Chapter 8

WaWa ("Easy") I/O

8.1 Overview

There is a fairrly coherent set of routines which attempt to hide many of the nasty details mentioned in the previous chapters. They perform most catalog file operations for the programmer and hide the details of calls to COMOFF, MINIT, MDISK, ZCREAT, et al. In many cases these cost memory and/or speed, but for computation-bound algorithms these are probably not important.

Any task which uses the WaWa package and creates scratch files should include the /CFILES/ common given in the INCLUDE DFIL.INC. The values of IBAD should be filled in using the contents of AIPS adverb BADDISK. This allows the scratch file creation routine to avoid putting files on user selectable disks.

8.2 Salient Features of the WaWa I/O package

1. Each main task calls a single setup routine; a maximum of 5 simultaneously open image files is allowed.
2. All the parameters needed to specify a cataloged file are gathered into a single array, called a namestring.
3. The WaWa package hides the interface between the parameter passing subroutines (e.g., GTPARM) and the I/O routines.
4. Many subroutine calls are combined so that e.g., ZPHFIL, CATDIR, CATIO, and MINIT, more or less disappear from sight.
5. A general clean-up subroutine for closing files and destroying scratch files is provided.
6. "Hidden" buffers large enough to hold a 2048-point image row are provided. These make double buffered I/O look more like FORTRAN I/O on the large mainframes.

8.3 Namestrings

In order to reduce the many arguments required for the fundamental AIPS I/O routines needed to specify the desired file the WaWa package uses a namestring. With a namestring it is possible to refer to any cataloged file by a character string. The name string used for WAWA I/O is a CHARACTER string of length 36 of the following form:

```
 1:12 C*12 Name
13:18 C*6 Class
19:20 C*2 Physical type
21:27 I7 Sequence number
28:29 I2 Disk number
30:36 I7 User id number
```

However, application software should not directly access the contents of this string since it is subject to change. Instead, the following routines convert to and from the WAWA name string:

- H2WAWA converts AIPS adverb values to a namestring.
- WAWA2A cracks a namestring into it's component parts.
- A2WAWA forms a namestring from CHARACTER and INTEGER variables.

The call sequences to these routines are described at the end of this chapter.

Some null values are allowed that cause defaults to be invoked.

1. A blank name means "any NAME".
2. A blank class means "any CLASS".
3. A sequence number = 0 means "any SEQ".
4. A disk number = 0 means "any DISK".
5. A blank Physical type means a physical type of "MA".
6. A user id = 0 means USID of NLUSER i.e., the task user. 32,000 means "any USID"

A value of "SC" for the Physical type means "scratch". In this case, all the package subroutines substitute internally (some do not alter the calling namestring) a NAME, CLASS, and USID unique to the main task and AIPS initiator (i.e., interactive AIPS 1, 2 or BATCH AIPS 6, 7, ...):

```
NAME = 'SCRATCH FILE'
CLASS = TSKNAM|NPOPS
USID = NLUSER
```

8.4 Subroutines

The following is a list of the WaWa package of routines with a short description of each. Detailed descriptions of the function and call sequence of these routines can be found at the end of this chapter.

- IOSET - Setup I/O; a maximum of 5 files can be opened at any time.
- FILOPN - Open a file, particularly associated files.
- OPENCF - Open a cataloged file.
- FILIO - Do I/O to a non-map file.
- MAPWIN - Set a multi-dimensional window on an open map.
- MAPXY - Set a 2-dim window on top plane of a map.
- MAPIO - Read or write to a map.
- FILCLS - Close a map or non-map file.
- FILCR - Create a non-map file.
- MAPCR - Create a map file.
- FILDES - Destroy either a map or non-map file.
- UNSCR - Destroy all scratch files.
- CLENUP - Call UNSCR and close any still open files.

- MAPMAX - Find MAX & MIN of a map and enter into catalog.
- FILNUM - Find WaWa pointers to open file (used for history).
- GETHDR - Retrieve catalog header for an open cataloged file.
- SAVHDR - Save header in catalog for an open cataloged file.
- HDRINF - Retrieve specified items from map header.
- TSKBEG - Combination of IOSET and some task startup chores.
- TSKEND - Some task cleanup chores.

8.5 Things WaWa Can't Do Well or At All

There are several applications for which the WaWa routines are inadequate. The non-map I/O routines are much inferior to the other AIPS non-map I/O routines. Other applications, such as uv data handling and plotting, are not provided for at all. History files may be written in tasks using WaWa I/O, but it requires digging in the the WaWa commons. The following sections suggest possible courses of action.

8.5.1 Non-map files

The WaWa package is not overly useful for non-map I/O at the moment. The user will want to consult the chapter on disk I/O and the routines TABINI and TABIO for more useful software.

8.5.2 UV data files

No help here. See the chapter on disk I/O.

8.5.3 Plotting

The WaWa package has no plotting capability. See the chapter in this manual on plotting.

8.5.4 History

The WaWa package has no capacity to copy or write into history files. See the chapter on tasks and in particular the routines HISCOP and HIADD. In addition, you will need to determine the catalog slot numbers of the relevant files from the /WAWAIO/ common variable FILTAB(POCAT,*) (file must be open to do so). Use FILNUM. The task HGEOM provides a useful example of history writing within the WaWA I/O system.

8.5.5 More than 5 I/O Streams at a Time

If a task may need to have more than 5 map or non-map I/O streams open at the same time, then serious restructuring of the WaWa commons is needed. You are better off ignoring WaWa I/O and using the standard I/O described in the chapter on disk I/O.

8.5.6 I/O to Tapes

No help here. See the chapter on device I/O.

8.6 Additional goodies and "helpful" hints

A number of features have been added to the WaWa package to increase it usefulness. These will be discussed in the following sections. Also on occasion the programmer will have to find some of the things the WaWa package has hidden; a discussion of where WaWa hides useful information is also given in the following sections.

8.6.1 Use of LUNs

The LUN used does convey meaning. Legal values range from 9 through 30. However, values 16 through 25 convey an implication that the file is a map file, value 9 is reserved for the TV, and values 10 through 15 may get you into trouble. Use 26–30 for non-maps.

8.6.2 WaWa commons

The WaWa package hides many things in several commons. Frequently the programmer needs to know the contents of these commons. The following sections describe the contents of the commons.

Information common

The primary common in the WaWa package is obtained by the INCLUDE DITB.INC.. The text of this and other relevant includes are shown at the end of this chapter. The name of the primary WaWa I/O common is /WAWAIO/ and its contents are as follows:

```
WRIT        C*4   'WRIT'     I/O control strings
REED        C*4   'READ'
CLWR        C*4   'CLWR'     Catalog control strings
CLRD        C*4   'CLRD'
REST        C*4   'REST'
OPEN        C*4   'OPEN'
CLOS        C*4   'CLOS'
SRCH        C*4   'SRCH'
INFO        C*4   'INFO'
UPDT        C*4   'UPDT'
FINI        C*4   'FINI'     I/O control string
CSTA        C*4   'CSTA'     Catalog control string
INDEF       R     'INDE'     Blanked floating point pixel

SUBNAM     C*6(8) Subroutine names: CATDIR, CATIO, MINIT,
                  MDISK, ZCLOSE, ZCREAT, ZDESTR, ZOPEN

LINT        I     Number integer values in one IO buffer
LREAL       I     Number real values in one IO buffer
NFIL        I     Number simultaneous open map files
EFIL        I     Size of FILTAB ( 5 + NFIL) - number of
                  simultaneous files of all types
QUACK       I     0 => restart AIPS at end, 1 => already done

POLUN       I     FILTAB pointer for LUN value (1)
POFIN       I     FILTAB pointer for I/O table pointer value
                  (2)
POVOL       I     FILTAB pointer for disk number value (3)
POCAT       I     FILTAB pointer for cat location value (4)
POIOP       I     FILTAB pointer for opcode number (5):
                  values 1 => write, 2=> read, <0 => new win
```

```
POASS        I     FILTAB pointer for is it associated file
                   (6): 1 => assoc, 0 => main file
POBPX        I     FILTAB pointer for bytes/pixel code (7)
PODIM        I     FILTAB pointer for # axes (8)
PONAX        I     FILTAB pointer for # points on each of 7
                   axes (9)
POBLC        I     FILTAB pointer for Bottom left corner (16)
POTRC        I     FILTAB pointer for Top right corner (23)
PODEP        I     FILTAB pointer for current depth in I/O on
                   axes 2 - 7 (30), Area (36) used for integer
                   map (input) blanking code.
POBL         I     FILTAB pointer for block offset start I/O
                   in the current plane (37)

FILTAB(38,EFIL)  I     Table to hold all the values pointed
                       at by the PO... pointers above: (e.g.,
                       the cat number is = FILTAB (POCAT, n)
                       where n is found by finding that
                       FILTAB (POLUN, n) which = desired LUN
                       (Only for open files!!)
```

Catalog and Buffer Commons

There are 2 other commons which are used heavily. They are /MAPHDR/ which is a work area for map headers containing the equivalenced arrays CATBLK, CATH, CATR, and CATD. The contents of this common are changed frequently by the basic WaWa I/O routines, but it can be used, for example, to get the catalog header record after a call to FILOPN or OPENCF. This common may be obtained by the include DCAT.INC. The other common, called /WAWABU/ from INCLUDE DBUF.INC, contains:

```
RMAX     R(10)    1-5 used by MAPIO for scale factor
RMIN     R(10)    1-5 used by MAPIO for offset
WBUFF    I(256)   scratch buffer for catalog access
RBUF     R(*)     I/O buffers for map I/O.
```

The areas RMAX and RMIN for subscripts 6 through 10 could be used by a programmer, for example, to keep track of max/min. If no map file is currently open, RBUF is a large and useful scratch area of core.

Declaration of Commons

If a WaWa I/O task (or any other task for that matter) is to be overlaid on some computers, then all commons must be declared in the main program. For the WaWa system, this may be done by the following list of includes:

```
INCLUDE 'INCS:DBUF.INC'     WaWa buffer/table sizes
INCLUDE 'INCS:DITB.INC'     WaWa I/O common
INCLUDE 'INCS:DDCH.INC'     System parms
INCLUDE 'INCS:DHDR.INC'     Header pointers
INCLUDE 'INCS:DMSG.INC'     Messages, POPS #, ...
INCLUDE 'INCS:DCAT.INC'     Catalog header
INCLUDE 'INCS:DFIL.INC'     Gives BADDISK
```

8.6.3 Error return codes

A uniform system of error code numbers has been adopted in the WaWa I/O package. These codes are consistent with the error codes used by many I/O routines, but not with the other error codes in the multitudinous collection of AIPS routines. They are:

```
 1 => File not open
 2 => Input parameter error
 3 => I/O error ("other")
 4 => End of file (hardware generated, see 9)
 5 => Beginning of medium
 6 => End of medium
 7 => buffer too small
 8 => Illegal data type
 9 => Logical end of file (software generated, not hardware)
10 => Catalog operation error
11 => Catalog status error
12 => Map not in catalog
13 => EXT file not in catalog
14 => No room in header/catalog
16 => Illegal window specification
17 => Illegal window specification for writing a file
21 => Create: file already exists
22 => Create: volume unavailable
23 => Create: space unavailable
24 => Create: "other"
25 => Destroy: "other"
26 => Open: "other"
```

8.7 INCLUDEs

There are several types of INCLUDE file which are distinguished by the first character of their name. Different INCLUDE file types contain different types of Fortran declaration statements as described in the following list.

- Pxxx.INC. These INCLUDE files contain declarations for parameters and the PARAMETER statements.
- Dxxx.INC. These INCLUDE files contain Fortran type (with dimension) declarations, COMMON and EQUIVALENCE statments.
- Vxxx.INC. These contain Fortran DATA statements.
- Zxxx.INC. These INCLUDE files contain declarations which may change from one computer or installation to another.

8.7.1 DBUF.INC

```
C                                                           Include DBUF.
      REAL      RBUF(20480), RMAX(10), RMIN(10)
      INTEGER   WBUFF(256), IBUF(1)
      COMMON /WAWABU/ RMAX, RMIN, WBUFF, RBUF
      EQUIVALENCE (RBUF(1), IBUF(1))
C                                                           End DBUF.
```

8.7.2 DCAT.INC

```
C                                                           Include DCAT.
C                                        catalog header common
      INTEGER   CATBLK(256)
      REAL      CATR(256)
      HOLLERITH CATH(256)
      DOUBLE PRECISION CATD(128)
      COMMON /MAPHDR/ CATBLK
      EQUIVALENCE (CATBLK, CATR, CATH, CATD)
C                                                           End DCAT.
```

8.7.3 DFIL.INC

```
C                                                           Include DFIL.
C                                        AIPS system catalog and scratch
      INTEGER   NSCR, SCRVOL(128), SCRCNO(128), IBAD(10), LUNS(10),
     *   NCFILE, FVOL(128), FCNO(128), FRW(128), CCNO
      LOGICAL   RQUICK
      COMMON /CFILES/ RQUICK, NSCR, SCRVOL, SCRCNO, NCFILE, FVOL, FCNO,
     *   FRW, CCNO, IBAD, LUNS
C                                                           End DFIL.
```

8.7.4 DITB.INC

```
C                                                           Include DITB.
C                                        Wawa I/O common
      REAL    INDEF
      CHARACTER WRIT*4, REED*4, CLWR*4, CLRD*4, REST*4, OPEN*4, CLOS*4,
     *   SRCH*4, INFO*4, UPDT*4, FINI*4, CSTA*4, SUBNAM(8)*6
      INTEGER    LINT, LREAL, NFIL, EFIL, QUACK,
     *   POLUN, POFIN, POVOL, POCAT, POIOP, POASS, POBPX,
     *   PODIM, PONAX, POBLC, POTRC, PODEP, POBL, FILTAB(38,10)
      COMMON /WAWCHR/  WRIT, REED, CLWR, CLRD, REST, OPEN, CLOS,
     *   SRCH, INFO, UPDT, FINI, CSTA, SUBNAM
      COMMON /WAWAIO/ INDEF,  LINT, LREAL, NFIL, EFIL, QUACK,
     *   POLUN, POFIN, POVOL, POCAT, POIOP, POASS, POBPX,
     *   PODIM, PONAX, POBLC, POTRC, PODEP, POBL, FILTAB
C                                                           End DITB.
```

8.8 Detailed Descriptions of the Subroutines

8.8.1 A2WAWA

WaWa IO system: Packs Wawa-IO Namestring having format A12, A6, A2, I7, I2, I7 for NAME, CLASS, PTYPE, SEQ, VOL, USID from its component parts

```
A2WAWA (NAME, CLASS, SEQ, PTYPE, VOL, USID, NAMEST)
Inputs:
```

```
   NAME    C*12   file name
   CLASS   C*6    file class (6 chars)
   SEQ     I      file sequence number
   PTYPE   C*2    file physical type (2 chars)
   VOL     I      file disk number
   USID    I      user number
Outut:
   NAMEST  C*36   WaWa Namestring
```

8.8.2 CLENUP

WaWa IO system: Close all files opened with FILOPN. Destroy scratch files.

```
  CLENUP
     no arguments
```

8.8.3 FILCLS

WaWa IO system: Close a file opened by FILOPN, taking care of catalog bookkeeping and flush last write buffers if any.

```
FILCLS (LUN)
Inputs:
   LUN       I      Logical unit no. of file to close
```

8.8.4 FILCR

WaWa IO system: Create an associated or scratch non-map file

```
FILCR (NAMS, TYPE, NBLOCK, VER, ERROR)
Inputs:
   NAMS      C*36  NAMESTRING specifying catalog block to which
                   file is associated: NAME,CLASS,CATTYPE,SEQ,VOL,
                   USID.  NAME,CLASS,USID ignored for scratch files.
   TYPE      C*2   Associated file type for non-scratch files
                   Ignored for scratch files
In/out:
   NBLOCK    I      Number of 512-type blocks in file: in requested,
                    out actual
Outputs:
   VER       I      Version number of file created
   ERROR     I      Error code: 0 => ok
                          10 => catalog error
                          12 => map not in catalog
                          14 => no room for another ext. type
                          21 => ZCREAT: file already exists
                          22 => ZCREAT: volume unavailable
                          23 => Disk space unavailable
                          24 => Other create errors
Common: /MAPHDR/ modfified extensively for scratch file create
   a little for associated file
```

8.8.5 FILDES

WaWa IO system: Destroy the file specified by NAMS, TYPE, VER

```
FILDES (NAMS, ASSOC, TYPE, VER, ERROR)
Inputs:
   NAMS      C*36  NAMESTRING specifying catalog block to which
                   file is associated: NAME,CLASS,CATTYPE,SEQ,VOL,
                   USID.  NAME,CLASS,USID ignored for scratch files.
   ASSOC     L     File is an associated file,i.e. not cataloged
                   ASSOC will be taken as FALSE if NAMS(8)='SCxx'
   TYPE      C*2   Associated file type; ignored if ASSOC is false
   VER       I     Associated file version; ignored if ASSOC is fal
Outputs:
   ERROR     I      Error code:  0 => o.k.
                      10 => catalog error
                      11 => map too busy to destroy
                      12 => map not found in catalog
                      13 => extension file not in catalog
                      25 => other destroy errors
```

8.8.6 FILIO

WaWa IO system: Read or Write a single record from/to a non-map file which has been opened with FILOPN (256 integers). Adds a 'READ' status to catlg on first call.

```
FILIO (OP, LUN, REC, DATA, ERROR)
Inputs:
   OP          C*4      READ or WRIT
   LUN         I        File Logical Unit Number
   REC         I        Which record out of file (1-relative)
In/Out:
   DATA(256)   I        Data record to input or output
Output:
   ERROR       I        Error return from ZFI3
                        0 => o.k.
                        1 => file not open
                        2 => input error e.g. file not opened for
                             desired operation
                        3 => i/o error
                        4 => end of file
                        5 => beginning of medium
                        6 => end of medium (from IO system)
                       10 => catalog error
```

8.8.7 FILNUM

WaWa IO system: find the FILTAB entry for a file

```
FILNUM (LUN, IFIL, ERROR)
Inputs:
   LUN     I       Logical unit number of file
Outputs:
```

```
   IFIL    I       Entry number (2nd subscript to FILTAB)
   ERROR   I       Error code: 0 => ok, 1 => file not open
```

8.8.8 FILOPN

WaWa IO system: Open the file specified by NAMS and associate it with Logical Unit number LUN.

```
FILOPN (LUN, NAMS, ASSOC, TYPE, VER, ERROR)
Inputs:
   LUN       I     Logical Unit Number
   ASSOC     L     File is an associated file,i.e. not cataloged
                   ASSOC will be taken as FALSE if NAMS(8)='SCxx'
   TYPE      C*2   Associated file type; ignored if ASSOC is false
   VER       I     Associated file version; ignored if ASSOC is fal
In/Out:
   NAMS      C*36  NAMESTRING specifying catalog block to which
                   file is associated: NAME,CLASS,CATTYPE,SEQ,VOL,
                   USID.  NAME,CLASS,USID ignored for scratch files.
Outputs:
   ERROR     I     Error code:  0 => o.k.
                        2 => input error: bad or in use LUN
                       10 => catalog error
                       12 => map not found
                       13 => extension file not in catalog
                       14 => no room in FILTAB
                       22 => volume not available
                       26 => open error
```

8.8.9 GETHDR

WaWa IO system: Retrieve the catalog header block for a file that is already open (via FILOPN or OPENCF)

```
GETHDR (LUN, CAT, ERROR)
Inputs:
   LUN            I      Logical Unit No. of file
Outputs:
   CAT(256)       I      Returned Header block
   ERROR          I      Error code: 0 => ok
                                     1 => file not open
                                    10 => catlg error
```

8.8.10 HDRINF

WaWa IO system: Return a number of items from the header block of an open, cataloged file.

```
HDRINF (LUN, WTYPE, SITEM, NITEM, OUTPUT, ERROR)
Inputs:
   LUN       I       Logical Unit No. of file
   WTYPE     I       Data type: 1 = I,   2 = R   3 = D   6 = C*8
   SITEM     I       Index # of 1st item wanted, indexed in a
                     system appropriate to WTYPE (R   for C*8)
   NITEM     I       Number of items requested
```

```
Outputs:
   OUTPUT(*) ???     Array into which items go
   ERROR     I       Error code: 0 => ok
                                 1 => file not open
                                 2 => nonsense input parms
                                10 => catlg read error
Common /MAPHDR/ receives the header read from catlg file
```

8.8.11 H2WAWA

WaWa IO system: packs AIPS adverb values (Holleriths, floating points) into a WaWa IO Namestring having format A12, A6, A2, I7, I2, I7 for NAME, CLASS, PTYPE, SEQ, VOL, USID

```
H2WAWA (NAME, CLASS, SEQ, PTYPE, VOL, USID, NAMEST)
Inputs:
   NAME    H(3)   file name
   CLASS   H(2)   file class (6 chars)
   SEQ     R      file sequence number
   PTYPE   H      file physical type (2 chars)
   VOL     R      file disk number
   USID    R      user number
Output:
   NAMEST  C*36   WaWa Namestring
```

8.8.12 IOSET

This routine initializes the I/O tables; calls ZDCHIN; allocates buffer space for map I/O to 5 files adequate for 2048 real or 1024 complex pixels per line.

```
IOSET

no calling arguments
```

8.8.13 MAPCR

WaWa IO system: Create and catalog a map whose catalog description is defined by the namestring NAMS, and whose size is specified by the KIDIM and KINAX parameters in the Header.

```
MAPCR (ONAMS, NAMS, HDR, ERROR)
Inputs:
   ONAMS  C*36    Namestring of related "input" file - must be
                  complete and correct; used to complete defaults
                  in NAMS (typically the input file namestring).
In/Out:
   NAMS   C*36    Namestring NAME:CLASS:TYPE:SEQ:VOL:USID of map to
                  be created; can contain blanks, wildcards...
   HDR    I(256)  Catalog header for map, containing enough info to
                  define size.  The updated header is returned for
                  real images, not SC files
Outputs:
   ERROR  I       Error code: 0 => ok
```

```
                         10 => catalog error
                         14 => no room in catalog
                         21 => file already exists
                         23 => create error
```

8.8.14 MAPIO

WaWa IO system: Do I/O from a file opened using FILOPN to area DATA

```
MAPIO (OP, LUN, DATA, ERROR)
Inputs:
   OP            C*4     'READ' or 'WRIT'
   LUN           I       File logical unit no.
Input/output:
   DATA(*)       R       Data in or out
Output:
   ERROR         I       Error code: 0 => ok
                             1 => file not open
                             2 => bad input parms
                           3-6 => IO errors
                             8 => Bad data type (ie write integers)
                             9 => IO is complete (software
                                  generated EOF)
                            10 => catalog read/write error
                            11 => Catalog status error
```

8.8.15 MAPMAX

WaWa IO system: Determine max and min of a map opened by FILOPN and update CAT block accordingly

```
MAPMAX (LUN, XMAX, XMIN, ERROR)
Inputs:
   LUN     I   Logical Unit No. of map
Outputs:
   XMAX    R   Maximum in map
   XMIN    R   Minimum
   ERROR   I   Error codes: 0 => ok
                            1 => file not open
                            2 => input parms error
                          3-6 => IO errors
                           10 => catalog read/... error
```

8.8.16 MAPWIN

WaWa IO system: Set or reset parameters for a window on MAP I/O File must be opened first with FILOPN.

```
MAPWIN (LUN, BLC, TRC, ERROR)
Inputs:
   LUN     I      Logical Unit No. of file (must be open)
   BLC     R(7)   Lower bounds of map subrectangle
   TRC     R(7)   Upper bounds of map subrectangle
```

```
Outputs:
   ERROR   I      Error codes: 0 => ok
                       1 => file not open
                      10 => catalog error
                      16 => bad window specification

                      17 => partial row specified on write.
```

8.8.17 MAPXY

WaWa IO system: Set windows so that MAPIO returns a subrectangle of the top plane of a map

```
MAPXY (LUN, WIN, ERROR)
Inputs:
   LUN    I      Logical Unit No. of an open map
   WIN    R(4)   Corners of rectangle.  If WIN(1)=0.0, whole top
                 plane is taken.
Output:
   ERROR  I      As returned by MAPWIN
```

8.8.18 OPENCF

WaWa IO system: Open a MAIN (i.e. Cataloged) file and associate it with Logical Unit Number LUN.

```
OPENCF (LUN, NAMS, ERROR)
Inputs:
   LUN        I      Logical Unit No.
In/out:
   NAMS       C*36   Catalog identification NAMESTRING:
                     NAME:CLASS:PTYPE:SEQIN:VOL:USID
                     NAME, CLASS, & USID ignored if PTYPE = 'SC'.
OUTPUTS:
   ERROR      I      Error codes: 0 => o.k.
                     Otherwise as returned by FILOPN
```

8.8.19 SAVHDR

WaWa IO system: Save the catalog header block for a file that is already open (via FILOPN or OPENCF)

```
SAVHDR (LUN, CATBLK, ERROR)
Inputs:
   LUN      I        Logical Unit No. of file
   CATBLK   I(256)   Saved Header block
Outputs:
   ERROR    I        Error code: 0 => ok
                        1 => file not open
                        10 => catlg error
```

8.8.20 TSKBEG

WaWa IO system: Do most of the operations necessary to begin a task: Calls IOSET, calls GTPARM to get parameters, and, if appropriate, calls RELPOP. For ≤ 5 simultaneously open map files. You should end with TSKEND.

```
TSKBEG (PRGNAM, NPARM, RPARM, ERROR)
Inputs:
   PRGNAM   C*6    Task name
   NPARM    I      No. of real parameters passed by AIPS
OUTPUTS:
   RPARM    R(*)   Array to receive passed parameters
   ERROR    I      Error return: 0 => Okay
                      0 < ERROR <10  => Error return from GTPARM
```

8.8.21 TSKEND

WaWa IO system: Terminate a task, including calls to CLENUP and RELPOP if appropriate. Also close down messages.

```
TSKEND (IRET)
Inputs:
   IRET      I      A return code passed to AIPS if task was run
                    in wait mode: 0 => ok, else => failure.
```

8.8.22 UNSCR

WaWa IO system: Destroy all scratch files created by this task.

```
  UNSCR
     no arguments
```

8.8.23 WAWA2A

WaWa IO system: unpacks Wawa-IO Namestring having format A12, A6, A2, I7, I2, I7 for NAME, CLASS, PTYPE, SEQ, VOL, USID into component parts

```
WAWA2A (NAMEST, NAME, CLASS, SEQ, PTYPE, VOL, USID)
Input:
   NAMEST  C*36   WaWa Namestring
Outputs:
   NAME    C*12   file name
   CLASS   C*6    file class
   SEQ     I      file sequence number
   PTYPE   C*2    file physical type
   VOL     I      file disk number
   USID    I      user number
```