READ
      MVC  ADD1,=C'USER0001'      USER IDENTIFICATION
      MVC  ADD3,=C'PASSWORD'      USER PASSWORD
      MVC  RB(8),=C'UPD=1,2.'     FILES 1 AND 2 TO BE UPDATED
      CALL ADABAS,(CB,FB,RB)      CALL ADABAS
      CLC  RC,=H'0'               CHECK RESPONSE CODE
```

```
      BE   EX3A                  BRANCH IF RESPONSE CODE 0
      CLC  RC,=H'9'              CHECK FOR RESPONSE CODE 9
      BE   EXMP3                 BRANCH TO REPEAT OPEN
      B    EX3ERR                BRANCH IF NOT 0 OR 9
EX3A  EQU  *
      CLC  CID,=F'0'             CHECK IF ET DATA FROM
*                                PREVIOUS SESSION EXISTS
      BE   EX3B                  BRANCH IF NO ET DATA
*     . . .
```

Display ET Data

Display the ET data contained in the record buffer on the terminal screen to inform the user of the last successfully processed transaction of the previous user session:

```
      B    EX3C                  BRANCH TO BEGIN TRANS. PROCESS.
EX3B  EQU  *
```

No ET Data Received

If no ET data was received, a message is displayed indicating that no transactions were successfully processed during the previous user session.

Transaction Processing

This section is executed for each user transaction:

```
EX3C EQU  *
*    . . .                           ACCEPT INPUT FROM TERMINAL  . . .
```

Issue Find Command

Issue the Find command for file 1 to determine if a record exists with the field AA equal to the input key entered:

```
EX3D EQU  *
     MVC  CCODE,=C'S4'              FIND WITH HOLD COMMAND
     MVC  CID,=C'    '              ISN LIST NOT TO BE SAVED
     MVC  FNR,=H'1'                 FILE 1
     MVC  ISNLL,=F'0'               ALL QUALIFY. ISNS TO BE RETURNED
     MVI  FB,C'.'                   NO READ OF DATA STORAGE
     MVC  SB(3),=C'AA.'             SEARCH CRITERION
     MVC  VB(8),INPUT               SEARCH VALUE
     CALL ADABAS,(CB,FB,RB,SB,VB,IB) CALL ADABAS
     CLC  RC,=H'0'                  CHECK RESPONSE CODE
     BE   EX3E                      BRANCH IF RESPONSE CODE 0
     B    EX3ERR                    BRANCH TO ERROR ROUTINE
EX3E EQU  *
     CLC  ISNQ,=F'0'                CHECK NUMBER OF RECORDS FOUND
     BNE  EX3F                      BRANCH IF RECORD FOUND
```

Issue Message if No Record is Found

If no record is found, the user program issues a message requesting a correction:

```
B    EX3C                      RETURN TO ACCEPT USER INPUT
```

Delete Record from File 1

The ISN of the record to be deleted is already in the ISN field and in hold status as a result of the S4 command.

```
EX3F EQU  *
     MVC  CCODE,=C'E4'                DELETE COMMAND
     CALL ADABAS,(CB)                 CALL ADABAS
     CLC  RC,=H'0'                    CHECK RESPONSE CODE
     BE   EX3G                        BRANCH IF RESPONSE CODE 0
     CLC  RC,=H'9'                    CHECK IF CURRENT TRANS. HAS BEEN
*                                     BACKED OUT BY ADABAS
     BE   EX3D                        IF YES, BRANCH TO REPEAT S4
     B    EX3ERR                      BRANCH TO ERROR ROUTINE
```

Add a New Record to File 2

```
EX3G EQU  *
     MVC  CCODE,=C'N1'               ADD NEW RECORD
     MVC  FNR,=H'2'                  FILE 2
     MVC  FB(6),=C'RA.'              VALUE BEING PROVIDED FOR RA
     MVC  RB(8),INPUT                VALUE FOR FIELD RA
     CALL ADABAS,(CB,FB,RB)          CALL ADABAS
     CLC  RC,=H'0'                   CHECK RESPONSE CODE
     BE   EX3I                       BRANCH IF RESPONSE CODE 0
     CLC  RC,=H'9'                   WAS TRANSACTION BACKED OUT?
     BE   EX3D                       IF YES, RETURN TO REISSUE TRANS.
```

Unable to Add a New Record

If the attempt to add a new record is not successful, the transaction is backed out and the user is notified that an error condition exists.

```
     MVC  CCODE,=C'BT'               BACKOUT TRANSACTION
     CALL ADABAS,(CB)                CALL ADABAS
     CLC  RC,=H'0'                   CHECK IF RESPONSE CODE 0
     BE   EX3H                       BRANCH IF 0
```

Backout Not Successful

When the backout is not successful, a message is issued indicating that result.

```
     B    EX3ERR                     BRANCH TO ERROR ROUTINE
EX3H EQU  *
```

Backout Successful

When the backout is successful, a message is issued indicating that after an error was detected, the transaction was backed out.

```
     B    EX3ERR                     BRANCH TO ERROR ROUTINE
```

Updates Successfully Executed : Issue ET Command with ET Data

When the updates have been successfully executed, an ET command with ET data is issued.

```
EX3I EQU  *
     MVC  CCODE,=C'ET'        END OF TRANSACTION COMMAND
     MVI  COPT2,C'E'          ET DATA TO BE WRITTEN
     MVC  RB(8),INPUT         ET DATA CONSISTS OF INPUT KEY OF THIS TRANSACTION
     CALL ADABAS,(CB,FB,RB)   CALL ADABAS
     CLC  RC,=H'0'            CHECK RESPONSE CODE
     BE   EX3C                IF RESPONSE CODE 0, RETURN TO RECEIVE INPUT FOR
*                             THE NEXT TRANSACTION
     CLC  RC,=H'9'            CHECK IF CURRENT TRANSACTION HAS BEEN BACKED OUT
*                             BY ADABAS
     BE   EX3D                IF CURRENT TRANSACTION HAS BEEN BACKED OUT,
*                             RETURN TO REISSUE TRANSACTION
```

Error Routine

```
EX3ERR EQU  *
*      .                     NONZERO RESPONSE CODE RECEIVED
*      .                     DISPLAY ERROR MESSAGE
*      .                     TERMINATE USER PROGRAM
       . . .
INPUT  DS   CL8              KEY ENTERED FROM TERMINAL
```

ACB COBOL Examples

This section contains examples of using direct Adabas calls in COBOL. The previously defined Adabas files are used in each example.

```
*
 *** CONTROL BLOCK
     01  CONTROL-BLOCK.
         02 FILLER                    PIC X(2) VALUE SPACES.
         02 COMMAND-CODE              PIC X(2) VALUE SPACES.
         02 COMMAND-ID                PIC X(4) VALUE SPACES.
         02 FILE-NUMBER               PIC S9(4) COMP VALUE +0.
         02 RESPONSE-CODE             PIC S9(4) COMP VALUE +0.
         02 ISN                       PIC S9(8) COMP VALUE +0.
         02 ISN-LOWER-LIMIT           PIC S9(8) COMP VALUE +0.
         02 ISN-QUANTITY              PIC S9(8) COMP VALUE +0.
         02 FORMAT-BUFFER-LENGTH      PIC S9(4) COMP VALUE +100.
         02 RECORD-BUFFER-LENGTH      PIC S9(4) COMP VALUE +250.
         02 SEARCH-BUFFER-LENGTH      PIC S9(4) COMP VALUE +50.
         02 VALUE-BUFFER-LENGTH       PIC S9(4) COMP VALUE +100.
         02 ISN-BUFFER-LENGTH         PIC S9(4) COMP VALUE +20.
         02 COMMAND-OPTION-1          PIC X VALUE SPACE.
         02 COMMAND-OPTION-2          PIC X VALUE SPACE.
         02 ADDITIONS-1               PIC X(8) VALUE SPACES.
         02 ADDITIONS-2               PIC X(4) VALUE SPACES.
         02 ADDITIONS-3               PIC X(8) VALUE SPACES.
         02 ADDITIONS-4               PIC X(8) VALUE SPACES.
         02 ADDITIONS-5               PIC X(8) VALUE SPACES.
         02 COMMAND-TIME              PIC S9(8) COMP VALUE +0.
         02 USER-AREA                 PIC X(4) VALUE SPACES.
*
*** USER BUFFER AREAS
    01 FORMAT-BUFFER                  PIC X(100) VALUE SPACES.
    01 RECORD-BUFFER                  PIC X(250) VALUE SPACES.
    01 SEARCH-BUFFER                  PIC X(50)  VALUE SPACES.
    01 VALUE-BUFFER                   PIC X(100) VALUE SPACES.
    01 ISN-BUFFER                     PIC X(20)  VALUE SPACES.
*
```

```
*** ADDITIONAL FIELDS USED IN THE EXAMPLES
    01 PROGRAM-WORK-AREA.
       05         COMM-ID PIC X(4).
       05         COMM-ID-X REDEFINES COMM-ID PIC S9(8) COMP.
       05         INPUT-KEY PIC X(8).
       05         RECORD-BUFFER-EX2.
          10      RECORD-BUFFER-A PIC X(8).
          10      RECORD-BUFFER-B PIC S9(3) COMP-3.
       05         RECORD-BUFFER-EX3.
          10      OPEN-RECORD-BUFFER.
             15   OPEN-RECORD-BUFFER-X PIC X(8).
             15   FILLER PIC S9(8) COMP.
          10      FILLER PIC X(18).
          10      UPDATED-XC PIC X(6).
          10      LAST-XD PIC X(8).
          10      FILLER PIC X(5).
       05         USER-DATA.
          10      RESTART-XD PIC X(8).
          10      RESTART-ISN PIC S9(8) COMP.
       05         SYNC-CHECK-SWITCH PIC 9 VALUE 0.
       05         AB-VALUE PIC S9(4) COMP-3 VALUE +500.
*
```

This section provides the following examples:

- Example 1
- Example 2
- Example 3 : User Session with ET Logic

Example 1

- Find the set of records in file 2 with XB = 99.
- Read each record selected using the GET NEXT option.

Issue Open Command

```
EXMP1.
        MOVE      'OP' TO COMMAND-CODE.
        MOVE      'ACC.' TO RECORD-BUFFER.
        CALL      'ADABAS'
                  USING CONTROL-BLOCK, FORMAT-BUFFER, RECORD-BUFFER.
        IF RESPONSE-CODE NOT EQUAL TO 0  GO TO EX1ERR.
```

Issue Find Command

```
MOVE      'S1' TO COMMAND-CODE.
        MOVE      'S101' TO COMMAND-ID.
        MOVE      2 TO FILE-NUMBER.
        MOVE      0 TO ISN-LOWER-LIMIT.
        MOVE      0 TO ISN-BUFFER-LENGTH.
        MOVE      '.' TO FORMAT-BUFFER.
        MOVE      'XB,3,U.' TO SEARCH-BUFFER.
        MOVE      '099' TO VALUE-BUFFER.
        CALL      'ADABAS' USING CONTROL-BLOCK, FORMAT-BUFFER,
```

```
                RECORD-BUFFER, SEARCH-BUFFER, VALUE-BUFFER.
        IF RESPONSE-CODE NOT EQUAL TO 0  GO TO EX1ERR.
    EX1A.
        IF ISN-QUANTITY = 0  GO TO EX1EXIT.
```

Read Each Qualifying Record

```
EX1B.
        MOVE      'L1' TO COMMAND-CODE.
        MOVE      0 TO ISN.
        MOVE      'N' TO COMMAND-OPTION-2.
        MOVE      'RG.' TO FORMAT-BUFFER.
    EX1C.
        CALL      'ADABAS'
                  USING CONTROL-BLOCK, FORMAT-BUFFER, RECORD-BUFFER.
        IF RESPONSE-CODE = 0 GO TO EX1D.
        IF RESPONSE-CODE = 3 GO TO EX1EXIT.
    EX1D.
        . . . PROCESS RECORD . . .
        GO TO EX1C.
```

Error Routine

```
EX1ERR.
*       .DISPLAY ERROR MESSAGE
*       .TERMINATE USER PROGRAM
```

Issue Close Command

```
EX1EXIT.
        MOVE      'CL' TO COMMAND-CODE.
        CALL      'ADABAS' USING CONTROL-BLOCK.
        IF RESPONSE-CODE NOT EQUAL TO 0  GO TO EX1ERR.
```

Example 2

- All records in file 1 are to be read in physical sequential order.
- Each record read is to be updated with the following values:
  - Field AA = ABCDEFGH
  - Field AB = 500
- User is to have exclusive control of file 1.

Issue Open Command

```
EXMP2.
        MOVE      'OP' TO COMMAND-CODE.
        MOVE      'EXU=1.' TO RECORD-BUFFER.
        CALL      'ADABAS' USING
                  CONTROL-BLOCK, FORMAT-BUFFER, RECORD-BUFFER.
        IF RESPONSE-CODE NOT EQUAL TO 0  GO TO EX2ERR.
```

Issue Read Physical Sequential Command

```
EX2A.
        MOVE      'L201' TO COMMAND-ID.
        MOVE      1 TO FILE-NUMBER.
        MOVE      0 TO ISN.
        MOVE      'GA.' TO FORMAT-BUFFER.
    EX2B.
        MOVE      'L2' TO COMMAND-CODE.
        CALL      'ADABAS' USING
                  CONTROL-BLOCK, FORMAT-BUFFER, RECORD-BUFFER.
        IF RESPONSE-CODE = 0  GO TO EX2C.
        IF RESPONSE-CODE = 3  GO TO EX2EXIT.
        GO TO EX2ERR.
```

Update Record

- The same fields are to be updated as were read.
- The same CID and format buffer can be used for the update command.
- The ISN of the record to be updated is already in the ISN field as a result of the L2 command.

```
EX2C.
        MOVE      'A1' TO COMMAND-CODE.
        MOVE      'ABCDEFGH' TO RECORD-BUFFER-A.
        MOVE      AB-VALUE TO RECORD-BUFFER-B.
        CALL      'ADABAS' USING
                  CONTROL-BLOCK, FORMAT-BUFFER, RECORD-BUFFER-EX2.
        IF RESPONSE-CODE NOT EQUAL TO 0  GO TO EX2ERR.
        GO TO EX2B.
```

Error Routine

```
EX2ERR.
        . DISPLAY ERROR MESSAGE
        . TERMINATE USER PROGRAM
```

Close User Session

```
EX2EXIT.
        MOVE      'CL' TO COMMAND-CODE.
        CALL      'ADABAS' USING CONTROL-BLOCK.
        IF RESPONSE-CODE NOT EQUAL TO 0  GO TO EX2ERR.
```

Example 3 : User Session with ET Logic

During user session initialization, the user program is to display information indicating the last successfully processed transaction of the previous user session.

For each user transaction, the user program is to

- accept from a terminal 8 characters of input to be used as the key for updating files 1 and 2; and
- issue the Find command for file 1 to determine if a record exists with field AA = input key.

If no record is found, the user program is to issue a message. If a record is found, the user program is to

- delete the record from file 1; and
- add a new record to file 2: Field RA = input key entered.

Other fields are to contain a null value.

If the record cannot be successfully added, the user program is to issue a BT command and display an error message.

If both updates are successful, the user program is to issue an ET command.

Session Initialization

This section of the program is only executed during user session initialization.

- The OP command is issued with ET data of the previous session being read.
- A message is displayed on the terminal screen identifying the last successfully processed transaction of the user's previous session.

```
    EX3.
         MOVE    'OP' TO COMMAND-CODE.
         MOVE    'E' TO COMMAND-OPTION-2.
         MOVE    'USER0002' TO ADDITIONS-1.
         MOVE    'PASSWORD' TO ADDITIONS-3.
         MOVE    'UPD=1,2.' TO RECORD-BUFFER.
         CALL    'ADABAS' USING
                 CONTROL-BLOCK, FORMAT-BUFFER, RECORD-BUFFER.
         IF RESPONSE-CODE = 9  GO TO EX3.
         IF RESPONSE-CODE NOT EQUAL TO 0
              GO TO EX3ERR.
    EX3A.
         MOVE    COMMAND-ID TO COMM-ID.
         IF COMM-ID-X = +0
              GO TO EX3B.
* Display ET data (contained in RECORD BUFFER) on screen to inform user of
* last successfully processed transaction of previous user session.
               . . .DISPLAY ET DATA. . .
              GO TO EX3C.
    EX3B.
*** No ET data received.
*   Display message that no transactions were successfully processed during
*   the previous user session
              . . .DISPLAY MESSAGE . . .
*** Transaction processing.
*   This section is executed for each user transaction.
    EX3C.
*             . . .ACCEPT INPUT FROM TERMINAL. . .
*   Issue Find command for file 1 to determine if record exists with field AA
*   equal to input key entered.
    EX3D.
         MOVE    'S4' TO COMMAND-CODE.
         MOVE    SPACES TO COMMAND-ID.
         MOVE    1 TO FILE-NUMBER.
         MOVE    0 TO ISN-LOWER-LIMIT.
         MOVE    '.' TO FORMAT-BUFFER.
         MOVE    'AA.' TO SEARCH-BUFFER.
```

```
         MOVE    INPUT-KEY TO VALUE-BUFFER.
         CALL    'ADABAS' USING
                 CONTROL-BLOCK, FORMAT-BUFFER, RECORD-BUFFER,
                 SEARCH-BUFFER, VALUE-BUFFER, ISN-BUFFER.
         IF RESPONSE-CODE = 0
              GO TO EX3E.
         GO TO EX3ERR.
EX3E.
         IF ISN-QUANTITY NOT EQUAL TO ZEROS
              GO TO EX3F.
***No records found, issue message requesting correction.
              . . .ISSUE MESSAGE . . .
         GO TO EX3C.
*** Delete record from file 1.
*  ISN of record to be deleted is already in ISN field and in hold
status
*  as a result of the S4 command.
    EX3F.
         MOVE     E3' TO COMMAND-CODE.
         CALL     'ADABAS' USING CONTROL-BLOCK.
         IF RESPONSE-CODE = 0
              GO TO EX3G.
         IF RESPONSE-CODE = 9
              GO TO EX3D.
         GO TO EX3ERR.
*** Add new record to file 2.
    EX3G.
         MOVE     'N1' TO COMMAND-CODE.
         MOVE     2 TO FILE-NUMBER.
         MOVE     'RA.' TO FORMAT-BUFFER.
         MOVE     INPUT-KEY TO RECORD-BUFFER.
         CALL     'ADABAS' USING
                  CONTROL-BLOCK, FORMAT-BUFFER, RECORD-BUFFER.
         IF RESPONSE-CODE = 0
              GO TO EX3I.
         IF RESPONSE-CODE = 9
              GO TO EX3D.
*** Attempt to add new record not successful.
*  Backout transaction.
*  Notify user that error condition exists.
         MOVE     'BT' TO COMMAND-CODE.
         CALL     'ADABAS' USING control-block.
         IF RESPONSE-CODE = 0
              GO TO EX3H.
*** Backout not successful.
*   Issue message indicating that the backout was not successful
         GO TO EX3ERR.
    EX3H.
*** Backout successful.
*   Issue message indicating the error condition detected while while
adding a
*   new record
         GO TO EX3ERR.
*** Updates successfully executed.
*  Issue ET command with ET data.
  EX3I.
         MOVE     'ET' TO COMMAND-CODE.
         MOVE     'E' TO COMMAND-OPTION-2.
         MOVE     INPUT-KEY TO RECORD-BUFFER.
         CALL     'ADABAS' USING
                  CONTROL-BLOCK, FORMAT-BUFFER, RECORD-BUFFER.
         IF RESPONSE-CODE = 0
```

```
             GO TO EX3C.
         IF RESPONSE-CODE = 9
             GO TO EX3D.
*** Error Routine
    EX3ERR.
*        . DISPLAY ERROR MESSAGE
*        . TERMINATE USER PROGRAM
               . . .
```

ACB PL/I Examples

This section contains examples of using direct Adabas calls in PL/ I . The previously defined Adabas files are used in each example.

```
/*** CONTROL BLOCK ***/
DCL 1    CONTROL_BLOCK,
    02   FILLER1               CHAR (2) INIT (' '),
    02   COMMAND_CODE          CHAR (2) INIT (' '),
    02   COMMAND_ID            CHAR (4) INIT (' '),
    02   FILE_NUMBER           BIN FIXED (15) INIT (0),
    02   RESPONSE_CODE         BIN FIXED (15) INIT (0),
    02   ISN                   BIN FIXED (31) INIT (0),
    02   ISN_LOWER_LIMIT       BIN FIXED (31) INIT (0),
    02   ISN_QUANTITY          BIN FIXED (31) INIT (0),
    02   FORMAT_BUFFER_LENGTH  BIN FIXED (15) INIT (100),
    02   RECORD_BUFFER_LENGTH  BIN FIXED (15) INIT (250),
    02   SEARCH_BUFFER_LENGTH  BIN FIXED (15) INIT (50),
    02   VALUE_BUFFER_LENGTH   BIN FIXED (15) INIT (100),
    02   ISN_BUFFER_LENGTH     BIN FIXED (15) INIT (20),
    02   COMMAND_OPTION_1      CHAR(1) INIT (' '),
    02   COMMAND_OPTION_2      CHAR(1) INIT (' '),
    02   ADDITIONS_1           CHAR(8) INIT (' '),
    02   ADDITIONS_2           CHAR(4) INIT (' '),
    02   ADDITIONS_3           CHAR(8) INIT (' '),
    02   ADDITIONS_4           CHAR(8) INIT (' '),
    02   ADDITIONS_5           CHAR(8) INIT (' '),
    02   COMMAND_TIME          BIN FIXED (31) INIT (0),
    02   USER_AREA             CHAR(4) INIT (' ');

/*** USER BUFFER AREAS ***/
DCL FORMAT_BUFFER    CHAR(100),
    RECORD_BUFFER    CHAR(250),
    SEARCH_BUFFER    CHAR(50),
    VALUE_BUFFER     CHAR(100),
    ISN_BUFFER       CHAR(20);
*
*

/***    ADDITIONAL FIELDS USED IN THE EXAMPLES ***/
DCL
    COMM_ID_X  BIN FIXED(31);
    COMM_ID    CHAR(4) BASED (ADDR(COMM_ID_X));
DCL      INPUT_KEY CHAR(8);
DCL      SYNC_CHECK_SWITCH CHAR(1) INIT('0');
DCL 1 RECORD_BUFFER_EX2,
         2   RECORD_BUFFER_A  CHAR(8),
         2   RECORD_BUFFER_B  DEC FIXED(3,0),
         2   FILLER3 CHAR(240);
DCL 1 RECORD_BUFFER_EX3,
         2   OPEN_RECORD_BUFFER,
             3   OPEN_RECORD_BUFFER_X CHAR(8),
```

```
             3   FILLER4 BIN FIXED(31),
          2   FILLER5 CHAR(18),
          2   UPDATED_XC CHAR(6),
          2   LAST_XD CHAR(8),
          2   FILLER6 CHAR(5),
     1 USER_DATA,
          2   RESTART_XD CHAR(8),
          2   RESTART_ISN BIN FIXED(31);
DCL       ADABAS ENTRY OPTIONS(ASM);
```

This section provides the following examples:

- Example 1
- Example 2
- Example 3

Example 1

- Find the set of records in file 2 with XB = 99.
- Read each record selected using the GET NEXT option.

Issue Open Command

```
*** Issue Open Command **/
EXMP1:
    COMMAND_CODE = 'OP';
    RECORD_BUFFER = 'ACC.';
    CALL ADABAS (CONTROL_BLOCK,FORMAT_BUFFER,RECORD_BUFFER);
    IF RESPONSE_CODE > 0        THEN GOTO EX1ERR;
```

Issue Find Command

```
/*** Issue Find Command ***/
    COMMAND_CODE = 'S1';
    COMMAND_ID = 'S101';
    FILE_NUMBER = 2;
    ISN_LOWER_LIMIT = 0;
    ISN_BUFFER_LENGTH = 0;
    FORMAT_BUFFER = '.';
    SEARCH_BUFFER = 'XB,3,U.';
    VALUE_BUFFER = '099';
    CALL ADABAS (CONTROL_BLOCK, FORMAT_BUFFER,
         RECORD_BUFFER, SEARCH_BUFFER, VALUE_BUFFER);
    IF RESPONSE_CODE > 0 THEN GOTO EX1ERR;
EX1A:
    IF ISN_QUANTITY = 0 THEN GOTO EX1EXIT;
EX1B:
    COMMAND_CODE = 'L1';
    ISN = 0;
    COMMAND_OPTION_1 = 'N';
    FORMAT_BUFFER  = 'RG.';
EX1C:
    CALL ADABAS (CONTROL_BLOCK,FORMAT_BUFFER,RECORD_BUFFER);
    IF RESPONSE_CODE = 0 THEN
         GOTO EX1D;
    IF RESPONSE_CODE = 3 THEN
```

```
         GOTO EX1EXIT;
EX1D:
    . . .PROCESS RECORD . . .
         GOTO EX1C;
```

Error Routine

```
/*** Error Routine ***/
EX1ERR:
/*   . DISPLAY ERROR MESSAGE */
/*   . TERMINATE USER PROGRAM */
```

Issue Close Command

```
/** Issue Close Command **/
EX1EXIT:
    COMMAND_CODE = 'CL';
    CALL ADABAS (CONTROL_BLOCK);
    IF RESPONSE_CODE > 0 THEN
         GOTO EX1ERR;
```

Example 2

- All records in file 1 are to be read in physical sequential order.
- Each record read is to be updated with the following values:
  - Field AA = ABCDEFGH
  - Field AB = 500
- User is to have exclusive control of file 1.

Issue Open Command

```
/*** Issue Open Command ***/
EXMP2:
    COMMAND_CODE = 'OP';
    RECORD_BUFFER = 'EXU=1.';
    CALL ADABAS (CONTROL_BLOCK,FORMAT_BUFFER,RECORD_BUFFER);
    IF RESPONSE_CODE > 0 THEN   GOTO EX2ERR;
```

Issue Read Physical Sequence Command

```
/*** Issue Read Physical Seq.  Command ***/
EX2A:
    COMMAND_ID = 'L201';
    FILE_NUMBER = 1;
    ISN = 0;
    FORMAT_BUFFER = 'GA.';
EX2B:
    COMMAND_CODE = 'L2';
    CALL ADABAS (CONTROL_BLOCK,FORMAT_BUFFER,RECORD_BUFFER);
    IF RESPONSE_CODE = 0 THEN   GOTO EX2C;
    IF RESPONSE_CODE = 3 THEN   GOTO EX2EXIT;
    GOTO EX2ERR;
```

Update Record

```
/*** Update record.  ***/
/* Same fields are to be updated as were read.  */
/* Same CID and FORMAT BUFFER can be used for update.  */
/* ISN of record to be updated is already in ISN field as a result of */
/* the L2 command.  */
EX2C:
    COMMAND_CODE = 'A1';
    RECORD_BUFFER_A = 'ABCDEFGH';
    RECORD_BUFFER_B = 500;
    CALL ADABAS (CONTROL_BLOCK,FORMAT_BUFFER,
                 RECORD_BUFFER_EX2);
    IF RESPONSE_CODE > 0 THEN   GOTO EX2ERR;
    GOTO EX2B;
```

Error Routine

```
/*** Error Routine ***/
EX2ERR:
/*   . DISPLAY ERROR MESSAGE */
/*   . TERMINATE USER PROGRAM */
```

Close User Session

```
/* Close User Session */
EX2EXIT:
    COMMAND_CODE = 'CL';
    CALL ADABAS (CONTROL_BLOCK);
    IF RESPONSE_CODE > 0 THEN   GOTO EX2ERR;
```

Example 3

This example illustrates a user session with ET logic. The user program is to perform the following functions:

1. During user session initialization, display information indicating the last successfully processed transaction of the previous user session.
2. For each user transaction:

- Accept from a terminal 8 characters of input that is used as the key for updating files 1 and 2.
- Issue a Find command for file 1 to determine if a record exists with field AA = input key.
- If no record is found, issue a message.
- If a record is found:
    - Delete the record from file 1;
    - Add a new record to file 2: Field RA = input key entered. Other fields to contain null value.
    - If the record cannot be successfully added, issue a BT command, display error message.

- If both updates are successful, issue an ET command.

Session Initialization

This section of the program is only executed during user session initialization.

- The OP command is issued with ET data of the previous session being read.
- A message is displayed on the terminal screen identifying the last successfully processed transaction of the user's previous session.

```
EX3:
    COMMAND_CODE = 'OP';
    COMMAND_OPTION_2 = 'E';
    ADDITIONS_1 = 'USER0003';
    ADDITIONS_3 = 'PASSWORD';
    RECORD_BUFFER = 'UPD=1,2.';
    CALL ADABAS (CONTROL_BLOCK,FORMAT_BUFFER,RECORD_BUFFER);
    IF RESPONSE_CODE = 9 THEN   GOTO EX3;
    IF RESPONSE_CODE > 0 THEN
         GOTO EX3ERR;
EX3A:
    COMM_ID = COMMAND_ID;
    IF COMM_ID_X = 0 THEN
         GOTO EX3B;
/*  Display ET data (contained in RECORD BUFFER) on screen to inform user of
    last successfully processed transaction of previous user session. */
         . . .DISPLAY ET DATA. . .
    GOTO EX3C;
EX3B:
/*                          */
/*** No ET data received.  */
/*   Display message that no transactions were successfully processed during
    the previous user session. */
         . . .DISPLAY MESSAGE . . .
/*                              */
/*** Transaction processing.  ***/
/* This section is executed for each user transaction.  */
EX3C:
         . . .ACCEPT INPUT FROM TERMINAL. . .
/*                                                    */
/* Issue Find command for file 1 to determine if rec exists with field AA
   equal to input key entered. */
EX3D:
    COMMAND_CODE = 'S4';
    COMMAND_ID = ' ';
    FILE_NUMBER = 1;
    ISN_LOWER_LIMIT = 0;
    FORMAT_BUFFER = '.';
    SEARCH_BUFFER = 'AA.';
    VALUE_BUFFER = INPUT_KEY;
    CALL ADABAS (CONTROL_BLOCK,FORMAT_BUFFER,RECORD_BUFFER,
         SEARCH_BUFFER,VALUE_BUFFER,ISN_BUFFER);
    IF RESPONSE_CODE = 0 THEN
         GOTO EX3E;
    GOTO EX3ERR;
EX3E:
    IF ISN_QUANTITY > 0 THEN
         GOTO EX3F;
/*                                                    */
/* No record found, issue message requesting correction.  */
```

```
         . . .ISSUE MESSAGE . . .
    GOTO EX3C;
/*                                */
/* Delete record from file 1.  */
/* ISN of record to be deleted is already in ISN field and in hold
status
   as a result of the S4 command. */
EX3F:
    COMMAND_CODE = 'E4';
    CALL ADABAS (CONTROL_BLOCK);
    IF RESPONSE_CODE = 0 THEN
         GOTO EX3G;
    IF RESPONSE_CODE = 9 THEN
         GOTO EX3D;
    GOTO EX3ERR;
/***Add new record to file 2.  */
EX3G:
    COMMAND_CODE = 'N1';
    FILE_NUMBER = 2;
    FORMAT_BUFFER = 'RA.';
    RECORD_BUFFER = INPUT_KEY;
    CALL ADABAS (CONTROL_BLOCK,FORMAT_BUFFER,RECORD_BUFFER);
    IF RESPONSE_CODE = 0 THEN
         GOTO EX3I;
    IF RESPONSE_CODE = 9 THEN
         GOTO EX3D;
/*                                  */
/*  Attempt to add new record not successful. Backout transaction and
notify
    user that error condition exists. */
    COMMAND_CODE = 'BT';
    CALL ADABAS (CONTROL_BLOCK);
    IF RESPONSE_CODE = 0 THEN
         GOTO EX3H;
/*                              */
/*  Backout not successful.   */
     . . .ISSUE MESSAGE INDICATING BACKOUT NOT SUCCESSFUL . .
    GO TO EX3ERR.
/*                                */
EX3H:
/*** Backout successful.    ***/
/* Issue message indicating error condition detected while adding new
record.*/
         . . .ISSUE MESSAGE. . .
    GOTO EX3ERR;
/*                                     */
/*** Updates successfully executed.  ***/
/*   Issue ET command with ET data.    */
EX3I:
    COMMAND_CODE = 'ET';
    COMMAND_OPTION_2 = 'E';
    RECORD_BUFFER = INPUT_KEY;
    CALL ADABAS (CONTROL_BLOCK,FORMAT_BUFFER,RECORD_BUFFER);
    IF RESPONSE_CODE = 0 THEN
         GOTO EX3C;
    IF RESPONSE_CODE = 9 THEN
         GOTO EX3D;
/*                                     */
/***    Error Routine    ***/
```

```
EX3ERR:
/*  . DISPLAY ERROR MESSAGE */
/*  . TERMINATE USER PROGRAM */
         . . .
```

ACB Fortran Example

This section contains an example of using direct Adabas calls in FORTRAN. The previously defined Adabas files are used in each example.

```
C   *** CONTROL BLOCK ***
    INTEGER*4     CB(20),CID,ISN,ISNL,ISNQ
    INTEGER*4     ADD1(2),ADD2,ADD3(2),ADD4(2),ADD5(2)
    INTEGER*4     CTIME,UAREA
    INTEGER*2     CBI(40),CCODE,FNR,RC,FBL,RBL,SBL,VBL,IBL
    LOGICAL*1     CBL(80),COPT1,COPT2
    EQUIVALENCE       (CB(1),CBI(1),CBL(1))
    EQUIVALENCE       (CID,CB(2)),(ISN,CB(4))
    EQUIVALENCE       (ISNL,CB(5)),(ISNQ,CB(6))
    EQUIVALENCE       (ADD1(1),CB(10)),(ADD2,CB(12)),(ADD3(1),CB(13))
    EQUIVALENCE       (ADD4(1),CB(15),(ADD5(1),CB(17))
    EQUIVALENCE       (CTIME,CB(19)),(UAREA,CB(20))
    EQUIVALENCE       (CCODE,CBI(2)),(FNR,CBI(5)),(RC,CBI(6))
    EQUIVALENCE       (FBL,CBI(13)),(RBL,CBI(14)),(SBL,CBI(15))
    EQUIVALENCE       (VBL,CBI(16)),(IBL,CBI(17))
    EQUIVALENCE       (COPT1,CBL(35)),(COPT2,CBL(36))


C   *** USER BUFFER AREAS ***
    INTEGER*4 FB(25),RB(50),SB(10),VB(10),IB(50)
 *
 *
C   *** ADDITIONAL FIELDS USED IN THIS EXAMPLE  ***
    LOGICAL*1 BLANK/1H /,COPH/1HH/,PERIOD/1H./,COPN/1HN/
    INTEGER*2 S1/2HS1/,L1/2HL1/,CL/2HCL/
    INTEGER*4 CID1/4HS101/,FB1/4H.  /,FB2/4HRG. /,SB1/4HXB,3/
INTEGER*4 SB2/4H,U. /,VB1/4H099 /
```

This section provides the following example:

- Example 1

Example 1

- Find the set of records in file 2 with XB = 99.
- Read each record selected using the GET NEXT option.

Initialize Control Block

```
c*** Initialize Control Block
    DO 5 I=1,80
    CBL(I)=BLANK
5   CONTINUE
    DO 10 I=3,6
    CB(I)=0
10  CONTINUE
    CBI(13)=100
    CBI(14)=200
```

```
    CBI(15)=40
    CBI(16)=40
    CBI(17)=200
    CBI(19)=0
```

Issue Find Command

```
c***Issue FIND Command
    CCODE=S1
    CID=CID1
    FNR=2
    ISNL=0
    COPT1=COPH
    FB(1)=FB1
    SB(1)=SB1
    SB(2)=SB2
    VB(1)=VB1
    CALL ADABAS(CB,FB,RB,SB,VB,IB)
    IF(RC.NE.0) GO TO 50
    IF(ISNQ.EQ.0) GO TO 100
```

Read Each Record Selected

```
c***Read Each Record Selected
15  CONTINUE
    CCODE=L1
    ISN=0
    COPT1=COPN
    FB(1)=FB2
    CALL ADABAS(CB,FB,RB)
    IF(RC.EQ.0) GO TO 30
    IF(RC.EQ.3) GO TO 100
    PRINT 60,RC,CCODE
60  FORMAT(1H0,'ADABAS ERROR CODE',I4,' FROM '.A2,' COMMAND')
    GO TO 50
30  CONTINUE
C   ...PROCESS RECORD...
    GO TO 15
50  CONTINUE
    STOP
100 CONTINUE
    ...
```