MODIFICATIONS OF THE U.S. GEOLOGICAL SURVEY MODULAR, FINITE-DIFFERENCE, GROUND-WATER FLOW MODEL TO READ AND WRITE GEOGRAPHIC INFORMATION SYSTEM FILES

U.S. GEOLOGICAL SURVEY
Open-File Report 92-50

Prepared in Cooperation with
INTERGOVERNMENTAL RESOURCE CENTER,
CLARK COUNTY, WASHINGTON
# CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract</td>
<td>1</td>
</tr>
<tr>
<td>Introduction</td>
<td>1</td>
</tr>
<tr>
<td>Purpose and scope</td>
<td>2</td>
</tr>
<tr>
<td>Organization and Nomenclature</td>
<td>2</td>
</tr>
<tr>
<td>MODFLOWARC design</td>
<td>2</td>
</tr>
<tr>
<td>Installation and operation of MODFLOWARC</td>
<td>3</td>
</tr>
<tr>
<td>Instructions for input and output data for MODFLOWARC</td>
<td>3</td>
</tr>
<tr>
<td>Basic package input</td>
<td>5</td>
</tr>
<tr>
<td>Basic package output</td>
<td>8</td>
</tr>
<tr>
<td>Block-centered flow package input</td>
<td>10</td>
</tr>
<tr>
<td>River package Input</td>
<td>13</td>
</tr>
<tr>
<td>Recharge package input</td>
<td>15</td>
</tr>
<tr>
<td>Well package input</td>
<td>17</td>
</tr>
<tr>
<td>Drain package input</td>
<td>19</td>
</tr>
<tr>
<td>Evapotranspiration package input</td>
<td>21</td>
</tr>
<tr>
<td>General-head boundary package input</td>
<td>23</td>
</tr>
<tr>
<td>Streamflow-routing package input</td>
<td>25</td>
</tr>
<tr>
<td>Utility modules input</td>
<td>27</td>
</tr>
<tr>
<td>Documentation of MODFLOWARC</td>
<td>28</td>
</tr>
<tr>
<td>AML MODFLOWARC program</td>
<td>28</td>
</tr>
<tr>
<td>MODFLOWARC.F77 program</td>
<td>32</td>
</tr>
<tr>
<td>File opening module MODFIL in MODFLOWARC</td>
<td>34</td>
</tr>
<tr>
<td>Subroutine MODFLOW in MODFLOWARC</td>
<td>37</td>
</tr>
<tr>
<td>Basic package modules</td>
<td>41</td>
</tr>
<tr>
<td>BAS1RPARC</td>
<td>41</td>
</tr>
<tr>
<td>BAS1OCARC</td>
<td>44</td>
</tr>
<tr>
<td>BAS1OTARCC</td>
<td>47</td>
</tr>
<tr>
<td>SBAS1DARC</td>
<td>49</td>
</tr>
<tr>
<td>SBAS1HARC</td>
<td>51</td>
</tr>
<tr>
<td>SBAS1IARC</td>
<td>54</td>
</tr>
<tr>
<td>Block-centered flow package modules</td>
<td>56</td>
</tr>
<tr>
<td>BCF1RPARC</td>
<td>56</td>
</tr>
<tr>
<td>BCF1BDARC</td>
<td>60</td>
</tr>
<tr>
<td>SBCF1FARC</td>
<td>63</td>
</tr>
<tr>
<td>SBCF1BARC</td>
<td>67</td>
</tr>
<tr>
<td>River package modules</td>
<td>70</td>
</tr>
<tr>
<td>RIV1RPARC</td>
<td>70</td>
</tr>
<tr>
<td>RIV1BDARC</td>
<td>74</td>
</tr>
<tr>
<td>Recharge package modules</td>
<td>77</td>
</tr>
<tr>
<td>RCH1RPARC</td>
<td>77</td>
</tr>
<tr>
<td>RCH1BDARC</td>
<td>78</td>
</tr>
<tr>
<td>Well package modules</td>
<td>82</td>
</tr>
<tr>
<td>WEL1RPARC</td>
<td>82</td>
</tr>
<tr>
<td>WEL1BDARC</td>
<td>85</td>
</tr>
<tr>
<td>Drain package modules</td>
<td>88</td>
</tr>
<tr>
<td>DRN1RPARC</td>
<td>88</td>
</tr>
<tr>
<td>DRN1BDARC</td>
<td>91</td>
</tr>
</tbody>
</table>
## ILLUSTRATIONS

Figure 1-30.--Flowcharts showing only modified program elements for:

1. BAS1RPARC module ................................................. .41
2. BAS1OCARC module ................................................ .44
3. BAS1OTARC module ................................................ .47
4. SBAS1DARC module ................................................ .49
5. SBAS1HARC module ................................................ .52
6. SBAS1IARC module ................................................ .54
7. BCF1RPARC module ................................................ .57
8. BCF1BDARC module ................................................ .60
9. SBCF1FARC module ................................................ .63
10. SBCF1BARC module ................................................ .67
11. RIV1RPARC module ................................................ .71
12. RIV1BDARC module ................................................ .74
13. RCH1RPARC module ................................................ .77
14. RCH1BDARC module ................................................ .79
15. WEL1RPARC module ................................................ .82
16. WEL1BDARC module ................................................ .85
17. DRN1RPARC module ................................................ .88
18. DRN1BDARC module ................................................ .91
19. EVT1RPARC module ................................................ .94

<table>
<thead>
<tr>
<th>CONTENTS--Continued</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td>Evapotranspiration package modules</td>
</tr>
<tr>
<td>EVT1RPARC</td>
</tr>
<tr>
<td>EVT1BDARC</td>
</tr>
<tr>
<td>General-head boundary package modules</td>
</tr>
<tr>
<td>GHBI1RPARC</td>
</tr>
<tr>
<td>GHBI1BDARC</td>
</tr>
<tr>
<td>Streamflow-routing package modules</td>
</tr>
<tr>
<td>STR1RPARC</td>
</tr>
<tr>
<td>STR1BDARC</td>
</tr>
<tr>
<td>STR1SRARC</td>
</tr>
<tr>
<td>Utility modules</td>
</tr>
<tr>
<td>U2DRELARC</td>
</tr>
<tr>
<td>U1DRELARC</td>
</tr>
<tr>
<td>U2DINTARC</td>
</tr>
<tr>
<td>U2DBDARC</td>
</tr>
<tr>
<td>U2DBSRARC</td>
</tr>
<tr>
<td>INFO utility modules</td>
</tr>
<tr>
<td>File utility module</td>
</tr>
<tr>
<td>References cited</td>
</tr>
<tr>
<td>Appendix A</td>
</tr>
<tr>
<td>Sample problem</td>
</tr>
<tr>
<td>Opening data and control files for sample problem</td>
</tr>
<tr>
<td>Input data and control for sample problem</td>
</tr>
<tr>
<td>Output data for sample problem</td>
</tr>
<tr>
<td>Output control for sample problem</td>
</tr>
<tr>
<td>Results for sample problem</td>
</tr>
</tbody>
</table>

## CONTENTS--Continued
<table>
<thead>
<tr>
<th>Illustration</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>20.</td>
<td>EVT1BDARC module</td>
<td>96</td>
</tr>
<tr>
<td>21.</td>
<td>GHB1RPARC module</td>
<td>99</td>
</tr>
<tr>
<td>22.</td>
<td>GHB1BDARC module</td>
<td>102</td>
</tr>
<tr>
<td>23a.</td>
<td>STR1RPARC module</td>
<td>105</td>
</tr>
<tr>
<td>23b.</td>
<td>STR1RPARC module</td>
<td>106</td>
</tr>
<tr>
<td>24.</td>
<td>STR1BDARC module</td>
<td>112</td>
</tr>
<tr>
<td>25.</td>
<td>STR1SRARC module</td>
<td>117</td>
</tr>
<tr>
<td>26.</td>
<td>U2DRELARC module</td>
<td>122</td>
</tr>
<tr>
<td>27.</td>
<td>U1DRELARC module</td>
<td>126</td>
</tr>
<tr>
<td>28.</td>
<td>U2DINTARC module</td>
<td>130</td>
</tr>
<tr>
<td>29.</td>
<td>UBUDSVARC module</td>
<td>135</td>
</tr>
<tr>
<td>30.</td>
<td>ULASAVARC module</td>
<td>137</td>
</tr>
</tbody>
</table>
MODIFICATIONS OF THE U.S. GEOLOGICAL SURVEY MODULAR, FINITE-Difference, GROUND-
WATER FLOW MODEL TO READ AND WRITE GEOGRAPHIC INFORMATION SYSTEM FILES

By Leonard L. Orzol and Timothy S. McGrath

ABSTRACT

This report documents modifications to the U.S. Geological Survey modular, three-dimensional, finite-difference, ground-water flow model, commonly called MODFLOW, so that it can read and write files used by a geographic information system (GIS). The modified model program is called MODFLOW ARC.

Simulation programs such as MODFLOW generally require large amounts of input data and produce large amounts of output data. Viewing data graphically, generating head contours, and creating or editing model data arrays such as hydraulic conductivity are examples of tasks that currently are performed either by the use of independent software packages or by tedious manual editing, manipulating, and transferring data. Programs such as GIS programs are commonly used to facilitate preparation of the model input data and analyze model output data; however, auxiliary programs are frequently required to translate data between programs. Data translations are required when different programs use different data formats. Thus, the user might use GIS techniques to create model input data, run a translation program to convert input data into a format compatible with the ground-water flow model, run the model, run a translation program to convert the model output into the correct format for GIS, and use GIS to display and analyze this output. MODFLOW ARC, avoids the two translation steps and transfers data directly to and from the ground-water-flow model.

This report documents the design and use of MODFLOW ARC and includes instructions for data input/output of the Basic, Block-centered flow, River, Recharge, Well, Drain, Evapotranspiration, General-head boundary, and Streamflow-routing packages. The modification to MODFLOW and the Streamflow-Routing package was minimized. Flow charts and computer-program code describe the modifications to the original computer codes for each of these packages. Appendix A contains a discussion on the operation of MODFLOW ARC using a sample problem.

INTRODUCTION

The increased use and importance of ground water as a source of water supply has led to the need for understanding and managing ground-water resources. A three-dimensional, finite-difference ground-water-flow model, developed by McDonald and Harbaugh (1988) and referred to as "MODFLOW", is used extensively by the U.S. Geological Survey (USGS) and others to simulate ground-water flow. Requirements for constructing and operating the model include compiling and manipulating large and unwieldy data sets. The graphic and analytical capabilities of a geographic information system (GIS) facilitate manipulating, editing, viewing data graphically, or generating contours for large input and output array data sets. ARC/INFO1 software is used in numerous USGS offices and in many other agencies to manage these large spatial and relational data sets.

1ARC/INFO is a registered trademark of Environment Systems Research Institute. The use of trade or product names in this report is for identification purposes only and does not constitute endorsement by the U.S. Geological Survey.
ARC/INFO software can facilitate the initial preparation of data arrays for a ground-water-flow model. The initial data are frequently derived from a collection of non-girded observations such as well pumpage or hydraulic conductivity. ARC/INFO can be used to process the data into the girded form required for input to the ground-water-flow model. The data can be graphically displayed if desired at any stage of the process.

ARC/INFO software also can be useful for manipulating data during model calibration. The user can graphically display the girded data sets, interactively select areas that need correction, and edit these values. Using this technique, sensitivity analysis of a ground-water-flow model becomes less unwieldy. The results of various simulations can be displayed, and differences can be analyzed.

A two-step approach commonly is required to use data stored in ARC/INFO datafiles in a ground-water-flow model. In the first step, data are rewritten from the binary format used by ARC/INFO software to the American Standard Code for Information Interchange (ASCII) format needed for input to the ground-water-flow model. After running the ground-water flow model, the second step consists of rewriting the output data from the simulation in a format acceptable for input into ARC/INFO datafiles. MODFLOW was modified so that it can directly read ARC/INFO data. The modified program, called MODFLOWARC, was developed using FORTRAN subroutines available within the ARC/INFO software to take data stored in the binary format of ARC/INFO datafiles and transfer these data into the ground-water flow model. MODFLOWARC performs no data modifications or transformations.

Purpose and Scope

The purpose of this report is to serve as a user’s manual for operating MODFLOWARC, an enhanced version of MODFLOW that can transfer data directly between ARC/INFO software and the ground-water-flow model. MODFLOWARC is designed for individuals who have a working knowledge of the flow model developed by McDonald and Harbaugh (1988). This knowledge of the ground-water flow model must include the input data used by the program code, and the initial control records within ASCII files that are used to direct the operation of individual packages (grouped subroutines or modules) within the flow model. Each package deals with a specific aspect of the simulation, such as flow to or from rivers.

Organization and Nomenclature

The documentation of this report includes: (1) a description of the overall design of the extensions to MODFLOW contained in MODFLOWARC; (2) discussions of input instructions and modifications for the different packages of the ground-water-flow model by McDonald and Harbaugh (1988) and the Streamflow-Routing package added by Prudic (1988); (3) a discussion of the installation and operation on different computer systems; (4) an appendix describing a test problem using MODFLOWARC.

The names of variables, modules, and submodules used to explain the operations of MODFLOWARC were derived from names used in the original ground-water-flow model (McDonald and Harbaugh, 1988). As an example, input data describing IBOUND arrays are read by the BAS1RP module within the Basic package. The equivalent module within the MODFLOWARC is called BAS1RPARC and the variable name searched for in the ARC/INFO datafiles is IBOUND. Code that has been added to the original FORTRAN program code of the model will be referred to as an “arc-section.”

MODFLOWARC DESIGN

The design of MODFLOWARC parallels the design of the ground-water flow model program MODFLOW. The suffix, ARC, was added to the new and modified modules. Modifications to the code of MODFLOW were minimized.

MODFLOWARC uses FORTRAN subroutines provided as part of ARC/INFO. This set of subroutines, collectively described as a low level, machine independent module (Environment Systems
Research Institute, Inc., 1989), opens, closes, sorts, and performs various other functions on ARC/INFO files. A significant advantage of the subroutines is that the item names of data stored in ARC/INFO files (the names for the arrays of values of input/output data) and their individual formats are easily obtainable. The item names adopted within the new modules or submodules of MODFLOWARC match the variable names used by the flow model. To read ARC/INFO data, MODFLOWARC opens an ARC/INFO file and the item names are retrieved and verified for the matching ground-water, flow-model, program-variable name. The matched item’s format is also retrieved. In this manner the user does not need to supply the format. When writing ARC/INFO data, MODFLOWARC specifies formats. A limitation is that ARC/INFO software does not support scientific notation.

INSTALLATION AND OPERATION OF MODFLOWARC

MODFLOWARC should work on any computer that has ARC/INFO software (version 5.01). It has been tested on a Prime computer and a Data General Aviion computer. The MODFLOWARC modules use FORTRAN library subroutines within ARC/INFO software (ISP module). These library subroutines are available from ESRI (Environment Systems Research Institute) and must be installed on the system along with the necessary software to compile and load FORTRAN programs. The ARC/INFO libraries are usually found on a Prime computer system under an ARC50>LIB directory or on a computer using UNIX under an /ARC50/LIB/ directory.

Installation involves two steps: (1) compiling the modules of MODFLOWARC and (2) linking the compiled modules of MODFLOWARC with the computer system libraries, if needed, and ARC/INFO libraries to produce an executable program.

Operation of MODFLOWARC follows a three-step process: (1) user activates the ARC/INFO software by issuing the "arc" command; (2) user runs the AML program MODFLOWARC.AML within the ARC/INFO software by next issuing the command "&r modflowarc filename_argument" (filename_argument is discussed below); and (3) the AML program, MODFLOWARC.AML, passes program control to MODFLOWARC.F77. The latter two steps without user commands.

The user supplies a filename_argument when running MODFLOWARC.AML at the "arc" prompt. The filename_argument consists of an ASCII file containing unit numbers and filenames of the files that must be opened for a model simulation. Each line in this filename_argument file is a unit number followed by the associated filename in free format with the filename surrounded by single quotes such as ‘modflow.list’. The user builds this file in a definite order. The first record consists of the unit number and filename for the Basic (BAS) package input. The last record is the unit number and filename where all printer output is directed. All remaining unit numbers and filenames are included in any order between these two entries. If unformatted files are used for a model simulation, the user specifies the unit numbers of these files as negative. For an example, if the starting heads were recorded in an unformatted file in a previous simulation, then the user sets the unit number for the file containing the starting heads data as negative for the next simulation.

INSTRUCTIONS FOR INPUT AND OUTPUT DATA FOR MODFLOWARC

During the data input phase, MODFLOWARC reads array control records similar to the original control records of the ground-water flow model, except that an additional variable follows: either the ITMP variable for the Drain, Well, River, General-head boundary packages; or the print variable IPRN for Basic, Block-centered, Recharge, and Evapotranspiration packages; or the print variable IPTFLG for Streamflow-routing package (Prudic, 1988). This additional variable is a complete path to the file within the ARC/INFO database containing input data to be read. This specified path is dependent on the computer in use. If this additional input variable is blank, then the module operates exactly in the manner of the “original” module.

The input data within each ARC/INFO file is organized by items and each item has a user defined data format. MODFLOWARC uses the names of these items to locate and read input arrays. The names of
the items are preset within MODFLOWARC and are described in the following input examples for each package of the ground-water flow model. For example, the Basic (BAS) package of MODFLOWARC reads in the IBOUND array values using the root name IBOUND_ and attaches a suffix representing the layer number such as IBOUND_1 for the IBOUND array values for layer 1. The user defines the data format within each ARC/INFO file for input arrays as either integer, floating point, or numeric. MODFLOWARC uses the complete path supplied by the user to locate the directory and file containing the input data; then retrieves the array values using the preset item names. One record is needed for each model cell within the specified ARC/INFO file for two and three-dimensional arrays, however, for one-dimensional arrays one record is needed for each model layer. For example, the three-dimensional IBOUND array will need a record for each cell and the one-dimensional TRPY array will need a record for each layer. In the example inputs for each package that follow this section of the report, the input item names and record structure are described for the one-, two-, and three-dimensional arrays.

The input for the Streamflow-routing package (Prudic, 1988) is slightly different than the input to the packages in the original model. The original Streamflow-routing package reads input data for streams, diversions, and tributaries from one file. ARC/INFO datafiles are not easily constructed to store these data in such a form. MODFLOWARC needs these data separated into one ARC/INFO datafile for stream data, one for diversion data, if needed, and one for tributaries data, if needed. The user must add a suffix to the names of these ARC/INFO datafiles containing these data, such as STREAMS_1 for stream data. The suffix contains the stress period during which these stream data are read. The under-score preceding the stress period is mandatory. MODFLOWARC reads the specified path to the stream data (specified path after the print flag IPTFLG) and uses the suffix to locate and read the ARC/INFO datafiles containing diversion and tributary data. The user formulates the filenames for the ARC/INFO datafiles containing diversion and tributary data by adding this suffix to the root TRIB_ for diversion data and DIV_ for tributary data. For example, stream data is stored in an ARC/INFO datafile, STREAMS_1, while the tributary and diversion data are stored in two ARC/INFO datafiles called TRIB_1 and DIV_1 (where 1 represents the stress period 1).

When recording output data from MODFLOWARC, the user sets package record/print flags such as IWELCB for the Well (WEL) package, head and drawdown output flag, IHEDFL, and cell_by_cell flow-term flag, ICBCFL, like the original output operations of MODFLOW and the Streamflow-routing package. To record cell_by_cell budget data in unformatted files or ARC/INFO files, the user sets the individual package record/print flag such as IWELCB for the Well (WEL) package greater than 0. In the original model, these record/print flags are set to unit numbers for recording output data and to less than 0 to print output data. Also, the user sets ICBCFL to greater than 0 to record output data to unformatted files and to less than 0 to record output data to ARC/INFO files. To record heads and drawdown output data, the user sets head and drawdown output flag, IHEDFL, to greater than 0 to record output data to unformatted files and to less than 0 to record output data to ARC/INFO files.

However, the user must supply an additional item, OUTPATH, to the control record for each of the output control modules BAS1RPARC and BAS1OCARC. BAS1RPARC module reads the print formats and unit numbers for the head and drawdown output data: IHEDFM, IHDNFM, IHEDUN, and IHDNUN. This additional variable is a path name to the directory where output head and drawdown data from the ground-water flow model are recorded. If IHDDFL is set to less than 0 for head and drawdown output arrays, the OUTPATH variable immediately follows after the unit number for drawdown, IHDNUN. BAS1OCARC module reads the head/drawdown output code, the output flag for head/drawdown data, budget print flag, and the cell_by_cell flow-term flag: INCODE, IHHDFL, IBUDFL, and ICBCFL. If the cell-by-cell flow-term flag, ICBCFL, is set to less than 0, then the OUTPATH variable immediately follows the ICBCFL. This additional variable is a path name to the directory where output budget data from the ground-water-flow model are recorded.

Specific Arc/Info output file names for the individual packages within the ground-water-flow model are created by the MODFLOWARC modules. MODFLOWARC formulates filenames from a root name derived from the package name such as WELBUD is from the Well (WEL) package and then appends a
suffix. This suffix is composed of the stress period and the time step within the stress period when data was recorded. The stress period and time step are appended together in this order and are separated by underscores. For example, the root name WELBUD is used for well output budget data and is appended by stress period 1 and time step 2 and output data is recorded in an ARC/INFO file named WELBUD_1_2.

With MODFLOWARC the output for the Streamflow-routing package (Prudic, 1988) has been enhanced; however, the user can still record output data for stream outflow and leakage separately. To record stream outflow and leakage data, the user sets stream record/print flags, ISTCB1 and ISTCB2, to greater than 0. If the cell-by-cell flow-term flag, ICBCFL, is greater than 0, stream output data is recorded in two unformatted files, and if the ICBCFL is less than 0, stream output data is recorded in two separate ARC/INFO files, LKGBUD_1_2 (stream leakage) and FLOBUD_1_2 (stream outflow) (stress period 1 and time step 2). These data are recorded for each cell and not by stream segment or reach within each segment (see Prudic (1988) for discussion of stream segment and reaches). However, input data for the streamflow-routing package is grouped by stream segment and reach. During model calibration, the user may wish to calibrate streamflow output data by stream segment and reach. An additional module is incorporated into MODFLOWARC to record streamflow output data by stream segment and reach in an ARC/INFO file. For an example, streamflow output data by stream segment and reach is recorded to ARC/INFO file, STRSEG_1_2, for stress period 1 and time step 2. This additional module is called STR1SRARC. The user sets streamflow print flag, IPTFLG, to less than 0 and cell_by_cell flow-term flag, ICBCFL, to less than 0 to record streamflow output data by stream segment and reach to an ARC/INFO file.

The following sections describe how to modify the sample inputs for each package to activate the modules of MODFLOWARC. The modifications are in bold print. The original variable names and format are maintained for each package. If no changes occur, the definitions for each module, submodule, and code variables are not repeated. New program variables or options are defined.

### Basic Package Input

Input is read from the fortran unit specified by the user in an ASCII file that is read by the main program, MODFLOWARC. The user specifies this ASCII file as an argument when issuing the command 

```
&modflowarc filename_argument
```

The following information is sample input to the BAS1RPARC module of MODFLOWARC and excludes output control records. Modification to MODFLOW are shown in bold type.

**FOR EACH SIMULATION**

**BAS1DF**

1. Data HEADNG(32) Format 20A4
2. Data HEADNG (continued) Format 12A4
3. Data NLAY NROW NCOL NPER ITMUNI Format I10 I10 I10 I10 I10
4. Data IUINT(24) Format 24I34

**BAS1AL**

5. Data IAPART ISTRT

---

5
<table>
<thead>
<tr>
<th>Format</th>
<th>I10</th>
<th>I10</th>
</tr>
</thead>
</table>

**BAS1RPARC**

6. **Data** 
Module **U2DINTARC**  
(Arc/Info item root name is **IBOUND_**xx; xx are for appropriate layer and one record for each cell)

7. **Data**  
Format **F10.0**

6. **Data**  
Module **U2DRELARC**  
(Arc/Info item root name is **SHEAD_**xx; xx are for appropriate layer and one record for each cell)

**BAS1ST**

8. **Data**  
Format **F10.0**  
**I10**  
**F10.0**
DATA ITEM | EXPLANATION | INPUT RECORD
--- | --- | ---
1 | {HEADING} | SAMPLE—3 LAYERS, 15 ROWS, 15 COLUMNS; STEADY STATE; CONSTANT HEADS COLUMN 1, LAYERS 1 AND 2; RECHARGE, WELLS AND DRAINS
2 | {HEADING} CONTINUED | NLAY, NROW, NCOL, NPER, ITMUNI | 3 15 15 1
3 | [IUNIT TABLE] | IAPART, ISTRRT | 11 12 13 00 00 00 18 19 00 00 20
4 | CONTROL RECORD FOR IBOUND ARRAY LAYER 1 | 5 1 (15I3) | 3GWINF>SAMPLE>INFO!ARC!IBOUND
5 | CONTROL RECORD FOR IBOUND ARRAY LAYER 2 | 5 1 (15I3) | 3GWINF>SAMPLE>INFO!ARC!IBOUND
6 | CONTROL RECORD FOR IBOUND ARRAY LAYER 3 | 0 1 | IBOUND-3
7 | [HNOFLO] | 999.99 | 3GWINF>SAMPLE>INFO!ARC!HEADS
8 | CONTROL RECORD FOR STARTING HEAD ARRAY LAYER 1 | 5 1 (15I3) | HEAD-2
9 | CONTROL RECORD FOR STARTING HEAD ARRAY LAYER 2 | 0 0.0 | HEAD-3
10 | CONTROL RECORD FOR STARTING HEAD ARRAY LAYER 3 | 0 0.0 | HEAD-3
11 | STRESS PERIOD 1—[PERLEN, NSTP, TSMULT] | 86400. 1 1.

The boundary arrays for layers 1 and 2 are read from items IBOUND_1 and IBOUND_2 in Arc/Info file GWINF>SAMPLE>INFO!ARC!IBOUND. The character string identifying the path to IBOUND must start in column 51 of that record. If the user adds comments to these control records, the path must be separated from the comment with a blank space. The starting heads for layer 1 are read from item SHEAD_1 in Arc/Info file HEADS.
Basic Package Output

Output control is read from the unit number specified by IUNIT(12). The following information is sample input set for output control to the BAS1RPARC and BAS1OCARC modules of MODFLOWARC. Modification to MODFLOW are shown in bold type.

FOR EACH SIMULATION

BAS1RPARC

1. Data IHEDFM IDDNFM IHEDUN IDDNUN OUTPATH
   Format I10 I10 I10 I10 A80

FOR EACH TIME STEP

BAS1OCARC

2. Data INCODE IHDDFL IBUDFL ICBCFL OUTPATH
   Format I10 I10 I10 I10 A80

3. Data Hdpr Ddpr Hdsv Ddsv
   Format I10 I10 I10 I10

IHDDFL— is a head and drawdown output flag.
   If IHDDFL = 0, neither heads nor drawdowns are printed or saved on disk.
   If IHDDFL > 0, heads and drawdowns are printed or saved according to the flags for each
   layer specified in input item 3 (ASCII option).
   If IHDDFL < 0, heads and drawdowns are printed or saved according to the flags for
   each layer specified in input item 3 (ARC/INFO option).

ICBCFL— is a cell-by-cell flow-term flag.
   If ICBCFL = 0, cell-by-cell flow terms are not printed or saved on disk.
   If ICBCFL > 0, cell-by-cell flow terms are printed or saved according to the flags for each
   layer specified in input item 3 (ASCII option).
   If ICBCFL < 0, cell-by-cell flow terms are printed or saved according to the flags for each
   layer specified in input item 3 (ARC/INFO option).

OUTPATH— is a directory path to the ARC/INFO subdirectory where the output values for head,
   drawdown, and cell-by-cell flow terms for the individual packages, if activated, of the ground-water
   flow model are recorded during each model simulation.
The user specifies a path (minus the file name of the ARC/INFO file) where output data are recorded (see above data items 1 or 2). The character string (A80) identifying this path must start in column 41 of that record. If a user omits this path, then the original program flow is maintained. The file name is not needed because these names are created by MODFLOWARC modules. Each file name includes the stress period and time step as part of the name such as DDNBUD_1_1, which infers that the drawdown values were called for and output was specified for stress period 1 and for time step 1. The array values for head, drawdown, and cell-by-cell flow terms are sent to ARC/INFO files if the flags IHDDFL and ICBCFL are negative and each package record/input flag such as IWELCB for the Well (WEL) package are greater that zero. The array values are recorded in these ARC/INFO files under the item name LAYER_xx created by MODFLOWARC modules (xx represents the appropriate layer).

<table>
<thead>
<tr>
<th>DATA ITEM</th>
<th>EXPLANATION</th>
<th>INPUT RECORD</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 [IHEDFM, IDDNFM, IHEDUN, IDDNUN]</td>
<td>4 8 76</td>
<td>77GWINF&gt;SAMPLE&gt;INFO!ARC!</td>
</tr>
<tr>
<td>2 TIME STEP 1--{INCODE, IHDDFL, IBUDFL, ICBCFL}</td>
<td>1</td>
<td>-1 1</td>
</tr>
<tr>
<td>3 LAYER 1--{HDPR, DDPR, HDSV, DDSV}</td>
<td>1</td>
<td>1 1 1</td>
</tr>
<tr>
<td>3 LAYER 2--{HDPR, DDPR, HDSV, DDSV}</td>
<td>1</td>
<td>1 1 1</td>
</tr>
<tr>
<td>3 LAYER 3--{HDPR, DDPR, HDSV, DDSV}</td>
<td>1</td>
<td>1 1 1</td>
</tr>
</tbody>
</table>
Block-centered Flow Package Input

Input control for the Block-centered flow Package is read from IUNIT(1). The following information is sample input to the BCF1RPARC module of MODFLOWARC and excludes output control records. Modification to MODFLOW are shown in bold type.

FOR EACH SIMULATION

BCF1AL

1. Data ISS IBCFCB
   Format I10 I10

2. Data LAYCON(NLAY)
   Format 40I2

BCF1RPARC

3. Data TRPY(NLAY)
   Module U1DRELARC
   (Arc/Info item root name is TRPY; one record for each model layer)

4. Data DELR(NCOL)
   Module U1DRELARC
   (Arc/Info item root name is DELR; one record for every column)

5. Data DELC(NROW)
   Module U1DRELARC
   (Arc/Info item root name is DELC; one record for every row)

IF THE SIMULATION IS TRANSIENT

6. Data SF1(NCOL,NROW)
   Module U2DRELARC
   (Arc/Info item root name is SF1_xx; xx are for appropriate layer and one record for each cell)

IF THE LAYER TYPE CODE (LAYCON) IS ZERO OR TWO

7. Data TRAN(NCOL,NROW)
   Module U2DRELARC
   (Arc/Info item root name is TRAN_xx; xx are for appropriate layer and one record for each cell)

IF THE LAYER TYPE CODE (LAYCON) IS ONE OR THREE

8. Data HY(NCOL,NROW)
   Module U2DRELARC
   (Arc/Info item root name is HY_xx; xx are for appropriate layer and one record for each cell)
9. Data Module BOT(NCOL,NROW) U2DRELARC
   (Arc/Info item root name is BOT_{xx}; xx are for appropriate layer and one record for each cell)

IF THIS IS NOT THE BOTTOM LAYER

10. Data Module VCONT(NCOL,NROW) U2DRELARC
    (Arc/Info item root name is VCONT_{xx}; xx are for appropriate layer and one record for each cell)

IF THE SIMULATION IS TRANSIENT AND THE LAYER TYPE CODE (LAYCON) IS TWO OR THREE

11. Data Module SF2(NCOL,NROW) U2DRELARC
    (Arc/Info item root name is SF2_{xx}; xx are for appropriate layer and one record for each cell)

IF THE LAYER TYPE CODE (LAYCON) IS TWO OR THREE

12. Data Module TOP(NCOL,NROW) U2DRELARC
    (Arc/Info item root name is TOP_{xx}; xx are for appropriate layer and one record for each cell)
The array values for DELR (grid spacing in the row direction), DELC (grid spacing in the column direction), TRPY (ratio of transmissivity along columns to transmissivity along rows for each layer), SF1 (primary storage coefficient), TRAN (transmissivity along rows), HY (hydraulic conductivity along rows), BOT (elevation of the aquifer bottom), VCONT (vertical conductivity), SF2 (secondary storage coefficient), and TOP (elevation of the aquifer) are read from Arc/Info files. The character string identifying this path must start in column 51 of the control record. If the user adds comments to these control records, the path must be separated from the comment with a blank space.
River Package Input

Input is read from unit specified in IUNIT(4). The following information is sample input to the RIV1RPARC module of MODFLOWARC and excludes output control records. Modification to MODFLOW are shown in bold type.

FOR EACH SIMULATION

RIV1AL

1. Data MXRIVR IRIVCB
   Format I10 I10

FOR EACH STRESS PERIOD

RIV1RPARC

2. Data ITMP RIVPATH
   Format I10 A80

3. Data LAYER ROW COLUMN STAGE COND RBOT
   Format User specified when ARC/INFO file was created
   (Arc/Info item root names are LAYER, ROW, COLUMN, STAGE, COND, and RBOT)

RIVPATH is a complete path to an ARC/INFO file containing values for layer, row, column, stage, cond, and rbot variables, respectively.
<table>
<thead>
<tr>
<th>DATA ITEM</th>
<th>EXPLANATION</th>
<th>INPUT RECORD</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 {MXRIVER, IRIVCB}</td>
<td>________________________________</td>
<td>5 1</td>
</tr>
<tr>
<td>2 {ITMP, RIVPATH}</td>
<td>________________________________</td>
<td>5GWINFO&gt;SAMPLE&gt;INFO!ARC!RIVERS</td>
</tr>
</tbody>
</table>

The River package arrays are read from the Arc/INFO file GWINFO>SAMPLE>INFO!ARC!RIVERS. The character string identifying this path must start in column 11 of the control record. If the user adds comments to these control records, the path must be separated from the comment with a blank space.
Recharge Package Input

Input is read from unit specified in IUNIT(8). The following information is sample input to the RCH1RPARC module of MODFLOWARC and excludes output control records. Modification to MODFLOW are shown in bold type.

FOR EACH SIMULATION

RCH1AL

1. Data NRCHOP IRCHCB
   Format I10 I10

FOR EACH STRESS PERIOD

RCH1RPARC

2. Data INRECH INIRCH
   Format I10 I10

3. Data RECH(NCOL,NROW)
   Module U2DRELARC
   (Arc/Info item root name is RECH; one record for each cell)

IF THE RECHARGE OPTION IS EQUAL TO 2

4. Data IRCH(NCOL,NROW)
   Module U2DIINTARC
   (Arc/Info item root name is IRCH; and one record for each cell)
<table>
<thead>
<tr>
<th>ITEM</th>
<th>EXPLANATION</th>
<th>INPUT RECORD</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>{NRCHOP, IRCHCB}</td>
<td>1 99</td>
</tr>
<tr>
<td>2</td>
<td>{INRECH, INIRCH}</td>
<td>1 0</td>
</tr>
<tr>
<td>3</td>
<td>CONTROL RECORD FOR RECHARGE ARRAY</td>
<td>5 3.E-8(15I3) 3GWINF&gt;SAMPLE&gt;INFO!ARC!RECH</td>
</tr>
</tbody>
</table>

The recharge array is read from the Arc/Info file GWINF>SAMPLE>INFO!ARC!RECH. The character string identifying this path must start in column 51 of the control record. If the user adds comments to these control records, the path must be separated from the comment with a blank space.
Well Package Input

Input is read from unit specified in IUNIT(2). The following information is sample input to the WEL1RPARC module of MODFLOWARC and excludes output control records. Modification to MODFLOW are shown in bold type.

FOR EACH SIMULATION

WEL1AL

1. Data MXWELL IWELCB
   Format I10 I10

FOR EACH STRESS PERIOD

WEL1RPARC

2. Data ITMP WELPATH
   Format I10 A80

3. Data LAYER ROW COLUMN Q
   Format User specified when ARC/INFO file was created
   (Arc/Info item root names are LAYER, ROW, COLUMN, and Q)

WELPATH is a complete path to an ARC/INFO file containing array values for rates of recharge to or discharge from pumping wells for the Well package for each stress period.
<table>
<thead>
<tr>
<th>DATA ITEM</th>
<th>EXPLANATION</th>
<th>INPUT RECORD</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 {MXWELL, IWELCB}</td>
<td>________________________________</td>
<td>15 77</td>
</tr>
<tr>
<td>2 {ITMP, WELPATH}</td>
<td>________________________________</td>
<td>15GWINF&gt;SAMPLE&gt;INFO!ARC!WELLS</td>
</tr>
</tbody>
</table>

The array values are read from the Arc/Info file GWINF>SAMPLE>INFO!ARC!WELLS. The character string identifying this path must start in column 11 of the control record. If the user adds comments to these control records, the path must be separated from the comment with a blank space.
Drain Package Input

Input is read from unit specified in IUNIT(3). The following information is sample input to the DRN1RPARC module of MODFLOWARC and excludes output control records. Modification to MODFLOW are shown in bold type.

FOR EACH SIMULATION

DRN1AL

1. Data MXDRN IDRNCB
   Format I10 I10

FOR EACH STRESS PERIOD

DRN1RPARC

2. Data ITMP DRNPATH
   Format I10 A80

3. Data LAYER ROW COLUMN ELEVATION COND
   Format User specified when ARC/INFO file was created
   (Arc/Info item root names are LAYER, ROW, COLUMN, ELEVATION, and COND)

   DRNPATH is a complete path to an ARC/INFO file containing values for layer, row, column, elevation, and cond variables, respectively.
<table>
<thead>
<tr>
<th>DATA ITEM</th>
<th>EXPLANATION</th>
<th>INPUT RECORD</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 MXDRN, IDRNCB</td>
<td>_________________________________</td>
<td>9 77</td>
</tr>
<tr>
<td>2 ITMP, DRNPATH</td>
<td>_________________________________</td>
<td>9GWINFSAMPLEINFOARC!DRAINS</td>
</tr>
</tbody>
</table>

Each Drain array is read from the Arc/Info file GWINFSAMPLE>INFO!ARC!DRAINS. The character string identifying this path must start in column 11 of the control record. If the user adds comments to these control records, the path must be separated from the comment with a blank space.
Evapotranspiration Package Input

Input is read from unit specified in IUNIT(5). The following information is sample input to the EVT1RPARC module of MODFLOWARC and excludes output control records. Modification to MODFLOW are shown in bold type.

FOR EACH SIMULATION

EVT1AL

1. Data NEVTOP IEVTCB
   Format I10 I10

FOR EACH STRESS PERIOD

EVT1RPARC

2. Data INSURF INEVTR INEXDP INIEVT
   Format I10 I10 I10 I10

3. Data SURF(NCOL,NROW)
   Module U2DRELARC
   (Arc/Info item name is SURF; one record for each cell)

4. Data EVTR(NCOL,NROW)
   Module U2DRELARC
   (Arc/Info item name is EVTR; one record for each cell)

5. Data EXDP(NCOL,NROW)
   Module U2DRELARC
   (Arc/Info item name is EXDP; one record for each cell)

IF THE ET OPTION IS EQUAL TO TWO

6. Data IEVT(NCOL,NROW)
   Module U2DINTARC
   (Arc/Info item name is IEVT; one record for each cell)
<table>
<thead>
<tr>
<th>DATA ITEM</th>
<th>EXPLANATION</th>
<th>INPUT RECORD</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 {NEVTOP, IEVTCB}</td>
<td></td>
<td>9  77</td>
</tr>
<tr>
<td>2 {INSURF, INEVTR, INEXDP, INIEVT}</td>
<td></td>
<td>1  1  1</td>
</tr>
<tr>
<td>3 CONTROL RECORD FOR ET SURFACE ARRAY</td>
<td></td>
<td>9  1.0(10F5.0) 4GWINF&gt;SAMPLE&gt;INFO!ARC!ET</td>
</tr>
<tr>
<td>4 CONTROL RECORD FOR ET RATE ARRAY</td>
<td></td>
<td>9  1.0(10F4.0) 4GWINF&gt;SAMPLE&gt;INFO!ARC!ET</td>
</tr>
<tr>
<td>5 CONTROL RECORD FOR EXTINCTION DEPTH ARRAY</td>
<td></td>
<td>9  1.0(10F5.0) 4GWINF&gt;SAMPLE&gt;INFO!ARC!ET</td>
</tr>
</tbody>
</table>

The Evapotranspiration arrays, SURF (elevations of ET surfaces), EVTR (maximum ET rates), and EXDP (ET extinction depth), are read from the Arc/Info file GWINF>SAMPLE>INFO!ARC!ET. Evapotranspiration array, IEVT (layer indicator array), was omitted. The character string identifying this path must start in column 51 of the control record. If the user adds comments to these control records, the path must be separated from the comment with a blank space.
General-Head Boundary Package Input

Input is read from unit specified in IUNIT(2). The following information is sample input to the GHB1RPARC module of MODFLOWARC and excludes output control records. Modification to MODFLOW are shown in bold type.

FOR EACH SIMULATION

GHB1AL

1. Data MXBND IGHBCB
   Format I10 I10

FOR EACH STRESS PERIOD (reads from ARC/INFO file)

GHB1RPARC

2. Data ITMP GHBPATH
   Format I10 A80

3. Data LAYER ROW COLUMN BOUNDARYHEAD COND
   Format User specified when ARC/INFO file was created
   (Arc/Info item root names are LAYER, ROW, COLUMN, BOUNDARYHEAD, and COND)

GHBPATH is a complete path to an ARC/INFO file containing values for rates of flow to and from general-head boundaries for the General-Head Boundary package.
<table>
<thead>
<tr>
<th>DATA ITEM</th>
<th>EXPLANATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>{MXBND, IGHBCB}</td>
</tr>
<tr>
<td>2</td>
<td>{ITMP, GHBPATH}</td>
</tr>
</tbody>
</table>

The General-Head Boundary arrays, LAYER, ROW, COLUMN, BOUNDARYHEAD, and COND, are read from the Arc/Info file GWINFOSAMPLEINFO!ARC!BOUNDARY. The character string representing this path must start in column 11 of that line. If the user adds comments to these control records, the user must separate the path from the comment by inserting a blank space after the former.
Streamflow-Routing Package Input

Input is read from a user specified unit number of the control record from the Basic package (user specified). MODFLOWARC does not allow streamflow array values to be stored in both ARC/INFO files and ASCII files. A sample input data set follows and excludes output control records. Modifications are in bold type.

FOR EACH SIMULATION

STR1AL

1. Data MXSTRM NSS NTRIB NDIV ICALC CONST ISTCB1 ISTCB2
   Format I10 I10 I10 I10 I10 F10.0 I10 I10

FOR EACH STRESS PERIOD

STR1RPARC

2. Data ITMP IRDFLG IPTFLG STRPTH
   Format I10 I10 I10 A80
3. Data LAYER ROW COLUMN SEG REACH FLOW STAGE COND SBOT STOP
   Format User specified when ARC/INFO file was created
   (Arc/Info item names are LAYER, ROW, COLUMN, SEG, REACH, FLOW,
   STAGE, COND, SBOT, and STOP)
4. Data WIDTH SLOPE ROUGH
   Format User specified when ARC/INFO file was created
   (Arc/Info item names are WIDTH, SLOPE, and ROUGH)
5. Data ITRIB(1) ITRIB(2) ... ... ... ITRIB(NTRIB)
   Format User specified when ARC/INFO file was created
   (Arc/Info item root name is ITRIB_xx; for each appropriate tributary)
6. Data IUPSEG
   Format User specified when ARC/INFO file was created
   (Arc/Info item names are IUPSEG; for each appropriate segment)

IPTFLG--is a flag.

If IPTFLG = 0 or blank, prints the streamflow rates and leakage data. These data are not the budget arrays that are printed per cell, but the streamflow rates and leakage arrays for each reach that are printed per segment and per reach.

If IPTFLG > 0, heads and drawdowns are printed or saved according to the flags for each layer specified in input item 3 (ASCII option).

If IPTFLG < 0, records the streamflow rates and leakage data. These data are not the budget arrays that are record per cell, but the streamflow rates and leakage arrays for each reach that are record per segment and per reach. The new module STR1SRARC of MODFLOWARC records these data to an ARC/INFO file only (ARC/INFO option).

STRPTH is a complete path to an ARC/INFO file containing values for layer, row, column, seg, reach, flow, stage, cond, sbot, and stop variables, respectively (if width, slope, and rough are needed).
The control record for the STR1RPARC module contains three parameters: ITMP (a flag and counter for the stream data to be read), IRDFLG (flag for either suppress printing or printing of input stream data), and IPTFLG (flag for either suppress printing or printing of output stream results). In order to activate the "arc-section" code, an additional parameter must be included within the record after the IPTFLG parameter. The additional parameter is a path to the ARC/INFO file containing array values. The character string identifying this path must start in column 31 of that record. If comments are added to these control records a blank space must separate the path from the comment. If the user wants to record streamflow output data by stream segment and reach using the STR1SRARC module in MODFLOWARC into an ARC/INFO file, the user sets streamflow print flag, IPTFLG, to less than 0 and cell_by_cell flow-term flag, ICBCFL, to less than 0.
Utility Modules Input

The utility modules are called by Basic, Block-centered, Recharge, and Evapotranspiration packages during the data input phase. The control records for these utility modules contain four parameters: LOCAT (the location of the data-the fortran unit number from which values of the arrays will be read), CNSTNT or ICONST (the multiplier constant for the array values), FMTIN (the fortran format of the array values), and IPRN (the print format code for the array values). Modification to MODFLOW are shown in bold type.

FOR REAL ARRAY READER (U2DRELARC or U1DRELARC)

<table>
<thead>
<tr>
<th>Data</th>
<th>LOCAT</th>
<th>CNSTNT</th>
<th>FMTIN</th>
<th>IPRN</th>
<th>INFOPATH</th>
</tr>
</thead>
<tbody>
<tr>
<td>Format</td>
<td>I10</td>
<td>F10.0</td>
<td>5A4</td>
<td>I10</td>
<td>A80</td>
</tr>
</tbody>
</table>

FOR INTEGER ARRAY READER (U2DINTARC)

<table>
<thead>
<tr>
<th>Data</th>
<th>LOCAT</th>
<th>ICONST</th>
<th>FMTIN</th>
<th>IPRN</th>
<th>INFOPATH</th>
</tr>
</thead>
<tbody>
<tr>
<td>Format</td>
<td>I10</td>
<td>I10</td>
<td>5A4</td>
<td>I10</td>
<td>A80</td>
</tr>
</tbody>
</table>

INFOPATH is a complete path to an ARC/INFO file containing values for the arrays for the Basic, Block-centered, Recharge, and Evapotranspiration packages.

The utility modules UBUDSV and ULASAV were entirely rewritten and renamed to UBUDSVARC and ULASAVARC. These modules are activated by setting the cell-by-cell flow-term flag ICBCFL to -1. ICBCFL is set to 1 for unformatted output and set to 0 when the user wants no output.

FOR EACH SIMULATION

BAS1RPARC

1. Data | IHEDFM | IDDNFM | IHEDUN | IDDNUN | OUTPATH |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Format</td>
<td>I10</td>
<td>I10</td>
<td>I10</td>
<td>I10</td>
<td>A80</td>
</tr>
</tbody>
</table>

FOR EACH TIME STEP

BAS1OCARC

2. Data | INCODE | IHDDFL | IBUDFL | ICBCFL | OUTPATH |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Format</td>
<td>I10</td>
<td>I10</td>
<td>I10</td>
<td>I10</td>
<td>A80</td>
</tr>
</tbody>
</table>

IHDDFL is a head and drawdown output flag.
- If IHDDFL = 0, neither heads nor drawdowns are printed or saved on disk.
- If IHDDFL > 0, heads and drawdowns are printed or saved according to the flags for each layer specified in input item 3 (ASCII option).
- If IHDDFL < 0, heads and drawdowns are printed or saved according to the flags for each layer specified in input item 3 (ARC/INFO option).

ICBCFL is a cell-by-cell flow-term flag.
- If ICBCFL = 0, cell-by-cell flow terms are not printed or saved on disk.
- If ICBCFL > 0, cell-by-cell flow terms are printed or saved according to the flags for each layer specified in input item 3 (ASCII option).
- If ICBCFL < 0, cell-by-cell flow-terms are printed or saved according to the flags for each layer specified in input item 3 (ARC/INFO option).

OUTPATH is a path (minus the name of the ARC/INFO file) to the file where the output values for head, drawdown, cell-by-cell flow-term flag, and volumetric budget values for the individual packages, if
activated, of the flow model are written during each model run.

FOR EACH TIME STEP

BAS1OCARC

<table>
<thead>
<tr>
<th>Data</th>
<th>Hdpr</th>
<th>Ddpr</th>
<th>Hdsv</th>
<th>Ddsv</th>
</tr>
</thead>
<tbody>
<tr>
<td>Format</td>
<td>I10</td>
<td>I10</td>
<td>I10</td>
<td>I10</td>
</tr>
</tbody>
</table>

Other definitions are given in McDonald and Harbaugh (1988).

DOCUMENTATION OF MODFLOWARC

The following sections describe: the Arc Macro Language (AML) program, MODFLOWARC.AML, that the user initializes to start a simulation and the program, MODFLOWARC.F77, that is operated from this AML. In the following sections flowcharts show where modifications to each of the ground-water flow packages and the Streamflow-Routing package occurred within the original flowcharts as described by McDonald and Harbaugh (1988) and Prudic (1988). Only that part of the program that was modified or replaced is shown in the flowchart. Numbers in the upper right corner of the flowchart symbols match the original numbering in the flowcharts from McDonald and Harbaugh (1988) and Prudic (1988). The modifications within program code are shown in bold type and any variables that have been added to the code are described.

AML MODFLOWARC Program

The user initializes a simulation of the ground-water-flow model by activating the program MODFLOWARC.AML. This AML program is listed below. Discussion of the specific commands within this program is omitted and the user should consult the ARC/INFO manual on AML commands. The user must modify the line within the AML program shown in bold type to indicate the pathname of the program MODFLOWARC.F77.

The program MODFLOWARC.AML controls the operation of the modules of MODFLOWARC.F77. The user must first issue the command "arc" at the Primos level or Unix level and this command initiates the ARC/INFO software. The importance of this command is that the Primos search-rules are set thereby allowing the modules of MODFLOWARC to load the proper system libraries and to call routines within these system libraries; otherwise the routines within libraries can not be located and the programs fail. Documentation of the MODFLOWARC program follows.
AML name: Modflowarc.aml
Language: AML ARC Macro Language

Purpose: Enable the user to run Modflow program (Ground-water flow model by McDonald and Harbaugh, 1988) using the TASK function. Created for the ARC - McDonald/Harbaugh ground water model interface.

Arguments: none

Variable name, Type, Definition

modflowarc_files character user specified file that contains all filenames and fortran unit numbers that need to be opened in order to run the Ground_Water Flow_Model by McDonald and Harbaugh (1988).

History:

Leonard Orzol 11/06/89 USGS-WRD Portland OR Original Coding
Leonard Orzol 10/25/91 USGS-WRD Portland OR Version 3.0

** args modflowarc_files
&s homepath [show &workspace]
&s slash /
&s computer_flag [index %homepath% %slash%]
&if %computer_flag% <= 0 &then
 &do
 &s slash >
 &s fortran_path %fortran_path%%slash%progs%slash%
 &s fortran_program modflowarc.run
 &end
&else
 &do
 &s fortran_path %fortran_path%%slash%progs%slash%
 &s fortran_program modflowarc
 &end

&watch modflowarc.watch

&if [null %modflowarc_files%] &then ~
 &call usage

&select %modflowarc_files%
 &when help, HELP
 &do
 &call helping
 &return
 &end
 &otherwise
 &call disclaimer
 &end

&s fortran_program %fortran_path%%slash%%fortran_program%

&if [exists %fortran_program% -file] &then ~
 &do
 &s McModel [task %fortran_program% %modflowarc_files%]
 &select %McModel%
 &when 0'
 &call failing
 &otherwise
 &type Successful completion of Modflowarc program
 &end
 &end
&else
 &do
 &type %fortran_program% does exist....///
 &call failing
 &return
 &end

&workspace %homepath%
&watch &off
&return
/*
routine helping

&severity &warning &routine error_warning
&severity &error &routine error_fail

&type 
&type :===============================================:===============================================:
&type U.S. Geological Survey preliminary computer program
&type Modflowarc.aml and Modflowarc.f version %version%
&type :===============================================:===============================================:
&type Purpose:
&type Enables the user to pass a filename that contains all filenames
&type and fortran unit numbers that need to be opened in order to run
&type the Modular Three-Dimensional Finite-Difference Ground_Water Flow_Model
&type by McDonald and Harbaugh (1988) using the enhanced MODFLOWARC by
&type
&type Modflowarc.aml needs:
mandatory arguments;
1) <modflowarc_files> that contains the model files used during the
operation of MODFLOW (McDonald and Harbaugh, 1988) or MODFLOWARC
(Orzol and McGrath, 1991).

::: U.S. Geological Survey preliminary computer program
Modflowarc.aml and Modflowarc.f version %version%
Written in Arc/Info AML (rev 5.01) & Fortran77 last modified %date%
Program operates Prime and Suns and Dgs
Source code available from L.L.Orzol fts: 429-2256

Disclaimer:
Although this program has been used by the U.S. Geological Survey,
no warranty, expressed or implied, is made by the USGS as to
the accuracy and functioning of the program and related program
material nor shall the fact of distribution constitute any such
warranty, and no responsibility is assumed by the USGS in
connection therewith.

return

routine usage
Usage: [translate %program%] <modflowarc_files_file> or
[translate %program%] help\nwatch &off
&close_status [close -all]
stop

routine failing
Bailing out....[translate %program%] Failure....
workspace %homepath%
watch &off
&close_status [close -all]
stop

routine error_warning
severity &error &ignore
return

routine error_fail
severity &error &ignore
Bailing out....Interface Failure....
workspace %homepath%
watch &off
&close_status [close -all]
stop
MODFLOWARC.F77 Program

MODFLOWARC.F77 is termed a TASK function program in ARC/INFO (Environment Systems Research Institute Inc., 1989). Creation of the TASK function program code follows the programming rules of FORTRAN77. MODFLOWARC.F77 (1) initializes the ARC environment (initializing the various program modules within ARC/INFO software such as the ISP routines); (2) calls a subroutine that opens or closes the various files containing package control specified for MODFLOWARC; and (3) executes the model by calling a subroutine that is a modification of the MODFLOW main program. Following is the MODFLOWARC program.
CALL VINIT

CALL INFINT

CALL TTINIT

CALL AMLFNA (MFAFIL)

CALL MESCHR (MFAFIL,0)

CALL INFORM ('Opening modular model files using %1% as source',-1)

CALL MODFIL (MFAFIL, MAXFIL, USRFIL, USRUNT, NFILES, BASUNT, MODUNT, *9999)

OUTSTR='0'

CALL MODFLOW (BASUNT, MODUNT, *502)

OUTSTR='1'

CALL CLOSE_FILE (USRFIL(NUMFIL), *9999)

IF(OUTSTR.EQ.'1') THEN
    CALL INFORM ('\Successful Termination of Modflowarc',-1)
ELSE
    CALL INFORM ('\Abnormal Termination of Modflowarc',-1)
ENDIF

CALL AMLFNV (OUTSTR)

END
File Opening Module MODFIL in MODFLOWARC

MODFLOWARC code includes a module, MODFIL.F77, that opens, closes, and deletes files that must be used during a model simulation. The user builds an ASCII file that contains unit numbers and filenames. The first record of this file represents the unit number and filename for the Basic (BAS) package. The last record represents the unit number and filename where all printer output is directed. For input data that are read from or output that is recorded to unformatted files, the user specifies the unit number as a negative. Documentation of this module follows.
2) remaining model packages and data array filenames and fortran unit numbers, and finally
3) filename and fortran unit number for the operational output file that includes "runtime" information.

Language: FORTRAN, with ARC/INFO subroutine call version 5.01

Inputs:

MFAFIL : FORTRAN UNIT NUMBER FOR INPUT FILENAME USRFIL
MAXFIL : MAXIMUM NUMBER OF FILENAMES AND UNIT NUMBERS IN USRFIL FILE

Outputs:

BASUNT : FORTRAN UNIT NUMBER OF THE BASIC_PACKAGE
OUTUNT : FORTRAN UNIT NUMBER OF THE OPERATIONAL OUTPUT FILE
NFILES : NUMBER OF FILES ACTUALLY READ IN MFAFIL FILE
USRFILE : LIST OF PACKAGE/DATA_ARRAY FILES OPENED FOR USE DURING A MODFLOWARC RUN
USRUNT : A MATCHING LIST OF FORTRAN UNIT NUMBERS FOR EACH USRFIL FILE OPENED

History:

Leonard L. Orzol 10/25/91 Version 3.0 Coding, USGS WRD Portland, Or Fts: 429-2256

==============================================

CHARACTER(*) MFAFIL,USRFL(MAXFIL)
INTEGER USRUNT(MAXFIL)
INTEGER NFILES,BASUNT,OUTUNT

C1----Open MFAFIL file that contains fortran unit numbers and filenames
MFAUNT=99
NSTAT=1
ITALK=1
CALL OPEN_FILE (MFAUNT,MFAFIL,NSTAT,ITALK,*9999)

C2----Loop to read and write filenames and fortran unit numbers
NFILES=0
DO 200 NUMFIL=1,MAXFIL
   READ (MFAUNT,*,ERR=9990,END=202)
   & USRUNT(NUMFIL),
   & USRFIL(NUMFIL)
   NFILES=NUMFIL
200 CONTINUE
202 CALL CLOSE_FILE (MFAFIL,*9999)

C3----Retrieve Basic_Package and Basic_Package_Output fortran unit numbers and filenames
BASUNT=USRUNT(1)
OUTUNT=USRUNT(NFILES)
NSTAT=2
CALL DELETE_FILE (USRFL(NFILES),*9999)
CALL OPEN_FILE (OUTUNT, USRFIL(NFILES), NSTAT, ITALK, *9999)

C4----Print a message identifying the package
C
WRITE(OUTUNT,'(1H1,20X,''U.S. GEOLOGICAL SURVEY '',
&                       ''MODFLOWARC -- MODFLOW ARC/INFO ENHANCEMENT'',
&                       '' VERSION 3.0, 10/25/91'')')
C
C5----Write Modflowarc flow system files heading
C
WRITE(OUTUNT,'(/,1H1,36X,''MODFLOWARC PACKAGE AND DATA '',
&                       ''ARRAY FILE LIST'',
&                       '/,10X,85(''-''))')
C
C6----Write Modflowarc filenames and fortran unit numbers
C
      NSTAT=1
      MAXCHR=25
      DO 600 NUMFIL=1,NFILES-1
         CALL OPEN_FILE (USRUNT(NUMFIL),USRFIL(NUMFIL),NSTAT,ITALK,
                     *9999)
         NUMCHR=INDEX(USRFIL(NUMFIL),' ')-1
         IF(NUMCHR.LT.0) THEN
            NUMCHR=1
            LENCHR=25
         ELSE IF(NUMCHR.GT.MAXCHR) THEN
            NUMCHR=NUMCHR-MAXCHR+1
            LENCHR=NUMCHR+MAXCHR-1
         ELSE
            NUMCHR=1
            LENCHR=NUMCHR
          ENDIF
         IF(NUMFIL.LE.1) THEN
            WRITE(OUTUNT,'(10X,
&                       ''MODFLOWARC BASIC PACKAGE FILE (BAS) '',
&                       A25,'' BASFIL'')')
            USRFIL(NUMFIL) (NUMCHR:LENCHR)
         ELSE
            WRITE(OUTUNT,'(10X,
&                       ''MODFLOWARC FILE OPENED '',
&                       A25,'' FILE'')')
            USRFIL(NUMFIL) (NUMCHR:LENCHR)
          ENDIF
         WRITE(OUTUNT,'(10X,
&                       ''  ON FORTRAN UNIT NUMBER '',
&                       10X,I15,'' UNIT'')')
         USRUNT(NUMFIL)
      600 CONTINUE
      WRITE(OUTUNT,'(10X,
&                       ''MODFLOWARC OPERATIONAL OUTPUT FILE '',
&                       A25,'' OUTFIL'')')
      USRFIL(NFILES) (NUMCHR:LENCHR)
      WRITE(OUTUNT,'(10X,
&                       ''  ON FORTRAN UNIT NUMBER '',
&                       10X,I15,'' UNIT'')')
      USRUNT(NFILES)
      WRITE(OUTUNT,'(10X,85(''-''),//)')
GO TO 1000

C**********************************************************************C
C                  Program Errors
C**********************************************************************C

9990 WRITE(*,'(/,'' Unable to read from file '',A)')
&                       MFAFIL (1:INDEX(MFAFIL,' '-1) )
9991 WRITE(*,'(/,'' Abnormal Termination of Modfil_Subroutine'')')
9999 RETURN 1
Subroutine MODFLOW in MODFLOWARC

The modules of MODFLOWARC program consist of modifications to the "read and prepare" and "budget" modules of the packages from the ground-water model MODFLOW and the Streamflow-Routing package. Some of the "read and prepare" modules within the ground-water flow model call utility modules U2DREL, U2DINT, and U1DREL. These modules have been rewritten for MODFLOWARC and renamed to U2DRELARC, U2DINTARC, and U1DRELARC. "Budget" modules that contain the letters BD in their names such as BCF1BD.F77 call utility modules UBUDSV and ULASAV in MODFLOW. These modules have been completely replaced with code that writes only to ARC/INFO files and are renamed UBUDSVARC and ULASAVARC. A user can continue to send output data to the utility modules UBUDSV and ULASAV by properly setting the output flags of the Basic package and the record/print flags within individual packages.

MODFLOWARC code includes a subroutine that is a modified version of the original main fortran program, MODFLOW.F77, by McDonald and Harbaugh (1988). The modifications include "read and prepare" and "budget" modules such as BAS1RPARC.F77 or BCF1BDARC.F77 and BAS1OTARC.F77 for each of the MODFLOW packages. Documentation of the modifications to the main program follows.
SUBROUTINE MODFLOW (INBAS, IOUT, *)
PARAMETER (MAXCOM=600000)
COMMON /XCOM/ X(MAXCOM)
COMMON /FLWCOM/ LAYCON(80)
CHARACTER*80 HDPATH, BDPATH
CHARACTER*4 HEADNG, VBNM
DIMENSION HEADNG(32), VBNM(4, 20), VBVL(4, 20), IUNIT(24)
DOUBLE PRECISION DUMMY
EQUIVALENCE (DUMMY, X(1))
SAVE /XCOM/,
SAVE /FLWCOM/,
EXTERNAL INFORM, MESINT

C1------SET SIZE OF X ARRAY. REMEMBER TO REDIMENSION X.
LENX=MAXCOM

C2------ASSIGN BASIC INPUT UNIT AND PRINTER UNIT.
(NOW ASSIGNED BY USER IN MODFLOWARC)

C      INBAS=USER ASSIGNED
C      IOUT=USER ASSIGNED

C3------DEFINE PROBLEM__ROWS, COLUMNS, LAYERS, STRESS PERIODS, PACKAGES
CALL BAS1DF(ISUM, HEADNG, NPER, ITMUNI, TOTIM, NCOL, NROW, NLAY,
    1 NODES, INBAS, IOUT, IUNIT)

C4------ALLOCATE SPACE IN "X" ARRAY.
CALL BASIAL(ISUM, LENX, LCHNEW, LCHOLD, LCIBOU, LCCR, LCCC, LCCV,
    1 LCHCOP, LCRHS, LCDELR, LCDELCL, LCESTR, LCBUFF, LCIOFL,
    2 INBAS, ISTRT, NCOL, NROW, NLAY, IOUT)
IF(IUNIT(1).GT.0) CALL BCF1IAL(ISUM, LENX, LCSC1, LCHY,
    1 LCBOT, LCTOP, LCSC2, LCTRPY, IUNIT(1), ISS,
    2 NCOL, NROW, NLAY, IOUT, ICBCF)
IF(IUNIT(2).GT.0) CALL WELIAL(ISUM, LENX, LCWELL, MUXWELL, MUXWELLS,
    1 IUNIT(2), IOUT, ICWEL)
IF(IUNIT(3).GT.0) CALL DRNIAL(ISUM, LENX, LCDRAI, NDRRAIN, MXDRN,
    1 IUNIT(3), IOUT, IDCRI)
IF(IUNIT(8).GT.0) CALL RCH1IAL(ISUM, LENX, LCRCHL, LCRECH, NRCHOP,
    1 NCOL, NROW, IUNIT(8), IOUT, ICHRB)
IF(IUNIT(5).GT.0) CALL EVT1IAL(ISUM, LENX, LCEVT, LCEVTR, LCEXDP,
    1 LCSPRF, NCOL, NROW, NEVTOP, IUNIT(5), IOUT, IEVCB)
IF(IUNIT(4).GT.0) CALL RIV1IAL(ISUM, LENX, LCRIVR, MRIVER, NRIVER,
    1 IUNIT(4), IOUT, IRIVCB)
IF(IUNIT(13).GT.0) CALL SRIAL(ISUM, LENX, LCSTRM, ICSTRM, MXSTRM, STR1
    1 NSTRM, IUNIT(13), IOUT, ISTRM1, ISTRM2, NSS, NTRIB, STR1
    2 NDIV, ICALC, CONST, LCTBAR, LCSTRB, LCVAR) STR1
IF(IUNIT(7).GT.0) CALL GHBIAL(ISUM, LENX, LCHBND, MHBOND, MXBND,
    1 IUNIT(7), IOUT, IGHBCB)
IF(IUNIT(9).GT.0) CALL SIPIAL(ISUM, LENX, LCEL, LCEF, LGCL, LCV,
    1 LCDCG, LCLRC, LCW, MXITER, NPARN, NCOL, NROW, NLAY,
    2 IUNIT(9), IOUT)
IF(IUNIT(11).GT.0) CALL SORIAL(ISUM, LENX, LCA, LCRES, LCHDCG, LCLRC,
    1 LCIEQP, MXITER, NCOL, NLAY, NSLICE, MBW, IUNIT(11), IOUT)

C5------IF THE "X" ARRAY IS NOT BIG ENOUGH THEN STOP.
    IF(ISUM-1.GT.LENX) GO TO 9999

C6------READ AND PREPARE INFORMATION FOR ENTIRE SIMULATION.
CALL BASIRPARC(X(LCIBOU), X(LCHNEW), X(LCSTRAT), X(LCHOLD),
    1 ISTRAT, INBAS, HEADNG, NCOL, NROW, NLAY, NODES, VBVL, X(LCIIOFL),
    2 IUNIT(12), IHEDPFM, IDNHFPM, IHEDF, IDDNMH, IOUT, IDPATH,
    E *9999)
IF(IUNIT(1).GT.0) CALL BCF1RPARC(X(LCIBOU), X(LCHNEW), X(LCSC1),
    1 X(LCHY), X(LCCR), X(LCCC), X(LCCV), X(LCDELR),
    2 X(LCDELCL), X(LCBOT), X(LCTOP), X(LCSC2), X(LCTRPY),
    3 IUNIT(1), ISS, NCOL, NROW, NLAY, NODES, IOUT,
    E *9999)
IF(IUNIT(9).GT.0) CALL SIP1RP(NPARN, MXITER, ACCL, HCLOSE, X(LCW),
IUNIT(9), IPCALC, IPRSIG, IOUT)
IF(IUNIT(11).GT.0) CALL SOR1RP(MXITER, ACCL, HCLOSE, IUNIT(11),
1 IPRSOR, IOUT)

C
C7------SIMULATE EACH STRESS PERIOD.
DO 300 KPER=1,NPER
KPER=KPER

C
C7A------READ STRESS PERIOD TIMING INFORMATION.
CALL BAS1ST(NSTP, DELT, TSMULT, PERTIM, KPER, INBAS, IOUT)

C
C7B------READ AND PREPARE INFORMATION FOR STRESS PERIOD.
IF(IUNIT(2).GT.0) CALL WEL1RPARC(X(LCWELL), NWELLS, MXWELL,
1 E
IF(IUNIT(3).GT.0) CALL DRN1RPARC(X(LCDRAI), NDRAIN, MXDRN,
1 E
IF(IUNIT(8).GT.0) CALL RCH1RPARC(NRCHOP, X(LCIRCH), X(LCRECH),
1 X(LCDELR), X(LCDELCA), NROW,
2 NCOL, IUNIT(8), IOUT,
1 *9999)
IF(IUNIT(5).GT.0) CALL EVT1RPARC(NEVTOP, X(LCIEVT), X(LCEVTR),
1 X(LCEXDP), X(LCSURF), X(LCDELRC),
2 X(LCDEL), NCOL, NROW,
3 IUNIT(5), IOUT,
1 *9999)
IF(IUNIT(4).GT.0) CALL RIV1RPARC(X(LCRIVR), NRIVER, MXRIVR,
1 E
IF(IUNIT(13).GT.0) CALL STR1RPARC(X(LCSTRM), X(ICSTRM), NSTREM, STR1
1 MXSTRM, IUNIT(13), IOUT, X(LCTBAR), NDIV, NSS,
1 STR1 NTRIB, X(LCTRIB), ICALC, IPTFLG,
2 STR1 E
IF(IUNIT(7).GT.0) CALL GHB1RPARC(X(LCBNDS), NBOUND, MXBND,
1 E

C
C7C------SIMULATE EACH TIME STEP.
DO 200 KSTP=1,NSTP
KKSTP=KKSTP

C
C7C1------CALCULATE TIME STEP LENGTH. SET HOLD=HNEW..
CALL BAS1AD(DELT, TSMULT, TOTIM, PERTIM, X(LCHNEW), X(LCHOLD), KKSTP,
1 NCOL, NROW, NLAY)

C
C7C2------ITERATIVELY FORMULATE AND SOLVE THE EQUATIONS.
DO 100 KITER=1,MXITER
KKITER=KKITER

C
C7C2A------FORMULATE THE FINITE DIFFERENCE EQUATIONS.
CALL BAS1FM(X(LCHCOF), X(LCRHS), NODES)
IF(IUNIT(1).GT.0) CALL BCF1FM(X(LCHCOF), X(LCRHS), X(LCHOLD),
1 X(LCS1), X(LCHNEW), X(LCIBOU), X(LCCR), X(LCCV),
2 X(LCH), X(LCTRPY), X(LCBOT), X(LCTOP), X(LSC2),
3 X(LCDRL), X(LCDLR), DELT, ISS, KKITER, KKSTP, KPER, NCOL,
4 NROW, NLAY, IOUT)
IF(IUNIT(2).GT.0) CALL WEL1FM(NWELLS, MXWELL, X(LCRHS), X(LCWELL),
1 X(LCIBOU), NCOL, NROW, NLAY)
IF(IUNIT(3).GT.0) CALL DRN1FM(NDRAIN, MXDRN, X(LCDRAI), X(LCHNEW),
1 X(LCHCOF), X(LCRHS), X(LCIBOU), NCOL, NROW, NLAY)
IF(IUNIT(8).GT.0) CALL RCH1FM(NRCHOP, X(LCIRCH), X(LCRECH),
1 X(LCRHS), X(LCIBOU), NCOL, NROW, NLAY)
IF(IUNIT(5).GT.0) CALL EVT1FM(NEVTOP, X(LCIEVT), X(LCEVTR),
1 X(LCEXDP), X(LCSURF), X(LCDELRC), X(LCIBOU),
2 X(LCHNEW), NCOL, NROW, NLAY)
IF(IUNIT(4).GT.0) CALL RIV1FM(NRIVER, MXRIVR, X(LCRIVR), X(LCHNEW),
1 X(LCHCOF), X(LCRHS), X(LCIBOU), NCOL, NROW, NLAY)
IF(IUNIT(13).GT.0) CALL STR1FM(NSTREM, X(LCSTRM), X(ICSTRM), STR1
1 X(LCHNEW), X(LCHCOF), X(LCRHS), STR1
2 X(LCIBOU), MXSTRM, NCOL, NROW, NLAY, IOUT, NSS,
3 STR1 X(LCTBAR), NTRIB, X(LCTRIB), X(LCTVAR), ICALC, CONST) STR1
IF(IUNIT(7).GT.0) CALL GHB1FM(NBOUND,MXBND,X(LCBNDS),X(LCHCOF),
X(LCRHS),X(LCIBOU),NCOL,NROW,NLAY)

C7C2B---MAKE ONE CUT AT AN APPROXIMATE SOLUTION.

CALL MESINT (KKITER)
CALL INFORM ('Modflow solution try %1%',-1)
IF(IUNIT(9).GT.0) CALL SIP1AP(X(LCHNEW),X(LCIBOU),X(LCCR),X(LCCC),
X(LCCV),X(LCHDCG),X(LCLRCH),NPARM,KKITER,HCLOSE,ACCL,ICNVG,
KKSTP,KKPER,IPCALC,IPRSIP,MXITER,NSTP,NCOL,NROW,NLAY,NODES,
IPOUT)

C
C7C2C---IF CONVERGENCE CRITERION HAS BEEN MET STOP ITERATING.
IF(ICNVG.EQ.1) GO TO 110
100 CONTINUE
KITER=MXITER
110 CONTINUE

C7C3----DETERMINE WHICH OUTPUT IS NEEDED.
CALL BASIOARC (NSTP,KKSTP,ICNVG,X(LCIOFL),NLAY,
IBUDFL,ICBCFL,IHDFL,IUNIT(12),IOUT,BDPATH,
*9999)

C7C4----CALCULATE BUDGET TERMS. SAVE CELL-BY-CELL FLOW TERMS.
MSUM=1
IF(IUNIT(1).GT.0) CALL BCF1BDARC (VBNM,VBVL,MSUM,X(LCHNEW),
X(LCIBOU),X(LCHOLD),X(LCSC1),X(LCCR),X(LCCX),X(LCCT),
X(LCTOP),X(LCS2),DELT,ISS,NCOL,NROW,NLAY,KKSTP,KKPER,
ICBCFB,ICBCFL,X(LBUFF),IOUT,BDPATH,
*9999)
IF(IUNIT(2).GT.0) CALL WEL1BDARC (NWELLS,MXWELL,VBNM,VBVL,MSUM,
X(LCWELL),X(LCIWOU),DELT,NCOL,NROW,NLAY,KKSTP,KKPER,iewelcb,
ICBCFL,X(LBUFF),IOUT,BDPATH,
*9999)
IF(IUNIT(3).GT.0) CALL DRN1BDARC (NDRAIN,MXDRN,VBNM,VBVL,MSUM,
X(LCDRAI),DELT,X(LCHNEW),NCOL,NROW,NLAY,X(LCIBOU),KKSTP,
2 KKPER,IDRNCB,ICBCFL,X(LBUFF),IOUT,BDPATH,
*9999)
IF(IUNIT(8).GT.0) CALL RCH1BDARC (NRCHOP,X(LICIRCH),X(LCIRCH),
X(LCIBOU),NROW,NCOL,NLAY,DELT,VBVL,VBNM,MSUM,KKSTP,KKPER,
ICRCBC,ICBCFL,X(LBUFF),IOUT,BDPATH,
*9999)
IF(IUNIT(5).GT.0) CALL EVT1BDARC (NEVTOP,X(LCEVTR),X(LCEVT),
X(LCEXDP),X(LCSURF),X(LCIBOU),X(LCHNEW),NCOL,NROW,NLAY,
2 DELT,VBVL,VBNM,MSUM,KKSTP,KKPER,IEVTCB,ICBCFL,X(LBUFF),
*9999)
IF(IUNIT(4).GT.0) CALL RIV1BDARC (NRIVER,MXIRIVR,X(LCIRVR),
X(LCIBOU),X(LCHNEW),NCOL,NROW,NLAY,DELT,VBVL,VBNM,MSUM,
2 KKSTP,KKPER,IRIVCB,ICBCFL,X(LBUFF),IOUT,BDPATH,
*9999)
IF(IUNIT(13).GT.0) CALL STR1BDARC (NSTREM,X(LCSTRM),X(ICSTRM),
X(LCIBOU),MXSTRM,X(LCHNEW),NCOL,NROW,NLAY,DELT,VBVL,
2 VBVM,MSUM,KKSTP,KKPER,ISTCB1,ISTCB2,ICBCFL,X(LBUFF),IOUT,
*9999)
NTRIB,NSS,X(LCTRB2),X(LCTVAR),ICALC,CONST,
4 IPTFLG,BDPATH,
*9999)
IF(IUNIT(13).GT.0) CALL STR1SRARC (NSTREM,X(LCSTRM),X(ICSTRM),
1 MXSTRM,KKSTP,KKPER,ICALC,
2 IPTFLG,ICBCFL,BDPATH,IOUT,
*9999)
IF(IUNIT(7).GT.0) CALL GHB1BDARC (NBOUND,MXBND,VBNM,VBVL,MSUM,
1 X(LCBNDS),DELX,X(LCHNEW),NCOL,NROW,NLAY,X(LCIBOU),KKSTP,
2 KKPER,IGHBCB,ICBCFL,X(LBUFF),IOUT,BDPATH,
*9999)

C7C5---PRINT AND OR SAVE HEADS AND DRAWDOWNS. PRINT OVERALL BUDGET.
CALL BAS1OTARC (X(LCNEW), X(LCSTRT), ISTRT, X(LCBUFF), X(LCIOFL),
1       MSUM, X(LCIBOU), VBNM, VBVL, KKSTP, KKPER, DELT,
2       PERTIM, TOTIM, ITMUNI, NCOL, NROW, NLAY, ICNVG,
3       IHDDFL, IBUDFL, IHEDFM, IHEDUN, IDDNFM, IDDNUN, IOUT, HDPATH,
        *9999)

C
C7C6----IF ITERATION FAILED TO CONVERGE THEN STOP.
C   IF(ICNVG.EQ.0) STOP
200 CONTINUE
300 CONTINUE
C
C8------END PROGRAM
C   STOP
   RETURN
C
CE------ERRORS
C
9999 CALL INFORM ("Abnormal Termination of Modflow_Program",-1)
   RETURN 1
END

Basic Package modules

The BAS (Basic package) consists of eight primary modules and five submodules; of these, three primary modules (BAS1RPARC, BAS10CARC, and BAS1OTARC) and three submodules (SBAS1DARC, SBAS1HARC, and SBAS1IARC) were changed as indicated below.

BAS1RPARC

This module reads and prepares data for the BASIC package from either ASCII or ARC/INFO files and calls submodules U2DINTARC and U2DRELARC. Output control calls the modified submodule SBAS1IARC. The flowchart for part of BAS1RPARC is shown in figure 1, and the documentation of the module follows.

--- Diagram ---

Figure 1.—Modified program elements for the BAS1RPARC module.
SUBROUTINE BAS1RPARC (IBOUND,HNEW,STRT,HOLD,ISTRT,INBAS,
  **      HEADNG,NCOL,NROW,NLAY,NODES,VBVL,IOFLG,INOC,IHDPM,**
  **      INBAS,IOFLG,INOC,IHEDFM,**
  **      IHDUN,IDDNFM,IDDNUN,IOUT,HDPATH,**
      E                *)

C-----VERSION 3.0 25OCTOBER1991 BAS1RPARC
C                 MODIFIED BY LEONARD L. ORZOL
C
C     ******************************************************************
C     READ AND INITIALIZE BASIC MODEL ARRAYS
C     ******************************************************************
C
C        SPECIFICATIONS:
C        ---------------------------------------------------------------
CHARACTER*80 HDPATH
CHARACTER*16 INFOITEM
CHARACTER*4 HEADNG,ANAME
DOUBLE PRECISION HNEW,HNOFLO
C
DIMENSION HNEW(NODES),IBOUND(NODES),STRT(NODES),HOLD(NODES),
  1 ANAME(6,2),VBVL(4,20),IOFLG(NLAY,4),HEADNG(32)
C
DATA ANAME(1,1),ANAME(2,1),ANAME(3,1),ANAME(4,1),ANAME(5,1),
  1 ANAME(6,1) /'    ',',    ','  BO','UND','RY A','RRAY'/
DATA ANAME(1,2),ANAME(2,2),ANAME(3,2),ANAME(4,2),ANAME(5,2),
  1 ANAME(6,2) /'    ',',    ','    ',',IAL ','HEAD'/
---------------------------------------------------------------
C
C1------PRINT SIMULATION TITLE, CALCULATE # OF CELLS IN A LAYER.
C
WRITE(IOUT,1) HEADNG
1  FORMAT(1H1,32A4)
NCR=NCOL*NROW
C
C2------READ BOUNDARY ARRAY(IBOUND) ONE LAYER AT A TIME.
C
DO 100 K=1,NLAY
  100 CONTINUE
K=K
LOC=1+(K-1)*NCR
INFOITEM='IBOUND'
CALL U2DINTARC (INFOITEM,IBOUND(LOC),ANAME(1,1),NROW,NCOL,KK,
  1 INBAS,IOUT, *9999)
C
C3------READ AND PRINT HEAD VALUE TO BE PRINTED FOR NO-FLOW CELLS.
C
READ(INBAS,2) TMP
2  FORMAT(F10.0)
HNOFLO=TMP
WRITE(IOUT,3) TMP
3     FORMAT(1H0,'AQUIFER HEAD WILL BE SET TO ',1PG11.5,
  1 ' AT ALL NO-FLOW NODES (IBOUND=0).')
C
C4------READ STARTING HEADS.
C
DO 300 K=1,NLAY
  300 CONTINUE
K=K
LOC=1+(K-1)*NCR
INFOITEM='SHEAD'
CALL U2DRELARC (INFOITEM,HOLD(LOC),ANAME(1,2),NROW,NCOL,KK,
  1 INBAS,IOUT, *9999)
C
C5------COPY INITIAL HEADS FROM HOLD TO HNEW.
C
DO 400 I=1,NODES
  400 CONTINUE
C6------IF STARTING HEADS ARE TO BE SAVED THEN COPY HOLD TO STRT.
C
    IF(ISTRT.EQ.0) GO TO 590
DO 500 I=1,NODES
    STRT(I)=HOLD(I)
500 CONTINUE
C
C7------INITIALIZE VOLUMETRIC BUDGET ACCUMULATORS TO ZERO.
C
590 DO 600 I=1,20
    DO 600 J=1,4
        VBVL(J,I)=0.
600 CONTINUE
C
C8------SET UP OUTPUT CONTROL.
C
CALL SBAS1IARC (NLAY,ISTRT,IOFLG,INOC,IOUT,IHEDFM,
E       &      IDDNFM,IHEDUN,IDDNUN,HDPATH,
            *9999)
C
C9------RETURN
C
1000 RETURN
C
CE------ERRORS
C
9999 CALL INFORM (‘\\Abnormal Termination of Baslrp_Arc_Subroutine’,-1)
RETURN 1
END

Added variables for module BASIRPARC

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>HDPATH</td>
<td>Package</td>
<td>The directory path to the ARC/INFO subdirectory where head and drawdown arrays are recorded.</td>
</tr>
<tr>
<td>INFOITEM</td>
<td>Module</td>
<td>The name of the INFO item either primary or redefined within the ARC/INFO file containing the information.</td>
</tr>
</tbody>
</table>
The BAS1OCARC module sets flags for the budget and output procedures for the BASIC package (fig. 2). Two new variables were added to the program code: BUFFER and OUTPATH. The major change consisted of an alternate branch marked 3B (fig. 2 numbering follows the numbering of the comment lines within the program code) in the following program code where the program control checks the control record for the variable OUTPATH.

```
SUBROUTINE BAS1OCARC (NSTP,KSTP,ICNVG,IOFLG,NLAY, &                           IBUDFL,ICBCFL,IHDDFL,INOC,IOUT,OUTPATH, &                                                      E                      *)
C
C-----VERSION 3.0 25OCTOBER1991 BAS1OCARC
C               MODIFIED BY LEONARD L. ORZOL
C
C     ******************************************************************
C     OUTPUT CONTROLLER FOR HEAD, DRAWDOWN, AND BUDGET
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
DIMENSION IOFLG(NLAY,4)
CHARACTER*132 BUFFER
CHARACTER*80 OUTPATH
C     ------------------------------------------------------------------
C
C1------TEST UNIT NUMBER (INOC (INOC=IUNIT(12))) TO SEE IF
C1------OUTPUT CONTROL IS ACTIVE.
    IF(INOC.NE.0)GO TO 500
C
C2------IF OUTPUT CONTROL IS INACTIVE THEN SET DEFAULTS AND RETURN.
    IHDDFL=0
    IF(ICNVG.EQ.0 .OR. KSTP.EQ.NSTP)IHDDFL=1
    IBUDFL=0
    IF(ICNVG.EQ.0 .OR. KSTP.EQ.NSTP)IBUDFL=1
```
ICBCFL=0
GO TO 1000

C3-------READ AND PRINT OUTPUT FLAGS AND CODE FOR DEFINING IOFLG.
C
500  READ(INOC,'(A132)',ERR=9990,END=9991) BUFFER
C
C3A-------READ AND PRINT README FOR DEFINING IOFLG.
C
IF(BUFFER (41:41).EQ.'' .OR. BUFFER (41:41).EQ.' ') THEN
   READ(BUFFER,'(4I10)',ERR=9992) INCODE,IHDDFL,IBUDFL,ICBCFL 
   IF(IHDDFL.LT.0) GO TO 9993 
   IF(ICBCFL.LT.0) GO TO 9994
C
C3B-------READ AND PRINT OUTPUT FLAGS AND CODE FOR DEFINING IOFLG AND
C----------OUTPUT PATH FOR INFO FILES
C
ELSE
   READ(BUFFER,'(4I10,A80)',ERR=9993) &
      INCODE,IHDDFL,IBUDFL,ICBCFL,OUTPATH
ENDIF

C4------DECODE INCODE TO DETERMINE HOW TO SET FLAGS IN IOFLG.
IF(INCODE) 100,200,300
C
C5------USE IOFLG FROM LAST TIME STEP.
100 WRITE(IOUT,101)
   101 FORMAT(1H1,'REUSING PREVIOUS VALUES OF IOFLG')
   GO TO 600

C6------READ IOFLG FOR LAYER 1 AND ASSIGN SAME TO ALL LAYERS
200 READ(INOC,201,ERR=9996,END=9997) (IOFLG(1,M),M=1,4)
   201 FORMAT(4I10)
   DO 210 K=1,NLAY
      IOFLG(K,1)=IOFLG(1,1)
      IOFLG(K,2)=IOFLG(1,2)
      IOFLG(K,3)=IOFLG(1,3)
      IOFLG(K,4)=IOFLG(1,4)
   210 CONTINUE
   WRITE(IOUT,211) (IOFLG(1,M),M=1,4)
   211 FORMAT(1H0,'OUTPUT FLAGS FOR ALL LAYERS ARE THE SAME:'/
            1X,'         HEAD    DRAWDOWN  HEAD  DRAWDOWN'/
            1X,'LAYER  PRINTOUT  PRINTOUT  SAVE    SAVE'/
            1X,41('-'))/1X,I5,I10,I8,I8)
   GO TO 600

C7------READ IOFLG IN ENTIRETY
300 READ(INOC,301,ERR=9996,END=9997) ((IOFLG(K,I),I=1,4),K=1,NLAY)
   301 FORMAT(4I10)
   WRITE(IOUT,302)
   302 FORMAT(1H0,'OUTPUT FLAGS FOR EACH LAYER:'/
            1X,'         HEAD    DRAWDOWN  HEAD  DRAWDOWN'/
            2X,'LAYER  PRINTOUT  PRINTOUT  SAVE    SAVE'/
            3X,41('-'))/K,(IOFLG(K,I),I=1,4),K=1,NLAY)
   303 FORMAT(1X,I4,I8,I10,I8,I8)

C8------THE LAST STEP IN A STRESS PERIOD AND STEPS WHERE ITERATIVE
C8------PROCEDURE FAILED TO CONVERGE GET A VOLUMETRIC BUDGET.
600 IF(ICNVG.EQ.0 .OR. KSTP.EQ.NSTP) IBUDFL=1
C
C9------RETURN
1000 RETURN
C
CE------ERRORS
C
9990 CALL INFORM ('\Unable to read Basic Output Control Package',-1)
CALL INFORM
& ('\Abnormal Termination of Basloc_Arc_Subroutine',-1)
RETURN 1
9991 CALL INFORM ('\End_of_file; ' //
& 'Missing input to Basic Output Control Package',-1)
& ('\Abnormal Termination of Basloc_Arc_Subroutine',-1)
RETURN 1
9992 CALL INFORM ('\Unable to read INCODE, IHDDFL, IBUDFL, and ' //
& 'ICBCFL from output control line',-1)
& ('\Abnormal Termination of Basloc_Arc_Subroutine',-1)
RETURN 1
9993 CALL INFORM ('\IHDDFL has been set to less than 0.',-1)
CALL INFORM (' This flag indicates to record head/drawdown',-1)
CALL INFORM (' output data in ARC/INFO files, but path to',-1)
CALL INFORM (' output directory, OUTPATH, is missing on ' //
& 'output control line',-1)
& ('\Abnormal Termination of Basloc_Arc_Subroutine',-1)
RETURN 1
9994 CALL INFORM ('\ICBCFL has been set to less than 0.',-1)
CALL INFORM (' This flag indicates to record package',-1)
CALL INFORM (' output data in ARC/INFO files, but path to',-1)
CALL INFORM (' output directory, OUTPATH, is missing on ' //
& 'output control line',-1)
& ('\Abnormal Termination of Basloc_Arc_Subroutine',-1)
RETURN 1
9995 CALL INFORM ('\Unable to read INCODE, IHDDFL, IBUDFL, ICBCFL',-1)
& 'OUTPATH from output control line',-1)
& ('\Abnormal Termination of Basloc_Arc_Subroutine',-1)
RETURN 1
9996 CALL INFORM ('\Unable to read layer output flags',-1)
CALL INFORM ('\Abnormal Termination of Basloc_Arc_Subroutine',-1)
RETURN 1
9997 CALL INFORM ('\End_of_file; Missing layer output layers',-1)
CALL INFORM ('to Basic Output Control Package',-1)
& ('\Abnormal Termination of Basloc_Arc_Subroutine',-1)
RETURN 1
END

Added variables for module BAS1OCARC

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>BUFFER</td>
<td>Module</td>
<td>The control record is read into this variable and is parsed into the appropriate variables: INCODE, IHDDFL, IBUDFL, ICBCFL, and OUTPATH.</td>
</tr>
<tr>
<td>OUTPATH</td>
<td>Package</td>
<td>The directory path to the ARC/INFO subdirectory where cell-by-cell flow-term arrays are recorded depending on flags set in the component of flow packages such as IWELCB.</td>
</tr>
</tbody>
</table>
This module directs the output procedures for the BASIC package. One new variable, OUTPATH, was added to the program code. The major change consists of replacing the submodules SBAS1H with SBAS1HARC and SBAS1D with SBAS1DARC marked 3B in figure 3. Documentation of the program code follows.

```fortran
SUBROUTINE BAS1OTARC (HNEW, STRT, ISTRT, BUFF, IOFLG, MSUM, IBOUND,
&                        VBNM, VBVL, KSTP, KPER, DELT, PERTIM, TOTIM, ITMUNI, NCOL, NROW, NLAY,
&                        ICNVG, IHDDFL, IBUDFL, IHEDFM, IHEDUN, IDDNFM, IDDNUN, IOUT, OUTPATH,
&                        *9999)
C
C------VERSION 3.0 25OCTOBER1991 BAS1OTARC
C          MODIFIED BY LEONARD L. ORZOL
C
C     ******************************************************************
C     OUTPUT TIME, VOLUMETRIC BUDGET, HEAD, AND DRAWDOWN
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
CHARACTER*80 OUTPATH
CHARACTER*4 VBNM
DOUBLE PRECISION HNEW
C
DIMENSION HNEW(NCOL,NROW,NLAY), STRT(NCOL,NROW,NLAY),
1          VBNM(4,20), VBVL(4,20), IOFLG(NLAY,4),
2          IBOUND(NCOL,NROW,NLAY), BUFF(NCOL,NROW,NLAY)
C     ------------------------------------------------------------------
C
C1------CLEAR PRINTOUT FLAG (IPFLG)
C
IPFLG=0
C
C2------IF ITERATIVE PROCEDURE FAILED TO CONVERGE PRINT MESSAGE
C
IF(ICNVG.EQ.0) WRITE(IOUT,1) KSTP, KPER
1 FORMAT(1H0,10X,'****FAILED TO CONVERGE IN TIME STEP',I3,
1      ' OF STRESS PERIOD',I3,'****')
C
C3------IF HEAD AND DRAWDOWN FLAG (IHDDFL) IS SET WRITE HEAD AND
C3------DRAWDOWN IN ACCORDANCE WITH FLAGS IN IOFLG.
C
IF(IHDDFL.EQ.0) GO TO 100
C
CALL SBAS1HARC (HNEW, BUFF, IOFLG, KSTP, KPER, NCOL, NROW, NLAY, IOUT,
&                      IHEDFM, IHEDUN, IHDDFL, OUTPATH, IPFLG, PERTIM, TOTIM,
&                      *9999)
CALL SBAS1DARC (HNEW, BUFF, IOFLG, KSTP, KPER, NCOL, NROW, NLAY, IOUT,
&                      IDDNFM, IDDNUN, IHDDFL, OUTPATH, STRT, ISTRT, IBOUND,
&                      IPFLG, PERTIM, TOTIM,
&                      *9999)
C
```

Figure 3.—Modified program elements for BAS1OTARC module.
C
C4------PRINT TOTAL BUDGET IF REQUESTED
C
100 IF(IBMFL.EQ.0) GO TO 120
   CALL SBAS1V (MSUM,VBNM,VBLK,KSTP,KPER,IOUT)
      IPFLG=1
C
C5------END PRINTOUT WITH TIME SUMMARY AND FORM FEED IF ANY PRINTOUT
C5------WILL BE PRODUCED.
C
120 IF(IPFLG.EQ.0) RETURN
   CALL SBAS1T (KSTP,KPER,DELT,PERTIM,TOTIM,ITMUNI,IOUT)
   WRITE(IOUT,101)
101     FORMAT(1H1)
C
C6------RETURN
C
RETURN
C
CE------ERRORS
C
9999 CALL INFORM ('\Abnormal Termination of Baslot_Arc_Subroutine',-1)
   RETURN 1
END

Added variables for module BAS1OTARC

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>OUTPATH</td>
<td>Package</td>
<td>The directory path to the ARC/INFO subdirectory where head and drawdown arrays are recorded.</td>
</tr>
</tbody>
</table>
This submodule directs the output procedures for drawdown within the BASIC package and is called by module BAS1OTARC. Six new variables were added to the program code: IDDNFL, OUTPATH, DDNPATH, INFONAME, INFOITEM, and QFILE. The major change consists of replacing program code to direct storage of the drawdown values into either unformatted files, ARC/INFO files, or INFO files. When the flag, IDDNFL, is set to a negative value, submodule SBAS1DARC calls the utility module ULASAVARC that writes the drawdown array into an ARC/INFO file for each layer specified by the user (fig. 4). Documentation of submodule SBAS1DARC follows.

SUBROUTINE SBAS1DARC (HNEW, BUFF, IOFLG, KSTP, KPER, NCOL, NROW, NLAY, IOUT, IDDNFM, IDDNUN, IDDNFL, DDNPATH, STRT, ISTRT, IBOUND, IPFLG, PERTIM, TOTIM, *)
C
C-----VERSION 3.0 25OCTOBER1991 SBAS1DARC
C MODIFIED BY LEONARD L. ORZOL
C
C ***********************************************
C CALCULATE PRINT AND RECORD DRAWDOWNS
C ***********************************************
C
SPECIFICATIONS
-------------------------------------------
CHARACTER*128 OUTPATH
CHARACTER*80 DDNPATH
CHARACTER*32 INFONAME
CHARACTER*16 INFOITEM
CHARACTER*4 TEXT
DOUBLE PRECISION HNEW
LOGICAL QFILE

DIMENSION HNEW(NCOL, NROW, NLAY), IOFLG(NLAY, 4), TEXT(4), 1 BUFF (NCOL, NROW, NLAY), STRT(NCOL, NROW, NLAY), 2 IBOUND (NCOL, NROW, NLAY)

DATA TEXT(1), TEXT(2), TEXT(3), TEXT(4) /'    ',' DRAW', 1 'DOWN'/
C1-------FOR EACH LAYER CALCULATE DRAWDOWN IF PRINT OR RECORD
C1-------IS REQUESTED
DO 59 K=1,NLAY
C
C2-------IS DRAWDOWN NEEDED FOR THIS LAYER?
    IF(IOFLG(K,2).EQ.0 .AND. IOFLG(K,4).EQ.0) GO TO 59
C
C3-------DRAWDOWN IS NEEDED. WERE STARTING HEADS SAVED?
    IF(ISTRT.NE.0) GO TO 53
C
C4-------STARTING HEADS WERE NOT SAVED. PRINT MESSAGE AND STOP.
    WRITE(IOUT,52)
52 FORMAT(1H0,'CANNOT CALCULATE DRAWDOWN BECAUSE START',
           ' HEADS WERE NOT SAVED')
    GO TO 9999
C
C5-------CALCULATE DRAWDOWN FOR THE LAYER.
53 DO 58 I=1,NROW
     DO 58 J=1,NCOL
         HSG=HNEW(J,I,K)
         BUFF(J,I,K)=HSING
         IF(IBOUND(J,I,K).NE.0) BUFF(J,I,K)=STRT(J,I,K)-HSING
58 CONTINUE
59 CONTINUE
C
C6-------FOR EACH LAYER: DETERMINE IF DRAWDOWN SHOULD BE PRINTED.
C6-------IF SO THEN CALL ULAPRS OR ULAPRN TO PRINT DRAWDOWN.
DO 69 K=1,NLAY
    KK=K
    IF(IOFLG(K,2).EQ.0) GO TO 69
    IF(IDDNFM.LT.0) CALL ULAPRS(BUFF(1,1,K),TEXT(1),KSTP,KPER,
                               NCOL,NROW,KK,-IDDNFM,IOUT)
    IF(IDDNFM.GE.0) CALL ULAPRN(BUFF(1,1,K),TEXT(1),KSTP,KPER,
                               NCOL,NROW,KK,IDDNFM,IOUT)
    IPFLG=1
69 CONTINUE
C
C7-------FOR EACH LAYER: DETERMINE IF DRAWDOWN SHOULD BE RECORDED.
C7-------IF SO THEN CALL ULASAV OR ULASAVARC TO RECORD DRAWDOWN.
    IFIRST=1
    QFILE=.TRUE.
    IF(IDDNUN.LE.0) GO TO 80
    DO 79 K=1,NLAY
        KK=K
        IF(IOFLG(K,4).LE.0) GO TO 79
        IF(IFIRST.EQ.1) THEN
            IF(IDDNFL.GE.0) WRITE(IOUT,74) IDDNUN,KSTP,KPER
            IF(IDDNFL.LT.0) WRITE(IOUT,75) DDNPATH (:INDEX(DDNPATH,' ')-1),
                          KPER,KSTP,KSTP,KPER
        ENDIF
        IFIRST=0
    74 FORMAT(1H0,'DRAWDOWN WILL BE SAVED ON UNIT',I3,
               ' AT END OF TIME STEP',I3,' STRESS PERIOD',I3)
    75 FORMAT(1H0,'DRAWDOWN SAVED IN ARC/INFO FILE ',A,'DRAWDOWN_','
               I3,'_',I3,' AT END OF TIME STEP',I3,' STRESS PERIOD',I3)
   69 CONTINUE
C
C7A-------IF IDDNFL>0 THEN CALL ULASAV TO RECORD DRAWDOWN IN UNFORMATTED FILE.
C
    IF(IDDNFL.GE.0) THEN
        CALL ULASAV(BUFF(1,1,K),TEXT(1),KSTP,KPER,
                   PERTIM,TOTIM,NCOL,NROW,KK,IDDNUM)
    ENDIF
C
C7B-------IF IDDNFL<0 THEN CALL ULASAVARC TO RECORD DRAWDOWN IN ARC/INFO FILE.
C
    ELSE
        INFOITEM='LAYER'
        INFONAME='DDNBUD'
        OUTPATH=DDNPATH (:INDEX(DDNPATH,' ')-1)//INFONAME
        CALL ULASAVARC (BUFF(1,1,K),INFOITEM,KSTP,KPER,
                         NCOL,NROW,KK,OUTPATH,QFILE,NLAY,IOFLG,
                         +9999)
    ENDIF
C
C8------FOR EACH LAYER: DETERMINE IF DRAWDOWN SHOULD BE RECORDED.
C8------IF SO THEN CALL ULASAV OR ULASAVARC TO RECORD DRAWDOWN.
    IF(IOFLG(K,4).EQ.0) GO TO 89
    IF(IDDNFL.LT.0) CALL ULASAVARC (BUFF(1,1,K),INFOITEM,KSTP,KPER,
                                     NCOL,NROW,KK,OUTPATH,QFILE,NLAY,IOFLG,
                                     *9999)
    ENDIF
C
QFILE=.FALSE.
ENDIF
79 CONTINUE
C
C8------RETURN
80 RETURN
C
CE------ERRORS
C
9999 CALL INFORM (‘\Abnormal Termination of Sbas1d_Arc_Subroutine’,-1)
RETURN 1
END

Added variables for module SBAS1DARC

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>DDNPATH</td>
<td>Package</td>
<td>The directory path to the ARC/INFO subdirectory where the values for the drawdown array are recorded (passed argument from module BAS1OTARC).</td>
</tr>
<tr>
<td>IDDNFL</td>
<td>Package</td>
<td>The drawdown flag that indicates whether the array values will be printed or recorded.</td>
</tr>
<tr>
<td>INFONAME</td>
<td>Submodule</td>
<td>The name of the ARC/INFO file where the values for the drawdown array are recorded (passed argument consisting of the root name DDNBUD which later will be appended by the stress period and time step).</td>
</tr>
<tr>
<td>INFOITEM</td>
<td>Submodule</td>
<td>The name of the ARC/INFO or INFO item where the values for the drawdown array are recorded (passed argument consisting of the root name LAYER which later will be appended by the layer number).</td>
</tr>
<tr>
<td>OUTPATH</td>
<td>Submodule</td>
<td>The path for the ARC/INFO file where the values for the drawdown array are recorded.</td>
</tr>
<tr>
<td>QFILE</td>
<td>Submodule</td>
<td>Logical flag indicating whether ARC/INFO file for drawdown values has been created (TRUE, create file; FALSE, do not create file).</td>
</tr>
</tbody>
</table>

SBAS1HARC

This submodule directs the output procedures for head within the BASIC package and is called by module BAS1OTARC. Six new variables were added to the program code: IHDDFL, OUTPATH, HEDPATH, INFONAME, INFOITEM, and QFILE. The major change consisted of replacing program code to direct storage of the head values in either unformatted files, ARC/INFO files, or INFO files. When the flag IHDDFL has been set to a negative value, submodule SBAS1DARC calls the utility module ULASAVARC (fig.5). Module ULASAVARC writes the head array into an ARC/INFO file for each layer that has been user specified. Documentation of the submodule follows.
Figure 5.—Modified program elements for the SBAS1HARC module.

```fortran
SUBROUTINE SBAS1HARC (HNEW, BUFF, IOFLG, KSTP, KPER, NCOL, NROW, NLAY,
&                    IOUT, IHEDFM, IHEDUN, IHDDFL, HEDPATH, IPFLG, PERTIM, TOTIM,
&                    E )
C-----VERSION 3.0 25OCTOBER1991 SBAS1HARC
C               MODIFIED BY LEONARD L. ORZOL
C
C     *******************************************************
C     PRINT AND RECORD HEADS
C     *******************************************************
C
C     SPECIFICATIONS
C     -------------------------------------------------------
DOUBLE PRECISION HNEW
DIMENSION HNEW(NCOL, NROW, NLAY), IOFLG(NLAY, 4), TEXT(4),
1   BUFF(NCOL, NROW, NLAY)
CHARACTER*128 OUTPATH
CHARACTER*80 HEDPATH
CHARACTER*32 INFONAME
CHARACTER*16 INFOITEM
CHARACTER*4 TEXT
LOGICAL QFILE

C     -------------------------------------------------------

DATA TEXT(1), TEXT(2), TEXT(3), TEXT(4) /'    ','    ','    ',
1       'HEAD'/

C     -------------------------------------------------------
C1------FOR EACH LAYER: PRINT HEAD IF REQUESTED.
      DO 39 K=1, NLAY
      KK=K

C2------TEST IOFLG TO SEE IF HEAD SHOULD BE PRINTED.
      IF(IOFLG(K,1).EQ.0) GO TO 39
      IPFLG=1

C3------COPY HEADS FOR THIS LAYER INTO BUFFER.
      DO 32 I=1, NROW
      DO 32 J=1, NCOL
      BUFF(J, I, 1)=HNEW(J, I, K)
      32 CONTINUE
```

52
CALL UTILITY MODULE TO PRINT CONTENTS OF BUFFER.

IF (IHEDFM.LT.0) CALL ULAPRS(BUFF,TEXT(1),KSTP,KPER,NCOL,NROW,
   &                 KK,-IHEDFM,IOUT)
   IF (IHEDFM.GE.0) CALL ULAPRN(BUFF,TEXT(1),KSTP,KPER,NCOL,NROW,
   &                 KK,IHEDFM,IOUT)

39 CONTINUE

IF UNIT FOR RECORDING HEADS <= 0: THEN RETURN.

IF (IHEDUN.LE.0) GO TO 50
   QFILE=.TRUE.

FOR EACH LAYER: RECORD HEAD IF REQUESTED.

DO 49 K=1,NLAY
   KK=K

CHECK IOFLG TO SEE IF HEAD FOR THIS LAYER SHOULD BE RECORDED.

   IF (IOFLG(K,3).LE.0) GO TO 49
   IF (IFIRST.EQ.1) THEN
      IF (IHDDFL.GE.0) WRITE(IOUT,41) IHEDUN,KSTP,KPER
      41       FORMAT(1H0,'HEAD WILL BE SAVED ON UNIT',
                       &                I3,' AT END OF TIME STEP',I3,'
                       &                      STRESS PERIOD',I3)
      IF (IHDDFL.LT.0) WRITE(IOUT,43) HEDPATH (:INDEX(HEDPATH,' ')-1),
      43       FORMAT(1H0,'HEADS SAVED WITHIN ARC/INFO FILE ',A,'
                       &                       HEAD_',I3,
                       &                      '_',I3,' AT END OF TIME STEP ',I3,'
                       &                      STRESS PERIOD',I3)
   ENDIF
   IFIRST=0

COPY HEADS FOR THIS LAYER INTO BUFFER.

   DO 44 I=1,NROW
      DO 44 J=1,NCOL
         BUFF(J,I,1)=HNEW(J,I,K)
   44 CONTINUE

RECORD CONTENTS OF BUFFER ON UNIT=IHEDUN

   IF (IHDDFL.GE.0) THEN
      CALL ULASAV(BUFF,TEXT(1),KSTP,KPER,PERTIM,TOTIM,NCOL,
      &             NROW,KK,IHEDUN)
   ELSE
      RECORD CONTENTS OF BUFFER ON HEDPATH ARC/INFO FILE
      INFOITEM='LAYER'
      INFONAME='HEDBUD'
      OUTPATH=HEDPATH (:INDEX(HEDPATH,' ')-1)//INFONAME
      CALL ULASAVARC (BUFF,INFOITEM,KSTP,KPER,
      &                     NCOL,NROW,KK,OUTPATH,QFILE,NLAY,IOFLG,
      E                     *9999)
   ENDIF
49 CONTINUE

RETURN

RETURN 1

CALL INFORM ('\Abnormal Termination of Sbas1h_Arc_Subroutine',-1)

END
Added variables for module SBAS1HARC

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>HEDPATH</td>
<td>Package</td>
<td>The directory path the ARC/INFO subdirectory where the values for the head array are recorded (passed argument from module BAS1OTARC).</td>
</tr>
<tr>
<td>IHDDFL</td>
<td>Package</td>
<td>The head flag that indicates whether the array values will be printed or recorded.</td>
</tr>
<tr>
<td>INFONAME</td>
<td>Submodule</td>
<td>The name of the ARC/INFO file where the values for the head array are recorded (passed argument consisting of the root name HEDBUD which later will be appended by the stress period and time step).</td>
</tr>
<tr>
<td>INFOITEM</td>
<td>Submodule</td>
<td>The name of the ARC/INFO or INFO item where the values for the head array are recorded (passed argument consisting of the root name LAYER which later will be appended by the layer number).</td>
</tr>
<tr>
<td>OUTPATH</td>
<td>Submodule</td>
<td>The path for the ARC/INFO file where the values for the head array are recorded.</td>
</tr>
<tr>
<td>QFILE</td>
<td>Submodule</td>
<td>Logical flag indicating whether ARC/INFO file for head values has been created (TRUE, create file; FALSE, do not create file).</td>
</tr>
</tbody>
</table>

This submodule initializes the output control system within the BASIC package and is called by module BAS1ORPARC. Two new variables were added to the program code: OUTPATH and BUFFER. The major change consists of replacing program code to direct this submodule to read BUFFER (the control record for output control). The variable OUTPATH is read from BUFFER when present. If OUTPATH is present, OUTPATH is the path for ARC/INFO files or INFO files where drawdown, head, and cell-by-cell flow-term array values are recorded. The changes are marked 2A and 2B in figure 6. Documentation of the program code follows.

![Diagram](attachment:image.png)

Figure 6.--Modified program elements for the SBAS1IARC module.
SUBROUTINE SBAS1IARC (NLAY, ISTRT, IOFLG, INOC, IOUT, IHEDFM,
                     IDDNFM, IHEDUN, IDDNUN, OUTPATH,
                     E)

C-----VERSION 3.0 25OCTOBER1991 SBAS1IARC
C MODIFIED BY LEONARD L. ORZOL
C
C************************************************************************
C SET UP OUTPUT CONTROL
C************************************************************************
C SPECIFICATIONS:
C************************************************************************
DIMENSION IOFLG(NLAY, 4)
CHARACTER*132 BUFFER
CHARACTER*80 OUTPATH
C************************************************************************
C1------TEST UNIT NUMBER FROM IUNIT (INOC) TO SEE IF OUTPUT
C1------CONTROL IS ACTIVE.
IF(INOC.LE.0) GO TO 600
C
C2------READ AND PRINT FORMATS FOR PRINTING AND UNIT NUMBERS FOR
C2------RECORDING HEADS AND DRAWDOWN. THEN RETURN.
C
500  READ(INOC,'(A132)',ERR=9990,END=9991) BUFFER
C2A------OUTPUT DIRECTED TO ASCII FILES
C IF(BUFFER (41:41).EQ.'' .OR. BUFFER (41:41).EQ.' ') THEN
READ (BUFFER,'(4I10)',ERR=9992) IHEDFM, IDDNFM, IHEDUN, IDDNUN
WRITE (IOUT,501) IHEDFM, IDDNFM
501    FORMAT (1H0,'HEAD PRINT FORMAT IS FORMAT NUMBER',I4,
                  &                      ' DRAWDOWN PRINT FORMAT IS FORMAT NUMBER',I4)
WRITE (IOUT,503) IHEDUN, IDDNUN
503    FORMAT (1H0,'HEADS WILL BE SAVED ON UNIT',I3,
                  &                      ' DRAWDOWNS WILL BE SAVED ON UNIT',I3)
C2B------OUTPUT DIRECTED TO ARC/INFO FILES
C ELSE
READ (BUFFER,'(4I10,A80)',ERR=9992)
&                          IHEDFM, IDDNFM, IHEDUN, IDDNUN, OUTPATH
WRITE(IOUT,501) IHEDFM, IDDNFM
WRITE (IOUT,505) OUTPATH (:INDEX(OUTPATH,' ')-1)
505    FORMAT (1H0,       'HEADS AND DRAWDOWN SAVED IN INFO FILES IN INFO DIRECTORY ',A)
ENDIF
C2C------TIME STEP DECLARATION FOR OUTPUT CONTROL
C WRITE(IOUT,507)
507    FORMAT(1H0,'OUTPUT CONTROL IS SPECIFIED EVERY TIME STEP')
GO TO 1000
C
C3------OUTPUT CONTROL IS INACTIVE. PRINT A MESSAGE LISTING DEFAULTS.
600   WRITE(IOUT,641)
641    FORMAT(1H0,'DEFAULT OUTPUT CONTROL -- THE FOLLOWING OUTPUT',
                  &                          ' COMES AT THE END OF EACH STRESS PERIOD:')
WRITE(IOUT,642)
642    FORMAT(1X,'TOTAL VOLUMETRIC BUDGET')
WRITE(IOUT,643)
643    FORMAT(1X,10X,'HEAD')
IF(ISTRT.NE.0)WRITE(IOUT,644)
644    FORMAT(1X,10X,'DRAWDOWN')
C
C4------SET THE FORMAT CODES EQUAL TO THE DEFAULT FORMAT.
IHEDFM=0
IDDNFM=0
C
C5------SET DEFAULT FLAGS IN IOFLG SO THAT HEAD (AND DRAWDOWN) IS
C5------PRINTED FOR EVERY LAYER.
ID=0
IF (ISTRT.NE.0) ID=1
670 DO 680 K=1,NLAY
  IOFLG(K,1)=1
  IOFLG(K,2)=ID
  IOFLG(K,3)=0
  IOFLG(K,4)=0
680 CONTINUE
GO TO 1000
C
C6------RETURN
1000  RETURN
C
CE------ERRORS
C
9990 CALL INFORM ("Unable to read Basic Output Control Package",-1)
   CALL INFORM ("Abnormal Termination of Sbasli_Arc_Subroutine",-1)
   RETURN 1
9991 CALL INFORM ("Missing input to Basic Output Control Package",-1)
   CALL INFORM ("Abnormal Termination of Sbasli_Arc_Subroutine",-1)
   RETURN 1
9992 CALL INFORM ("Unable to read IHEDFM, IDDNFM, IHEDUN, and " //
   "IDDNUN from output control line",-1)
   CALL INFORM ("Abnormal Termination of Sbasli_Arc_Subroutine",-1)
   RETURN 1
9993 CALL INFORM ("Unable to read IHEDFM, IDDNFM, IHEDUN, " //
   "IDDNUN, and OUTPATH from output control line",-1)
9999 CALL INFORM ("Abnormal Termination of Sbasli_Arc_Subroutine",-1)
   RETURN 1
END

Added variables for module SBAS1IARC

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>BUFFER</td>
<td>Module</td>
<td>The control record is read into this variable and is parsed into the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>appropriate variables: IHEDFM, IDDNFM, IHEDUN,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>IDDNUN, and OUTPATH.</td>
</tr>
<tr>
<td>OUTPATH</td>
<td>Package</td>
<td>The directory path to the ARC/INFO subdirectory where output</td>
</tr>
<tr>
<td></td>
<td></td>
<td>head and drawdown arrays are recorded.</td>
</tr>
</tbody>
</table>

Block-centered flow Package Modules

The BCF (Block-centered flow) package consists of four primary modules and five submodules; of these, two primary modules (BCF1RPARC and BCF1BDARC) and two submodules (SBCF1BARC and SBCF1FARC) indicated below were modified.

BCF1RPARC

This module reads and prepares data for the BCF package and calls submodules U1RELARC and U2DRELARC (fig. 7). Documentation of the modified code follows.
Figure 7.--Modified program elements for the BCF1RPARC module.
SUBROUTINE BCF1RPARC (IBOUND, HNEW, SC1, HY, CR, CC, CV, DELR, DELC, 
&                  BOT, TOP, SC2, TRPY, IN, ISS, NCOL, NROW, NLAY, NODES, IOUT, 
&                  E
*)

C-------VERSION 3.0 25OCTOBER1991 BCF1RPARC
C                MODIFIED BY LEONARD L. ORZOL
C
C     ******************************************************************
C     READ AND INITIALIZE DATA FOR BLOCK-CENTERED FLOW PACKAGE
C     ******************************************************************
C
C        SPECIFICATIONS:
C        ------------------------------------------------------------------
CHARACTER*4 ANAME
CHARACTER*16 INFOITEM
DOUBLE PRECISION HNEW

DIMENSION HNEW(NODES), SC1(NODES), HY(NODES), CR(NODES), CC(NODES), 
1 CV(NODES), ANAME(6,10), DELR(NCOL), DELC(NROW), BOT(NODES), 
1 TOP(NODES), SC2(NODES), TRPY(NLAY), IBOUND(NODES)

COMMON /FLWCOM/LAYCON(80)

DATA ANAME(1,1),ANAME(2,1),ANAME(3,1),ANAME(4,1),ANAME(5,1), 
1 ANAME(6,1) /'    ','PRIM','ARY ','STOR','AGE ','COEF'/
DATA ANAME(1,2),ANAME(2,2),ANAME(3,2),ANAME(4,2),ANAME(5,2), 
1 ANAME(6,2) /'    ','TRAN','SMIS','. AL','ONG ','ROWS'/
DATA ANAME(1,3),ANAME(2,3),ANAME(3,3),ANAME(4,3),ANAME(5,3), 
1 ANAME(6,3) /'    ','H','YD' ,',COND' ,'. AL' ,',ONG' ,',ROWS'/
DATA ANAME(1,4),ANAME(2,4),ANAME(3,4),ANAME(4,4),ANAME(5,4), 
1 ANAME(6,4) /'    ','VERT',' HYD' ,', CON' ,',D /T',', HICK',', NESS'/
DATA ANAME(1,5),ANAME(2,5),ANAME(3,5),ANAME(4,5),ANAME(5,5), 
1 ANAME(6,5) /'    ','    ','    ','    ','  BO','TTOM'/
DATA ANAME(1,6),ANAME(2,6),ANAME(3,6),ANAME(4,6),ANAME(5,6), 
1 ANAME(6,6) /'    ','    ','    ','    ','    ',' TOP'/
DATA ANAME(1,7),ANAME(2,7),ANAME(3,7),ANAME(4,7),ANAME(5,7), 
1 ANAME(6,7) /'    ','    ','    ','    ','    ','    '/
DATA ANAME(1,8),ANAME(2,8),ANAME(3,8),ANAME(4,8),ANAME(5,8), 
1 ANAME(6,8) /'    ','    ','    ','    ','    ','  DELR'/
DATA ANAME(1,9),ANAME(2,9),ANAME(3,9),ANAME(4,9),ANAME(5,9), 
1 ANAME(6,9) /'    ','    ','    ','    ','    ','  DELC'/
DATA ANAME(1,10),ANAME(2,10),ANAME(3,10),ANAME(4,10),ANAME(5,10), 
1 ANAME(6,10) /'    ','    ','    ','    ','    '/

C1------CALCULATE NUMBER OF NODES IN A LAYER AND READ TRPY,DELR,DELC
NIJ=NCOL*NROW

INFOITEM='TRPY'
CALL U1DRELARC (INFOITEM,TRPY,ANAME(1,8),NLAY,IN,IOUT,*9999)
INFOITEM='DELR'
CALL U1DRELARC (INFOITEM,DELR,ANAME(1,9),NCOL,IN,IOUT,*9999)
INFOITEM='DELC'
CALL U1DRELARC (INFOITEM,DELC,ANAME(1,10),NROW,IN,IOUT,*9999)

C2------READ ALL PARAMETERS FOR EACH LAYER
KT=0
KB=0
DO 200 K=1,NLAY
   KK=K

C2A------FIND ADDRESS OF EACH LAYER IN THREE DIMENSION ARRAYS.
   IF(LAYCON(K).EQ.1 .OR. LAYCON(K).EQ.3) KB=KB+1
   IF(LAYCON(K).EQ.2 .OR. LAYCON(K).EQ.3) KT=KT+1
   LOC=1+(K-1)*NIJ
   LOCB=1+(KB-1)*NIJ
   LOCT=1+(KT-1)*NIJ

C2B------READ PRIMARY STORAGE COEFFICIENT INTO ARRAY SC1 IF TRANSIENT
INFOITEMS='SF1'
   IF(ISS.EQ.0) CALL U2DRELARC (INFOITEM,SC1(LOC),ANAME(1,1),NROW, 
&                  NCOL, KK, IN, IOUT, &
E                             *9999)

C2C------READ TRANSMISSIVITY INTO ARRAY CC IF LAYER TYPE IS 0 OR 2
KT=0
KB=0
DO 200 K=1,NLAY
   KK=K
C
C2A------FIND ADDRESS OF EACH LAYER IN THREE DIMENSION ARRAYS.
   IF(LAYCON(K).EQ.1 .OR. LAYCON(K).EQ.3) KB=KB+1
   IF(LAYCON(K).EQ.2 .OR. LAYCON(K).EQ.3) KT=KT+1
   LOC=1+(K-1)*NIJ
   LOCB=1+(KB-1)*NIJ
   LOCT=1+(KT-1)*NIJ
C
C2B------READ PRIMARY STORAGE COEFFICIENT INTO ARRAY SC1 IF TRANSIENT
   INFOITEM='SF1'
   IF(ISS.EQ.0) CALL U2DRELARC (INFOITEM,SC1(LOC),ANAME(1,1),NROW,
   &                     NCOL,KK,IN,IOUT,
   E                   *9999)
C
C2C------READ TRANSMISSIVITY INTO ARRAY CC IF LAYER TYPE IS 0 OR 2
   IF(LAYCON(K).EQ.3 .OR. LAYCON(K).EQ.1) GO TO 100
   INFOITEM='TRAN'
   CALL U2DRELARC (INFOITEM,CC(LOC),ANAME(1,2),NROW,NCOL,KK,IN,
   &                 IOUT,
   E                   *9999)
   GO TO 110
C
C2D------READ HYDRAULIC CONDUCTIVITY(HY) AND BOTTOM ELEVATION(BOT)
C2D------IF LAYER TYPE IS 1 OR 3
100   INFOITEM='HY'
   CALL U2DRELARC (INFOITEM,Hy(LOCB),ANAME(1,3),NROW,NCOL,KK,IN,
   &                  IOUT,
   E                   *9999)
   INFOITEM='BOT'
   CALL U2DRELARC (INFOITEM,BOT(LOCB),ANAME(1,5),NROW,NCOL,KK,IN,
   &                 IOUT,
   E                   *9999)
C
C2E------READ VERTICAL HYCOND/THICK INTO ARRAY CV IF NOT BOTTOM LAYER
C2E------READ AS HYCOND/THICKNESS -- CONVERTED TO CONDUCTANCE LATER
110   IF(K.EQ.NLAY) GO TO 120
   INFOITEM='VCONT'
   CALL U2DRELARC (INFOITEM,CV(LOC),ANAME(1,4),NROW,NCOL,KK,IN,
   &                  IOUT,
   E                   *9999)
C
C2F------READ SECONDARY STORAGE COEFFICIENT INTO ARRAY SC2 IF TRANSIENT
C2F------AND LAYER TYPE IS 2 OR 3
120   IF(LAYCON(K).NE.3 .AND. LAYCON(K).NE.2) GO TO 200
   IF(ISS.EQ.0) THEN
      INFOITEM='SF2'
      CALL U2DRELARC (INFOITEM,SC2(LOCT),ANAME(1,7),NROW,NCOL,KK,IN,
      &                 IOUT,
      E                   *9999)
   ENDIF
C
C2G------READ TOP ELEVATION(TOP) IF LAYER TYPE IS 2 OR 3
   INFOITEM='TOP'
   CALL U2DRELARC (INFOITEM,TOP(LOCT),ANAME(1,6),NROW,NCOL,KK,IN,
   &                  IOUT,
   E                   *9999)
   200 CONTINUE
C
C3------PREPARE AND CHECK BCF DATA
   CALL SBCF1N(HNEW,IBOUND,SC1,SC2,CR,CC,CV,TRPY,DELR,DELC,ISS,
   1                     NCOL,NCOL,NROW,NLAY,IOUT)
C
C4------RETURN
   RETURN
Added variables for module BCF1RPARC

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>INFOITEM</td>
<td>Module</td>
<td>The name of the INFO item either primary or redefined within the ARC/INFO file containing the information.</td>
</tr>
</tbody>
</table>

**BCF1BDARC**

This module calculates cell-by-cell flow terms and overall volumetric for the BCF package calls submodules UBUDSVARC, SBCF1FARC and SBCF1BARC (fig. 8). Documentation of the module changes follows.

---

Figure 8.—Modified program elements for the BCF1BDARC module.
SUBROUTINE BCF1BDARC (VBNM, VBVL, MSUM, HNEW, IBOUND, HOLD, SC1, CR, CC,
&                  CV, TOP, SC2, DELT, ISS, NCOL, NROW, NLAY, KSTP, KPER, IBCFCB,
&                  ICBCFL, BUFF, IOUT, BCFPATH,
E                      *)
C-----VERSION 3.0 25OCTOBER1991 BCF1BDARC
C               MODIFIED BY LEONARD L. ORZOL
C
C     ******************************************************************
C     COMPUTE BUDGET FLOW TERMS FOR BCF -- STORAGE, CONSTANT HEAD, AND
C     FLOW ACROSS CELL WALLS
C     ******************************************************************
C
C     SPECIFICATIONS:
C     *------------------------------------------------------------------
CHARACTER*128 OUTPATH
CHARACTER*80 BCFPATH
CHARACTER*32 INFONAME
CHARACTER*16 INFOITEM
CHARACTER*4 VBNM, TEXT
DOUBLE PRECISION HNEW
LOGICAL QFILE

DIMENSION HNEW(NCOL,NROW, NLAY), IBOUND(NCOL,NROW, NLAY),
1   HOLD(NCOL,NROW, NLAY), SC1(NCOL, NROW, NLAY),
2   CR(NCOL,NROW, NLAY), CC(NCOL,NROW, NLAY),
3   CV(NCOL,NROW, NLAY), VBNM(4,20), VBVL(4,20),
4   SC2(NCOL,NROW, NLAY),
5   TOP(NCOL,NROW, NLAY), BUFF(NCOL,NROW, NLAY)

COMMON /FLWCOM/LAYCON(80)

DIMENSION TEXT(4)

DATA TEXT(1),TEXT(2),TEXT(3),TEXT(4) /'    ','    ',' STO','RAGE'/

C     *------------------------------------------------------------------
C
C1------INITIALIZE BUDGET ACCUMULATORS
STOIN=0.
STOUT=0.

C2------IF CELL-BY-CELL FLOWS ARE NEEDED THEN SET FLAG IBD.
IBD=0
IF(ICBCFL.NE.0 .AND. IBCFCB.GT.0) IBD=1

C3------IF STEADY STATE THEN SKIP ALL STORAGE CALCULATIONS
IF(ISS.NE.0) GO TO 305

C4------IF CELL-BY-CELL FLOWS ARE NEEDED (IBD IS SET) CLEAR BUFFER
IF(IBD.EQ.0) GO TO 220
DO 210 K=1, NLAY
DO 210 I=1, NROW
DO 210 J=1, NCOL
BUFF(J,I,K)=0.
210 CONTINUE

C5------RUN THROUGH EVERY CELL IN THE GRID
220 KT=0
DO 300 K=1, NLAY
LC=LAYCON(K)
IF(LC.EQ.3 .OR. LC.EQ.2) KT=KT+1
DO 300 I=1, NROW
DO 300 J=1, NCOL
300 CONTINUE

C6------CALCULATE FLOW FROM STORAGE (VARIABLE HEAD CELLS ONLY)
IF(BOUND(J,I,K).LE.0) GO TO 300
HSING=HNEW(J,I,K)

C6A------CHECK LAYER TYPE TO SEE IF ONE STORAGE CAPACITY OR TWO
IF(LC.NE.3 .AND. LC.NE.2) GO TO 285

C6B------TWO STORAGE CAPACITIES
TP=TOP(J,I,KT)
SYA = SC2(J, I, KT)
SCFA = SC1(J, I, K)
SOLD = SYA
IF (HOLD(J, I, K) .GT. TP) SOLD = SCFA
SNEW = SYA
IF (HSING .GT. TP) SNEW = SCFA
STRG = SOLD * (HOLD(J, I, K) - TP) + SNEW * TP - SNEW * HSING
GO TO 288
C
C6------ONE STORAGE CAPACITY
285 SC = SC1(J, I, K)
      STRG = SC * HOLD(J, I, K) - SC * HSING
C
C7------STORE CELL-BY-CELL FLOW IN BUFFER AND ADD TO ACCUMULATORS
288 IF (IBD .EQ. 1) BUFF(J, I, K) = STRG / DELT
      IF (STRG) 292, 300, 294
292 STOUT = STOUT - STRG
      GO TO 300
294 STOIN = STOIN + STRG
C
300 CONTINUE
C
C8------IF IBD FLAG IS SET RECORD THE CONTENTS OF THE BUFFER
C
C8A------SET RECORD THE CONTENTS OF THE BUFFER IN UNIFORMED FILE
      IF (IBD .EQ. 1 .AND. ICBCFL .GE. 0) CALL UBUDSV(KSTP, KPER, TEXT,
      & IBCFCB, BUFF, NCOL, NROW, NLAY, IOUT)
C
C8B------SET RECORD THE CONTENTS OF THE BUFFER IN ARC INFO FILE
      IF (IBD .EQ. 1 .AND. ICBCFL .LT. 0) THEN
          INFOITEM = 'LAYER'
          INFONAME = 'STOBUD'
          OUTPATH = BCFPATH (:INDEX(BCFPATH, ' ') - 1) // INFONAME
          CALL UBUDSVARC(KSTP, KPER, INFOITEM, OUTPATH, BUFF, NCOL, NROW,
          & NLAY, IOUT, *9999)
      ENDIF
C
C9------ADD TOTAL RATES AND VOLUMES TO VBVL & PUT TITLES IN VBNM
305 VBVL(1, MSUM) = VBVL(1, MSUM) + STOIN
      VBVL(2, MSUM) = VBVL(2, MSUM) + STOUT
      VBVL(3, MSUM) = STOIN / DELT
      VBVL(4, MSUM) = STOUT / DELT
      VBNM(1, MSUM) = TEXT(1)
      VBNM(2, MSUM) = TEXT(2)
      VBNM(3, MSUM) = TEXT(3)
      VBNM(4, MSUM) = TEXT(4)
      MSUM = MSUM + 1
C
C10------CALCULATE FLOW FROM CONSTANT HEAD NODES
C
      CALL SBCF1FARC(VBNM, VBVL, MSUM, HNEW, IBOUND, CR, CC, CV, TOP, DELT,
      & NCOL, NROW, NLAY, KSTP, KPER, IBD, IBCFCB, ICBCFL, BUFF, IOUT,
      & OUTPATH, *9999)
C
C11------CALCULATE AND SAVE FLOW ACROSS CELL BOUNDARIES IF C-B-C
C11------FLOW TERMS ARE REQUESTED.
C
      IF (IBD .NE. 0) CALL SBCF1BARC(HNEW, IBOUND, CR, CC, CV, TOP,
      & NCOL, NROW, NLAY, KSTP, KPER, IBCFCB, ICBCFL, BUFF, IOUT,
      & OUTPATH, *9999)
C
C12------RETURN
      RETURN
C
CE------ERRORS
C
9999 CALL INFORM ('\Abnormal Termination of Bcf1bd_Arc_Subroutine', -1)
      RETURN 1
END
Added variables for module BCF1BDARC

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>BCFPATH</td>
<td>Package</td>
<td>The directory path to the ARC/INFO subdirectory where cell-by-cell flow terms are recorded.</td>
</tr>
<tr>
<td>INFONAME</td>
<td>Module</td>
<td>The name of the ARC/INFO file where cell-by-cell flow terms array are recorded (passed argument consisting of the root name CHDBUD, FRFBUD, FFFBUD, or FLFBUD which later will be appended by the stress period and time step).</td>
</tr>
<tr>
<td>INFOITEM</td>
<td>Module</td>
<td>The name of the ARC/INFO or INFO item where cell-by-cell flow terms are recorded (passed argument consisting of the root name LAYER which later will be appended by the layer number).</td>
</tr>
<tr>
<td>OUTPATH</td>
<td>Module</td>
<td>The path for the ARC/INFO file where cell-by-cell flow terms are recorded.</td>
</tr>
</tbody>
</table>

**SBCF1FARC**

This submodule calculates flow terms from constant-head cells for the BCF package calls submodules UBUDSVARC or UBUDSV. The values for program variables, IBD and ICBCFL, determine which submodule is called (fig. 9). Documentation of the changes follow.

![Diagram](image-url)

Figure 9.--Modified program elements for the SBCF1FARC module.
SUBROUTINE SBCF1FARC (VBNM, VBVL, MSUM, HNEW, IBOUND, CR, CC, CV, TOP, &
   DELT, NCOL, NROW, NLAY, KSTP, KPER, IBD, IBCFCB, ICBCFL, BUFF, IOUT, &
   CHDPATH, *)
C-----VERSION 3.0 25OCTOBER1991 SBCF1FARC
C                  MODIFIED BY LEONARD L. ORZOL
C
C     ******************************************************************
C     COMPUTE FLOW FROM CONSTANT HEAD NODES
C     ******************************************************************
C
C     SPECIFICATIONS:
C     ----------------------------------------------------------------
CHARACTER*128 OUTPATH
CHARACTER*80 CHDPATH
CHARACTER*32 INFONAME
CHARACTER*16 INFOITEM
CHARACTER*4 VBNM, TEXT
DOUBLE PRECISION HNEW, HD
LOGICAL QFILE
C
DIMENSION HNEW(NCOL,NROW,NLAY), IBOUND(NCOL,NROW,NLAY),
1     CR(NCOL,NROW,NLAY), CC(NCOL,NROW,NLAY),
2     CV(NCOL,NROW,NLAY), VBNM(4,20), VBVL(4,20),
3     TOP(NCOL,NROW,NLAY),BUFF(NCOL,NROW,NLAY)
C
COMMON /FLWCOM/LAYCON(80)
C
DIMENSION TEXT(4)
C
DATA TEXT(1),TEXT(2),TEXT(3),TEXT(4) /'   C','ONST','ANT ','HEAD'/
C     ----------------------------------------------------------------
C
C1------CLEAR BUDGET ACCUMULATORS
CHIN=0.
CHOUT=0.
C
C2------CLEAR BUFFER IF CELL-BY-CELL FLOW TERM FLAG(IBD) IS SET
IF(IBD.EQ.0) GO TO 8
DO 5 K=1,NLAY
   DO 5 I=1,NROW
      DO 5 J=1,NCOL
         BUFF(J,I,K)=0.
      5 CONTINUE
C
C3------FOR EACH CELL IF IT IS CONSTANT HEAD COMPUTE FLOW ACROSS 6
C3-----FACES.
C
8 KT=0
   DO 200 K=1,NLAY
      LC=LAYCON(K)
      IF(LC.EQ.3 .OR. LC.EQ.2) KT=KT+1
      DO 200 I=1,NROW
         DO 200 J=1,NCOL
            BUFF(J,I,K)=0.
       200 CONTINUE
C
C4------IF CELL IS NOT CONSTANT HEAD SKIP IT & GO ON TO NEXT CELL.
   IF (IBOUND(J,I,K).GE.0) GO TO 200
C
C5------CLEAR FIELDS FOR SIX FLOW RATES.
   X1=0.
   X2=0.
   X3=0.
   X4=0.
   X5=0.
   X6=0.
C
C6------FOR EACH FACE OF THE CELL CALCULATE FLOW THROUGH THAT FACE
C6-----OUT OF THE CONSTANT HEAD CELL AND INTO THE FLOW DOMAIN.
C6-----COMMENTS 7-11 APPEAR ONLY IN THE SECTION HEADED BY COMMENT 6A
C6-----BUT THEY APPLY IN A SIMILAR MANNER TO THE SECTIONS HEADED
C6-----BY COMMENTS 6B-6F.
C
C6A------CALCULATE FLOW THROUGH THE LEFT FACE
C
C7-----IF THERE IS NOT A VARIABLE HEAD CELL ON THE OTHER SIDE OF THIS
C7-----FACE THEN GO ON TO THE NEXT FACE.
   IF(J.EQ.1) GO TO 30
   IF(IBOUND(J-1,I,K).LE.0)GO TO 30
   HDIFF=HNEW(J,I,K)-HNEW(J-1,I,K)
C
C8-----CALCULATE FLOW THROUGH THIS FACE INTO THE ADJACENT CELL.
   X1=HDIFF*CR(J-1,I,K)
C
C9-----TEST TO SEE IF FLOW IS POSITIVE OR NEGATIVE
   IF (X1) 10,30,20
C
C10-----IF NEGATIVE ADD TO CHOUT(FLOW OUT OF DOMAIN TO CONSTANT HEAD).
   10 CHOUT=CHOUT-X1
       GO TO 30
C
C11-----IF POSITIVE ADD TO CHIN(FLOW INTO DOMAIN FROM CONSTANT HEAD).
   20 CHIN=CHIN+X1
C
C6B----CALCULATE FLOW THROUGH THE RIGHT FACE
   30 IF(J.EQ.NCOL) GO TO 60
   IF(IBOUND(J+1,I,K).LE.0) GO TO 60
   HDIFF=HNEW(J,I,K)-HNEW(J+1,I,K)
   X2=HDIFF*CR(J,I,K)
   IF(X2)40,60,50
   40 CHOUT=CHOUT-X2
       GO TO 60
   50 CHIN=CHIN+X2
C
C6C----CALCULATE FLOW THROUGH THE BACK FACE.
   60 IF(I.EQ.1) GO TO 90
   IF (IBOUND(J,I-1,K).LE.0) GO TO 90
   HDIFF=HNEW(J,I,K)-HNEW(J,I-1,K)
   X3=HDIFF*CC(J,I-1,K)
   IF(X3) 70,90,80
   70 CHOUT=CHOUT-X3
       GO TO 90
   80 CHIN=CHIN+X3
C
C6D----CALCULATE FLOW THROUGH THE FRONT FACE.
   90 IF(I.EQ.NROW) GO TO 120
   IF(IBOUND(J,I+1,K).LE.0) GO TO 120
   HDIFF=HNEW(J,I,K)-HNEW(J,I+1,K)
   X4=HDIFF*CC(J,I,K)
   IF (X4) 100,120,110
   100 CHOUT=CHOUT-X4
       GO TO 120
   110 CHIN=CHIN+X4
C
C6E----CALCULATE FLOW THROUGH THE UPPER FACE
   120 IF(K.EQ.1) GO TO 150
   IF (IBOUND(J,I,K-1).LE.0) GO TO 150
   HD=HNEW(J,I,K)
   IF(LC.NE.3 .AND. LC.NE.2) GO TO 122
   TMP=HD
   IF(TMP.LT.TOP(J,I,KT)) HD=TOP(J,I,KT)
   122 HDIFF=HD-HNEW(J,I,K-1)
   X5=HDIFF*CV(J,I,K-1)
   IF (X5) 130,150,140
   130 CHOUT=CHOUT-X5
       GO TO 150
   140 CHIN=CHIN+X5
C
C6F----CALCULATE FLOW THROUGH THE LOWER FACE.
   150 IF(K.EQ.NLAY) GO TO 180
   IF(IBOUND(J,I,K+1).LE.0) GO TO 180
   HD=HNEW(J,I,K+1)
   IF(LAYCON(K+1).NE.3 .AND. LAYCON(K+1).NE.2) GO TO 152
   TMP=HD
   IF(TMP.LT.TOP(J,I,KT+1)) HD=TOP(J,I,KT+1)
   152 HDIFF=HNEW(J,I,K)-HD
   X6=HDIFF*CV(J,I,K)
IF(X6) 160,180,170
160 CHOUT=CHOUT-X6
GO TO 180
170 CHIN=CHIN+X6
C
C12-----SUM UP FLOWS THROUGH SIX SIDES OF CONSTANT HEAD CELL.
180 RATE=X1+X2+X3+X4+X5+X6
C
C13-----PRINT THE INDIVIDUAL RATES IF REQUESTED(IBCFCB<0).
     IF(IBCFCB.LT.0.AND.ICBCFL.NE.0) WRITE(IOUT,900) (TEXT(N),N=1,4),
1     KPER,KSTP,K,I,J,RATE
900 FORMAT(1H0,4A4,' PERIOD',I3,' STEP',I3,' LAYER',I3,
1    ' ROW',I4,' COL',I4,' RATE ',G15.7)
C
C14----IF CELL-BY-CELL FLAG SET STORE SUM OF FLOWS FOR CELL IN BUFFER
     IF(IBD.EQ.1) BUFF(J,I,K)=RATE
C
200 CONTINUE
C
C15----IF CELL-BY-CELL FLAG SET THEN RECORD CONTENTS OF BUFFER
C
C15A----RECORD CONTENTS OF BUFFER IN UNFORMATTED FILE
     IF(IBD.EQ.1 .AND. ICBCFL.GE.0) CALL UBUDSV(KSTP,KPER,TEXT(1),
 &     IBCFCB,BUFF,NCOL,NROW,NLAY,IOUT)
C
C15B----RECORD CONTENTS OF BUFFER IN ARC INFO FILE
     IF(IBD.EQ.1 .AND. ICBCFL.LT.0) THEN
       INFOITEM='LAYER'
       INFONAME='CHDBUD'
       OUTPATH=CHDPATH (:INDEX(CHDPATH,' ')-1)//INFONAME
       CALL UBUDSVARC (KSTP,KPER,INFOITEM,OUTPATH,BUFF,NCOL,NROW,NLAY,
 &                   IOUT,
 &                   *9999)
 ENDIF
C
C16-----SAVE TOTAL CONSTANT HEAD FLOWS AND VOLUMES IN VBVL TABLE
C16-----FOR INCLUSION IN BUDGET. PUT LABELS IN VBNM TABLE.
     VBVL(1,MSUM)=VBVL(1,MSUM)+CHIN*DELT
     VBVL(2,MSUM)=VBVL(2,MSUM)+CHOUT*DELT
     VBVL(3,MSUM)=CHIN
     VBVL(4,MSUM)=CHOUT
C
     ---SETUP VOLUMETRIC BUDGET NAMES
     VBNM(1,MSUM)=TEXT(1)
     VBNM(2,MSUM)=TEXT(2)
     VBNM(3,MSUM)=TEXT(3)
     VBNM(4,MSUM)=TEXT(4)
C
     MSUM=MSUM+1
C
C
C17-----RETURN
     RETURN
C
CE------ERRORS
8999 CALL INFORM ('\Abnormal Termination of Sbcf1f_Arc_Subroutine',-1)
     RETURN 1
     END

Added variables for module SBCF1FARC

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHDPATH</td>
<td>Module</td>
<td>The path for the ARC/INFO file where cell-by-cell flow terms are recorded (passed argument from module BCF1BDARC).</td>
</tr>
</tbody>
</table>
INFONAME Submodule The name of the ARC/INFO file where cell-by-cell flow terms are recorded (passed argument consisting of the root name CHDBUD which later will be appended by the stress period and time step).

INFOITEM Submodule The name of the ARC/INFO or INFO item where cell-by-cell flow terms are recorded (passed argument consisting of the root name LAYER which later will be appended by the layer number).

OUTPATH Submodule The path for the ARC/INFO file where cell-by-cell flow terms are recorded.

SBCF1BARC

This submodule calculates flow terms from cell-by-cell flow terms (RIGHTFACE, LOWERFACE, and FRONTFACE) for the BCF package and calls submodules UBUDSVARC or UBUDSV. The values for program variables, IBD and ICBCFL, determine which submodule is called (fig. 10). Documentation of the submodule follows.

Figure 10.—Modified program elements for the SBCF1BARC module.
SUBROUTINE SBCF1BARC (HNEW, IBOUND, CR, CC, CV, TOP, NCOL, NROW, NLAY, & KSTP, KPER, IBCFCB, ICBCFL, BUFF, IOUT, CBCPATH, E *)

C C-----VERSION 3.0 25OCTOBER1991 SBCF1BARC
C MODIFIED BY LEONARD L. ORZOL
C
C     ******************************************************************
C     COMPUTE FLOW ACROSS EACH CELL WALL
C     ******************************************************************
C
C     SPECIFICATIONS:
C     ------------------------------------------------------------------
CHARACTER*128 OUTPATH
CHARACTER*80 CBCPATH
CHARACTER*32 INFONAME
CHARACTER*16 INFOITEM
CHARACTER*4 TEXT
DOUBLE PRECISION HNEW, HD

DIMENSION HNEW(NCOL, NROW, NLAY), IBOUND(NCOL, NROW, NLAY),
1 CR(NCOL, NROW, NLAY), CC(NCOL, NROW, NLAY),
2 CV(NCOL, NROW, NLAY), TOP(NCOL, NROW, NLAY),
3 BUFF(NCOL, NROW, NLAY)

COMMON /FLWCOM/LAYCON(80)

DIMENSION TEXT(12)

DATA TEXT(1), TEXT(2), TEXT(3), TEXT(4), TEXT(5), TEXT(6), TEXT(7),
1 /'FLOW','RIG','HTF','ACE ','FLOW','PRO','NTF','ACE ','FLOW','LOW','ERF','ACE '/

------------------------------------------------------------------

NCM1=NCOL-1
IF(NCM1.LT.1) GO TO 405

C1-----CLEAR THE BUFFER
DO 310 K=1,NLAY
DO 310 I=1,NROW
DO 310 J=1,NCOL
BUFF(J,I,K)=0.
310 CONTINUE

C2-----FOR EACH CELL CALCULATE FLOW THRU RIGHT FACE & STORE IN BUFFER
DO 400 K=1,NLAY
DO 400 I=1,NROW
DO 400 J=1,NCM1
IF((IBOUND(J,I,K).LE.0) .AND. (IBOUND(J+1,I,K).LE.0)) GO TO 400
HDIFF=HNEW(J,I,K)-HNEW(J+1,I,K)
BUFF(J,I,K)=HDIFF*CR(J,I,K)
400 CONTINUE

C3-----RECORD CONTENTS OF BUFFER
C
C3A-----RECORD CONTENTS OF BUFFER IN UNFORMATTED FILE
IF(ICBCFL.GE.0) THEN
CALL UBUDSV(KSTP, KPER, TEXT(1), IBCFCB, BUFF, NCOL, NROW, NLAY, IOUT)
ELSE
ENDIF

C3B-----RECORD CONTENTS OF BUFFER IN ARC INFO FILE
ELSE
INFOITEM='LAYER'
INFONAME='FRFBUD'
OUTPATH=CBCPATH (:INDEX(CBCPATH,' ') -1) // INFONAME
CALL UBUDSVARC (KSTP, KPER, INFOITEM, OUTPATH, BUFF, NCOL, NROW, NLAY,
& IOUT, *9999)
ENDIF

C4-----CLEAR THE BUFFER
405 NRM1=NROW-1
IF(NRM1.LT.1) GO TO 505
DO 410 K=1,NLAY
DO 410 I=1,NROW
DO 410 J=1,NCOL
BUFF(J,I,K)=0.
410 CONTINUE

C
C5-----FOR EACH CELL CALCULATE FLOW THRU FRONT FACE & STORE IN BUFFER
DO 500 K=1,NLAY
DO 500 I=1,NRM1
DO 500 J=1,NCOL
IF((IBOUND(J,I,K).LE.0) .AND. (IBOUND(J,I+1,K).LE.0)) GO TO 500
HDIFF=HNEW(J,I,K)-HNEW(J,I+1,K)
BUFF(J,I,K)=HDIFF*CC(J,I,K)
500 CONTINUE

C
C6-----RECORD CONTENTS OF BUFFER.
C
C6A-----RECORD CONTENTS OF BUFFER IN UNFORMATTED FILE
IF(ICBCFL.GE.0) THEN
   CALL UBUDSV(KSTP,KPER,TEXT(5),IBCFCB,BUFF,NCOL,NROW,NLAY,IOUT)
ENDIF

C6B-----RECORD CONTENTS OF BUFFER IN ARC INFO FILE
ELSE
   INFOITEM='LAYER'
   INFONAME='FFFBUD'
   OUTPATH=CBPATH (:INDEX(CBPATH,' ')-1)//INFONAME
   CALL UBUDSVARC (KSTP,KPER,INFOITEM,OUTPATH,BUFF,NCOL,NROW,NLAY,
   &               IOUT,*9999)
ENDIF

505 NLM1=NLAY-1
IF(NLM1.LT.1) GO TO 1000

C
C7-----CLEAR THE BUFFER
DO 510 K=1,NLAY
DO 510 I=1,NROW
DO 510 J=1,NCOL
BUFF(J,I,K)=0.
510 CONTINUE

C
C8-----FOR EACH CELL CALCULATE FLOW THRU LOWER FACE & STORE IN BUFFER
KT=0
DO 600 K=1,NLM1
   IF(LAYCON(K).EQ.3 .OR. LAYCON(K).EQ.2) KT=KT+1
   DO 600 I=1,NROW
   DO 600 J=1,NCOL
   IF((IBOUND(J,I,K).LE.0) .AND. (IBOUND(J,I,K+1).LE.0)) GO TO 600
   HD=HNEW(J,I,K+1)
   IF(LAYCON(K+1).NE.3 .AND. LAYCON(K+1).NE.2) GO TO 580
   TMP=HD
   IF(TMP.LT.TOP(J,I,KT+1)) HD=TOP(J,I,KT+1)
580 HDIFF=HNEW(J,I,K)-HD
BUFF(J,I,K)=HDDIFF*CV(J,I,K)
600 CONTINUE

C
C9-----RECORD CONTENTS OF BUFFER
C
C9A-----RECORD CONTENTS OF BUFFER IN UNFORMATTED FILE
IF(ICBCFL.GE.0) THEN
   CALL UBUDSV(KSTP,KPER,TEXT(9),IBCFCB,BUFF,NCOL,NROW,NLAY,IOUT)
ENDIF

C9B-----RECORD CONTENTS OF BUFFER IN ARC INFO FILE
ELSE
   INFOITEM='LAYER'
   INFONAME='LIFFBUD'
   OUTPATH=CBPATH (:INDEX(CBPATH,' ')-1)//INFONAME
   CALL UBUDSVARC (KSTP,KPER,INFOITEM,OUTPATH,BUFF,NCOL,NROW,NLAY,
   &               IOUT,*9999)
ENDIF
C10----RETURN  
1000 RETURN  
C  
CE------ERRORS  
C  
9999 CALL INFORM ('\Abnormal Termination of Sbcflb_Arc_Subroutine',-1)  
RETURN 1  
END  

Added variables for module SBCF1BARC  

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>CBCPATH</td>
<td>Package</td>
<td>The directory path to the ARC/INFO subdirectory where cell-by-cell flow terms are recorded (passed argument from module BCF1BDARC).</td>
</tr>
<tr>
<td>INFONAME</td>
<td>Submodule</td>
<td>The name of the ARC/INFO file where cell-by-cell flow terms are recorded (passed argument consisting of the root name FRFBUD, FFFBUD, or FLFBUD which later will be appended by the stress period and time step).</td>
</tr>
<tr>
<td>INFOITEM</td>
<td>Submodule</td>
<td>The name of the ARC/INFO or INFO item where cell-by-cell flow terms are recorded (passed argument consisting of the root name LAYER which later will be appended by the layer number).</td>
</tr>
<tr>
<td>OUTPATH</td>
<td>Submodule</td>
<td>The path for the ARC/INFO file where cell-by-cell flow terms are recorded.</td>
</tr>
</tbody>
</table>

River Package Modules

The RIV (River) package consists of four primary modules; of these, two primary modules indicated below were modified.

RIV1RPARC

This module directly reads and prepares data for the RIV package. The module reads the control record and searches for two variables, ITMP and RIVPATH, and reads values for river parameters (layer, row, column, stage, conductance, and bottom elevation) within the ARC/INFO file specified within the variable RIVPATH (fig. 11). Documentation of modifications follow.
Figure 11.--Modified program elements for the RIV1RPARC module.
SUBROUTINE RIVRPARC (RIVR,NRIVER,MXRIVR,IN,IOUT,
  &               E)

C-----VERSION 3.0 25OCTOBER1991 RIVRPARC
C               MODIFIED BY LEONARD L. ORZOL

C*****************************************************************
C       READ RIVER HEAD, CONDUCTANCE AND BOTTOM ELEVATION
C*****************************************************************

C SPECIFICATIONS:
-----------------------------------------------
DIMENSION RIVR(6,MXRIVR)
CHARACTER*132 BUFFER
CHARACTER*128 INFOPATH
CHARACTER*80 RIVPATH
CHARACTER*16 ITEMS(6)
INTEGER ACCESS,FNUM,NUMRIV,NITEMS,NUMREC
DATA ITEMS(1),ITEMS(2),ITEMS(3),ITEMS(4),ITEMS(5),ITEMS(6)
&      /'LAYER','ROW','COLUMN','STAGE','COND','RBOT'/
-----------------------------------------------

C1-------RIVER REACHES INFORMATION
C
READ(IN,'(A132)',ERR=9990,END=9991) BUFFER

C1A------READ ITMP(NUMBER OF RIVER REACHES OR FLAG SAYING REUSE DATA)
C
IF(BUFFER (11:11).EQ.'' .OR. BUFFER (11:11).EQ.' ') THEN
READ (BUFFER,'(I10)',ERR=9992) ITMP
IARC=0
C
C1B------READ ITMP(NUMBER OF RIVER REACHES OR FLAG SAYING REUSE DATA) AND
C---------RIVPATH TO ARC/INFO FILE
C
ELSE
READ (BUFFER,'(I10,A80)',ERR=9993) ITMP,RIVPATH
INFOPATH=RIVPATH (1:INDEX(RIVPATH,' ')-1)
IARC=1
ENDIF
C
C2------TEST ITMP.
IF(ITMP.GE.0)GO TO 50
C
C2A------IF ITMP <0 THEN REUSE DATA FROM LAST STRESS PERIOD.
WRITE(IOUT,7)
7 FORMAT(1H0,'REUSING RIVER REACHES FROM LAST STRESS PERIOD')
GO TO 260
C
C3------IF ITMP=> ZERO THEN IT IS THE NUMBER OF RIVER REACHES
50 NRIVER=ITMP
C
C4------IF NRIVER>MXRIVR THEN STOP.
IF(NRIVER.LE.MXRIVR)GO TO 100
WRITE(IOUT,99)NRIVER,MXRIVR
99 FORMAT(1H0,'NRIVER(',I4,') IS GREATER THAN MXRIVR(',I4,')')
C
C4A-----ABNORMAL STOP.
    GO TO 9999
C
C5------PRINT NUMBER OF RIVER REACHES IN THIS STRESS PERIOD.
100 WRITE(IOUT,1)NRIVER
    1 FORMAT(1H0,//1X,I5,' RIVER REACHES')
C
C6------IF THERE ARE NO RIVER REACHES THEN RETURN.
IF(NRIVER.EQ.0) GO TO 260
C
C7------READ AND PRINT DATA FOR EACH RIVER REACH.
WRITE(IOUT,3)
3 FORMAT(1H0,15X,'LAYER',5X,'ROW',5X,'COLUMN',5X,'STAGE',5X,'COND',
         1,' STAGE CONDUCTANCE BOTTOM ELEVATION RIVER REACH'
         2/1X,15X,80('—'))
C7A------READING RIVER REACHES INFORMATION FROM ASCII FILE
C
IF (IARC.LT.1) THEN
  DO 250 II=1, NRIVER
    READ (IN, 4) K, I, J, RIVR(4, II), RIVR(5, II), RIVR(6, II)
  4 FORMAT (3I10, 3F10.0)
    WRITE (IOUT, 5) K, I, J, RIVR(4, II), RIVR(5, II), RIVR(6, II), II
  5 FORMAT (1X, 15X, I4, I9, I8, G13.4, G14.4, G19.4, I10)
    RIVR(1, II) = K
    RIVR(2, II) = I
    RIVR(3, II) = J
  250 CONTINUE
C
C6B------READING WELL INFORMATION FROM ARC/INFO FILE
C
ELSE
C
C6BB------TESTS EXISTENCE OF INFO FILE
C
ACCESS = 1
    CALL INFO_OPENS (INFOPATH, ACCESS, FNUM, NUMREC, *9999)
C
C6BC------OPENS AND READS ITEMS (LAYER, ROW, COLUMN, STAGE, CONDUCTANCE AND
C----------BOTTOM ELEVATION)
C
NITEMS = 6
    NUMRIV = NRIVER * NITEMS
    CALL INFO_READMULT (FNUM, NUMREC, NUMRIV, ITEMS, NITEMS, RIVR, *9999)
    DO 600 II = 1, NRIVER
      K = RIVR(1, II)
      I = RIVR(2, II)
      J = RIVR(3, II)
      WRITE (IOUT, 5) K, I, J, RIVR(4, II), RIVR(5, II), RIVR(6, II), II
    600 CONTINUE
ENDIF
C
C8------RETURN
260 RETURN
C
CE------ERRORS
C
9990 CALL INFORM ('\Unable to read River input control line', -1)
    CALL INFORM ('\Abnormal Termination of Rivlrp_Arc_Subroutine', -1)
    RETURN 1
9991 CALL INFORM ('\Missing River input package control line', -1)
    CALL INFORM ('\Abnormal Termination of Rivlrp_Arc_Subroutine', -1)
    RETURN 1
9992 CALL INFORM ('\Unable to read ITMP from River input line', -1)
    CALL INFORM ('\Abnormal Termination of Rivlrp_Arc_Subroutine', -1)
    RETURN 1
9993 CALL INFORM ('\Unable to read ITMP or RIVPATH ' //
      & \'from River input line', -1)
9999 CALL INFORM ('\Abnormal Termination of Rivlrp_Arc_Subroutine', -1)
    RETURN 1
END

Added variables for module RIV1RPARC

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCESS</td>
<td>Module</td>
<td>Flag indicating whether access to the ARC/INFO file is read or write.</td>
</tr>
<tr>
<td>BUFFER</td>
<td>Module</td>
<td>The control record is read into this variable and is parsed into the appropriate variables: ITMP and RIVPATH.</td>
</tr>
</tbody>
</table>
This module calculates rates and volumes transferred between the aquifer and rivers for the RIV package, and calls submodule UBUDSVARC (fig. 12). Documentation of the changes follow.

Figure 12.--Modified program elements for the RIV1BDARC module.
SUBROUTINE RIV1BDARC (NRIVER, MXRIVR, RIVR, IBOUND, HNEW, & NCOL, NROW, NLAY, DELT, VBVL, VBNM, MSUM, KSTP, KPER, IRIVCB, & ICBCFL, BUFF, IOUT, RIVPATH, & E *)
C-----VERSION 3.0 25OCTOBER1991 RIV1BDARC
C MODIFIED BY LEONARD L. ORZOL
C
C     ******************************************************************
C     CALCULATE VOLUMETRIC BUDGET FOR RIVERS
C     ******************************************************************
C
C     SPECIFICATIONS:
C     ------------------------------------------------------------------
CHARACTER*128 OUTPATH
CHARACTER*80 RIVPATH
CHARACTER*32 INFONAME
CHARACTER*16 INFOITEM
CHARACTER*4 VBNM, TEXT
DOUBLE PRECISION HNEW
DIMENSION RIVR(6, MXRIVR), IBOUND (NCOL, NROW, NLAY),
1 HNEW(NCOL, NROW, NLAY), VBVL(4, 20), VBNM(4, 20),
2 BUFF (NCOL, NROW, NLAY)
DIMENSION TEXT(4)
DATA TEXT(1), TEXT(2), TEXT(3), TEXT(4) /'   R', 'IVER', ' LEA', ' KAGE'/
C     ------------------------------------------------------------------
C
C1------INITIALIZE CELL-BY-CELL FLOW TERM FLAG (IBD) AND
C1------ACCUMULATORS (RATIN AND RATOUT).
IBD=0
RATIN=0.
RATOUT=0.
C
C2------IF NO REACHES KEEP ZEROES IN ACCUMULATORS.
IF (NRIVER.EQ.0) GO TO 200
C
C3------TEST TO SEE IF CELL-BY-CELL FLOW TERMS ARE NEEDED.
IF (ICBCFL.EQ.0  .OR. IRIVCB.LE.0 ) GO TO 10
C
C3A------CELL-BY-CELL FLOW TERMS ARE NEEDED SET IBD AND CLEAR BUFFER.
IBD=1
DO 5 IL=1, NLAY
DO 5 IR=1, NROW
DO 5 IC=1, NCOL
BUFF(IC, IR, IL)=0.
5 CONTINUE
C
C4------FOR EACH RIVER REACH ACCUMULATE RIVER FLOW (STEPS 5-15)
10 DO 100 L=1, NRIVER
C
C5------GET LAYER, ROW & COLUMN OF CELL CONTAINING REACH.
IL=RIVR(1, L)
IR=RIVR(2, L)
IC=RIVR(3, L)
C
C6------IF CELL IS EXTERNAL MOVE ON TO NEXT REACH.
IF (IBOUND(IC, IR, IL).LE.0) GO TO 100
C
C7------GET RIVER PARAMETERS FROM RIVER LIST.
HRIV=RIVR(4, L)
CRIV=RIVR(5, L)
RBOT=RIVR(6, L)
HNEW=HNEW(IC, IR, IL)
C
C8------COMPARE HEAD IN AQUIFER TO BOTTOM OF RIVERBED.
C
C9------AQUIFER HEAD > BOTTOM THEN RATE=CRIV*(HRIV-HNEW).
IF (HNEW.GT.RBOT) RATE=CRIV*(HRIV-HNEW)
C
C10------AQUIFER HEAD < BOTTOM THEN RATE=CRIV*(HRIV-RBOT)
IF (HNEW.LE.RBOT) RATE=CRIV*(HRIV-RBOT)
C
C11------PRINT THE INDIVIDUAL RATES IF REQUESTED(IRIVCB<0).
IF(IRIVCB.LT.0.AND.ICBCFL.NE.0) WRITE(IOUT,900) (TEXT(N),N=1,4),
1 KPER,KSTP,L,IL,IR,RATE
900 FORMAT(1H0,4A4,'   PERIOD',I3,'   STEP',I3,'   REACH',I4,
1 '   LAYER',I3,'   ROW',I4,'   COL',I4,'   RATE',G15.7)
C
C12-----IF C-B-C FLOW TERMS ARE TO BE SAVED THEN ADD RATE TO BUFFER.
   IF(IBD.EQ.1) BUFF(IC,IR,IL)=BUFF(IC,IR,IL)+RATE
C
C13-----SEE IF FLOW IS INTO AQUIFER OR INTO RIVER.
   IF(RATE)94,100,96
C
C14-----AQUIFER IS DISCHARGING TO RIVER SUBTRACT RATE FROM RATOUT.
   94 RATOUT=RATOUT-RATE
   GO TO 100
C
C15-----AQUIFER IS RECHARGED FROM RIVER ADD RATE TO RATIN.
   96 RATIN=RATIN+RATE
   100 CONTINUE
C
C6-----IF C-B-C FLOW TERMS WILL BE SAVED CALL UBUDSV TO RECORD THEM.
C
C6A------RECORD IN UNFORMATTED FILE IWELCB
C   IF(IBD.EQ.1 .AND. ICBCFL.GE.0) CALL UBUDSV(KSTP,KPER,TEXT,IRIVCB,
&                          BUFF,NCOL,NROW,NLAY,IOUT)
C
C6B------RECORD IN ARC/INFO FILE
C   IF(IBD.EQ.1 .AND. ICBCFL.LT.0) THEN
     INFOITEM='LAYER'
     INFONAME='RIVBUD'
     OUTPATH=RIVPATH (:INDEX(RIVPATH,' ')-1)//INFONAME
     CALL UBUDSVARC (KSTP,KPER,INFOITEM,OUTPATH,BUFF,NCOL,NROW,NLAY,
&                  IOUT, *9999)
   ENDIF
C
C17-----MOVE RATES,VOLUMES & LABELS INTO ARRAYS FOR PRINTING.
   200 VBVL(3,MSUM)=RATIN
   VBVL(4,MSUM)=RATOUT
   VBVL(1,MSUM)=VBVL(1,MSUM)+RATIN*DELT
   VBVL(2,MSUM)=VBVL(2,MSUM)+RATOUT*DELT
   VBNM(1,MSUM)=TEXT(1)
   VBNM(2,MSUM)=TEXT(2)
   VBNM(3,MSUM)=TEXT(3)
   VBNM(4,MSUM)=TEXT(4)
C
C18-----INCREMENT BUDGET TERM COUNTER
   MSUM=MSUM+1
C
C19-----RETURN
   RETURN
C
CE------ERRORS
C   9999 CALL INFORM ('\Abnormal Termination of Riv1bd_Arc_Subroutine',-1)
   RETURN 1
END

Added variables for module RIV1BDARC

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>INFONAME</td>
<td>Module</td>
<td>The name of the ARC/INFO file where cell-by-cell flow terms are recorded (passed argument consisting of the root name RIVBUD which later will be appended by the stress period and time step).</td>
</tr>
<tr>
<td>INFOITEM</td>
<td>Module</td>
<td>The name of the ARC/INFO or INFO item where cell-by-cell flow</td>
</tr>
</tbody>
</table>
terms are recorded (passed argument consisting of the root name LAYER which later will be appended by the layer number).

OUTPATH Module The path for the ARC/INFO file where cell-by-cell flow terms are recorded.

RIVPATH Package The directory path to the ARC/INFO subdirectory where cell-by-cell flow terms are recorded.

Recharge Package Modules

The RCH (Recharge package) consists of four primary modules and five submodules; of these, two primary modules (RCH1RPARC and RCH1BDARC) indicated below were modified.

RCH1RPARC

This module reads and prepares data for the RCH package calls submodules U2RELARC and U2DINTARC (fig. 13). Documentation of these changes follows.

![Figure 13.--Modified program elements for the RCH1RPARC module.](image)

SUBROUTINE RCH1RPARC (NRCHOP, IRCH, RECH, DELR, DELC, NROW, NCOL, IN, IOUT, E)

C-----VERSION 3.0 25OCTOBER1991 RCH1RPARC
C MODIFIED BY LEONARD L. ORZOL

*********************************************************************************
READ RECHARGE RATES
*********************************************************************************

SPECIFICATIONS:
----------------------------------------------------------------------------------
CHARACTER*16 INFOITEM
CHARACTER*4 ANAME
DIMENSION IRCH (NCOL, NROW), RECH (NCOL, NROW),
1 ANAME (6, 2), DELR (NCOL), DELC (NROW)

DATA ANAME(1,1), ANAME(2,1), ANAME(3,1), ANAME(4,1), ANAME(5,1),
1 ANAME(6, 1) /'    ','RECH','ARGE',' LAY','ER I','NDEX'/
DATA ANAME(1,2), ANAME(2,2), ANAME(3,2), ANAME(4,2), ANAME(5,2),
1 ANAME(6, 2) /'    ','    ','    ','    ','RECH','ARGE'/

C1------READ FLAGS SHOWING WHETHER DATA IS TO BE REUSED.
READ(IN, 4, ERR=9990, END=9991) INRECH, INIRCH
4 FORMAT(2I10)
C
C2------TEST INRECH TO SEE WHERE RECH IS COMING FROM.
IF(INRECH.GE.0) GO TO 32
C
C2A------IF INRECH<0 THEN REUSE RECHARGE ARRAY FROM LAST STRESS PERIOD
WRITE(IOUT,3)
3 FORMAT(1H0,'REUSING RECH FROM LAST STRESS PERIOD')
GO TO 55
C
C3------IF INRECH=>0 THEN CALL U2DREL TO READ RECHARGE RATE.
32 INFOITEM= 'RECH'
   CALL U2DRELARC (INFOITEM, RECH, ANAME(1,2), NROW, NCOL, 0, IN, IOUT, *9999)
C
C4------MULTIPLY RECHARGE RATE BY CELL AREA TO GET VOLUMETRIC RATE.
   DO 50 IR=1, NROW
   DO 50 IC=1, NCOL
      RECH(IC,IR)=RECH(IC,IR)*DELR(IC)*DELC(IR)
50 CONTINUE
C
C5------IF NRCHOP=2 THEN A LAYER INDICATOR ARRAY IS NEEDED.
   55 IF (NRCHOP.NE.2) GO TO 60
C
C6------IF INIRCH<0 THEN REUSE LAYER INDICATOR ARRAY.
   IF(INIRCH.GE.0) GO TO 58
   WRITE(IOUT,2)
2 FORMAT(1H0,'REUSING IRCH FROM LAST STRESS PERIOD')
   GO TO 60
C
C7------IF INIRCH=>0 CALL U2DINTARC TO READ LAYER IND ARRAY(IRCH)
58 INFOITEM= 'IRCH'
   CALL U2DINTARC (INFOITEM, IRCH, ANAME(1,1), NROW, NCOL, 0, IN, IOUT, *9999)
C
C8------RETURN
   60 RETURN
C
CE------ERRORS
C
9990 CALL INFORM ('\Unable read Recharge input package control record',-1)
   CALL INFORM ('\Abnormal Termination of Rch1rp_Arc_Subroutine',-1)
   RETURN 1
9991 CALL INFORM ('\Missing Recharge input package control record',-1)
   CALL INFORM ('\Abnormal Termination of Rch1rp_Arc_Subroutine',-1)
   RETURN 1
9999 CALL INFORM ('\Abnormal Termination of Rch1rp_Arc_Subroutine',-1)
   RETURN 1
END

Added variables for module RCH1RPARC

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>INFOITEM</td>
<td>Module</td>
<td>The name of the ARC/INFO or INFO item where the values for the recharge array are stored (passed argument consisting of the root name RECH and IRCH which later will be appended within utility U2DRELARC and U2DINTARC modules.</td>
</tr>
</tbody>
</table>

RCH1BDARC

This module calculates the rate and accumulated volume of recharge for the RCH package and calls submodules UBUDSVARC (fig. 14). Documentation follows.
SUBROUTINE RCH1BDARC (NRCHOP, IRCH, RECH, IBOUND, NROW, NCOL, NLAY, 
& DELT, VBVL, VBNM, MSUM, KSTEP, KPER, IRCHCB, ICBCFL, BUFF, IOUT, RCHPATH, 
& *)
C
C-----VERSION 3.0 25OCTOBER1991 RCH1BDARC
C               MODIFIED BY LEONARD L. ORZOL
C     ******************************************************************
C     CALCULATE VOLUMETRIC BUDGET FOR RECHARGE
C     ******************************************************************
C
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
CHARACTER*128 OUTPATH
CHARACTER*80 RCHPATH
CHARACTER*32 INFONAME
CHARACTER*16 INFOITEM
CHARACTER*4 VBNM, TEXT
DIMENSION IRCH(NCOL, NROW), RECH(NCOL, NROW),
1 IBOUND(NCOL, NROW, NLAY), BUFF(NCOL, NROW, NLAY),
2 VBVL(4,20), VBNM(4,20)
DIMENSION TEXT(4)
DATA TEXT(1), TEXT(2), TEXT(3), TEXT(4) /'    ','    ','RECH','ARGE'/
C     ------------------------------------------------------------------

C1------CLEAR THE RATE ACCUMULATORS.
RATIN=0.
RATOUT=0.
C
C2------IF CELL-BY-CELL FLOW TERMS WILL BE SAVED THEN CLEAR THE BUFFER.
IBD=0
IF(ICBCFL.EQ.0 .OR. IRCHCB.LE.0) GO TO 5
IBD=1
DO 2 IL=1,NLAY
DO 2 IR=1,NROW
DO 2 IC=1,NCOL
BUFF(IC,IR,IL)=0.

Figure 14.--Modified program elements for the RCH1BDARC module.
2 CONTINUE

C3------IF NRCHOP=1 RECH GOES INTO LAYER 1. PROCESS EACH HORIZONTAL
C3------CELL LOCATION.
     5 IF(NRCHOP.NE.1) GO TO 15
C
---RECHARGE IS APPLIED TO TOP LAYER
     DO 10 IR=1,NROW
     DO 10 IC=1,NCOL

C3A------IF CELL IS EXTERNAL THEN DO NOT DO BUDGET FOR IT.
     IF(IBOUND(IC,IR,1).LE.0)GO TO 10
     Q=RECH(IC,IR)
C
C3B------IF CELL-BY-CELL FLOW TERMS WILL BE SAVED THEN ADD RECH TO BUFF
     IF(IBD.EQ.1) BUFF(IC,IR,1)=Q
C
C3C------IF RECH POSITIVE ADD IT TO RATIN ELSE ADD IT TO RATOUT.
     IF(Q) 8,10,7
     7 RATIN=RATIN+Q
     GO TO 10
     8 RATOUT=RATOUT-Q
     10 CONTINUE
     GO TO 100

C4------IF NRCHOP=2 RECH IS IN LAYER SHOWN IN INDICATOR ARRAY(IRCH).
C4------PROCESS HORIZONTAL CELL LOCATIONS ONE AT A TIME.
     15 IF(NRCHOP.NE.2)GO TO 25
     DO 20 IR=1,NROW
     DO 20 IC=1,NCOL

C4A-----GET LAYER INDEX FROM INDICATOR ARRAY(IRCH).
     IL=IRCH(IC,IR)
C
C4B-----IF CELL IS EXTERNAL DO NOT CALCULATE BUDGET FOR IT.
     IF(IBOUND(IC,IR,IL).LE.0) GO TO 20
     Q=RECH(IC,IR)
C
C4C-----IF C-B-C FLOW TERMS WILL BE SAVED THEN ADD RECHARGE TO BUFFER.
     IF(IBD.EQ.1) BUFF(IC,IR,IL)=Q
C
C4D------IF RECHARGE IS POSITIVE ADD IT TO RATIN ELSE ADD IT TO RATOUT.
     IF(Q) 18,20,17
     17 RATIN=RATIN+Q
     GO TO 20
     18 RATOUT=RATOUT-Q
     20 CONTINUE
     GO TO 100

C5------IF OPTION=3 RECHARGE IS INTO HIGHEST INTERNAL CELL. IT WILL NOT
C5------PASS THROUGH A CONSTANT HEAD CELL. PROCESS HORIZONTAL CELL
C5------LOCATIONS ONE AT A TIME.
     25 IF(NRCHOP.NE.3)GO TO 100
     DO 30 IR=1,NROW
     DO 30 IC=1,NCOL
     DO 28 IL=1,NLAY

C5A-----IF CELL IS CONSTANT HEAD MOVE ON TO NEXT HORIZONTAL LOCATION.
     IF(IBOUND(IC,IR,IL).LT.0) GO TO 30
C
C5B------IF CELL IS INACTIVE MOVE DOWN TO NEXT CELL.
     IF (IBOUND(IC,IR,IL).EQ.0)GO TO 28
     Q=RECH(IC,IR)
C
C5C------IF C-B-C FLOW TERMS TO BE SAVED THEN ADD RECHARGE TO BUFFER.
     IF(IBD.EQ.1) BUFF(IC,IR,IL)=Q
C
C5D------IF RECH IS POSITIVE ADD IT TO RATIN ELSE ADD IT TO RATOUT.
     IF(Q) 27,30,26
     26 RATIN=RATIN+Q
     GO TO 30
     27 RATOUT=RATOUT-Q
GO TO 30
28 CONTINUE
30 CONTINUE

C
100 CONTINUE

C6------IF C-B-C FLOW TERMS TO BE SAVED TO RECORD THEM.
C6A------RECORD IN UNFORMATTED FILE IRCHCB
C
IF (IBD.EQ.1 .AND. ICBCFL.GE.0) CALL UBUDSV(KSTP, KPER, TEXT, IRCHCB,
       &                       BUFF, NCOL, NROW, NLAY, IOUT)
C
C6B------RECORD IN ARC/INFO FILE
C
IF (IBD.EQ.1 .AND. ICBCFL.LT.0) THEN
   INFOITEM='LAYER'
   INFONAME='RCHBUD'
   OUTPATH=RCHPATH (:INDEX(RCHPATH, ' ')-1)//INFONAME
   CALL UBUDSVARC (KSTP, KPER, INFOITEM, OUTPATH, BUFF, NCOL, NROW, NLAY,
       &                   IOUT, *9999)
ENDIF
C
C7------MOVE TOTAL RECHARGE RATE INTO VBVL FOR PRINTING BY BAS1OT.
    VBVL(4,MSUM)=RATOUT
    VBVL(3,MSUM)=RATIN
C
C8------ADD RECHARGE FOR TIME STEP TO RECHARGE ACCUMULATOR IN VBVL.
    VBVL(2,MSUM)=VBVL(2,MSUM)+RATOUT*DELT
    VBVL(1,MSUM)=VBVL(1,MSUM)+RATIN*DELT
C
C9------MOVE BUDGET TERM LABELS TO VBNM FOR PRINT BY MODULE BAS_OT.
    VBNM(1,MSUM)=TEXT(1)
    VBNM(2,MSUM)=TEXT(2)
    VBNM(3,MSUM)=TEXT(3)
    VBNM(4,MSUM)=TEXT(4)
C
C10------INCREMENT BUDGET TERM COUNTER.
    MSUM=MSUM+1
C
C11------RETURN
    RETURN
C
CE------ERRORS
C
9999 CALL INFORM ('\Abnormal Termination of Rch1bd_Arc_Subroutine',-1)
    RETURN 1
END

Added variables for module RCH1BDARC

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>INFONAME</td>
<td>Module</td>
<td>The name of the ARC/INFO file where cell-by-cell flow terms are recorded (passed argument consisting of the root name RCHBUD which later will be appended by the stress period and time step).</td>
</tr>
<tr>
<td>INFOITEM</td>
<td>Module</td>
<td>The name of the ARC/INFO or INFO item where cell-by-cell flow terms are recorded (passed argument consisting of the root name LAYER which later will be appended by the layer number).</td>
</tr>
<tr>
<td>OUTPATH</td>
<td>Module</td>
<td>The path for the ARC/INFO file where cell-by-cell flow terms are recorded.</td>
</tr>
<tr>
<td>RCHPATH</td>
<td>Package</td>
<td>The directory path to the ARC/INFO subdirectory where cell-by-cell flow terms are recorded.</td>
</tr>
</tbody>
</table>
Well Package Modules

The WEL package (Well) consists of four primary modules; of these, two primary modules (WEL1RPARC and WEL1BDARC) indicated below were modified.

WEL1RPARC

This module directly reads and prepares data for the WEL package. This module searches the control record for two variables, ITMP and WELPATH and reads values for well parameters (layer, row, column, and Q) within the ARC/INFO file specified within the variable WELPATH (fig. 15). Documentation of the modified module follows.

Figure 15.--Modified program elements for the WEL1RPARC module.
SUBROUTINE WEL1RPARC (WELL,NWELLS,MXWELL,IN,IOUT, E *)

C-----VERSION 3.0 25OCTOBER1991 WEL1RPARC
C MODIFIED BY LEONARD L. ORZOL
C
C**********************************************************************
C READ NEW WELL LOCATIONS AND STRESS RATES
C**********************************************************************
C
C SPECIFICATIONS:
C--------------------------------------------------------------------------------
DIMENSION WELL(4,MXWELL)
CHARACTER*128 INFOPATH
CHARACTER*80 BUFFER
CHARACTER*80 WELPATH
CHARACTER*16 ITEMS(4)
INTEGER ACCESS,FNUM,IARC,NITEMS,NUMREC,NUMWEL
DATA ITEMS(1),ITEMS(2),ITEMS(3),ITEMS(4)/'LAYER','ROW','COLUMN','Q'/
--------------------------------------------------------------------------------
C
C1------READ CONTROL LINE FOR WELL INFORMATION
C READ(IN,'(A132)',ERR=9990,END=9991) BUFFER
C
C1A------READ ITMP(NUMBER OF WELLS OR FLAG SAYING REUSE WELL DATA)
C IF(BUFFER (11:11).EQ.'' .OR. BUFFER (11:11).EQ.' ') THEN
READ (BUFFER,'(I10)',ERR=9992) ITMP
IARC=0
C
C1B------READ ITMP(NUMBER OF WELLS OR FLAG SAYING REUSE WELL DATA) AND
---------WELPATH TO ARC/INFO FILE
C ELSE
READ (BUFFER,'(I10,A80)',ERR=9992) ITMP,WELPATH
INFOPATH=WELPATH (1:INDEX(WELPATH,' ')-1)
IARC=1
ENDIF
C
C2------TEST ITMP CONDITION
C IF(ITMP.GE.0) GO TO 50
C
C2A------IF ITMP LESS THAN ZERO REUSE DATA. PRINT MESSAGE AND RETURN.
WRITE(IOUT,6) 6 FORMAT(1H0,'REUSING WELLS FROM LAST STRESS PERIOD')
RETURN
C
C2B------ITMP=>0. SET NWELLS EQUAL TO ITMP.
50 NWELLS=ITMP
IF(NWELLS.LE.MXWELL) GO TO 100
C
C3------NWELLS>MXWELL. PRINT MESSAGE. STOP.
WRITE(IOUT,99) NWELLS,MXWELL
99 FORMAT(1H0,'NWELLS(',I4,') IS GREATER THAN MXWELL(',I4,')')
GO TO 999
C
C4------PRINT NUMBER OF WELLS IN CURRENT STRESS PERIOD.
100 WRITE (IOUT,2) NWELLS
2 FORMAT(1H0,10X,I4,' WELLS')
C
C5------IF THERE ARE NO ACTIVE WELLS IN THIS STRESS PERIOD THEN RETURN
IF(NWELLS.EQ.0) GO TO 260
C
C6------READ AND PRINT LAYER,ROW,COLUMN AND RECHARGE RATE.
C WRITE(IOUT,3)
3 FORMAT(1H0,47X,'LAYER    ROW    COL    STRESS RATE   WELL NO.'/
1,48X,45('-'))
C
C6A------READING WELL INFORMATION FROM ASCII FILE
IF(IARC.LT.1) THEN
    DO 250 II=1,NWELLS
        READ (IN,4) K, I, J, Q
        WRITE (IOUT,5) K, I, J, Q, II
    4       FORMAT(3I10,F10.0)
    5       FORMAT(48X,I3,I8,I7,G16.5,I8)
    WELL(1,II)=K
    WELL(2,II)=I
    WELL(3,II)=J
    WELL(4,II)=Q
    250    CONTINUE
ELSE

C6B------READING WELL INFORMATION FROM ARC/INFO FILE

C

C6BB------TESTS EXISTENCE OF INFO FILE

C

ACCESS=1
    CALL INFO_OPENS (INFOPATH,ACCESS,FNUM,NUMREC,
                      *9999)

C

C6BC------OPENS AND READS ITEMS (ROW,COLUMN,LAYER,STRESS RATE )

C

NITEMS=4
    NUMWEL=NWELLS*NITEMS
    CALL INFO_READMULT (FNUM,NUMREC,NUMWEL,ITEMS,NITEMS,WELL,
                             *9999)
    DO 600 II=1,NWELLS
        K=WELL(1,II)
        I=WELL(2,II)
        J=WELL(3,II)
        WRITE(IOUT,5) K, I, J, WELL(4,II), II
    600    CONTINUE
ENDIF

C7------RETURN
260 RETURN

CE------ERRORS

9990 CALL INFORM ("Unable read Well input package control record",-1)
9991 CALL INFORM ("Abnormal Termination of Wel1rp_Arc_Subroutine",-1)
9992 CALL INFORM ("Missing Well input package control record",-1)
9999 CALL INFORM ("Abnormal Termination of Wel1rp_Arc_Subroutine",-1)
END

Added variables for module WEL1RPARC

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCESS</td>
<td>Module</td>
<td>Flag indicating whether access to the ARC/INFO file is read or write.</td>
</tr>
<tr>
<td>BUFFER</td>
<td>Module</td>
<td>The control record is read into this variable and is parsed into the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>appropriate variables: ITMP and WELPATH.</td>
</tr>
<tr>
<td>FNUM</td>
<td>Module</td>
<td>Integer unit number used by this routine for the file specified by</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INFOPATH.</td>
</tr>
<tr>
<td>ITEMS</td>
<td>Module</td>
<td>The names of the INFO item array either primary or redefined</td>
</tr>
<tr>
<td></td>
<td></td>
<td>within the ARC/INFO file (specified by WELPATH) containing the information.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The item names are LAYER,</td>
</tr>
</tbody>
</table>
INFOPATH Module The path for the ARC/INFO file where the values for the well array are stored.

NITEMS Module Integer value for the number of items within the file specified by INFOPATH.

NUMREC Module Integer value for the number of records within the file specified by INFOPATH.

NUMWEL Module Integer value for the number of the record times the number of items that are needed within the file specified by INFOPATH.

WELPATH Module The complete path to the ARC/INFO file containing the item (INFOITEM) of interest.

WEL1BDARC

This module calculates rates and volumes recharged to, or discharged from flow system by pumping wells for WEL. Program code modifications were minimal, because program control passes to called submodule UBUDSVARC, instead of UBUDSV (fig. 16. Documentation of the modified module follows.

Figure 16.--Modified program elements for the WEL1BDARC module.
C     SPECIFICATIONS:
C     ---------------------------------------------------------------
CHARACTER*128 OUTPATH
CHARACTER*70 WELPATH
CHARACTER*32 INFONAME
CHARACTER*16 INFOITEM
CHARACTER*4 VBNM, TEXT
DIMENSION VBNM(4, MSUM), VBL(4, MSUM), WELL(4, MXWELL),
1       IBOUND(NCOL, NROW, NLAY), BUFF(NCOL, NROW, NLAY)
DIMENSION TEXT(4)
C     ---------------------------------------------------------------
DATA TEXT(1), TEXT(2), TEXT(3), TEXT(4) /'    ','    ','   W','ELLS'/
C     ---------------------------------------------------------------
C1------CLEAR RATIN AND RATOUT ACCUMULATORS.
RATIN=0.
RATOUT=0.
IBD=0
C2------IF THERE ARE NO WELLS DO NOT ACCUMULATE FLOW
IF(NWELLS.EQ.0) GO TO 200
C3------TEST TO SEE IF CELL-BY-CELL FLOW TERMS WILL BE RECORDED.
IF(ICBCFL.EQ.0 .OR. IWELCB.LE.0) GO TO 60
C4------IF CELL-BY-CELL FLOWS WILL BE SAVED THEN CLEAR THE BUFFER.
IBD=1
DO 50 IL=1, NLAY
DO 50 IR=1, NROW
DO 50 IC=1, NCOL
BUFF(IC, IR, IL)=0.
50 CONTINUE
C5------PROCESS WELLS ONE AT A TIME.
60 DO 100 L=1, NWELLS
     IR=WELL(2, L)
     IC=WELL(3, L)
     IL=WELL(1, L)
     Q=WELL(4, L)
C5A-----IF THE CELL IS EXTERNAL IGNORE IT.
     IF(IBOUND(IC, IR, IL).LE.0) GO TO 100
C5B-----PRINT THE INDIVIDUAL RATES IF REQUESTED(IWELCB<0).
     IF(IWELCB.LT.0.AND.ICBCFL.NE.0) WRITE(IOUT, 900) (TEXT(N), N=1, 4),
1    KPER, KSTP, L,IL,IR,IC,Q
900 FORMAT(1H0, 4A4, ' PERIOD', I3, ' STEP', I3, ' WELL', I4, ' LAYER', I3, ' ROW ', I4, ' COL', I4, ' RATE', G15.7)
C5C-----IF CELL-BY-CELL FLOWS ARE TO BE SAVED THEN ADD THEM TO BUFFER.
     IF(IBD.EQ.1 .AND. ICBCFL.GE.0) CALL UBUDSV(KSTP, KPER, TEXT, IWELCB, BUFF, NCOL, NROW, NLAY, IOUT)
C5D-----PUMPING RATE IS POSITIVE(RECHARGE). ADD IT TO RATIN.
     IF(Q) 80, 100, 80
     RATIN=RATIN+Q
80 CONTINUE
C5E-----PUMPING RATE IS NEGATIVE(DISCHARGE). ADD IT TO RATOUT.
     90 RATOUT=RATOUT-Q
100 CONTINUE
C6------IF CELL-BY-CELL FLOWS WILL BE SAVED TO RECORD THEM
C6A------RECORD IN UNFORMATTED FILE IWELCB
     IF(IBD.EQ.1 .AND. ICBCFL.GE.0) CALL UBUDSV(KSTP, KPER, TEXT, IWELCB, &
       BUFF, NCOL, NROW, NLAY, IOUT)
C6B------RECORD IN ARC/INFO FILE
     IF(IBD.EQ.1 .AND. ICBCFL.LT.0) THEN
       INFOITEM='LAYER'

86
INFONAME='WELBUD'
OUTPATH=WELPATH (:INDEX(WELPATH, ',')-1)//INFONAME
CALL UBUDSVARC (KSTP, KPER, INFOITEM, OUTPATH, BUFF, NCOL, NROW, NLAY, IOUT, *9999)
ENDIF

C
C7------MOVE RATES INTO VBVL FOR PRINTING BY MODULE BAS1OT.
200 VBVL(3,MSUM)=RATIN
    VBVL(4,MSUM)=RATOUT
C
C8------MOVE RATES TIMES TIME STEP LENGTH INTO VBVL ACCUMULATORS.
    VBVL(1,MSUM)=VBVL(1,MSUM)+RATIN*DELT
    VBVL(2,MSUM)=VBVL(2,MSUM)+RATOUT*DELT
C
C9------MOVE BUDGET TERM LABELS INTO VBNM FOR PRINTING.
    VBNM(1,MSUM)=TEXT(1)
    VBNM(2,MSUM)=TEXT(2)
    VBNM(3,MSUM)=TEXT(3)
    VBNM(4,MSUM)=TEXT(4)
C
C10------INCREMENT BUDGET TERM COUNTER(MSUM).
    MSUM=MSUM+1
C
C11------RETURN
    RETURN
C
CE------ERRORS
C
9999 CALL INFORM ('\Abnormal Termination of Wel1bd_Arc_Subroutine',-1)
    RETURN 1
END

Added variables for module WEL1BDARC

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>INFONAME</td>
<td>Module</td>
<td>The name of the ARC/INFO file where cell-by-cell flow terms are recorded (passed argument consisting of the root name WELBUD which later will be appended by the stress period and time step).</td>
</tr>
<tr>
<td>INFOITEM</td>
<td>Module</td>
<td>The name of the ARC/INFO or INFO item where cell-by-cell flow terms are recorded (passed argument consisting of the root name LAYER which later will be appended by the layer number).</td>
</tr>
<tr>
<td>OUTPATH</td>
<td>Module</td>
<td>The path for the ARC/INFO file where cell-by-cell flow terms are recorded.</td>
</tr>
<tr>
<td>WELPATH</td>
<td>Package</td>
<td>The directory path to the ARC/INFO subdirectory where cell-by-cell flow terms are recorded.</td>
</tr>
</tbody>
</table>
Drain Package Modules

The DRN package (Drain) consists of four primary modules; of these, two primary modules (DRN1RPARC and DRN1BDARC) indicated below were modified.

DRN1RPARC

This module reads and prepares data for the DRN package. This module searches the control record for two variables, ITMP and DRNPATH and then reads values for drain parameters (layer, row, column, elevation, and hydraulic conductance) within the ARC/INFO file specified within the variable DRNPATH (fig. 17). Documentation of these modifications follows.

Figure 17.--Modified program elements for the DRN1RPARC module.
SUBROUTINE DRN1RPARC (DRAI, NDRAIN, MXDRN, IN, IOUT, E *)

C -----VERSION 3.0 25OCTOBER1991 DRN1RPARC
MODIFIED BY LEONARD L. ORZOL

 Mbps READ DRAIN LOCATIONS, ELEVATIONS, AND CONDUCTANCES
******************************************************************
C
C     SPECIFICATIONS:
C     ------------------------------------------------------------------
DIMENSION DRAI(5,MXDRN)
CHARACTER*132 BUFFER
CHARACTER*128 INFOPATH
CHARACTER*80 DRNPATH
CHARACTER*16 ITEMS(5)
INTEGER ACCESS, FNUM, IARC, NITEMS, NUMDRN, NUMREC
DATA ITEMS(1), ITEMS(2), ITEMS(3), ITEMS(4), ITEMS(5) /'LAYER', 'ROW',
& 'COLUMN', 'ELEVATION', 'COND'/
------------------------------------------------------------------
C
C1------READ CONTROL LINE FOR DRAIN CELLS INFORMATION
C
READ(IN, '(A132)', ERR=9990, END=9991) BUFFER
C
C1A------READ ITMP(NUMBER OF DRAIN CELLS OR FLAG TO REUSE DATA)
C
IF(BUFFER (11:11).EQ.' ' .OR. BUFFER (11:11).EQ. '') THEN
  READ (BUFFER,'(I10)',ERR=9990,END=9992) ITMP
  IARC=0
C
C1B------READ ITMP(NUMBER OF DRNLS OR FLAG SAYING REUSE DRAIN DATA) AND
---------DRNPATH TO ARC/INFO FILE
C
ELSE
  READ (BUFFER,'(I10,A80)',ERR=9992) ITMP, DRNPATH
  INFOPATH=DRNPATH (1:INDEX(DRNPATH,' ')-1)
  IARC=1
ENDIF
C
C2------TEST ITMP
IF(ITMP.GE.0) GO TO 50
C
C2A------IF ITMP<0 THEN REUSE DATA FROM LAST STRESS PERIOD.
WRITE(IOUT,7)
7 FORMAT(1H0,'REUSING DRAINS FROM LAST STRESS PERIOD')
RETURN
C
C3------IF ITMP=>0 THEN IT IS THE NUMBER OF DRAINS.
50 NDRAIN=ITMP
IF(NDRAIN.LE.MXDRN) GO TO 100
C
C4------IF NDRAIN>MXDRN THEN STOP
WRITE(IOUT,99) NDRAIN, MXDRN
99 FORMAT(1H0,'NDRAIN(',I4,') IS GREATER THAN MXDRN(',I4,')')
GO TO 9999
C
C5------PRINT NUMBER OF DRAINS IN THIS STRESS PERIOD.
100 WRITE(IOUT,1) NDRAIN
1 FORMAT(1H0,//1X,I5, ' DRAINS')
C
C6------IF THERE ARE NO DRAINS THEN RETURN.
IF(NDRAIN.EQ.0) GO TO 260
C
C7------READ AND PRINT DATA FOR EACH DRAIN.
WRITE(IOUT,3)
3 FORMAT(1H0,15X,'LAYER', 5X,'ROW', 5X
& 'COLUMN', 'ELEVATION', 'COND', 9X, 'DRAIN NO.'/1X,15X,60(''-'))
C
C7A------READING DRAIN INFORMATION FROM ASCII FILE
IF(IARC.LT.1) THEN
   DO 250 II=1,NDRAIN
      READ (IN,4) K, I, J, DRAI(4,II), DRAI(5,II)
      WRITE (IOUT,5) K, I, J, DRAI(4,II), DRAI(5,II)
 4       FORMAT(3I10,2F10.0)
      DRAI(1,II)=K
      DRAI(2,II)=I
      DRAI(3,II)=J
   CONTINUE
C
C7B------READING DRAIN INFORMATION FROM ARC/INFO FILE
C
ELSE
   ACCESS=1
   CALL INFO_OPENS (INFOPATH, ACCESS, FNUM, NUMREC, *9999)
C
C7BC------OPENS AND READS ITEMS (ROW, COLUMN, LAYER, ELEVATION, CONDUCTANCE)
C
   NITEMS=5
   NUMDRN=NDRAIN*NITEMS
   CALL INFO_READMULT (FNUM, NUMREC, NUMDRN, ITEMS, NITEMS, DRAI, *9999)
C
ENDIF
C
C7BB------TESTS EXISTENCE OF INFO FILE
C
C
C8------RETURN
260 RETURN
C
CE------ERRORS
C
9990 CALL INFORM ('\Unable to read Drain input package control record',-1)
9991 CALL INFORM ('\Abnormal Termination of Drn1rp_Arc_Subroutine',-1)
RETURN 1
9992 CALL INFORM ('\Missing Drain input package control record',-1)
9999 CALL INFORM ('\Abnormal Termination of Drn1rp_Arc_Subroutine',-1)
RETURN 1
END

Added variables for module DRN1RPARC

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCESS</td>
<td>Module</td>
<td>Flag indicating whether access to the ARC/INFO file is read or write.</td>
</tr>
<tr>
<td>BUFFER</td>
<td>Module</td>
<td>The control record is read into this variable and is parsed into the appropriate variables: ITMP and DRNPATH.</td>
</tr>
<tr>
<td>DRNPATH</td>
<td>Module</td>
<td>The complete path to the ARC/INFO file containing the items (INFOITEM) of interest.</td>
</tr>
<tr>
<td>FNUM</td>
<td>Module</td>
<td>Integer unit number used by this routine for the file specified by INFOPATH.</td>
</tr>
<tr>
<td>IARC</td>
<td>Module</td>
<td>Flag indicating whether the &quot;arc-section&quot; of the program code will be activated. &gt; 0, ASCII file storage &lt; 0, ARC/INFO file storage</td>
</tr>
<tr>
<td>ITEMS</td>
<td>Module</td>
<td>The names of the INFO item array either primary or redefined within the ARC/INFO file (specified by DRNPATH) containing the information. The item names are LAYER,</td>
</tr>
</tbody>
</table>
INFOPATH Submodule The path for the ARC/INFO file where the values for the drain array are stored.

NITEMS Module Integer value for the number of items within the file specified by INFOPATH.

NUMDRN Module Integer value for the number of records times the number of items needed within the file specified by INFOPATH.

NUMREC Module Integer value for the number of records within the file specified by INFOPATH.

This module calculates rates and accumulated volume of drainage from the ground-water flow system for the DRN package. Program code modifications were minimal, because program control passes to called submodule UBUDSVARC, instead of UBUDSV (fig. 18. Documentation of these modifications follows.

Figure 18--Modified program elements for the DRN1BDARC module.

SUBROUTINE DRN1BDARC (NDRAIN, MXDRN, VBNM, VBVL, MSUM, DRAI, DELT, HNEW, & NCOL, NROW, NLAY, IBOUND, KSTP, KPER, IDRNCB, ICBCFL, BUFF, IOUT, DRNPATH, E *)

C-----VERSION 3.0 25OCTOBER1991 DRN1BDARC
C MODIFIED BY LEONARD L. ORZOL
C
C*************************************************************************
C CALCULATE VOLUMETRIC BUDGET FOR DRAINS
C*************************************************************************
C
SPECIFICATIONS:
C
------------------------------------------------------------------
CHARACTER*128 OUTPATH
CHARACTER*80 DRNPATH
CHARACTER*32 INFONAME
CHARACTER*16 INFOITEM
CHARACTER*4 VBNM,TEXT
DOUBLE PRECISION HNEW
C
DIMENSION VBNM(4,MSUM),VBVL(4,MSUM),DRAI(5,MXDRN),
  1  HNEW(NCOL,NROW,NLAY),IBOUND(NCOL,NROW,NLAY),
  2  BUFF(NCOL,NROW,NLAY)
DIMENSION TEXT(4)
C
DATA TEXT(1),TEXT(2),TEXT(3),TEXT(4) /'    ',',',',',' DR','AINS'/
C
C1------INITIALIZE CELL-BY-CELL FLOW TERM FLAG (IBD) AND
C1------ACCUMULATORS (RATIN AND RATOUT).
  RATOUT=0.
  IBD=0
C
C2------IF THERE ARE NO DRAINS THEN DO NOT ACCUMULATE DRAIN FLOW
  IF(NDRAIN.LE.0) GO TO 200
C
C3------TEST TO SEE IF CELL-BY-CELL FLOW TERMS ARE NEEDED.
  IF(ICBCFL.EQ.0 .OR. IDRNCB.LE.0) GO TO 60
C
C3B------CELL-BY-CELL FLOW TERMS ARE NEEDED SET IBD AND CLEAR BUFFER.
   IBD=1
   DO 50 IL=1,NLAY
   DO 50 IR=1,NROW
   DO 50 IC=1,NCOL
     BUFF(IC,IR,IL)=0.
   50 CONTINUE
C
C4------FOR EACH DRAIN ACCUMULATE DRAIN FLOW
  DO 100 L=1,NDRAIN
C
C5------GET LAYER, ROW & COLUMN OF CELL CONTAINING REACH.
   IL=DRAI(1,L)
   IR=DRAI(2,L)
   IC=DRAI(3,L)
C
C6------IF CELL IS EXTERNAL IGNORE IT.
   IF(IBOUND(IC,IR,IL).LE.0) GO TO 100
C
C7------GET DRAIN PARAMETERS FROM DRAIN LIST.
   EL=DRAI(4,L)
   C=DRAI(5,L)
   HHNEW=HNEW(IC,IR,IL)
C
C8------IF HEAD LOWER THAN DRAIN THEN FORGET THIS CELL.
   IF(HHNEW.LE.EL) GO TO 100
C
C9------HEAD HIGHER THAN DRAIN. CALCULATE Q=C*(EL-HHNEW).
   Q=C*(EL-HHNEW)
C9------SUBTRACT Q FROM RATOUT.
  RATOUT=RATOUT-Q
C
C10------PRINT THE INDIVIDUAL RATES IF REQUESTED(IDRNCB<0).
  IF(IDRNCB.LT.0.AND.ICBCFL.NE.0) WRITE(IOUT,900) (TEXT(N),N=1,4),
  1   KPER,KSTP,L,IL,IR,IC,Q
    900 FORMAT(1H0,4A4,'   PERIOD',I3,'   STEP',I3,'   DRAIN',I4,
    1     '   LAYER',I3,'   ROW',I4,'   COL',I4,'   RATE',G15.7)
C
C11------IF C-B-C FLOW TERMS ARE TO BE SAVED THEN ADD Q TO BUFFER.
   IF(IBD.EQ.1) BUFF(IC,IR,IL)=BUFF(IC,IR,IL)+Q
  100 CONTINUE
C
C12------IF C-B-C FLOW TERMS WILL BE SAVED TO RECORD THEM.
C
C12A------RECORD IN UNFORMATTED FILE IDRNCB
IF(IBD.EQ.1 .AND. ICBCFL.GE.0) CALL UBUDSV(KSTP,KPER,TEXT,IDRNCB,
&                                         BUFF,NCOL,NROW,NLAY,IOUT)

C12B------RECORD IN ARC/INFO FILE
C
IF(IBD.EQ.1 .AND. ICBCFL.LT.0) THEN
  INFOITEM='LAYER'
  INFONAME='DRNBUD'
  OUTPATH=DRNPATH (:INDEX(DRNPATH,' ')-1)//INFONAME
  CALL UBUDSVARC (KSTP,KPER,INFOITEM,OUTPATH,BUFF,NCOL,NROW,NLAY,
&                                                             IOUT,
  E                   *9999)
ENDIF
C
C13------MOVE RATES,VOLUMES & LABELS INTO ARRAYS FOR PRINTING.
200 VBVL(3,MSUM)=0.
    VBVL(4,MSUM)=RATOUT
    VBVL(2,MSUM)=VBVL(2,MSUM)+RATOUT*DELT
    VBNM(1,MSUM)=TEXT(1)
    VBNM(2,MSUM)=TEXT(2)
    VBNM(3,MSUM)=TEXT(3)
    VBNM(4,MSUM)=TEXT(4)
C
C14------INCREMENT BUDGET TERM COUNTER
MSUM=MSUM+1
C
C15------RETURN
RETURN
C
CE------ERRORS
C
9999 CALL INFORM ('\Abnormal Termination of Drn1bd_Arc_Subroutine',-1)
RETURN 1
END

Added variables for module DRN1BDARC

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>DRNPATH</td>
<td>Package</td>
<td>The directory path to the ARC/INFO subdirectory where cell-by-cell flow terms are recorded.</td>
</tr>
<tr>
<td>INFONAME</td>
<td>Module</td>
<td>The name of the ARC/INFO file where cell-by-cell flow terms are recorded (passed argument consisting of the root name DRNBUD which later will be appended by the stress period and time step).</td>
</tr>
<tr>
<td>INFOITEM</td>
<td>Module</td>
<td>The name of the ARC/INFO or INFO item where cell-by-cell flow terms are recorded (passed argument consisting of the root name LAYER which later will be appended by the layer number).</td>
</tr>
<tr>
<td>OUTPATH</td>
<td>Module</td>
<td>The path for the ARC/INFO file where cell-by-cell flow terms are recorded.</td>
</tr>
</tbody>
</table>
Evapotranspiration Package Modules

The EVT package (Evapotranspiration) consists of four primary modules and five submodules; of these, two primary modules (EVT1RPARC and EVT1BDARC) indicated below were modified.

EVT1RPARC

This module reads and prepares data for the EVT package calls submodules U2RELARC and U2DINTARC (fig. 19. Documentation of the modifications to the code follows.

SUBROUTINE EVT1RPARC (NEVTOP,IEVT,EVTR,EXDP,SURF,DELR,DELC,
1                        NCOL,NROW,IN,IOUT,
E                      *)

C-----VERSION 3.0 25OCTOBER1991 EVT1RPARC
C MODIFIED BY LEONARD L. ORZOL

**************************************************************************
READ EVAPOTRANSPIRATION DATA
**************************************************************************

SPECIFICATIONS:
-----------------------------------------------
CHARACTER*16 INFOITEM
CHARACTER*4 ANAME
DIMENSION IEVT(NCOL,NROW),EVTR(NCOL,NROW),EXDP(NCOL,NROW),
1          SURF(NCOL,NROW),ANAME(6,4),DELR(NCOL),DELC(NROW)

DATA ANAME(1,1),ANAME(2,1),ANAME(3,1),ANAME(4,1),ANAME(5,1),
1 ANAME(6,1) /'    ','    ','  ET',' LAY','ER I','NDEX'/
DATA ANAME(1,2),ANAME(2,2),ANAME(3,2),ANAME(4,2),ANAME(5,2),
1 ANAME(6,2) /'ET','SUR','FACE'/
DATA ANAME(1,3),ANAME(2,3),ANAME(3,3),ANAME(4,3),ANAME(5,3),
1 ANAME(6,3) /'EVA','POTR','ANSP','IRAT','ION ','RATE'/
DATA ANAME(1,4),ANAME(2,4),ANAME(3,4),ANAME(4,4),ANAME(5,4),
1 ANAME(6,4) /'EXTI','NCTI','ON D','EPTH'/
C
C1------READ FLAGS SHOWING WHETHER DATA IS TO BE REUSED.
READ(IN,6,ERR=9990,END=9991)INSURF,INEVTR,INEXDP,INIEVT
6 FORMAT(4I10)
C
C2------TEST INSURF TO SEE WHERE SURFACE ELEVATION COMES FROM.
IF(INSURF.GE.0)GO TO 32
C
C2A------IF INSURF<0 THEN REUSE SURFACE ARRAY FROM LAST STRESS PERIOD
WRITE(IOUT,3)
3 FORMAT(1H0,'REUSING SURF FROM LAST STRESS PERIOD')
GO TO 35
C
C3-------IF INSURF=>0 THEN READ SURFACE.
C
32 INFOITEM='SURF'
CALL U2DRELARC (INFOITEM,SURF,ANAME(1,2),NROW,NCOL,0,IN,IOUT,
E *9999)
C
C4------TEST INEVTR TO SEE WHERE MAX ET RATE COMES FROM.
35 IF(INEVTR.GE.0)GO TO 37
C
C4A-----IF INEVTR<0 THEN REUSE MAX ET RATE.
WRITE(IOUT,4)
4 FORMAT(1H0,'REUSING EVTR FROM LAST STRESS PERIOD')
GO TO 45
C
C5-------IF INEVTR=>0 THEN READ MAX ET RATE.
37 INFOITEM='EVTR'
CALL U2DRELARC (INFOITEM,EVTR,ANAME(1,3),NROW,NCOL,0,IN,IOUT,
E *9999)
C
C6------MULTIPLY MAX ET RATE BY CELL AREA TO GET VOLUMETRIC RATE
DO 40 IR=1,NROW
   DO 40 IC=1,NCOL
      EVTR(IC,IR)=EVTR(IC,IR)*DELR(IC)*DELC(IR)
40 CONTINUE
C
C7------TEST INEXDP TO SEE WHERE EXTINCTION DEPTH COMES FROM
45 IF(INEXDP.GE.0)GO TO 47
C
C7A------IF INEXDP<0 REUSE EXTINCTION DEPTH FROM LAST STRESS PERIOD
WRITE(IOUT,5)
5 FORMAT(1H0,'REUSING EXDP FROM LAST STRESS PERIOD')
GO TO 48
C
C8-------IF INEXDP=>0 THEN READ EXTINCTION DEPTH
47 INFOITEM='EXDP'
CALL U2DRELARC (INFOITEM,EXDP,ANAME(1,4),NROW,NCOL,0,IN,IOUT,
E *9999)
C
C9-------IF OPTION(NEVTOP) IS 2 THEN WE NEED AN INDICATOR ARRAY.
48 IF(NEVTOP.NE.2)GO TO 50
C
C10------IF INIEVT<0 THEN REUSE LAYER INDICATOR ARRAY.
   IF(INIEVT.GE.0)GO TO 49
   WRITE(IOUT,2)
5 FORMAT(1H0,'REUSING IEVT FROM LAST STRESS PERIOD')
GO TO 50
C
C11------IF INIEVT=>0 THEN READ INDICATOR ARRAY.
49 INFOITEM='IEVT'
   CALL U2DINTARC(INFOITEM,IEVT,ANAME(1,1),NROW,NCOL,0,IN,IOUT,
   E *9999)
   C

95
C12------RETURN
50 RETURN
C
CE------ERRORS
C
9990 CALL INFORM ('\Unable to read Evt input package control',-1)
   CALL INFORM ('\Abnormal Termination of Evt1rp_Arc_Subroutine',-1)
   RETURN 1
9991 CALL INFORM ('\Missing Evt input package control record',-1)
9999 CALL INFORM ('\Abnormal Termination of Evt1rp_Arc_Subroutine',-1)
   RETURN 1
END

Added variables for module EVT1RPARC

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>INFOITEM</td>
<td>Module</td>
<td>The name of the ARC/INFO or INFO item where the values for the evapotranspiration array are stored (passed argument consisting of the root name SURF, EVTR, EXDP, and IEVT which later will be appended within utility U2DRELARC and U2DINTARC modules.</td>
</tr>
</tbody>
</table>

EVT1BDARC

This module calculates the rate and accumulated volume of Evapotranspiration for the EVT package calls submodules UBUDSVARC (fig. 20. Documentation of the modified module follows.

![Diagram](image-url)

Figure 20.--Modified program elements for the EVT1BDARC module.

SUBROUTINE EVT1BDARC (NEVTOP,IEVT,EVTR,EXDP,SURF,IBOUND,HNEW,
& NCOL,NROW,NLAY,DELT,VBVL,VBNM,MSUM,KSTP,KPER,
& IEVTCB,ICBCFL,BUFF,IOUT,EVTPATH,
E *)
C-----VERSION 3.0 25OCTOBER1991 EVT1BDARC
C MODIFIED BY LEONARD L. ORZOL
C
C*************************************************************************
C CALCULATE VOLUMETRIC BUDGET FOR EVAPOTRANSPIRATION
C*************************************************************************
C SPECIFICATIONS:
---

CHARACTER*128 OUTPATH
CHARACTER*80 EVTPATH
CHARACTER*32 INFONAME
CHARACTER*16 INFOITEM
CHARACTER*4 VBNM, TEXT
DOUBLE PRECISION HNEW
DIMENSION IEVT(NCOL,NROW), EVTR(NCOL, NROW), EXDP(NCOL, NROW), 
  SURF(NCOL, NROW), IBOUND(NCOL, NROW, NLAY), 
  VBVL(4,20), VBNM(4,20), HNEW(NCOL, NROW, NLAY), 
  BUFF(NCOL, NROW, NLAY)
DIMENSION TEXT(4)
DATA TEXT(1), TEXT(2), TEXT(3), TEXT(4) /'    ','    ','    ','  ET'/
---

C1-------CLEAR THE RATE ACCUMULATOR.
RATOUT=0
C
C2-------IF CELL-BY-CELL FLOW TERMS WILL BE SAVED THEN CLEAR THE BUFFER.
  IBD=0
  IF(IEVTCB.LE.0 .OR. ICBCFL.EQ.0) GO TO 5
  IBD=1
  DO 4 IL=1, NLAY
     DO 4 IR=1, NROW
        DO 4 IC=1, NCOL
           BUFF(IC, IR, IL)=0.
     4 CONTINUE
C
C3------PROCESS EACH HORIZONTAL CELL LOCATION
  5 DO 10 IR=1, NROW
     DO 10 IC=1, NCOL
C
C4------SET THE LAYER INDEX EQUAL TO 1.
     IL=1
C
C5------IF OPTION 2 IS SPECIFIED THEN GET LAYER INDEX FROM IEVT ARRAY
     IF(NEVTOP.EQ.2) IL=IEVT(IC,IR)
C
C6------IF CELL IS EXTERNAL THEN IGNORE IT.
     IF(IBOUND(IC, IR, IL).LE.0) GO TO 10
     C=EVTR(IC, IR)
     S=SURF(IC, IR)
     H=HNEW(IC, IR, IL)
C
C7------IF AQUIFER HEAD => SURF, SET Q=MAX ET RATE
     IF(H.LT.S) GO TO 7
     Q=-C
     GO TO 9
C
C8------IF DEPTH=>EXTINCTION DEPTH, ET IS 0
     7 X=EXDP(IC, IR)
     D=S-H
     IF(D.GE.X) GO TO 10
C
C9------LINEAR RANGE . Q=-EVTR(H-EXEL)/EXDP
     Q=C*D/X-C
C
C10-----ACCUMULATE TOTAL FLOW RATE
     9 RATOUT=RATOUT-Q
C
C11------IF CELL-BY-CELL FLOW TERMS TO BE SAVED THE ADD Q TO BUFFER.
     IF(IBD.EQ.1) BUFF(IC, IR, IL)=Q
  10 CONTINUE
C12-----IF C-B-C TO BE SAVED CALL MODULE UBUDSV TO RECORD THEM.
C
C12A------RECORD IN UNFORMATTED FILE IEVTCB
C
        IF(IBD.EQ.1 .AND. ICBCFL.GE.0) CALL UBUDSV(KSTP,KPER,TEXT,IEVTCB,
 &                          BUFF,NCOL,NROW,NLAY,IOUT)
C
C12B------RECORD IN ARC/INFO FILE
C
        IF(IBD.EQ.1 .AND. ICBCFL.LT.0) THEN
            INFOITEM='LAYER'
            INFONAME='EVTBUD'
            OUTPATH=EVTPATH (:INDEX(EVTPATH,' ')-1)//INFONAME
            CALL UBUDSVARC (KSTP,KPER,INFOITEM,OUTPATH,BUFF,NCOL,NROW,NLAY,
 &                          IOUT,
 &                          *9999)
        ENDIF
C
C13-----MOVE TOTAL ET RATE INTO VBVL FOR PRINTING BY BAS1OT.
     VBVL(3,MSUM)=0.
     VBVL(4,MSUM)=RATOUT
C
C14-----ADD ET(ET_RATE TIMES STEP LENGTH) TO VBVL
     VBVL(1,MSUM)=0.
     VBVL(2,MSUM)=VBVL(2,MSUM)+RATOUT*DELT
C
C15-----MOVE BUDGET TERM LABELS TO VBNM FOR PRINT BY MODULE BAS1OT
     VBNM(1,MSUM)=TEXT(1)
     VBNM(2,MSUM)=TEXT(2)
     VBNM(3,MSUM)=TEXT(3)
     VBNM(4,MSUM)=TEXT(4)
C
C16-----INCREMENT BUDGET TERM COUNTER
     MSUM=MSUM+1
C
C17-----RETURN
     RETURN
C
CE------ERRORS
C
9999 CALL INFORM ("\Abnormal Termination ofEvt1bd_Arc_Subroutine",-1)
     RETURN 1
END

Added variables for module EVT1BDARC

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>EVTPATH</td>
<td>Package</td>
<td>The directory path to the ARC/INFO subdirectory where cell-by-cell flow terms are recorded.</td>
</tr>
<tr>
<td>INFONAME</td>
<td>Submodule</td>
<td>The name of the ARC/INFO file where cell-by-cell flow terms are recorded (passed argument consisting of the root name EVTBD which later will be appended by the stress period and time step).</td>
</tr>
<tr>
<td>INFOITEM</td>
<td>Submodule</td>
<td>The name of the ARC/INFO or INFO item where cell-by-cell flow terms are recorded (passed argument consisting of the root name LAYER which later will be appended by the layer number).</td>
</tr>
<tr>
<td>OUTPATH</td>
<td>Submodule</td>
<td>The path for the ARC/INFO file where cell-by-cell flow terms are recorded.</td>
</tr>
</tbody>
</table>
General-Head Boundary Package Modules

The GHB package (General-Head Boundary) consists of four primary modules; of these, two primary modules (GHB1RPARC and GHB1BDARC) indicated below were modified.

**GHB1RPARC**

This module reads and prepares data for the GHB. This module searches the control record for two variables, ITMP and GHBPATH, and to read values for well parameters (layer, row, column, and Q) within the ARC/INFO file specified within the variable GHBPATH (fig. 21). Documentation of the modified module follows.

![Diagram of GHB1RPARC module]

Figure 21.--Modified program elements for the GHB1RPARC module.
SUBROUTINE GHB1RPARC (BNDS,NBOUND,MXBND,IN,IOUT,
E)

C-----VERSION 3.0 25OCTOBER1991 GHB1RPARC
C               MODIFIED BY LEONARD L. ORZOL

C     ******************************************************************
C     READ DATA FOR GHB
C     ******************************************************************

C     SPECIFICATIONS:
------------------------------------------------------------------
DIMENSION BNDS(5,MXBND)
CHARACTER*132 BUFFER
CHARACTER*128 INFOPATH
CHARACTER*80 GHBPATH
CHARACTER*16 ITEMS(5)
INTEGER ACCESS,FNUM,IARC,NITEMS,NUMGHB,NUMREC
&      /'LAYER','ROW','COLUMN','BOUNDARYHEAD','COND'/
------------------------------------------------------------------

C1------GENERAL HEAD BOUNDARIES
C
READ(IN,'(A132)',ERR=9990,END=9991) BUFFER

C1A------READ ITMP(# OF GENERAL HEAD BOUNDS OR FLAG TO REUSE DATA.)
C
IF(BUFFER (11:11).EQ.'' .OR. BUFFER (11:11).EQ.' ') THEN
READ (BUFFER,'(I10)',ERR=9992) ITMP
IARC=0
C
C1B------READ ITMP(# OF GENERAL HEAD BOUNDS OR FLAG TO REUSE DATA) AND
C---------GHBPATH TO ARC/INFO FILE
C
ELSE
READ (BUFFER,'(I10,A80)',ERR=9992) ITMP,GHBPATH
INFOPATH=GHBPATH (1:INDEX(GHBPATH,' ')-1)
IARC=1
ENDIF

C2------TEST ITMP
IF(ITMP.GE.0) GO TO 50
C
C2A-----IF ITMP<0 THEN REUSE DATA FROM LAST STRESS PERIOD
WRITE(IOUT,7)
7 FORMAT(1H0,'REUSING HEAD-DEPENDENT BOUNDS FROM LAST STRESS',
1     ' PERIOD')
GO TO 260
C
C3------IF ITMP=>0 THEN IT IS THE # OF GENERAL HEAD BOUNDS.
50 NBOUND=ITMP
C
C4------IF MAX NUMBER OF BOUNDS IS EXCEEDED THEN STOP
IF(NBOUND.LE.MXBND) GO TO 100
WRITE(IOUT,99) NBOUND,MXBND
99 FORMAT(1H0,'NBOUND(',I4,') IS GREATER THAN MXBND(',I4,')')
C
C4A-----ABNORMAL STOP
GO TO 9999
C
C5------PRINT # OF GENERAL HEAD BOUNDS THIS STRESS PERIOD
100 WRITE(IOUT,1) NBOUND
1 FORMAT(1H0,15X,'LAYER',5X,'ROW',5X,'COLUMN',5X,'BOUNDARYHEAD',5X,'COND'/1X,15X,60('-'))
### C7A------READING GENERAL HEAD BOUNDARY INFORMATION FROM ASCII FILE

If (IARC.LT.1) then

```plaintext
DO 250 II=1,NBOUND
  READ (IN,4) K,I,J,BNDS(4,II),BNDS(5,II)
  WRITE (IOUT,5) K,I,J,BNDS(4,II),BNDS(5,II),II
  FNS(1,II)=K
  FNS(2,II)=I
  FNS(3,II)=J
250 CONTINUE
```

### C7B------READING GENERAL HEAD INFORMATION FROM ARC/INFO FILE

Else

```plaintext
ACCESS=1
CALL INFO_OPENS (INFOPATH,ACCESS,FNUM,NUMREC,*9999)
```

### C7BC------OPENS AND READS ITEMS (LAYER, ROW, COLUMN, BOUNDARY HEAD, AND CONDUCTANCE)

NITEMS=5
NUMGBH=NBOUND*NITEMS
CALL INFO_READMULT (FNUM,NUMREC,NUMGBH,ITEMS,NITEMS,BNDS,*9999)

```plaintext
DO 700 II=1,NBOUND
  K=BNDS(1,II)
  I=BNDS(2,II)
  J=BNDS(3,II)
  WRITE (IOUT,5) K,I,J,BNDS(4,II),BNDS(5,II),II
700 CONTINUE
```

### C7BA------TESTS EXISTENCE OF INFO FILE

```plaintext
ENDIF
```

### C8------RETURN

260 RETURN

### CE------ERRORS

```plaintext
9990 CALL INFORM (\"Unable to read Ghb input package control\",-1)
9991 CALL INFORM (\"Abnormal Termination of Ghblrp_Arc_Subroutine\",-1)
RETURN 1
9992 CALL INFORM (\"Missing Ghb input package control record\",-1)
9993 CALL INFORM (\"Abnormal Termination of Ghblrp_Arc_Subroutine\",-1)
RETURN 1
9999 CALL INFORM (\"Unable to read parameters from Ghb input record\",-1)
RETURN 1
```

---

Added variables for module GHB1RPARC:

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCESS</td>
<td>Module</td>
<td>Flag indicating whether access to the ARC/INFO file is read or write.</td>
</tr>
<tr>
<td>BUFFER</td>
<td>Module</td>
<td>The control record is read into this variable and is parsed into the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>appropriate variables: ITMP and GHBPATH.</td>
</tr>
<tr>
<td>GHBPATH</td>
<td>Module</td>
<td>The complete path to the ARC/INFO file containing the item</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(INFOITEM) of interest.</td>
</tr>
</tbody>
</table>
IARC  Module  Flag indicating whether the "arc-section" of the program code will be activated. > 0, ASCII file storage < 0, ARC/INFO file storage

ITEMS  Module  The names of the INFO item array either primary or redefined within the ARC/INFO file (specified by GHBPATH) containing the information. The item names are LAYER, ROW, COLUMN, BOUNDARYHEAD, and COND.

INFOPATH  Module  The path for the ARC/INFO file where the values for the drawdown array are stored.

NITEMS  Module  Integer value for the number of items within the file specified by INFOPATH.

NUMGHB  Module  Integer value for the number of records times the number of items that are needed within the file specified by INFOPATH.

NUMREC  Module  Integer value for the number of records within the file specified by INFOPATH.

GHB1BDARC

This module calculates rates and volumes of flow to and from general-head boundaries of the ground-water flow system for the GHB package and calls submodule UBUDSVARC (fig. 22). Documentation of the changes in program code follows.

![Diagram](image-url)

Figure 22.--Modified program elements for the GHB1BDARC module.
SUBROUTINE GHB1BDARC (NBOUND, MBND, VBNM, VBVL, MSUM, BNDS, DELT, HNEW, NCOL, NROW, NLAY, IBOUND, KSTP, KPER, IGHBCB, ICBCFL, BUFF, IOUT, GHBPATH, E *)

C ------VERSION 3.0 25OCTOBER1991 GHB1BDARC
C MODIFIED BY LEONARD L. ORZOL

C-************************************************************************
C CALCULATE VOLUMETRIC BUDGET FOR GHB
C-************************************************************************

SPECIFICATIONS:
-****************************************************************************
CHARACTER*128 OUTPATH
CHARACTER*80 GHBPATH
CHARACTER*32 INFONAME
CHARACTER*16 INFOITEM
CHARACTER*4 VBNM, TEXT
DOUBLE PRECISION HNEW
DIMENSION VBNM(4, MSUM), VBVL(4, MSUM), BNDS(5, MXBND),
1 HNEW(NCOL, NROW, NLAY), IBOUND(NCOL, NROW, NLAY),
2 BUFF(NCOL, NROW, NLAY)
DIMENSION TEXT(4)
DATA TEXT(1), TEXT(2), TEXT(3), TEXT(4) /' HEA', 'D DE', 'P BO', 'UNDS' /
-****************************************************************************

C1------INITIALIZE CELL-BY-CELL FLOW TERM FLAG (IBD) AND
C1------ACCUMULATORS (RATIN AND RATOUT)
IBD=0
RATOUT=0.
RATIN=0.

C2------IF NO BOUNDARIES THEN KEEP ZEROES IN ACCUMULATORS.
IF(NBOUND.EQ.0) GO TO 200

C3------TEST TO SEE IF CELL-BY-CELL FLOW TERMS ARE NEEDED.
IF(ICBCFL.EQ.0 .OR. IGHBCB.LE.0) GO TO 10

C3A------SINCE CELL-BY-CELL FLOW TERMS ARE NEEDED CLEAR BUFFER & SET
C3A------THE FLAG IBD.
IBD=1
DO 5 IL=1, NLAY
DO 5 IR=1, NROW
DO 5 IC=1, NCOL
BUFF(IC, IR, IL)=0.
5 CONTINUE

C4------FOR EACH GENERAL HEAD BOUND ACCUMULATE FLOW INTO AQUIFER
10 DO 100 L=1, NBOUND

C5------GET LAYER, ROW AND COLUMN OF EACH GENERAL HEAD BOUNDARY.
   IL=BNDS(1, L)
   IR=BNDS(2, L)
   IC=BNDS(3, L)

C6------IF CELL IS EXTERNAL THEN IGNORE IT.
   IF(IBOUND(IC, IR, IL).LE.0) GO TO 100

C7------GET PARAMETERS FROM BOUNDARY LIST.
   HNNEW=HNEW(IC, IR, IL)
   HB=BNDS(4, L)
   C=BNDS(5, L)

C8------CALCULATE THE FLOW RATE INTO THE CELL
   RATE=C*(HB-HNNEW)

C9------PRINT THE INDIVIDUAL RATES IF REQUESTED(IGHBCB<0).
   IF(IGHBCB.LT.0.AND.ICBCFL.NE.0) WRITE(IOUT, 900) (TEXT(N), N=1, 4),
      KPER, KSTP, L, IL, IR, IC, RATE
900 FORMAT(1H0, 4A4, ' PERIOD', I3, ' STEP', I3, ' BOUNDARY', I4,
      1 ' LAYER', I3, ' ROW', I4, ' COL', I4, ' RATE', G15.7)
C10------IF CELL-BY-CELL TERMS ARE TO BE SAVED THEN PUT RATE IN BUFFER
   IF(IBD.EQ.1) BUFF(IC,IR,IL)=BUFF(IC,IR,IL)+RATE
C
C11------SEE IF FLOW IS INTO AQUIFER OR OUT OF AQUIFER.
   IF(RATE)94,100,96
C
C12------FLOW IS OUT OF AQUIFER SUBTRACT RATE FROM RATOUT
   94 RATOUT=RATOUT-RATE
   GO TO 100
C
C13------FLOW IS INTO AQIFER ADD RATE TO RATIN
   96 RATIN=RATIN+RATE
   CONTINUE
C
C14------IF CELL-BY-CELL TERMS ARE TO BE SAVED THEN RECORD
C
C14A------RECORD IN UNFORMATTED FILE IGHBCB
C   IF(IBD.EQ.1 .AND. ICBCFL.GE.0) CALL UBUDSV(KSTP,KPER,TEXT,IGHBCB,
      &                          BUFF,NCOL,NROW,NLAY,IOUT)
C
C14B------RECORD IN ARC/INFO FILE
C   IF(IBD.EQ.1 .AND. ICBCFL.LT.0) THEN
      INFOITEM='LAYER'
      INFONAME='GHBBUD'
      OUTPATH=GHBPATH (:INDEX(GHBPATH, ' ')-1)//INFONAME
      CALL UBUDSVARC (KSTP,KPER,INFOITEM,OUTPATH,BUFF,NCOL,NROW,NLAY,
         &                         *9999)
   ENDIF
C
C15------MOVE RATES, VOLUMES AND LABELS INTO ARRAYS FOR PRINTING
   200 VBVL(3,MSUM)=RATIN
      VBVL(1,MSUM)=VBVL(1,MSUM)+RATIN*DELT
      VBVL(4,MSUM)=RATOUT
      VBVL(2,MSUM)=VBVL(2,MSUM)+RATOUT*DELT
      VBNM(1,MSUM)=TEXT(1)
      VBNM(2,MSUM)=TEXT(2)
      VBNM(3,MSUM)=TEXT(3)
      VBNM(4,MSUM)=TEXT(4)
C
C16------INCREMENT THE BUDGET TERM COUNTER
   MSUM=MSUM+1
C
C17------RETURN
   RETURN
C
CE------ERRORS
C
   9999 CALL INFORM ('$\Abnormal Termination of Ghb1bd_Arc_Subroutine\$',-1)
   RETURN 1
END

Added variables for module GHB1BDARC

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>GHBPATH</td>
<td>Package</td>
<td>The directory path to the ARC/INFO subdirectory where cell-by-cell flow terms are recorded.</td>
</tr>
<tr>
<td>INFONAME</td>
<td>Module</td>
<td>The name of the ARC/INFO file where cell-by-cell flow terms are recorded (passed argument consisting of the root name GHB1BDARC which later will be appended by the stress period and time step).</td>
</tr>
<tr>
<td>INFOITEM</td>
<td>Module</td>
<td>The name of the ARC/INFO or INFO item where cell-by-cell flow terms are recorded (passed argument consisting of the root</td>
</tr>
</tbody>
</table>
name LAYER which later will be appended by the layer number).

OUTPATH Module The path for the ARC/INFO file where cell-by-cell flow terms are recorded.

Streamflow-Routing Package Modules

The STR package (Streamflow-Routing) consists of four primary modules; of these, two primary modules (STR1RPARC and STR1BDARC) indicated below were modified and a new module (STR1SRARC) was created to record streamflow output data to ARC/INFO files by stream segment and reach.

STR1RPARC

This module reads and prepares data for the STR. This module searches the second control record for four variables: ITMP, IRDFLG, IPTFLG, and STRPATH, and to read values for well parameters (layer, row, column, seg, reach, flow, stage, cond, sbot, and stop) within the ARC/INFO file specified within the variable STRPATH (fig. 23). Documentation of the modified module follows.

![Diagram of modified program elements for the STR1RPARC module.](image)

Figure 23a.--Modified program elements for the STR1RPARC module.
Figure 23b.—Modified program elements for the STR1RPARC module.
SUBROUTINE STR1RPARC (STRM, ISTRM, NSTREM, MXSTRM, IN, IOUT, ITRBAR,
  &
  &  NDIV, NSS, NTRIB, IDIVAR, ICALC, IPTFLG,
  &
  &  *)
C
C-----VERSION 001 17JAN1989 STR1RPARC
C               BY D.E.PRUDIC
C-----VERSION 3.0 25OCTOBER1991 STR1RPARC
C               MODIFIED BY LEONARD L. ORZOL
C     *****************************************************************
C     READ STREAM DATA:  INCLUDES STREAM SEGMENT NUMBER, NODE SEQUENCE C
C            OF STREAM SEGMENT, FLUX INTO MODEL AT BOUNDARY, STREAM C
C            STAGE, STREAM BED CONDUCTANCE, AND STREAM BOTTOM ELEVATIONC
C     *****************************************************************
C
C SPECIFICATIONS:
C
REAL STREM(131072)
DIMENSION STRM(11,MXSTRM), ISTRM(5,MXSTRM), ITRBAR(NSS,NTRIB),
  1 IDIVAR(NSS)
CHARACTER*132 BUFFER
CHARACTER*128 INFOPATH
CHARACTER*80 STRPTH, PATH
CHARACTER*16 ITEM, ITEMS(13)
INTEGER FNUM, NUMREC, NITEMS, NUMSTR, ACCESS, STRDEX
COMMON /STRCMN/ STREM
DATA ITEMS(1), ITEMS(2), ITEMS(3), ITEMS(4), ITEMS(5), ITEMS(6),
  & ITEMS(7), ITEMS(8), ITEMS(9), ITEMS(10), ITEMS(11), ITEMS(12),
  & ITEMS(13)/'LAYER', 'ROW', 'COLUMN', 'SEG', 'REACH', 'FLOW', 'STAGE',
  & 'COND', 'SBOT', 'STOP', 'WIDTH', 'SLOPE', 'ROUGH'/
C     -----------------------------------------------------------------
C                                    --
C1A-----IF MXSTREAM IS LESS THAN 1 THEN STREAM IS INACTIVE. RETURN.  
C IF(MXSTRM.LT.1) RETURN
C
C1B------READ CONTROL LINE FOR WELL INFORMATION
C
READ(IN,'(A132)',ERR=9990,END=9991) BUFFER
C
C1BA------READ ITMP (NUMBER OF STREAM CELLS OR FLAG TO REUSE DATA) 
C IF(BUFFER (31:31).EQ.'' .OR. BUFFER (31:31).EQ.' ') THEN
C          READ(BUFFER,'(3I10)',ERR=9992) ITMP, IRDFLG, IPTFLG
C     IARC=0
C
C1BB------READ ITMP (NUMBER OF WELLS OR FLAG SAYING REUSE STREAM DATA)
C---------AND STRPTH TO ARC/INFO FILE
C
ELSE
READ(BUFFER,'(3I10,A80)',ERR=9992) ITMP, IRDFLG, IPTFLG, STRPTH
INFOPATH=STRPTH (1:INDEX(STRPTH,' ')-1)
IARC=1
ENDIF
C
C2A------IF ITMP <0 THEN REUSE DATA FROM LAST STRESS PERIOD.  
C IF(ITMP.GE.0)GO TO 50
WRITE(IOUT,2)
2 FORMAT(1H0,'REUSING STREAM NODES FROM LAST STRESS PERIOD')
RETURN
C
C2B------IF ITMP=> ZERO THEN IT IS THE NUMBER OF STREAM CELLS
C 50 NSTREM=ITMP
C
C3A------IF NSTREM=MXSTRM THEN STOP.
C IF(NSTREM.LE.MXSTRM)GO TO 100
WRITE(IOUT,99) NSTREM, MXSTRM
99 FORMAT(1H0,'NSTREM(',I4,') IS GREATER THAN MXSTRM(',I4,')')
RETURN 1
C
C3B------PRINT NUMBER OF STREAM CELLS IN THIS STRESS PERIOD.
C 100 IF(IRDFLG.GE.0) WRITE(IOUT,3)NSTREM
  3 FORMAT(1H0,//1X,I5,' STREAM NODES')
C4------IF THERE ARE NO STREAM CELLS THEN RETURN.
C   IF(NSTREM.EQ.0) RETURN
C5------READ AND PRINT DATA FOR EACH STREAM CELL.
C   IF(IRDFLG.GE.0) WRITE(IOUT,4)
   4 FORMAT(1H0,3X,'LAYER   ROW    COL    SEGMENT   REACH   STREAMFLOW
1      STREAM   STREAMBED     STREAMBED BOT  STREAMBED TOP',/27X,
2'NUMBER   NUMBER                   STAGE   CONDUCTANCE      ELEVAT
3TION      ELEVATION',/3X,110('-'))
C
C5A------OPENS ASCII FILE AND READS ITEMS (ROW,COLUMN,LAYER,SEG,
C---------REACH,FLOW,STAGE,COND,SBOT,STOP)
C   IF(IARC.LT.1) THEN
     DO 250 II=1,NSTREM
      READ(IN,5,ERR=9993)
      &                  K,I,J,ISTRM(4,II),ISTRM(5,II),STRM(1,II),
1                  STRM(2,II),STRM(3,II),STRM(4,II),STRM(5,II)
      5       FORMAT(5I5,F15.0,4F10.0)
      IF(IRDFLG.GE.0) WRITE(IOUT,6)K,I,J,ISTRM(4,II),ISTRM(5,II),
1        STRM(1,II),STRM(2,II),STRM(3,II),STRM(4,II),STRM(5,II)
      6       FORMAT(1X,3X,I4,2I7,2I9,7X,G11.4,G12.4,G11.4,4X,2G13.4)
      ISTRM(1,II)=K
      ISTRM(2,II)=I
      ISTRM(3,II)=J
     250    CONTINUE
C
C5B------OPENS ARC/INFO FILE AND READS ITEMS (ROW,COLUMN,LAYER,
C---------SEG,REACH,FLOW,STAGE,COND,SBOT,STOP)
C   ELSE
     ACCESS=1
     CALL INFO_OPENS (INFOPATH,ACCESS,FNUM,NUMREC,
E                          *9999)
     NITEMS=5
     DO 260 N=1,NITEMS
      CALL INFO_READREAL (FNUM,NUMREC,ITEMS(N),STREM,
E                          *9999)
     DO 270 I=1,NUMREC
      ISTRM(N,I)=STREM(I)
     270    CONTINUE
     260    CONTINUE
     NITEMS=10
     DO 280 N=6,NITEMS
      ITEM=ITEMS(N)
      CALL INFO_READREAL (FNUM,NUMREC,ITEM,STREM,
E                          *9999)
      DO 290 I=1,NUMREC
       STRM(N-5,I)=STREM(I)
      290    CONTINUE
     280    CONTINUE
     CALL INFO_CLOSING (FNUM)
     IF(IRDFLG.GE.0) WRITE(IOUT,6) (ISTRM(1,II),ISTRM(2,II),
1      ISTRM(3,II),ISTRM(4,II),ISTRM(5,II),STRM(1,II),
2      STRM(2,II),STRM(3,II),STRM(4,II),STRM(5,II), II=1,NSTREM)
   ENDDIF
C
C6------READ AND PRINT DATA IF HEADS IN STREAMS ARE TO BE CALCULATED
C   IF(ICALC.LE.0) GO TO 300
C   IF(IRDFLG.GE.0) WRITE(IOUT,7)
   7 FORMAT(1H0,3X,'LAYER',3X,'ROW',4X,'COL   ',3X,'SEGMENT',3X,
1      'REACH',8X,'STREAM',13X,'STREAM',10X,'ROUGH',/27X,'NUMBER',3X,
2      'NUMBER',8X,'WIDTH',14X,'SLOPE',10X,'COEF.',/3X,110('-'))
C
C6A------NON-ARC METHOD
C   IF(IARC.LT.1) THEN
     DO 600 II=1,NSTREM
      READ(IN,8,ERR=9994) STRM(6,II),STRM(7,II),STRM(8,II)
      8 FORMAT(3F10.0)
     600    CONTINUE
IF(IRDFLG.GE.0) WRITE(IOUT,9) ISTRM(1,II), ISTRM(2,II),
 1 ISTRM(3,II), ISTRM(4,II), ISTRM(5,II), STRM(6,II),
 2 STRM(7,II), STRM(8,II)
9 FORMAT(1X,8X,I4,I9,I8,I10,I20,10X,
 1                                 G12.4,5X,G13.4,4X,G12.4)
600 CONTINUE
C
C6B------ARC METHOD
C
ELSE
ACCESS=1
CALL INFO_OPENS (INFOPATH, ACCESS, FNUM, NUMREC,
 1                             *9999)
NITEMS=13
DO 610 N=11,NITEMS
ITEM=ITEMS(N)
CALL INFO_READREAL (FNUM, NUMREC, ITEM, STREM,
 1                             *9999)
DO 620 I=1,NUMREC
STREM(N-5,I)=STREM(I)
620       CONTINUE
610    CONTINUE
CALL INFO_CLOSING (FNUM)
IF(IRDFLG.GE.0) WRITE(IOUT,9) (ISTRM(1,II), ISTRM(2,II),
 1 ISTRM(3,II), ISTRM(4,II), ISTRM(5,II), STRM(6,II),
 2 STRM(7,II), STRM(8,II), II=1,NSTREM)
ENDIF
C                                                                      C
C7------INITIALIZE ALL TRIBUTARY STREAM SEGMENTS TO ZERO.              C
300 DO 320 IK=1,NSS
      DO 320 JK=1,NTRIB
        ITRBAR(IK,JK)=0
320 CONTINUE
C                                                                      C
C8-----INITIALIZE DIVERSION STREAM SEGMENT ARRAY TO ZERO.              C
DO 325 IK=1,NSS
      IDIVAR(IK)=0
325 CONTINUE
C                                                                      C
C9-----READ AND PRINT TRIBUTARY STREAM SEGMENTS                        C
IF(NTRIB.LE.0) GO TO 343
IF(IRDFLG.GE.0) WRITE(IOUT,10)NTRIB
10 FORMAT(1H0,30X,'MAXIMUM NUMBER OF TRIBUTARY STREAMS IS ',I5,//1X,
 1 20X,'STREAM SEGMENT',15X,'TRIBUTARY STREAM SEGMENT NUMBERS')
C
C9A----NON-ARC METHOD
C
IF(IARC.LT.1) THEN
DO 340 IK=1,NSS
      READ(IN,11,ERR=9995) (ITRBAR(IK,JK),JK=1,NTRIB)
11       FORMAT(10I5)
      IF(IRDFLG.GE.0) WRITE(IOUT,12)IK,(ITRBAR(IK,JK),JK=1,NTRIB)
12       FORMAT(20X,I5,20X,10I5)
340    CONTINUE
C
C9B----ARC METHOD
C
ELSE
ACCESS=1
STRPTH=INFOPATH
STRDEX=INDEX (STRPTH,’ ‘)
IF(STRDEX.LE.0) THEN
      STRDEX=INDEX (STRPTH,’!’)
      STRDEX=STRDEX+INDEX (STRPTH (STRDEX+1:LEN (STRPTH)),’!’)
ELSE
      STRDEX=STRDEX+INDEX (STRPTH (STRDEX+1:LEN (STRPTH)),’ ‘)
ENDIF
PATH=STRPTH (STRDEX+1:LEN (STRPTH))
STRPTH=STRPTH (1:STRDEX)
PATH=PATH (INDEX (PATH,’ ’):LEN (PATH))
INDEXPATH=INDEX (STRPTH (1:INDEX (STRPTH,’ ‘)-1))‘/’‘TRIB’//
6 PATH (1:INDEX (PATH,’ ‘)-1)
CALL INFO_OPENS (INFOPATH, ACCESS, FNUM, NUMREC, *9999)

DO 900 JK=1,NTRIB
   ICHAR=16
   IPER=0
   ISTP=0
   ITEM='ITRIB'
   CALL INFO_NAMING (ITEM, IPER, ISTP, JK, ICHAR)
   CALL INFO_READREAL (FNUM, NUMREC, ITEM, STREM, *9999)

DO 910 IK=1,NSS
   ITRBAR(IK,JK)=STREM(IK)
910 CONTINUE
900 CONTINUE
CALL INFO_CLOSING (FNUM)
IF(IRDFLG.GE.0) THEN
   DO 920 IK=1,NSS
      WRITE(IOUT,12) IK,(ITRBAR(IK,JK), JK=1,NTRIB)
920 CONTINUE
ENDIF
ENDIF
C
C----READ AND PRINT STREAM DIVERSIONS
C
IF(NDIV.LE.0) GO TO 350
IF(IRDFLG.GE.0) WRITE(IOUT,13)
13 FORMAT(1H0,10X,'DIVERSION SEGMENT NUMBER',10X,1H0,'UPSTREAM SEGMENT NUMBER')

C
C---NON-ARC METHOD
C
IF(IARC.LT.1) THEN
   DO 345 IK=1,NSS
      READ(IN,14,ERR=9996) IDIVAR(IK)
14       FORMAT(I10)
      IF(IRDFLG.GE.0) WRITE(IOUT,15) IK,IDIVAR(IK)
15       FORMAT(20X,I5,28X,I5)
345 CONTINUE
ENDIF

C
C----ARC METHOD
C
ELSE
   ACCESS=1
   STRPTH=INFOPATH
   STRDEX=INDEX(STRPTH,':')
   IF(STRDEX.LE.0) THEN
      STRDEX=INDEX(STRPTH,'!')
      STRDEX=STRDEX+INDEX(STRPTH (STRDEX+1:LEN(STRPTH)),'!')
   ELSE
      STRDEX=STRDEX+INDEX(STRPTH (STRDEX+1:LEN(STRPTH)),':')
   ENDIF
   STRPTH=STRPTH (1:STRDEX)
   PATH=STRPTH (STRDEX+1:LEN(STRPTH))
   INFOPATH=STRPTH(1:INDEX(STRPTH,'_')-1)//'DIV'//
   &   PATH (1:INDEX(PATH,'_')-1)
   CALL INFO_OPENS (INFOPATH, ACCESS, FNUM, NUMREC, *9999)
   ITEM='IUPSEG'
   CALL INFO_READINT (FNUM, NUMREC, ITEM, IDIVAR, *9999)
   CALL INFO_CLOSING (FNUM)
   IF(IRDFLG.GE.0) WRITE(IOUT,15) (IK,IDIVAR(IK), IK=1,NSS)
ENDIF
C
C----SET FLOW OUT OF REACH, FLOW INTO REACH, AND FLOW THROUGH STREAM BED TO ZERO.
C
DO 360 II =1,NSTREM
   STRM(9,II)=0.0
   STRM(10,II)=0.0
   STRM(11,II)=0.0
360 CONTINUE
C
110
9990 CALL INFORM ('\\Unable to read Stream input package ' //
& 'control record',-1)
CALL INFORM ('\\Abnormal Termination of Str1rp_Arc_Subroutine',-1)
RETURN 1
9991 CALL INFORM ('\\Missing Stream input package control record',-1)
CALL INFORM ('\\Abnormal Termination of Str1rp_Arc_Subroutine',-1)
RETURN 1
9992 CALL INFORM ('\\Unable to read parameters from Stream input ' //
& 'record',-1)
CALL INFORM ('\\Abnormal Termination of Str1rp_Arc_Subroutine',-1)
RETURN 1
9993 CALL INFORM ('\\Unable to read Hydraulic input parameters:','-1)
CALL INFORM ('\\ ROW, COLUMN, LAYER, SEG, REACH, FLOW, ' //
& 'STAGE, COND, SBOT, STOP',-1)
CALL INFORM ('\\Abnormal Termination of Str1rp_Arc_Subroutine',-1)
RETURN 1
9994 CALL INFORM ('\\Unable to read Hydraulic input parameters:','-1)
CALL INFORM ('\\ WIDTH, SLOPE, ROUGH',-1)
CALL INFORM ('\\Abnormal Termination of Str1rp_Arc_Subroutine',-1)
RETURN 1
9995 CALL INFORM ('\\Unable to read Tributary input parameters:','-1)
CALL INFORM ('\\Abnormal Termination of Str1rp_Arc_Subroutine',-1)
RETURN 1
9996 CALL INFORM ('\\Unable to read Diversions input parameters:','-1)
CALL INFORM ('\\Abnormal Termination of Str1rp_Arc_Subroutine',-1)
RETURN 1
9999 CALL INFORM ('\\Abnormal Termination of Str1rp_Arc_Subroutine',-1)
RETURN 1
END

Added variables for module STR1RPARC

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCESS</td>
<td>Module</td>
<td>Flag indicating whether access to the ARC/INFO file is read or write.</td>
</tr>
<tr>
<td>BUFFER</td>
<td>Module</td>
<td>The control record is read into this variable and is parsed into the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>appropriate variables: ITMP, IRDFLG, IPTFLG, and STRPATH.</td>
</tr>
<tr>
<td>PATH</td>
<td>Module</td>
<td>The root path to the ARC/INFO directory containing the files of interest.</td>
</tr>
<tr>
<td>STRPATH</td>
<td>Module</td>
<td>The complete path to the ARC/INFO file containing the item (INFOITEM) of</td>
</tr>
<tr>
<td></td>
<td></td>
<td>interest.</td>
</tr>
<tr>
<td>IARC</td>
<td>Module</td>
<td>Flag indicating whether the &quot;arc-section&quot; of the program code will</td>
</tr>
<tr>
<td></td>
<td></td>
<td>be activated. &gt; 0, ASCII file storage &lt; 0, ARC/INFO file storage</td>
</tr>
<tr>
<td>ITEMS</td>
<td>Module</td>
<td>The names of the INFO item array either primary or redefined within the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ARC/INFO file (specified by STRPATH) containing the information. The item</td>
</tr>
<tr>
<td></td>
<td></td>
<td>names are LAYER, ROW, COLUMN, SEG, REACH, FLOW, STAGE, COND, SBOT, and STOP</td>
</tr>
<tr>
<td></td>
<td></td>
<td>or WIDTH, SLOPE, and ROUGH or ITRIB_xx (where xx is tributary number) or IUPSEG.</td>
</tr>
<tr>
<td>INFOPATH</td>
<td>Submodule</td>
<td>The path for the ARC/INFO file where the values for the streamflow arrays</td>
</tr>
<tr>
<td></td>
<td></td>
<td>are stored.</td>
</tr>
<tr>
<td>NITEMS</td>
<td>Module</td>
<td>Integer value for the number of items within the file specified by</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INFOPATH.</td>
</tr>
<tr>
<td>NUMRECV</td>
<td>Module</td>
<td>Integer value for the number of records within the file specified by</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INFOPATH.</td>
</tr>
<tr>
<td>STREM</td>
<td>Module</td>
<td>Real array holding array values from the records within the file specified</td>
</tr>
<tr>
<td></td>
<td></td>
<td>by INFOPATH.</td>
</tr>
</tbody>
</table>
This module calculates rates and accumulated volumes of stream leakage into and out of aquifer for the STR package and calls submodule UBUDSVARC (fig. 24). Documentation of the changes in program code follows.

Figure 24.--Modified program elements for the STR1BDARC module.
SUBROUTINE STR1BDARC (NSTREM, STRM, ISTRM, IBOUND, MXSTRM, HNEW,
  1  NCOL, NROW, NLAY, DELT, VBVL, VBNM, MSUM, KPER, ISTCB1, ISTCB2,
  2  ICBCFL, BUFF, IOUT, NTRIB, NSS, ARTRIB, ITRBAR, IDIVAR, ICALC,
  3                                              CONST, IPTFLG, STRPTH,
 E                      *)
C-----VERSION 001 17JAN1989 STR1BDARC                                  C
C               BY D.E.PRUDIC
C-----VERSION 3.0 25OCTOBER1991 STR1BDARC
C               MODIFIED BY LEONARD L. ORZOL
C     *****************************************************************C
C     CALCULATE VOLUMETRIC BUDGET FOR STREAMS                          C
C     *****************************************************************C
C                                                                      C
C     SPECIFICATIONS:                                                  C
    -----------------------------------------------------------------C
CHARACTER*128 OUTPATH
CHARACTER*80 STRPTH
CHARACTER*32 INFONAME
CHARACTER*16 INFOITEM
CHARACTER*4 VBNM, TEXT, STRTXT
DOUBLE PRECISION HNEW
DIMENSION STRM(11,MXSTRM), ISTRM(5,MXSTRM), IBOUND(NCOL, NROW, NLAY),
  1  HNEW(NCOL, NROW, NLAY), VBVL(4,20), VBNM(4,20),
  2  BUFF(NCOL, NROW, NLAY), ARTRIB(NSS), ITRBAR(NSS, NTRIB),
  3          IDIVAR(NSS)
DIMENSION TEXT(4), STRTXT(4)
DATA TEXT(1), TEXT(2), TEXT(3), TEXT(4) /'  ST','REAM',' LEA','KAGE'/
DATA STRTXT(1), STRTXT(2), STRTXT(3), STRTXT(4) /'STRE','AM F',
  1                                              'LOW ','OUT '/
C     -----------------------------------------------------------------C
C                                                                      C
C1------SET IBD=1 IF BUDGET TERMS SHOULD BE SAVED ON DISK.             C
IBD=0
RATIN = 0.
RATOUT = 0.
C2------IF NO REACHES, KEEP ZEROS IN ACCUMULATORS.                     C
IF(NSTREM.EQ.0) GO TO 600
C                                                                      C
C3A-----TEST TO SEE IF CELL-BY-CELL TERMS ARE NEEDED.                  C
IF((ICBCFL.EQ.0).OR.(ISTCB1.LE.0)) GO TO 10
C                                                                      C
C3B-----CELL-BY-CELL TERMS ARE NEEDED, SET IBD AND CLEAR BUFFER.       C
IBD = 1
DO 5 IL=1,NLAY
  DO 5 IR=1,NROW
    DO 5 IC=1,NCOL
      BUFF(IC,IR,IL)=0.
  5 CONTINUE
C                                                                      C
C4------IF THERE ARE STREAMS THEN ACCUMULATE LEAKAGE TO OR FROM THEM.  C
10 DO 500 L=1,NSTREM
     LL=L-1
C                                                                      C
C5------DETERMINE REACH LOCATION.                                         C
IL=ISTRM(1,L)
IR=ISTRM(2,L)
IC=ISTRM(3,L)
C                                                                      C
C6------IF CELL IS EXTERNAL SKIP CALCULATIONS.                           C
IF(IBOUND(IC,IR,IL).LE.0)GO TO 500
C                                                                      C
C7------DETERMINE SEGMENT AND REACH NUMBER.                              C
ISTSG=ISTRM(4,L)
NREACH=ISTRM(5,L)
IF(NREACH.GT.1) GO TO 200
C                                                                      C
C8------SET FLOWIN EQUAL TO SEGMENT INFLOW IF FIRST REACH.              C
FLOWIN=STRM(1,L)
IF(ISTSG.GT.1) IFLG = ISTRM(4,LL)
C                                                                      C
C9------STORE OUTFLOW FROM PREVIOUS SEGMENT IN ARTRIB IF SEGMENT >1. C
IF(ISTSG.GT.1) ARTRIB(IFLG)=STRM(9,LL)

C

C10--IF SEGMENT IS A DIVERSION, COMPUTE FLOW OUT OF UPSTREAM SEGMENT. C
IF(IDIVAR(ISTSG).LE.0) GO TO 50
NDFLG=IDIVAR(ISTSG)
DUM=ARTRIB(NDFLG)-FLOWIN
IF(DUM.GE.0.0) ARTRIB(NDFLG)=DUM
IF(DUM.GE.0.0) GO TO 50
FLOWIN=0.
50 IF(FLOWIN.GE.0.0) GO TO 300

C

C11--SUM TRIBUTARY OUTFLOW AND USE AS INFLOW INTO DOWNSTREAM SEGMENT. C
FLOWIN =0.
DO 100 ITRIB=1,NTRIB
INODE=ITRBAR(ISTSG,ITRIB)
IF(INODE.LE.0) GO TO 100
FLOWIN=FLOWIN+ARTRIB(INODE)
100 CONTINUE

C

C12-----IF REACH >1, SET INFLOW EQUAL TO OUTFLOW FROM UPSTREAM REACH. C
200 IF(NREACH.GT.1) FLOWIN=STRM(9,LL)

C

C13-----COMPUTE STREAM STAGE IN REACH IF ICALC > 1. C
300 IF(ICALC.LE.0) GO TO 310
XNUM=((FLOWIN+STRM(9,L))/2.0)*STRM(8,L)
DNOM=CONST*STRM(6,L)*(SQRT(STRM(7,L)))
DEPTH=(XNUM/DNOM)**0.6
IF(DEPTH.LE.0) DEPTH=0.
STRM(2,L)=DEPTH+STRM(5,L)
310 HSTR=STRM(2,L)

C

C14----DETERMINE LEAKAGE THROUGH STREAMBED. C
IF(FLOWIN.LE.0.0) HSTR=STRM(5,L)
CSTR=STRM(3,L)
SBOT=STRM(4,L)
H=HNEW(IC,IR,IL)
T=HSTR-SBOT

C

C15----COMPUTE LEAKAGE AS A FUNCTION OF STREAM STAGE AND HEAD IN CELL. C
FLOBOT=CSTR*(HSTR-H)

C

C16----RECOMPUTE LEAKAGE IF HEAD IN CELL IS BELOW STREAMBED BOTTOM. C
IF(H.GT.SBOT) GO TO 312
FLOBOT=CSTR*T

C

C17----SET LEAKAGE EQUAL TO STREAM INFLOW IF LEAKAGE MORE THAN INFLOW. C
312 IF(FLOBOT.LE.FLOWIN) GO TO 320
FLOBOT=FLOWIN

C

C18-----STREAMFLOW OUT_EQUALS STREAMFLOW IN MINUS LEAKAGE. C
320 FLOWOUT=FLOWIN-FLOBOT
IF((ISTSG.GT.1).AND.(NREACH.EQ.1)) STRM(9,LL)=ARTRIB(IFLG)

C

C19----STORE STREAM INFLOW, OUTFLOW AND LEAKAGE FOR EACH REACH. C
STRM(9,L)=FLOWOT
STRM(10,L)=FLOWIN
STRM(11,L)=FLOBOT

C

C20----IF LEAKAGE FROM STREAMS IS TO BE SAVED THEN ADD RATE TO BUFFER. C
IF(IBD.EQ.1) BUFF(IC,IR,IL)=BUFF(IC,IR,IL)+FLOBOT

C

C21-----DETERMINE IF FLOW IS INTO OR OUT OF MODEL CELL. C
C SKIP ESTIMATE OF LEAKAGE FROM STREAM IF LEAKAGE IS ZERO. C
IF(FLOBOT).NE.0) GO TO 494

C

C22-----SUBTRACT FLOW RATE FROM RATOUT IF AQUIFER DISCHARGES TO STREAM.C
494 RATOUT=RATOUT-FLOBOT
GO TO 500

C

C23-----ADD FLOW RATE TO RATIN IF STREAM DISCHARGES TO AQUIFER. C
496 RATIN=RATIN+FLOBOT
500 CONTINUE
C24------IF BUDGET TERMS WILL BE SAVED THEN WRITE TO DISK.
C24A------RECORD IN UNFORMATTED FILE ISTCB1
C
   IF(IBD.EQ.1 .AND. ICBCFL.GE.0) CALL UBUDSV(KSTP,KPER,TEXT,
   1                        ISTCB1,BUFF,NCOL,NROW,NLAY,IOUT)
C
C24B------RECORD IN ARC/INFO FILE
C
   IF(IBD.EQ.1 .AND. ICBCFL.LT.0) THEN
      INFOITEM='LAYER'
      INFONAME='LKGBUD'
      OUTPATH=STRPTH (:INDEX(STRPTH,' ')-1)//INFONAME
      CALL UBUDSVARC (KSTP,KPER,INFOITEM,OUTPATH,BUFF,NCOL,NROW,NLAY,
      &   IOUT                  *9999)
   ENDIF
C
C25A------MOVE RATES INTO VBVL FOR PRINTING BY MODULE BAS_OT.
600 VBVL(3,MSUM)=RATIN
   VBVL(4,MSUM)=RATOUT
C25B------MOVE PRODUCT OF RATE AND TIME STEP INTO VBVL ACCUMULATORS.
   VBVL(1,MSUM)=VBVL(1,MSUM)+RATIN*DELT
   VBVL(2,MSUM)=VBVL(2,MSUM)+RATOUT*DELT
C
C25C------MOVE BUDGET TERM LABELS INTO VBNM FOR PRINTING BY BAS_OT.
   VBNM(1,MSUM)=TEXT(1)
   VBNM(2,MSUM)=TEXT(2)
   VBNM(3,MSUM)=TEXT(3)
   VBNM(4,MSUM)=TEXT(4)
C
C26------INCREASE BUDGET TERM COUNTER BY ONE.
   MSUM=MSUM+1
C
C27------RESET IBD COUNTER TO ZERO.
   IBD=0
C
C28------IF STREAM OUTFLOW FROM EACH REACH IS TO BE STORED IN ON DISK
   THEN STORE OUTFLOW RATES INTO BUFFER.
   IF((ICBCFL.EQ.0).OR.(ISTCB2.LE.0)) GO TO 625
      IBD = 1
      DO 605 IL=1,NLAY
      DO 605 IR=1,NROW
      DO 605 IC=1,NCOL
       605 BUFF(IC,IR,IL)=0.
C
C29------SAVE STREAMFLOWS OUT OF EACH REACH ON DISK.
   DO 615 L=1,NSTREM
      IC=ISTRM(3,L)
      IR=ISTRM(2,L)
      IL=ISTRM(1,L)
      IF(IBOUND(IC,IR,IL).LE.0) GO TO 615
      BUFF(IC,IR,IL)=BUFF(IC,IR,IL)+STRM(9,L)
   615 CONTINUE
C
C29A------RECORD IN UNFORMATTED FILE ISTCB2
C
   IF(ICBCFL.GE.0) CALL UBUDSV(KSTP,KPER,STRXTX,
   1                        ISTCB2,BUFF,NCOL,NROW,NLAY,IOUT)
C
C29B------RECORD IN ARC/INFO FILE
C
   IF(ICBCFL.LT.0) THEN
      INFOITEM='LAYER'
      INFONAME='FLOBUD'
      OUTPATH=STRPTH (:INDEX(STRPTH,' ')-1)//INFONAME
      CALL UBUDSVARC (KSTP,KPER,INFOITEM,OUTPATH,BUFF,NCOL,NROW,NLAY,
      &   IOUT                  *9999)
   ENDIF
C30-----PRINT STREAMFLOW RATES AND LEAKAGE FOR EACH REACH.

625 IF((ISTCB1.GE.0).OR.(ICBCFL.NE.0)) GO TO 800
IF(IPTFLG.GT.0) GO TO 800
IF(ICALC.GT.0) GO TO 700
WRITE(IOUT,650)
650 FORMAT(1H0,12X,'LAYER',6X,'ROW',5X,'COLUMN',5X,'STREAM',4X,
1'REACH',6X,'FLOW INTO',4X,'FLOW INTO',6X,'FLOW OUT OF'/43X,
2'NUMBER',3X,'NUMBER',4X,'STREAM REACH',4X,'AQUIFER',
36X,'STREAM REACH'//)
DO 690 L=1,NSTREM
IL=ISTRM(1,L)
IR=ISTRM(2,L)
IC=ISTRM(3,L)
WRITE(IOUT,675)IL,IR,IC,ISTRM(4,L),ISTRM(5,L),
1STRM(10,L),STRM(11,L),STRM(9,L)
675 FORMAT(1X,5X,5I10,8X,G9.3,5X,G9.3,8X,G9.3)
690 CONTINUE
GO TO 800
700 WRITE(IOUT,710)
710 FORMAT(1H0,12X,'LAYER',6X,'ROW',5X,'COLUMN',5X,'STREAM',4X,
1'REACH',6X,'FLOW INTO',4X,'FLOW INTO',6X,'FLOW OUT OF',5X,
2'HEAD IN'/43X,'NUMBER',3X,'NUMBER',4X,'STREAM REACH',
34X,'AQUIFER',6X,'STREAM REACH',5X,'STREAM'//)
DO 750 L=1,NSTREM
IL=ISTRM(1,L)
IR=ISTRM(2,L)
IC=ISTRM(3,L)
WRITE(IOUT,775)IL,IR,IC,ISTRM(4,L),ISTRM(5,L),
1STRM(10,L),STRM(11,L),STRM(9,L),STRM(2,L)
775 FORMAT(1X,5X,5I10,8X,G9.3,5X,G9.3,7X,G9.3,4X,F9.2)
750 CONTINUE
800 CONTINUE
C31-----RETURN.
RETURN
CE-------ERRORS
CE
9999 CALL INFORM ('\Abnormal Termination of Str1bd_Arc_Subroutine','-1)
RETURN 1
END

Added variables for module STR1BDARC

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>INFONAME</td>
<td>Module</td>
<td>The name of the ARC/INFO file where cell-by-cell flow terms are recorded (passed argument consisting of the root name LKGBUD or FLOBUD which later will be appended by the stress period and time step).</td>
</tr>
<tr>
<td>INFOITEM</td>
<td>MODodule</td>
<td>The name of the ARC/INFO or INFO item where cell-by-cell flow terms array are recorded (passed argument consisting of the root name LAYER which later will be appended by the layer number).</td>
</tr>
<tr>
<td>OUTPATH</td>
<td>Module</td>
<td>The path for the ARC/INFO file where cell-by-cell flow terms are recorded.</td>
</tr>
<tr>
<td>STRPATH</td>
<td>Module</td>
<td>The directory path to the ARC/INFO subdirectory where cell-by-cell flow terms array are recorded.</td>
</tr>
</tbody>
</table>
This module records rates and accumulated volumes of stream leakage into and out of stream segments and reaches for the STR package (fig. 25). Documentation of program code follows:

SUBROUTINE STR1HYARC (NSTREM, STRM, ISTRM, MXSTRM, & KSTP, KPER, ICALC, IPTFLG, ICBCFL, & STRPTH, IOUT, * )

C-----VERSION 3 25OCTOBER1991 STR1HYARC
C ORIGINAL CODING BY LEONARD L. ORZOL
C ***********************************************C

Figure 25.—Program elements for the STR1SRARC module.
PRODUCES AN ARC/INFO FILE THAT RECORDS STREAM LEAKAGE
AND STREAMFLOW BY LAYER, SEGMENT NUMBER, REACH NUMBER,
AND HEAD (HEAD WHEN ICALC > 0)

***************************************************************************

SPECIFICATIONS:

PARAMETER (MAXITM=9)
CHARACTER*128 STRFIL
CHARACTER*80 STRPTH
CHARACTER*32 INFONAME
CHARACTER*16 ITEM,ITEMS(MAXITM)
DIMENSION STRM(11,MXSTRM),ISTRM(5,MXSTRM)
COMMON STREM(131072)
INTEGER ACCESS,COPY,DECIML,DECIMLS(MAXITM),FNUM,KEYLEV,KEYTYP,
& LAYER,NCOORD,OCUR,OUTPUT,OUTPUTS(MAXITM),POSIT,PROTCT,
& REDEF,TYPE,TYPES(MAXITM),WIDTH,WIDTHS(MAXITM)

DATA ITEMS /'LAYER','ROW','COLUMN',
& 'SEG','REACH','FLOWOUT','FLOWIN','LEAKAGE','HEAD'/
DATA DECIMLS /5*-1,4*3/
DATA TYPES /5*3,4*4/
DATA WIDTHS /3,2,2,3,2,4*15/
DATA OUTPUTS /3,2,2,3,2,4*15/

------------------------------------------------------------------
IF(IPTFLG.LT.0 .AND. ICBCFL.LT.0) THEN

C1----Set local variables
IF(ICALC.LE.0) THEN
NITEMS=8
ELSE
NITEMS=9
ENDIF

C2----Creates attribute STRFIL file
ACCESS=3
NCHAR=128
LAYER=0
INFONAME='STRSEG'
STRFIL=STRPTH (:INDEX(STRPTH,' ')-1)//INFONAME
CALL INFO_NAMING (STRFIL,KPER,KSTP,LAYER,NCHAR)
CALL INFO_CREATE (I
I FNUM,
O STRFIL,COPY,
E *9999)

C3----Load item names into STRFIL file
DO 300 N=1,NITEMS
   ALTERN=''
   AFTER=''
   DECIML=DECIMLS(N)
   NINDEX=-1
   ITEM=ITEMS(N)
   KEYLEV=-1
   KEYTYP=-1
   OCCUR=-1
   OUTPUT=OUTPUTS(N)
   POSIT=0
   PROTCT=4
   READON=0
   REDEF=0
   TYPE=TYPES(N)
   WIDTH=WIDTHS(N)
   CALL INFO_ITEMS (I
I FNUM,ITEM,
I AFTER,ALTERN,DECIML,NINDEX,KEYLEV,KEYTYP,
I O E
CONTINUE CALL INFO_CLOSING (FNUM)

C4----Write stream and aquifer attributes to STRFIL file

CALL INFO_OPENS (I STRFIL, ACCESS,
O FNUM, NUMREC,
E *9999)

DO 400 L=1, NSTREM
STREM(L)=FLOAT(ISTRM(1, L))

400 CONTINUE

CALL INFO_Writes (FNUM, NSTREM, ITEMS(1), STREM(1), *9999)

DO 410 N=2, 5
DO 420 L=1, NSTREM
STREM(L)=FLOAT(ISTRM(N, L))

420 CONTINUE

CALL INFO_WRITE (FNUM, NSTREM, ITEMS(N), STREM(1), *9999)

410 CONTINUE

DO 430 N=6, 8
DO 440 L=1, NSTREM
STREM(L)=STRM(N+3, L)

440 CONTINUE

CALL INFO_WRITE (FNUM, NSTREM, ITEMS(N), STREM(1), *9999)

430 CONTINUE

IF (ICALC.GT.0) THEN
DO 450 L=1, NSTREM
STREM(L)=STRM(2, L)

450 CONTINUE

CALL INFO_WRITE (FNUM, NSTREM, ITEMS(MAXITM), STREM(1),
*9999)

ENDIF

CALL INFO_CLOSING (FNUM)

ENDIF

C

CR------RETURN.

RETURN

C

CE------ERRORS

C

9999 CALL INFORM

& ("Abnormal Termination of Str1srarc_Arc_Subroutine", -1)

RETURN 1

END

Variables for module STR1SRARC

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCESS</td>
<td>Module</td>
<td>Flag indicating whether access to the ARC/INFO file is read or write.</td>
</tr>
<tr>
<td>AFTER</td>
<td>Module</td>
<td>Character*16 Info item name where to add new item after.</td>
</tr>
<tr>
<td>ALTERN</td>
<td>Module</td>
<td>Character*16 Info variable holding alternate item name.</td>
</tr>
<tr>
<td>COPY</td>
<td>Module</td>
<td>Integer Info variable switches on item file procedure.</td>
</tr>
<tr>
<td>DECIML</td>
<td>Module</td>
<td>Integer Info variable holding number of decimal places.</td>
</tr>
<tr>
<td>DECIMLS</td>
<td>Module</td>
<td>Integer array holding number of decimal places.</td>
</tr>
<tr>
<td>FNUM</td>
<td>Module</td>
<td>Integer Info variable holding ISP channel number.</td>
</tr>
<tr>
<td>INFONAME</td>
<td>Module</td>
<td>The name of the ARC/INFO file where the values for the output data for streamflow by stream segment and reach are recorded.</td>
</tr>
<tr>
<td>ITEM</td>
<td>Module</td>
<td>The name of the ARC/INFO or INFO item where the values for</td>
</tr>
</tbody>
</table>
the array are recorded.

<table>
<thead>
<tr>
<th>Module</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ITEMS</td>
<td>The names of the INFO item array either primary or redefined within the ARC/INFO file (specified by STRPATH) containing the information. The item names are LAYER, ROW, COLUMN, SEG, REACH, FLOWOUT, FLOWOUT, LEAKAGE, and HEAD.</td>
</tr>
<tr>
<td>KEYLEV</td>
<td>Integer Info variable holding key level of items.</td>
</tr>
<tr>
<td>KEYTYP</td>
<td>Integer Info variable holding key type of items.</td>
</tr>
<tr>
<td>LAYER</td>
<td>Integer value for the layer number.</td>
</tr>
<tr>
<td>MAXITM</td>
<td>Integer value for the maximum number of items within the file specified by STRFIL.</td>
</tr>
<tr>
<td>NCHAR</td>
<td>Integer value for the maximum number of character in the filename STRFIL.</td>
</tr>
<tr>
<td>NITEMS</td>
<td>Integer value for the number of items within the file specified by STRFIL.</td>
</tr>
<tr>
<td>NUMREC</td>
<td>Integer value for the number of record within the file specified by STRFIL.</td>
</tr>
<tr>
<td>OCCUR</td>
<td>Integer Info variable holding occurrence count of item.</td>
</tr>
<tr>
<td>OUTPUT</td>
<td>Integer Info variable holding output width of item.</td>
</tr>
<tr>
<td>OUTPUTS</td>
<td>Integer array holding output width of item.</td>
</tr>
<tr>
<td>POSIT</td>
<td>Integer Info variable starting column redefined items.</td>
</tr>
<tr>
<td>PROTCT</td>
<td>Integer Info variable holding protection level of item.</td>
</tr>
<tr>
<td>READON</td>
<td>Integer Info variable holds read access level of item.</td>
</tr>
<tr>
<td>RECLEN</td>
<td>Integer Info variable record length of ARC/INFO files.</td>
</tr>
<tr>
<td>REDEF</td>
<td>Integer Info switch for normal(0) or redefine(1) item.</td>
</tr>
<tr>
<td>STREM</td>
<td>Real Info array holding the output data for streamflow by stream segment and reach.</td>
</tr>
<tr>
<td>STRFIL</td>
<td>The path for the ARC/INFO file where the values for the output data for streamflow by stream segment and reach are recorded.</td>
</tr>
<tr>
<td>STRPATH</td>
<td>The directory path to the ARC/INFO subdirectory where the values for the output data for streamflow by stream segment and reach are recorded.</td>
</tr>
<tr>
<td>TYPE</td>
<td>Integer Info variable holds item type (integer,...).</td>
</tr>
<tr>
<td>TYPES</td>
<td>Integer array holding item types (integer,...).</td>
</tr>
<tr>
<td>WIDTH</td>
<td>Integer Info variable holding input width of item.</td>
</tr>
<tr>
<td>WIDTHS</td>
<td>Integer array holding input width of item.</td>
</tr>
</tbody>
</table>
Utility Modules

Utility modules perform general tasks common to several different packages. The name of these carries a prefix "U" and a suffix "ARC". Five of the original eight utility modules were either changed by adding "arc-sections" code, or entirely rewritten.

Modified utility modules:

U2DRELARC: Reads an ASCII or ARC/INFO file consisting of a two-dimensional array of real numbers (format of these items is user specified at the time of creation)) for each cell in the grid. An "arc-section" code has been added to allow program flow to branch to either method: read an ASCII file (original program flow) or read an ARC/INFO file. The "arc-section" code is activated by a new parameter by placing a new parameter on the control record. The new parameter consists of the complete path to ARC/INFO file containing the array values. The item value containing the array values is derived from variable names used by the ground-water manual.

U2DINTARC: Reads an ASCII or ARC/INFO file consisting of a two-dimensional array of integers (format of these items is user specified at the time of creation)) for each cell in the grid. An "arc-section" code has been added to allow program flow to branch to either method: read an ASCII file (original program flow) or read an ARC/INFO file. The "arc-section" code is activated by a new parameter by placing a new parameter on the control record. The new parameter consists of the complete path to ARC/INFO file containing the array values. The item value containing the array values is derived from variable names used by the ground-water manual.

U1DRELARC: Reads an ASCII or ARC/INFO file consisting of a one-dimensional array of real numbers (format of these items is user specified at the time of creation)) for each cell in the grid. An "arc-section" code has been added to allow program flow to branch to either method: read an ASCII file (original program flow) or read an ARC/INFO file. The "arc-section" code is activated by a new parameter by placing a new parameter on the control record. The new parameter consists of the complete path to ARC/INFO file containing the array values. The item value containing the array values is derived from variable names used by the ground-water manual.

Completely rewritten utility modules:

UBUDSVARC: Writes an ARC/INFO file consisting of an array with a real number (floating point number, field length of sixteen and three decimal digits) for each cell in the grid.

ULASAVARC: Writes an ARC/INFO file consisting of an array with a real number (floating point number, field length of sixteen and three decimal digits) for each cell in a layer.
U2DRELARC

This module reads two-dimensional arrays when activated from calling "read and prepare" modules of other packages. "Arc-section" code was added to read the control record for a new parameter INFOPATH that specifies a complete path to an ARC/INFO file (fig. 26). Documentation of the modified module follows.

Figure 26.--Modified program elements for the U2DRELARC module.
SUBROUTINE U2DRELARC (INFOITEM, A, ANAME, II, JJ, K, IN, IOUT, E)
C
C-----VERSION 3 25OCTOBER1991 U2DRELARC
C MODIFIED BY LEONARD L. ORZOL
C
C **********************************************************************************
C ROUTINE TO INPUT 2-D REAL DATA MATRICES
A IS ARRAY TO INPUT
ANAME IS 24 CHARACTER DESCRIPTION OF A
II IS NO. OF ROWS
JJ IS NO. OF COLS
K IS LAYER NO. (USED WITH NAME TO TITLE PRINTOUT UNLESS K IS 0)
IN IS INPUT UNIT
IOUT IS OUTPUT UNIT
**********************************************************************************
C
SPECIFICATIONS:
---------------------------------------------------------------------
CHARACTER*(*) INFOITEM
CHARACTER*132 BUFFER
CHARACTER*128 INFOPATH
CHARACTER*82 NAMPATH
CHARACTER*16 ITEM
CHARACTER*8 USER
CHARACTER*4 ANAME
CHARACTER*20 FMTIN
DIMENSION A(JJ,II),ANAME(6)
INTEGER FNUM,NCHAR,NCOLUMNS,NROWS,NUMREC,ACCESS
EXTERNAL UCOLNO,ULAPRW
EXTERNAL INFO_CLOSING,INFO_NAMING,INFO_OPENS,INFO_READREAL
DATA ITEM /' '/
DATA INFOPATH /' '/
---------------------------------------------------------------------
C
C1------READ ARRAY CONTROL RECORD.
C
READ(IN,'(A132)',ERR=9990,END=9991) BUFFER
C
C1A------READ ARRAY FROM ASCII FILE LOCAT.
C
IF(BUFFER (51:51).EQ.'' .OR. BUFFER (51:51).EQ.' ') THEN
   READ (BUFFER,'(I10,F10.0,A20,I10)',ERR=9992)
&    LOCAT,CNSTNT,FMTIN,IPRN
   IARC=0
C
C1B------READ ARRAY FROM ARC/INFO FILE NAMPATH.
C
ELSE
   READ (BUFFER,'(I10,F10.0,A20,I10,A82)',ERR=9992)
&    LOCAT,CNSTNT,FMTIN,IPRN,NAMPATH
   INFOPATH=NAMPATH (1:INDEX(NAMPATH,' ')-1)
   IARC=1
ENDIF
C
C2------USE LOCAT TO SEE WHERE ARRAY VALUES COME FROM.
IF(LOCAT) 200,50,90
C
C3------IF LOCAT=0 THEN SET ALL ARRAY VALUES EQUAL TO CNSTNT. RETURN
50 DO 80 I=1,II
   DO 80 J=1,JJ
80    A(J,I)=CNSTNT
   IF(K.GT.0) WRITE(IOUT,2) ANAME,CNSTNT,K
   IF(K.LE.0) WRITE(IOUT,3) ANAME,CNSTNT
   RETURN
C
C4------IF LOCAT>0 THEN READ RECORDS.
C
C
C4A------IF LOCAT>0 THEN READ FORMATTED RECORDS USING FORMAT FMTIN.

90 IF(IARC.LT.1) THEN
   IF(K.GT.0) WRITE(IOUT,11) ANAME,K,LOCAT,FMTIN
   11     FORMAT(1H0,///30X,6A4,' FOR LAYER',I3,' WILL BE READ ON UNIT',
       &   I3,' USING FORMAT: ',A20/30X,96('-'))
   IF(K.LE.0) WRITE(IOUT,13) ANAME,LOCAT,FMTIN
   13     FORMAT(1H0,///30X,6A4,' WILL BE READ ON UNIT',
       &   I3,' USING FORMAT: ',A20/30X,83('-'))
   DO 100 I=1,II
       READ(LOCAT,FMTIN,ERR=9993,END=9994) (A(J,I),J=1,JJ)
   100 CONTINUE

C4B------IF LOCAT>0 THEN READ RECORDS USING ARC/INFO ROUTINES.

   ELSE
      IF(K.GT.0) THEN
          WRITE(IOUT,15) ANAME,K,INFOPATH (:INDEX(INFOPATH,' ')-1)
          15       FORMAT(1H0,///10X,6A4,' FOR LAYER',I3,
              &                  ' WILL BE READ FROM INFO FILE ',A,/10X,112('-'))
      ELSE
          WRITE(IOUT,17) ANAME,INFOPATH (:INDEX(INFOPATH,' ')-1)
          17       FORMAT(1H0,///10X,6A4,' WILL BE READ FROM INFO FILE',
              &                                                  A,/10X,112('-'))
      ENDIF

C4BB-----TESTS EXISTENCE OF INFO FILE AND ITEM NAME.

   ACCESS=1
   CALL INFO_OPENS (INFOPATH,ACCESS,FNUM,NUMREC,*9999)
   NCHAR=16
   IPER=0
   ISTP=0
   ITEM=INFOITEM
   KK=ABS(K)
   CALL INFO_NAMING (ITEM,IPER,ISTP,KK,NCHAR)

C4BC------OPENS AND READS INFO FILE.

   CALL INFO_READREAL (FNUM,NUMREC,ITEM,A,*9999)
   CALL INFO_CLOSING (FNUM)
   ENDIF

GO TO 300

C5------LOCAT<0 THEN READ UNFORMATTED RECORD CONTAINING ARRAY VALUES

200   LOCAT=-LOCAT
   IF(K.GT.0) WRITE(IOUT,201) ANAME,K,LOCAT
   201   FORMAT(1H0,///30X,6A4,' LAYER',I3,
       &   ' WILL BE READ UNFORMATTED ON UNIT',I3/30X,73('-'))
   IF(K.LE.0) WRITE(IOUT,202) ANAME,LOCAT
   202   FORMAT(1H0,///30X,
       &   ' WILL BE READ UNFORMATTED ON UNIT',I3/30X,60('-'))

C5A------READ AN UNFORMATTED DUMMY RECORD FIRST.

   READ(LOCAT)
   READ(LOCAT,ERR=9993,END=9994) A

C6------IF CNSTNT NOT ZERO THEN MULTIPLY ARRAY VALUES BY CNSTNT.

300   IF(CNSTNT.EQ.0.) GO TO 320
       DO 310 I=1,II
           A(J,I)=A(J,I)*CNSTNT
       310 CONTINUE

C7------IF PRINT CODE (IPRN) =>0 THEN PRINT ARRAY VALUES.

320   IF(IPRN.LT.0) RETURN
       CALL ULAPRW(A,ANAME,0,0,JJ,II,0,IPRN,IOUT)

C8------RETURN

RETURN
CE------ERRORS
C

9990 CALL MESINT (K)
CALL MESCHR (INFOITEM, 0)
CALL INFORM ('\Unable to read input control line for ' // &
'layer %1% and item %2%', -1)
&
CALL INFORM
& ('\Abnormal Termination of U2drelarc_Subroutine\ ', -1)
RETURN 1

9991 CALL MESINT (K)
CALL MESCHR (INFOITEM, 0)
CALL INFORM ('\End-of-file reached for layer %1% and item %2%' // &
'; Missing input package control line', -1)
CALL INFORM
& ('\Abnormal Termination of U2drelarc_Subroutine\ ', -1)
RETURN 1

9992 CALL MESCHR (BUFFER, 0)
CALL MESINT (K)
CALL MESCHR (INFOITEM, 0)
CALL INFORM ('\Unable to read from control line %1% for ' // &
'layer %2% and item %3%; Reformat control line', -1)
CALL INFORM
& ('\Abnormal Termination of U2drelarc_Subroutine\ ', -1)
RETURN 1

9993 CALL MESINT (K)
CALL MESCHR (INFOITEM, 0)
CALL INFORM ('\Unable to read for layer %1% ' // &
' and item %2%; Bad format', -1)
CALL INFORM
& ('\Abnormal Termination of U2drelarc_Subroutine\ ', -1)
RETURN 1

9994 CALL MESCHR (INFOITEM, 0)
CALL INFORM
& ('\End-of-file reached for layer %1% and item %2%', -1)
9999 CALL INFORM
& ('\Abnormal Termination of U2drelarc_Subroutine\ ', -1)
RETURN 1
END

Added variables for module U2DRELARC

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCESS</td>
<td>Module</td>
<td>Flag indicating whether access to the ARC/INFO file is read or write.</td>
</tr>
<tr>
<td>BUFFER</td>
<td>Module</td>
<td>The control record is read into this variable and is parsed into the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>appropriate variables: LOCAT, CNSTNT or ICONST, IPRN, and INFOPATH.</td>
</tr>
<tr>
<td>IARC</td>
<td>Module</td>
<td>Flag indicating whether the 'arc-section' of the program code will</td>
</tr>
<tr>
<td></td>
<td></td>
<td>be activated. &gt; 0, ASCII file storage &lt; 0, ARC/INFO file storage</td>
</tr>
<tr>
<td>INFOITEM</td>
<td>Module</td>
<td>The name of the ARC/INFO or INFO item where the values for the array are</td>
</tr>
<tr>
<td></td>
<td></td>
<td>stored (passed argument consisting of the root name).</td>
</tr>
<tr>
<td>INFOPATH</td>
<td>Module</td>
<td>The complete path to the ARC/INFO file containing the item</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(INFOITEM) of interest.</td>
</tr>
<tr>
<td>ITEM</td>
<td>Module</td>
<td>The name of the ARC/INFO or INFO item where the values for the array are</td>
</tr>
<tr>
<td></td>
<td></td>
<td>stored.</td>
</tr>
<tr>
<td>NAMPATH</td>
<td>Module</td>
<td>The complete path to the ARC/INFO file containing the item</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(INFOITEM) of interest that is read.</td>
</tr>
<tr>
<td>NUMREC</td>
<td>Module</td>
<td>Integer value for the number of record within the file specified by</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INFOPATH.</td>
</tr>
</tbody>
</table>
This module reads one-dimensional arrays and searches the control record for a new parameter INFOPATH that specifies a complete path to an ARC/INFO file (fig. 27). Documentation of the changes in program code follows.

Figure 27.—Modified program elements for the U1DRELARC module.
SUBROUTINE U1DRELARC (INFOITEM, A, ANAME, JJ, IN, IOUT, E *)

C
C-----VERSION 3 25OCTOBER1991 U1DRELARC
C MODIFIED BY LEONARD L. ORZOL
C
C ********************************************************************************
C ROUTINE TO INPUT 1-D REAL DATA MATRICES
A IS ARRAY TO INPUT
ANAME IS 24 CHARACTER DESCRIPTION OF A
JJ IS NO. OF ELEMENTS
IN IS INPUT UNIT
IOUT IS OUTPUT UNIT
********************************************************************************

SPECIFICATIONS:
-----------------------------------------------------------------
CHARACTER*(*) INFOITEM
CHARACTER*132 BUFFER
CHARACTER*128 INFOPATH
CHARACTER*82 NAMPATH
CHARACTER*16 ITEM
CHARACTER*4 ANAME
CHARACTER*20 FMTIN
DIMENSION A(JJ), ANAME(6)
EXTERNAL INFO_CLOSING, INFO_OPENS, INFO_READREAL
DATA ITEM /' '/
DATA INFOPATH /' '/
-----------------------------------------------------------------

C1------READ ARRAY CONTROL RECORD.
READ(IN, '(A132)', ERR=9990, END=9991) BUFFER

C1A------READ ARRAY FROM ASCII FILE LOCAT.
IF (BUFFER (51:51).EQ.'' .OR. BUFFER (51:51).EQ.' ') THEN
READ (BUFFER,'(I10,F10.0,A20,I10)', ERR=9992)
& LOCAT,CNSTNT,FMTIN,IPRN
  IARC=0
ELSE
READ (BUFFER,'(I10,F10.0,A20,I10,A82)', ERR=9992)
& LOCAT,CNSTNT,FMTIN,IPRN,NAMPATH
  INFOPATH=NAMPATH (1:INDEX(NAMPATH,' ')-1)
  IARC=1
ENDIF

C2------USE LOCAT TO SEE WHERE ARRAY VALUES COME FROM.
IF (LOCAT.GT.0) GO TO 90

C3------IF LOCAT=0 THEN SET ALL ARRAY VALUES EQUAL TO CNSTNT. RETURN
DO 80 J=1,JJ
  A(J)=CNSTNT
5 WRITE(IOUT,7) ANAME, CNSTNT
7 FORMAT(1H0,52X,6A4,' =',G15.7)
RETURN

C4------IF LOCAT>0 THEN READ RECORDS.

C4A------IF NAMPATH IS ABSENT THEN READ FORMATTED RECORDS USING FORMAT FMTIN.
90 IF (IARC.LT.1) THEN
  WRITE(IOUT,9) ANAME, LOCAT, FMTIN
  FORMAT(1H0,///30X,6A4,' WILL BE READ ON UNIT',I3,
 & USING FORMAT: ',A20/30X,79('-'/))
  READ (LOCAT, FMTIN, ERR=9993, END=9994) (A(J), J=1, JJ)

C4B------IF INFOPATH IS PRESENT THEN READ RECORDS USING ARC/INFO ROUTINES.
C
ELSE
WRITE(IOUT,11) ANAME,INFOPATH (:INDEX(INFOPATH,' ')=-1)
11 FORMAT(1H0,///10X,6A4,' WILL BE READ FROM INFO FILE',
     & A,/10X,112(''-'))
C
C4BB------TESTS EXISTENCE OF INFO FILE AND ITEM NAME.
C
ACCESS=1
CALL INFO_OPENS (INFOPATH,ACCESS,FNUM,NUMREC,*9999)
C
C4BC------OPENS AND READS INFO FILE.
C
ITEM=INFOITEM
CALL INFO_READREAL (FNUM,NUMREC,ITEM,A,*9999)
CALL INFO_CLOSING (FNUM)
ENDIF
C
C5------IF CNSTNT NOT ZERO THEN MULTIPLY ARRAY VALUES BY CNSTNT.
   IF(CNSTNT.EQ.0.) GO TO 120
DO 100 J=1,JJ
100 A(J)=A(J)*CNSTNT
C
C6------IF PRINT CODE (IPRN) =>0 THEN PRINT ARRAY VALUES.
   IF(IPRN.LT.0) RETURN
WRITE(IOUT,1001) (A(J),J=1,JJ)
1001 FORMAT((1X,1PG12.5,9(1X,G12.5))
C
C7------RETURN
C
RETURN
C
CE------ERRORS
C
9990 CALL MESCHR (INFOITEM,0)
CALL INFORM ('\Unable to read input control line for ' //
     & 'item %1%','-1)
& CALL INFORM ('\Abnormal Termination of U1drelarc_Subroutine\ ',-1)
RETURN 1
9991 CALL MESCHR (INFOITEM,0)
CALL INFORM ('\End-of-file reached for item %1% //
     & 'Missing input package control line','-1)
& CALL INFORM ('\Abnormal Termination of U1drelarc_Subroutine\ ',-1)
RETURN 1
9992 CALL MESCHR (BUFFER,0)
CALL MESCHR (INFOITEM,0)
CALL INFORM ('\Unable to read from control line %1% for ' //
     & 'item %2% ; Reformat control line','-1)
& CALL INFORM ('\Abnormal Termination of U1drelarc_Subroutine\ ',-1)
RETURN 1
9993 CALL MESCHR (INFOITEM,0)
CALL INFORM ('\Unable to read for item %1% ; Bad format','-1)
CALL INFORM ('\Abnormal Termination of U1drelarc_Subroutine\ ',-1)
RETURN 1
9994 CALL MESCHR (INFOITEM,0)
CALL INFORM ('\End-of-file reached for item %1%','-1)
9999 CALL INFORM ('\Abnormal Termination of U1drelarc_Subroutine\ ','-1)
RETURN 1
END
Added variables for module U1DRELARC

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCESS</td>
<td>Module</td>
<td>Flag indicating whether access to the ARC/INFO file is read or write.</td>
</tr>
<tr>
<td>BUFFER</td>
<td>Module</td>
<td>The control record is read into this variable and is parsed into the appropriate variables: LOCAT, CNSTNT or ICONST, IPRN, and INFOPATH.</td>
</tr>
<tr>
<td>IARC</td>
<td>Module</td>
<td>Flag indicating whether the &quot;arc-section&quot; of the program code will be activated. &gt; 0, ASCII file storage &lt; 0, ARC/INFO file storage</td>
</tr>
<tr>
<td>INFOITEM</td>
<td>Submodule</td>
<td>The name of the ARC/INFO or INFO item where the values for the array are stored (passed argument consisting of the root name).</td>
</tr>
<tr>
<td>INFOPATH</td>
<td>Module</td>
<td>The complete path to the ARC/INFO file containing the item (INFOITEM) of interest.</td>
</tr>
<tr>
<td>ITEM</td>
<td>Submodule</td>
<td>The name of the ARC/INFO or INFO item where the values for the array are stored.</td>
</tr>
<tr>
<td>NAMPATH</td>
<td>Module</td>
<td>The complete path to the ARC/INFO file containing the item (INFOITEM) of interest that is read.</td>
</tr>
<tr>
<td>NUMREC</td>
<td>Module</td>
<td>Integer value for the number of record within the file specified by INFOPATH.</td>
</tr>
</tbody>
</table>

U2DINTARC

This module reads one-dimensional arrays and searches the control record for a new parameter INFOPATH that specifies a complete path to an ARC/INFO file (fig. 28). Documentation of the changes in program code follows.
Figure 28.--Modified program elements for the U2DINTARC module.

SUBROUTINE U2DINTARC (INFOITEM, IA, ANAME, II, JJ, K, IN, IOUT, E *)

---VERSION 3 25OCTOBER1991 U2DINTARC
MODIFIED BY LEONARD L. ORZOL

*****************************************************************************
ROUTINE TO INPUT 2-D INTEGER DATA MATRICES
IA IS ARRAY TO INPUT
ANAME IS 24 CHARACTER DESCRIPTION OF IA
II IS NO. OF ROWS
JJ IS NO. OF COLS
K IS LAYER NO. (USED WITH NAME TO TITLE PRINTOUT UNLESS K IS 0)
IN IS INPUT UNIT
IOUT IS OUTPUT UNIT
**SPECIFICATIONS:**

- `CHARACTER*(*) INFOITEM`
- `CHARACTER*132 BUFFER`
- `CHARACTER*128 INFOPATH`
- `CHARACTER*82 NAMPATH`
- `CHARACTER*16 ITEM`
- `CHARACTER*4 ANAME`
- `CHARACTER*20 FMTIN`
- `DIMENSION IA(JJ,II),ANAME(6)`
- `INTEGER FNUM,NCOLUMNS,NROWS,NUMREC,ACCESS`
- `EXTERNAL UCOLNO`
- `EXTERNAL INFO_CLOSING,INFO_NAMING,INFO_OPENS,INFO_READINT`

**READ ARRAY CONTROL RECORD.**

```c
READ(IN,'(A132)',ERR=9990,END=9991) BUFFER
```

**READ ARRAY FROM ASCII FILE LOCAT.**

```c
IF(BUFFER (51:51).EQ.'' .OR. BUFFER (51:51).EQ.' ') THEN
  READ (BUFFER,'(I10,I10,A20,I10)',ERR=9992)
  & LOCAT,ICONST,FMTIN,IPRN
  IARC=0
ELSE
  READ (BUFFER,'(I10,I10,A20,I10,A82)',ERR=9992)
  & LOCAT,ICONST,FMTIN,IPRN,NAMPATH
  INFOPATH=NAMPATH (1:INDEX(NAMPATH,' ')-1)
  IARC=1
ENDIF
```

**USE LOCAT TO SEE WHERE ARRAY VALUES COME FROM.**

```c
IF(LOCAT) 200,50,90
```

**READ RECORDS.**

```c
IF(K.GT.0) WRITE(IOUT,7) ANAME,ICONST,K
  FORMAT(1H0,///30X,6A4,' =',I15,' FOR LAYER',I3)
IF(K.LE.0) WRITE(IOUT,9) ANAME,LOCAT,FMTIN
  FORMAT(1H0,///30X,6A4,' WILL BE READ ON UNIT',I3,' USING FORMAT: ',A20/30X,96('-'))
DO 100 I=1,II
  READ(LOCAT,FMTIN,ERR=9993,END=9994) (IA(J,I),J=1,JJ)
100 CONTINUE
```

**READ RECORDS USING ARC/INFO ROUTINES.**

```c
IF(K.GT.0) THEN
  WRITE(IOUT,15) ANAME,K,INFOPATH (:INDEX(INFOPATH,' ')-1)
```

131
15    FORMAT(1H0,///10X,6A4, ' FOR LAYER',I3, &
   1         ' WILL BE READ FROM INFO FILE ',A,///10X,112(''),)
   ELSE
   WRITE(IOUT,17) ANAME,INFOPATH (:INDEX(INFOPATH,' ') -1)
17    FORMAT(1H0,///10X,6A4, ' WILL BE READ FROM INFO FILE ', &
   1         A,///10X,112(''))
ENDIF
C4BB-----TESTS EXISTENCE OF INFO FILE AND ITEM NAME.
C
NUMREC=II*JJ
ACCESS=1
CALL INFO_OPENS (INFOPATH,ACCESS,FNUM,NUMREC,*9999)
NCHAR=16
IPER=0
ISTP=0
ITEM=INFOITEM
KK=ABS(K)
CALL INFO_NAMING (ITEM,IPER,ISTP,KK,NCHAR)
C
C4BC-----OPENS AND READS INFO FILE.
C
CALL INFO_READINT (FNUM,NUMREC,ITEM,IA,*9999)
CALL INFO_CLOSING (FNUM)
ENDIF
GO TO 300
C
C5------LOCAT<0 THEN READ UNFORMATTED RECORD CONTAINING ARRAY VALUES
200 LOCAT=-LOCAT
   IF(K.GT.0) WRITE(IOUT,201) ANAME,K,LOCAT
201 FORMAT(1H0,///30X,6A4,', LAYER',I3, &
   1    ' WILL BE READ UNFORMATTED ON UNIT',I3/30X,73('-'))
   IF(K.LE.0) WRITE(IOUT,202) ANAME,LOCAT
202 FORMAT(1H0,///30X,6A4, &
   1    ' WILL BE READ UNFORMATTED ON UNIT',I3/30X,60('-'))
C
C5A------READ AN UNFORMATTED DUMMY RECORD FIRST.
READ(LOCAT)
READ(LOCAT,ERR=9993,END=9994) IA
C
C6------IF ICONST NOT ZERO THEN MULTIPLY ARRAY VALUES BY ICONST.
300 IF(ICONST.EQ.0) GO TO 320
   DO 310 I=1,II
   DO 310 J=1,JJ
      IA(J,I)=IA(J,I)*ICONST
310 CONTINUE
C
C7------IF PRINT CODE (IPRN) =>0 THEN PRINT ARRAY VALUES.
320 IF(IPRN.LT.0) RETURN
   IF(IPRN.GT.5) IPRN=0
   IPRN=IPRN+1
C
C8------PRINT COLUMN NUMBERS AT TOP OF PAGE.
   IF(IPRN.EQ.1) CALL UCOLNO(1,JJ,0,10,12,IOUT)
   NL=125/IPRN/5*5
   IF(IPRN.GT.1) CALL UCOLNO(1,JJ,4,NL,IPRN,IOUT)
C
C9------PRINT EACH ROW IN THE ARRAY.
   DO 110 I=1,II
C
C10-----SELECT THE FORMAT
      GO TO(101,102,103,104,105,106), IPRN
C
101 WRITE(IOUT,1001) I,(IA(J,I),J=1,JJ)
1001 FORMAT(1H0,I3,2X,I11,9(1X,I11)/(5X,10(1X,I11)))
      GO TO 110
C
102 WRITE(IOUT,1002) I,(IA(J,I),J=1,JJ)
1002 FORMAT(1H0,I3,1X,60(1X,I1)/100(1X,I1))
      GO TO 110
C
C----------------FORMAT 40I2
103 WRITE(IOUT,1003) I, (IA(J,I), J=1, JJ)
1003 FORMAT(1H0, I3, 1X, 40(I2)/(5X, 40(I2)))
        GO TO 110
C----------------FORMAT 30I3
104 WRITE(IOUT,1004) I, (IA(J,I), J=1, JJ)
1004 FORMAT(1H0, I3, 1X, 30(I3)/(5X, 30(I3)))
        GO TO 110
C----------------FORMAT 25I4
105 WRITE(IOUT,1005) I, (IA(J,I), J=1, JJ)
1005 FORMAT(1H0, I3, 1X, 25(I4)/(5X, 25(I4)))
        GO TO 110
110 CONTINUE
C----------------FORMAT 20I5
106 WRITE(IOUT,1006) I, (IA(J,I), J=1, JJ)
1006 FORMAT(1H0, I3, 1X, 20(I5)/(5X, 20(I5)))
        CONTINUE
C11------RETURN
C
RETURN
CCE------ERRORS
C
9990 CALL MESINT (K)
   CALL MESCHR (INFOITEM, 0)
   CALL INFORM ("Unable to read input control line for \"layer \%1\% and item \%2\%;\", -1)
   CALL INFORM ("Abnormal Termination of U2dintarc_Subroutine\", -1)
   RETURN 1
9991 CALL MESINT (K)
   CALL MESCHR (INFOITEM, 0)
   CALL INFORM ("End-of-file reached for layer \%1\% and item \%2\% \"Missing input package control line\", -1)
   CALL INFORM ("Abnormal Termination of U2dintarc_Subroutine\", -1)
   RETURN 1
9992 CALL MESCHR (BUFFER, 0)
   CALL MESINT (K)
   CALL MESCHR (INFOITEM, 0)
   CALL INFORM ("Unable to read from control line \%1\% for \"layer \%2\% and item \%3\%; Reformat control line\", -1)
   CALL INFORM ("Abnormal Termination of U2dintarc_Subroutine\", -1)
   RETURN 1
9993 CALL MESINT (K)
   CALL MESCHR (INFOITEM, 0)
   CALL INFORM ("Unable to read for layer \%1\% \"and item \%2\%; Bad format\", -1)
   CALL INFORM ("Abnormal Termination of U2dintarc_Subroutine\", -1)
   RETURN 1
9994 CALL MESINT (K)
   CALL MESCHR (INFOITEM, 0)
   CALL INFORM ("End-of-file reached for layer \%1\% and item \%2\%;\", -1)
   CALL INFORM ("Abnormal Termination of U2dintarc_Subroutine\", -1)
9999 CALL INFORM ("Abnormal Termination of U2dintarc_Subroutine\", -1)
   RETURN 1
END
<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCESS</td>
<td>Module</td>
<td>Flag indicating whether access to the ARC/INFO file is read or write.</td>
</tr>
<tr>
<td>BUFFER</td>
<td>Module</td>
<td>The control record is read into this variable and is parsed into the appropriate variables: LOCAT, CNSTNT or ICONST, IPRN, and INFOPATH.</td>
</tr>
<tr>
<td>IARC</td>
<td>Module</td>
<td>Flag indicating whether the &quot;arc-section&quot; of the program code will be activated. &gt; 0, ASCII file storage &lt; 0, ARC/INFO file storage</td>
</tr>
<tr>
<td>INFOITEM</td>
<td>Submodule</td>
<td>The name of the ARC/INFO or INFO item where the values for the array are stored (passed argument consisting of the root name).</td>
</tr>
<tr>
<td>INFOPATH</td>
<td>Module</td>
<td>The complete path to the ARC/INFO file containing the item (INFOITEM) of interest.</td>
</tr>
<tr>
<td>ITEM</td>
<td>Submodule</td>
<td>The name of the ARC/INFO or INFO item where the values for the array are stored.</td>
</tr>
<tr>
<td>NAMPATH</td>
<td>Module</td>
<td>The complete path to the ARC/INFO file containing the item (INFOITEM) of interest that is read.</td>
</tr>
<tr>
<td>NUMREC</td>
<td>Module</td>
<td>Integer value for the number of records within the file specified by INFOPATH.</td>
</tr>
</tbody>
</table>

**UBUDSVARC**

This module records three-dimensional cell-by-cell flow terms when activated by budget modules of the ground-water flow model packages (fig. 29). Program code was entirely rewritten to transfer control to the INFO utility modules of MODFLOWARC.
Figure 29.--Modified program elements for the UBUDSVARC module.

```fortran
SUBROUTINE UBUDSVARC (KSTP, KPER, INFOITEM, OUTPATH, BUFF, NCOL, NROW,
&                  NLAY, IOUT, 
E               *)

*****VERSION 3 25OCTOBER1991 UBUDSVARC
WRITTEN BY LEONARD L. ORZOL

******************************************************************************
RECORD CELL-BY-CELL FLOW TERMS FOR ONE COMPONENT OF FLOW.
******************************************************************************

SPECIFICATIONS:
------------------------------------------------------------------
CHARACTER*(*) INFOITEM, OUTPATH
CHARACTER*128 INFOPATH
CHARACTER*16 ITEM
```
DIMENSION BUFF(NCOL,NROW,NLAY)
INTEGER FNUM,NCHAR,NCOLUMNS,NROWS,NUMREC,ACCESS
EXTERNAL INFO_ADDITEM,INFO_CLOSING,INFO_CREATES,INFO_NAMING
EXTERNAL INFO_OPENS,INFO_ROWCOLUMN,INFO_WRITE
DATA ITEM // /
DATA INFOPATH // /

C----------------------------------------------------------------------
C
C1------SETTING ARC/INFO FILE INFOPATH AND VARIABLES
C
NROWS=NROW
NCOLUMNS=NCOL
INFOPATH=OUTPATH
ACCESS=3

C2------CREATING ARC/INFO FILE INFOPATH AND ITEMS
C
NCHAR=128
LAYER=0
CALL INFO_NAMING (INFOPATH,KPER,KSTP,LAYER,NCHAR)
CALL INFO_CREATES (INFOPATH,NROWS,NCOLUMNS,FNUM,*9999)

C3------ADDING ITEM NAMES BY LAYER TO ARC/INFO FILE
C
DO 100 LAYER=1,NLAY
NCHAR=16
IPER=0
ISTP=0
ITEM=INFOITEM
CALL INFO_NAMING (ITEM,IPER,ISTP,LAYER,NCHAR)
CALL INFO_ADDITEM (FNUM,ITEM,*9999)
100 CONTINUE

C4------WRITING ROWCOLUMN VALUES INTO INFOPATH ARC/INFO FILE
C
CALL INFO_CLOSING (FNUM)
CALL INFO_OPENS (INFOPATH,ACCESS,FNUM,NUMREC,*9999)
CALL INFO_ROWCOLUMN (FNUM,NROWS,NCOLUMNS,*9999)
NUMREC=NROWS*NCOLUMNS

C5------LOOP TO WRITE BUFF BY LAYER INTO INFOPATH ARC/INFO FILE
C
DO 200 ILAY=1,NLAY
NCHAR=16
IPER=0
ISTP=0
ITEM=INFOITEM
CALL INFO_NAMING (ITEM,IPER,ISTP,ILAY,NCHAR)
CALL INFO_WRITE (FNUM,NUMREC,ITEM,BUFF(1,1,ILAY),*9999)
200 CONTINUE
CALL INFO_CLOSING (FNUM)

WRITE(IOUT,1) INFOPATH (:INDEX(INFOPATH,' ')-1),KSTP,KPER
1    FORMAT(1X,'BUDGET VALUES SAVED IN INFO FILE ',
       &                 A,' AT END OF TIME STEP',I3,', STRESS PERIOD',I3)

C6------RETURN
C
RETURN

CE------ERRORS
C
9999 CALL INFORM
   & ('\Abnormal Termination of Ubudsvarc_Subroutine\ ',',-1)
RETURN 1
END
Added variables for module UBUDSVARC

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>FNUM</td>
<td>Module</td>
<td>Integer unit number used by the this routine for the file specified by INFOPATH.</td>
</tr>
<tr>
<td>ITEM</td>
<td>Module</td>
<td>The name of the ARC/INFO or INFO item where the values for the array are recorded (passed argument consisting of the root name).</td>
</tr>
<tr>
<td>INFOPATH</td>
<td>Module</td>
<td>The complete path to the ARC/INFO file where cell-by-cell flow terms are recorded.</td>
</tr>
<tr>
<td>NUMREC</td>
<td>Module</td>
<td>Integer value for the number of records within the file specified by INFOPATH.</td>
</tr>
<tr>
<td>OUTPATH</td>
<td>Module</td>
<td>The path to the ARC/INFO file where cell-by-cell flow terms are recorded (passed argument consisting of the directory path and file root name).</td>
</tr>
</tbody>
</table>

ULASAVARC

This module records three-dimensional arrays by layer when activated from calling budget modules of Basic (BAS) package (fig. 30). Program code was entirely rewritten to transfer control to the INFO utility modules of MODFLOWARC.

Figure 30.--Modified program elements for the ULASAVARC module.
SUBROUTINE ULASAVARC (BUF, INFOITEM, KSTP, KPER, NCOL, & NROW, ILAY, OUTPATH, QFILE, NLAY, IOFLG, E *)

C-----VERSION 3 25OCTOBER1991 ULASAVARC
  WRITTEN BY LEONARD L. ORZOL

*****************************************************************************
SAVE 1 LAYER ARRAY ON DISK
*****************************************************************************

SPECIFICATIONS:

DIMENSION BUF(NCOL, NROW), IOFLG(NLAY, 4)
CHARACTER*(*) INFOITEM, OUTPATH
CHARACTER*128 INFOPATH
CHARACTER*16 ITEM
INTEGER FNUM,NCHAR,NCOLUMNS,NROWS,NUMREC,ACCESS
LOGICAL QFILE
EXTERNAL INFO_ADDITEM, INFO_CLOSING, INFO_CREATES, INFO_NAMING
EXTERNAL INFO_OPENS, INFO_ROWCOLUMN, INFO_WRITE
DATA ITEM /' '/
DATA INFOPATH /' '/

-----------------------------------------------
C1-------SETTING ARC/INFO FILE AND VARIABLE NAMES

ACCESS=3
NROWS=NROW
NCOLUMNS=NCOL
INFOPATH=OUTPATH
NCHAR=128
LAYER=0
CALL INFO_NAMING (INFOPATH, KPER, KSTP, LAYER, NCHAR)

C2A-------CREATING ARC/INFO FILE INFOPATH AND ITEMS

IF(QFILE) THEN
  CALL INFO_CREATES (INFOPATH, NROWS, NCOLUMNS, FNUM, *9999)
ENDIF

C2B-------ADDING ITEM NAMES TO ARC/INFO FILE

DO 10 LAYER=1, NLAY
  IF(IOFLG(LAYER,3).LE.0) GO TO 10
  NCHAR=16
  IPER=0
  ISTP=0
  ITEM=INFOITEM
  CALL INFO_NAMING (ITEM, IPER, ISTP, LAYER, NCHAR)
  CALL INFO_ADDITEM (FNUM, ITEM, *9999)
10  CONTINUE

CALL INFO_CLOSING (FNUM)
CALL INFO_OPENS (INFOPATH, ACCESS, FNUM, NUMREC, *9999)

C2C-------WRITING ROWCOLUMN INTO ARC/INFO FILE

CALL INFO_ROWCOLUMN (FNUM, NROWS, NCOLUMNS, *9999)
CALL INFO_CLOSING (FNUM)

ENDIF

C3-------OPENING AGAIN AND WRITING INTO ARC/INFO FILE INFOPATH

CALL INFO_OPENS (INFOPATH, ACCESS, FNUM, NUMREC, *9999)
NCHAR=16
IPER=0
ISTP=0
ITEM=INFOITEM
CALL INFO_NAMING (ITEM, IPER, ISTP, ILAY, NCHAR)
CALL INFO_WRITE (FNUM, NUMREC, ITEM, BUF, *9999)
CALL INFO_CLOSING (FNUM)

C4-------RETURN
Added variables for module ULASASVARC

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>FNUM</td>
<td>Module</td>
<td>Integer unit number used by this routine for the file specified by INFOPATH.</td>
</tr>
<tr>
<td>INFOPATH</td>
<td>Module</td>
<td>The complete path to the ARC/INFO file where output head and drawdown arrays are recorded.</td>
</tr>
<tr>
<td>ITEM</td>
<td>Submodule</td>
<td>The name of the ARC/INFO or INFO item where the values for the array are recorded (passed argument consisting of the root name).</td>
</tr>
<tr>
<td>ITEMS</td>
<td>Submodule</td>
<td>List of names of the ARC/INFO or INFO items (ROWCOLUMN, ROW, and COLUMN) within the new created file.</td>
</tr>
<tr>
<td>NUMREC</td>
<td>Module</td>
<td>Integer value for the number of record within the file specified by INFOPATH.</td>
</tr>
<tr>
<td>OUTPATH</td>
<td>Module</td>
<td>The path to the ARC/INFO file where output head and drawdown arrays are recorded (passed argument consisting of the directory path and file root name).</td>
</tr>
<tr>
<td>QFILE</td>
<td>Submodule</td>
<td>Logical flag indicating whether ARC/INFO file for array values has been created (TRUE, create file FALSE, do not create file).</td>
</tr>
</tbody>
</table>

INFO Utility Module

The INFO utility module is a collection of routines that is used repeatedly to open ARC/INFO files, check and retrieve item names and their format, and read or write array values from or into the item fields within these files. The INFO utility module performs general tasks common to several different modules, submodules, or other utility modules of the ground-water model packages. The INFO module calls the ISP module within Arc/Info software and passes the variable names and their values between the ground-water model and the ARC/INFO file. Each of the INFO routines is described below along with a list of routines and modules, and documentation for program code, and a list of variables used in the routines.

INFO utility module routines

INFO_OPENS: Opens the ARC/INFO file (INFONAME) and checks the existence of the ARC/INFO file (INFONAME) using the path supplied by the user (INFOPATH).

INFO_READINT: Checks the existence of the item name (ITEM) of interest within the ARC/INFO file (INFONAME). Reads the integer values of this item (ITEM) within the ARC/INFO file (INFONAME) and returns array values.

INFO_READREAL: Checks the existence of the item name (ITEM) of interest within the ARC/INFO file (INFONAME). Reads the real values of this item (ITEM) within the ARC/INFO file (INFONAME) and returns array values.

INFO_READMULT: Checks the existence of each of these item names (ITEMS) of interest within the ARC/INFO file (INFONAME). Reads an ARC/INFO file (INFONAME) and returns array values for a number of items within
INFO_CREATES: Creates an ARC/INFO file (INFONAME) along with the items ROW and COLUMN plus a redefine item ROWCOLUMN composed of the values of ROW and COLUMN.

INFO_ADDITEM: Adds a new item to the ARC/INFO file (INFONAME).

INFO_WRITE: Writes array values for item (ITEM) info the active ARC/INFO file (INFONAME).

INFO_CLOSING: Closes the current active ARC/INFO file (INFONAME).

INFO_NAMING: Builds the name for ARC/INFO file or item by taking the root and appending the stress step and time step to this root name (an example is drawdown, the root name for an item, then appending _1, stress step, and then appending _1, time step within stress period, thus the final name is drawdown_1_1).

POWER: Finds the number of significant digits within the variables, ROW and COLUMN.

FORMATING: Builds the fortran format for the number of significant digits.

<table>
<thead>
<tr>
<th>Modules</th>
<th>Routines</th>
</tr>
</thead>
<tbody>
<tr>
<td>RIV1RPARC</td>
<td>INFO_OPENS</td>
</tr>
<tr>
<td>WEL1RPARC</td>
<td></td>
</tr>
<tr>
<td>DRN1RPARC</td>
<td></td>
</tr>
<tr>
<td>GHB1RPARC</td>
<td></td>
</tr>
<tr>
<td>UBUDSVARC</td>
<td></td>
</tr>
<tr>
<td>ULASAVARC</td>
<td></td>
</tr>
<tr>
<td>U1DRELARC</td>
<td></td>
</tr>
<tr>
<td>U2DRELARC</td>
<td></td>
</tr>
<tr>
<td>U2DINTARC</td>
<td></td>
</tr>
<tr>
<td>U2DINTARC</td>
<td>INOF_READINT</td>
</tr>
<tr>
<td>U1DRELARC</td>
<td>INFO_READREAL</td>
</tr>
<tr>
<td>U2DRELARC</td>
<td></td>
</tr>
<tr>
<td>RIV1RPARC</td>
<td>INFO_READMULT</td>
</tr>
<tr>
<td>WEL1RPARC</td>
<td></td>
</tr>
<tr>
<td>DRN1RPARC</td>
<td></td>
</tr>
<tr>
<td>GHB1RPARC</td>
<td></td>
</tr>
<tr>
<td>UBUDSVARC</td>
<td>INFO_CREATES</td>
</tr>
<tr>
<td>ULASAVARC</td>
<td>INFO_ROWCOLUMN</td>
</tr>
<tr>
<td>UBUDSVARC</td>
<td></td>
</tr>
<tr>
<td>ULASAVARC</td>
<td></td>
</tr>
<tr>
<td>RIV1RPARC</td>
<td>INFO_ADDITEM</td>
</tr>
<tr>
<td>WEL1RPARC</td>
<td></td>
</tr>
<tr>
<td>DRN1RPARC</td>
<td></td>
</tr>
<tr>
<td>GHB1RPARC</td>
<td></td>
</tr>
<tr>
<td>UBUDSVARC</td>
<td></td>
</tr>
<tr>
<td>ULASAVARC</td>
<td></td>
</tr>
<tr>
<td>U1DRELARC</td>
<td></td>
</tr>
<tr>
<td>U2DRELARC</td>
<td></td>
</tr>
<tr>
<td>U2DINTARC</td>
<td></td>
</tr>
</tbody>
</table>
C** Purpose: Although this program has been used by the U.S. Geological Survey, no warranty, expressed or implied, is made by the USGS as to the accuracy and functioning of the program and related program material nor shall the fact of distribution constitute any such warranty, and no responsibility is assumed by the USGS in connection therewith.

C* Purpose: A set of subroutines that deal with Arc/Info files (create, add items, etc.).

Subroutines:

INFO_ADDITEM : Creates an floating point item in an existing Arc/Info file that contains zero records.
INFO_CLOSING : Closes an open Arc/Info file.
INFO_CREATES : Creates an Arc/Info file with ROW, COLUMN, and redefined ROWCOLUMN items.
INFO_CREATE : Creates an itemless Arc/Info file (can copy an items table file from an existing file).
INFO_ITEM : Creates an user specified item in an existing empty Arc/Info file that contains zero records.
INFO_JOIN : Joins two Arc/Info files.
INFO_MWRITE : Writes values into multiple items of an existing Arc/Info file.
INFO_NAMES : Formulates Arc/Info file names from coverage name.
INFO_OPENS : Opens existing Arc/Info file.
INFO_READINT : Reads integer item from existing Arc/Info file.
INFO_READMULT : Reads real items from existing Arc/Info file (across a record).
INFO_READREAL : Reads real item from existing Arc/Info file.
INFO_ROWCOLUMN : Loads values into ROWCOLUMN item in existing Arc/Info file.
INFO_WRITE : Writes values into an item of an existing Arc/Info file.

Secondary Subroutines:

FORMATING : Formulates integer format.
INFO_NAMING : Formulates Arc/Info file name using stress period and Time step.
POWER : Computes power of ten of integer item.

.. Language: FORTRAN, with ARC/INFO subroutine call version 5.0

.. History:

USGS WRD Portland, Or
Fts: 429-2256

-------------------------------------------------------------------------
C**********************************************************************C
C<><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><>C
SUBROUTINE INFO_ADDITEM (  
                     FNUM,ITEM,  
                      *)
-----------------------------------------------------------------------
C
---VERSION 2.0 INFO_ADDITEM 18JUNE1990
C
Purpose:

    Adds floating point (4 12 F 3) item to an open Arc/Info data file (empty of item values).

.. Language: FORTRAN, with ARC/INFO subroutine call version 5.0

Inputs:

FNUM      : FILE TABLE NUMBER (OPEN)
ITEM      : NAME OF ITEM

Outputs:

.. History:

Leonard L. Orzol 06/18/90 Modified Coding.
USGS WRD Portland, Or
Fts: 429-2256
C---------------------------------------------
C SPECIFICATIONS:
---------------------------------------------
C
CHARACTER*16 ALTERN, AFTER, ITEM
CHARACTER*1 CHARTYPE
INTEGER DECIML, FNUM, IERROR, INDEX, KEYLEV
INTEGER KEYTYP, OCCUR, OUTPUT, POSIT, PROTCT
INTEGER READON, REDEF, TYPE, WIDTH, YESNO
C************************************************************C
REDEF=0
ALTERN=''
AFTER=''
TYPE=6
WIDTH=4
OUTPUT=12
DECIML=3
PROTCT=4
READON=0
KEYLEV=-1
KEYTYP=-1
INDEX=-1
OCCUR=-1
POSIT=0
C************************************************************C
CALL INFADI (FNUM, REDEF, ITEM, ALTERN, AFTER, TYPE,
& WIDTH, OUTPUT, DECIML, PROTCT, READON,
& KEYLEV, KEYTYP, INDEX, OCCUR, POSIT, IERROR)
C************************************************************C
C******************** ERRORS **********************C
C************************************************************C
IF (IERROR.LT.0) THEN
CALL MESCHR (ITEM, 0)
CALL INFORM
& ('\Unable to add Info item %1% (INFO_ADDITEM).', -1)
CALL INFCLS (FNUM)
CALL INFORM
& ('\Abnormal Termination of Info_Utility_Subroutines', -1)
RETURN 1
ENDIF
C************************************************************C
RETURN
END
C**********************************************************************C
C<><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><>C
SUBROUTINE INFO_CLOSING (I
FNUM)
C-----------------------------------------------------------------------
C
C---VERSION 2.0 INFO_CLOSING 18JUNE1990
C
C::: Purpose:
C
C::: Closes an open Arc/Info data file.
C
C::: Language: FORTRAN, with ARC/INFO subroutine call version 5.0
C
C::: Inputs:
C
FNUM : FILE TABLE NUMBER (OPEN)

C::: Outputs:

C::: History:

Leonard L. Orzol 06/18/90 Modified Coding.
USGS WRD Portland, Or

143
SUBROUTINE INFO_CREATES (INFOPATH,NROWS,NCOLS,FNUM,*)

C---VERSION 2.0 INFO_CREATES 18JUNE1990
C
C .. Purpose:
C       Create Arc/Info data file INFOPATH and include items ROW, COLUMN,
C       and redefined ROWCOLUMN.
C
C .. Language: FORTRAN, with ARC/INFO subroutine call version 5.0
C
C .. History:
C       Leonard L. Orzol 06/18/90 Modified Coding.
C       USGS WRD Portland, Or
C       Fts: 429-2256

C------------------------------------------
C        SPECIFICATIONS:
C------------------------------------------
C
CHARACTER*(*) INFOPATH
CHARACTER*128 DIRECT
CHARACTER*32 INFONAME,STR
CHARACTER*16 ALTERN,AFTER,ITEM
CHARACTER*8 USER
CHARACTER*1 CHARTYPE
INTEGER ACCESS,COPY,DECIML,FNUM,IERROR,INDEX
INTEGER KEYLEV,KEYTYP,NROWS,NCOLS,OCCUR,OUTPUT,POSIT,PROTCT
INTEGER READON,REDEF,TYPE,WIDTH,YESNO
C**********************************************************************C
C             Parcelling INFOPATH into DIRECT, USER, and INFONAME
C**********************************************************************C
CALL INFPTH (INFOPATH,DIRECT,USER,INFONAME)
C**********************************************************************C
C                Checking existence of INFONAME INFO file              C
C**********************************************************************C
CALL INFEXF (INFONAME,DIRECT,USER,IERROR)
C**************************************************C
IF(IERROR.LT.0 .OR. IERROR.GT.0) THEN

DELEXT=0
CALL INFERS (INFONAME, DIRECT, USER, DELEXT, IERROR)
IF (IERROR.EQ.-1) THEN
  CALL MESCHR (INFONAME, 0)
  CALL INFORM
& ('\Unable to erase Info file %1% (INFO_CREAT).',-1)
  CALL INFCLS (FNUM)
  CALL INFORM
& ('\Abnormal Termination of Info_Utility_Subroutines’,-1)
  RETURN 1
ENDIF
ENDIF
C**********************************************************************C
C           CREATING AND OPENING INFONAME INFO FILE
C**********************************************************************C
COPY=0
C************************************************************C
CALL INFDEF (INFONAME, DIRECT, USER, COPY, FNUM, IERROR)
C**************************************************C
C******************** ERRORS **********************C
C**************************************************C
IF (IERROR.EQ.-1) THEN
  CALL MESCHR (INFONAME, 0)
  CALL INFORM
& ('\Info file %1% already exists (INFO_CREAT).',-1)
  CALL INFCLS (FNUM)
  CALL INFORM
& ('\Abnormal Termination of Info_Utility_Subroutines’,-1)
  RETURN 1
ELSE IF (IERROR.LT.-1) THEN
  CALL MESCHR (DIRECT, 0)
  CALL INFORM
& ('\Unable to find Info directory %1% (INFO_CREAT).’,-1)
  CALL INFCLS (FNUM)
  CALL INFORM
& ('\Abnormal Termination of Info_Utility_Subroutines’,-1)
  RETURN 1
ELSE
C**********************************************************************C
C               ADDING ITEM NAME ROW TO INFONAME FILE
C**********************************************************************C
ITEM='ROW'
  CALL POWER (NROWS,MULT1)
  WIDTH=MULT1
  OUTPUT=WIDTH
  REDEF=0
  ALTERN=''
  AFTER=''
  TYPE=3
  DECIML=-1
  PROTCT=4
  REASON=0
  KEYLEV=-1
  KEYTYP=-1
  INDEX=-1
  OCCUR=-1
  POSIT=0
C**********************************************************************C
  CALL INFADI (FNUM, REDEF, ITEM, ALTERN, AFTER, TYPE,
  & WIDTH, OUTPUT, DECIML, PROTCT, REASON,
  & KEYLEV, KEYTYP, INDEX, OCCUR, POSIT, IERROR)
C**********************************************************************C
C******************** ERRORS **********************C
C**********************************************************************C
IF (IERROR.LT.0) THEN
  CALL MESCHR (ITEM, 0)
  CALL MESCHR (INFONAME, 0)
  CALL INFORM
& ('\Info item %1% already exists in Info file %2% ' //
& '(INFO_CREAT).’,-1)
  CALL INFCLS (FNUM)
  CALL INFORM
ENDIF
! ("\Abnormal Termination of Info_Utility_Subroutines",-1)
RETURN 1
ENDIF
C******************************************************************************C
C               ADDING ITEM NAME COLUMN TO INFONAME FILE
C******************************************************************************C
CALL POWER (NCOLS,MULT2)
ITEM='COLUMN'
WIDTH=MULT2
OUTPUT=WIDTH
ALTERN=''
C******************************************************************************C
CALL INFADI (FNUM,REDEF,ITEM,ALTERN,AFTER,TYPE,
& WIDTH,OUTPUT,DECIML,PROTCT,READON,
& KEYLEV,KEYTYP,INDEX,OCCUR,POSIT,IERROR)
C******************************************************************************C
C******************** ERRORS **********************C
C******************************************************************************C
IF (IERROR.LT.0) THEN
CALL MESCHR (ITEM,0)
CALL MESCHR (INFONAME,0)
CALL INFORM
& ("\Info item %1% already exists in Info file %2% '/
& '(INFO_CREATES),',-1)
CALL INFCLS (FNUM)
CALL INFORM
& ("\Abnormal Termination of Info_Utility_Subroutines",-1)
RETURN 1
ENDIF
C******************************************************************************C
C               ADDING ROWCOLUMN ITEM TO INFONAME FILE
C******************************************************************************C
ITEM='ROWCOLUMN'
REDEF=1
ALTERN=''
AFTER=''
WIDTH=MULT1+MULT2
OUTPUT=WIDTH
POSIT=1
C******************************************************************************C
CALL INFADI (FNUM,REDEF,ITEM,ALTERN,AFTER,TYPE,
& WIDTH,OUTPUT,DECIML,PROTCT,READON,
& KEYLEV,KEYTYP,INDEX,OCCUR,POSIT,IERROR)
C******************************************************************************C
C******************** ERRORS **********************C
C******************************************************************************C
IF (IERROR.LT.0) THEN
CALL MESCHR (ITEM,0)
CALL MESCHR (INFONAME,0)
CALL INFORM
& ("\Info item %1% already exists in Info file %2% '/
& '(INFO_CREATES),',-1)
CALL INFCLS (FNUM)
CALL INFORM
& ("\Abnormal Termination of Info_Utility_Subroutines",-1)
RETURN 1
ENDIF
ENDIF
RETURN
END
C******************************************************************************C
C<><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><>C
SUBROUTINE INFO_CREATE (I, PATH,COPY,
O, FNUM,
E)
C---------------------------------------------------------------------C
C---VERSION 2.0 INFO_CREATE 18JUNE1990
C******************************************************************************C
.. Purpose:
    Create a itemless Arc/Info data file INFOPATH.

.. Language: FORTRAN, with ARC/INFO subroutine call version 5.0

Inputs:

PATH : COMPUTER PATH TO ARC/INFO FILE
COPY : COPY FLAG
    => 0 EMPTY ITEMS FILE IS CREATED
    => ^= 0 ANOTHER FILE TABLE NUMBER; COPYS FROM THAT ITEMS FILE

Outputs:

FNUM : FILE TABLE NUMBER (OPEN)

.. History:

Leonard L. Orzol  06/18/90   Modified Coding.
USGS WRD Portland, Or
Fts: 429-2256

---

**SPECIFICATIONS:**

---

CHARACTER*(*) PATH
CHARACTER*128 INFOPATH,DIRECT
CHARACTER*32 INFONAME
CHARACTER*8 USER
INTEGER COPY,FNUM,IERROR
INTEGER READON,REDEF,TYPE,WIDTH,YESNO

C Parcelling INFOPATH into DIRECT, USER, and INFONAME
INFOPATH=PATH (1:INDEX(PATH, ‘ ‘)-1)
CALL INFPTH (INFOPATH,DIRECT,USER,INFONAME)

C Checking existence of INFONAME INFO file
CALL INFEXF (INFONAME,DIRECT,USER,IERROR)

IF(IERROR.LT.0 .OR. IERROR.GT.0) THEN
DELEXT=0
CALL INFERS (INFONAME,DIRECT,USER,DELEXT,IERROR)
ELSE
CALL MESCHR (INFONAME,0)
CALL INFORM
& (‘\Unable to erase Info file %1% (INFO_CREATE).’,-1)
CALL INFCLS (FNUM)
CALL INFORM
& (‘\Abnormal Termination of Info_Utility_Subroutines’,-1)
RETURN 1
ENDIF

C Creating and opening INFONAME INFO file
COPY=0
CALL INFDEF (INFONAME,DIRECT,USER,COPY,FNUM,IERROR)

C Errors
IF(IERROR.EQ.-1) THEN
CALL MESCHR (INFONAME,0)
CALL INFORM
& (‘\Info file %1% already exists (INFO_CREATE).’,-1)
CALL INFCLS (FNUM)
CALL INFORM
& ('\Abnormal Termination of Info_Utility_Subroutines',-1)
RETURN 1
ELSE IF(IERROR.LT.-1) THEN
CALL MESCHR (DIRECT,0)
CALL INFORM
& ('\Unable to find Info directory %1% (INFO_CREATE).',-1)
CALL INFCLS (FNUM)
CALL INFORM
& ('\Abnormal Termination of Info_Utility_Subroutines',-1)
RETURN 1
ELSE
RETURN
ENDIF
RETURN
END

C**********************************************************************
C
C<><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><>C

SUBROUTINE INFO_ITEMS (FNUM,ITEM,AFTER,ALTERN,DECIML,INDEX,KEYLEV,KEYTYP, 
OCCUR,OUTPUT,POSIT,PROTCT,READON,REDEF,TYPE,WIDTH, *)

C-----------------------------------------------------------------------
C
C---VERSION 2.0 INFO_ITEMS 18JUNE1990
C
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
C .. Purpose:
C
C       Adds item to an open Arc/Info data file (empty of item values).
C
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
C .. Language: FORTRAN, with ARC/INFO subroutine call version 5.0
C
C Inputs:
C
C AFTER : ITEM NAME AFTER WHICH THE NEW ITEM WILL BE ADDED
C ALTERN : ALTERNATE NAME OF NEW ITEM
C DECIML : NUMBER OF DECIMAL PLACES
C => -1 NO DECIMAL PLACES
C => ^= VAILD DECIMAL PLACES
C FNUM : FILE TABLE NUMBER (OPEN)
C INDEX : INDEX NUMBER
C ITEM : NAME OF ITEM
C KEYLEV : KEY LEVEL
C => -1 NO KEY
C => ^= VAILD KEY
C KEYTYP : KEY TYPE
C => -1 NO KEY
C => 0 DATA KEY
C => 1 BIT KEY
C OCCUR : OCCURRENCE COUNT
C => -1 NONE
C => ^= VAILD COUNT
C OUTPUT : ITEM OUTPUT WIDTH
C POSIT : STARTING COLUMN FOR A REDEFINED ITEM
C PROTCT : PROTECTION LEVEL
C READON : READ ONLY FLAG
C REDEF : ITEM TYPE FLAG
C => 0 NORMAL ITEM
C => 1 REDEFINED ITEM
C TYPE : ITEM TYPE NUMBER
C => 1 D DATE
C => 2 C CHARACTER
C => 3 I INTEGER
C => 4 N NUMBERIC
=> 5 B BINARY
=> 6 F FLOATING POINT
WIDTH   : ITEM WIDTH

Outputs:

.. History:

Leonard L. Orzol  06/18/90 Modified Coding.
USGS WRD Portland, Or
Fts: 429-2256

SPECIFICATIONS:

CHARACTER*16 ALTERN,AFTER,ITEM
CHARACTER*1 CHARTYPE
INTEGER DECIML,FNUM,IERROR,INDEX,KEYLEV
INTEGER KEYTYP,OCCUR,OUTPUT,POSIT,PROTCT
INTEGER READON,REDEF,TYPE,WIDTH,YESNO

CALL INFADI (FNUM,REDEF,ITEM,ALTERN,AFTER,TYPE,
& WIDTH,OUTPUT,DECIML,PROTCT,READON,
& KEYLEV,KEYTYP,INDEX,OCCUR,POSIT,IERROR)

ERRORS

IF(IERROR.LT.0) THEN
   CALL MESCHR (ITEM,0)
   CALL INFORM
   & ('\Unable to add Info item %1% (INFO_ITEMS).',-1)
   CALL INFCLS (FNUM)
   CALL INFORM
   & ('\Abnormal Termination of Info_Utility_Subroutines',-1)
RETURN 1
ENDIF

RETURN

SUBROUTINE INFO_JOIN (INFILE,JONFIL,OUTFIL,RELATE,START,JTYPE,*)

---VERSION 2.0 INFO_JOIN 13JULY1990

Purpose:

Joins two Arc/Info files.

Language: FORTRAN, with ARC/INFO subroutine call version 5.0

Inputs:

INFILE   : INPUT INFO DATA FILE
JONFIL   : JOIN INFO DATA FILE
OUTFIL   : OUTPUT INFO DATA FILE
RELATE   : RELATE ITEM
START    : START ITEM WHERE ITEMS WILL BE INSERTED
JTYPE    : TYPE OF RELATE
  => 1 ORDERED
  => 2 LINEAR (DEFAULT)
  => 3 LINKED
C . History:
C
Leonard L. Orzol 07/13/90 Modified Coding.
USGS WRD Portland, Or
Fts: 429-2256

C-------------------------------------------------------------------------
C  SPECIFICATIONS:
C ------------------------------------------------------------------
C
REAL*8 INFOVAL
REAL VAL(NUMREC,NITEMS)
CHARACTER*(*) ITEMS(NITEMS)
CHARACTER*32 STR
CHARACTER*16 ITEM
INTEGER FNUM,INREC(2000),ITEMAR4(4),RECNUM
C**************************************************************************C

C C<><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><>
SUBROUTINE INFO_MWRITE (FNUM,NUMREC,ITEMS,NITEMS,VAL,E*)
C-----------------------------------------------------------------------
C
C---VERSION 2.0 INFO_MWRITE 18JUNE1990
C
C::: Purpose:
C
                Writes item values into an open Arc/Info data file.
C::: Language: FORTRAN, with ARC/INFO subroutine call version 5.0
C
C . Inputs:
C
FNUM : FILE TABLE NUMBER (OPEN)
ITEMS : NAME OF ITEMS
NUMREC : NUMBER OF RECORDS IN DATA FILE
VAL : INTEGER ARRAY VALUES (NUMREC,NITEMS)

C . Outputs:
C
C . History:
C
Leonard L. Orzol 06/18/90 Modified Coding.
USGS WRD Portland, Or
Fts: 429-2256

C-------------------------------------------------------------------------
C       Loop to read item format in INFONAME info file       C
C**********************************************************************C
DO 100 N=1,NITEMS
ITEM=ITEMS(N)
CALL INFEXI (FNUM,ITEM,ITEMAR4,IERROR)
C**********************************************************************C
C******************** ERRORS ****************************************C
C**********************************************************************C
IF(IERROR.LT.1) THEN
   CALL MESCHR (ITEM,0)
   CALL INFORM
   & ('"Unable to write Info item %1% (INFO_MWRITE).',-1)
   CALL INFCLS (FNUM)
   CALL INFORM
   & ('"Abnormal Termination of Info_Utility_Subroutines',-1)
   RETURN 1
ENDIF
C**********************************************************************C
C           LOOP TO WRITE ITEM IN INFONAME INFO FILE         C
C**********************************************************************C
DO 110 RECNUM=1,NUMREC
INFOVAL=VAL(RECNUM,N)
C**********************************************************************C
C            Getting Records From INFONAME Info File         C
C**********************************************************************C
CALL INFGET (FNUM,RECNUM,INREC,IERROR)
C**********************************************************************C
C******************** ERRORS ****************************************C
C**********************************************************************C
IF(IERROR.NE.0) THEN
   CALL MESCHR (ITEM,0)
   CALL MESINT (RECNUM,0)
   CALL INFCLS (FNUM)
   CALL INFORM ('"Unable to get record %2% for Info ' //
    & ' item %1% during INFO_MWRITE.',-1)
   RETURN 1
ENDIF
C**********************************************************************C
C                       Encoding items                       C
C**********************************************************************C
CALL INFENC (ITEMAR4,INFOVAL,STR,8000,INREC,IERROR)
C**********************************************************************C
C******************** ERRORS ****************************************C
C**********************************************************************C
IF(IERROR.LT.0) THEN
   CALL MESCHR (ITEM,0)
   CALL MESINT (RECNUM,0)
   CALL INFCLS (FNUM)
   CALL INFORM ('"Encoding for record %2%, Illegal item type for ' //
    & ' Info item %1% (INFO_MWRITE).',-1)
   CALL INFCLS (FNUM)
   CALL INFORM ('"Abnormal Termination of Info_Utility_Subroutines',-1)
   RETURN 1
ENDIF
C**********************************************************************C
C               Write records to INFO file                   C
C**********************************************************************C
CALL INFPUT (FNUM,RECNUM,INREC,IERROR)
C**********************************************************************C
C******************** ERRORS ****************************************C
C**********************************************************************C
IF(IERROR.NE.0) THEN
   CALL MESCHR (ITEM,0)
   CALL MESINT (RECNUM,0)
   CALL INFCLS (FNUM)
   CALL INFORM ('"Unable to write record %2% for Info item %1% ' //
    & ' (INFO_MWRITE).',-1)
   CALL INFCLS (FNUM)
   CALL INFORM ('"Abnormal Termination of Info_Utility_Subroutines',-1)
   RETURN 1
ENDIF
SUBROUTINE INFO_NAMES (INFOPATH, TYPE, INFONM)

C**********************************************
C     Specifying an Arc/Info data file name for a coverage INFOPATH.
C**********************************************

C.........Inputs:
C INFOPATH : COMPUTER PATH TO ARC/INFO COVERAGE
C TYPE     : FILE TYPE
C          => 1 .PAT
C          => 2 .AAT
C          => 3 .XAT (.PAT UNTIL POINT ATTRIBUTES)
C          => 4 .BND
C          => 5 .TIC
C          => 6 .CODE
C          => 7 .PRF
C          => 8 .ARF
C          => 9 .XRF
C          => 10 .TRN
C          => 11 .ADD
C          => 12 .REJ
C          => 13 .TXT
C
C.........Outputs:
C INFONM   : INFO FILENAME GENERATED
C
C.........History:
C Leonard L. Orzol 06/18/90 Modified Coding.
C USGS WRD Portland, Or
C Fts: 429-2256
C
C**********************************************

CALL INFPTH (INFOPATH, DIRECT, USER, INFONAME)
CALL INFNAM (INFONAME, TYPE, INFONM)

RETURN
SUBROUTINE INFO_OPENS (I INFOPATH, ACCESS, 
O FNUM, NUMREC, 
E *)
C**********************************************************************C
C<><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><>C
C---VERSION 2.0 INFO OPENS 18JUNE1990
C
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
C .. Purpose:
C
Vertifying Existence and opening of Arc/Info file INFOPATH.
C
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
C .. Language: FORTRAN, with ARC/INFO subroutine call version 5.0
C
C Inputs:
C
ACCESS   : WRITE/READ FLAG
==> 1 READ ONLY
==> 2 READ/WRITE, DELETE INDEXES ON WRITE
==> 3 READ/WRITE
INFOPATH : COMPUTER PATH TO ARC/INFO FILE
Outputs:
FNUM     : FILE TABLE NUMBER
NUMREC   : NUMBER OF RECORDS IN DATA FILE
C
C .. History:
C
Leonard L. Orzol  06/18/90     Modified Coding.
USGS WRD Portland, Or
Fts: 429-2256
C
C**********************************************************************C
C             Parcelling INFOPATH into DIRECT, USER, and INFONAME
C**********************************************************************C
CALL INFPTH (INFOPATH, DIRECT, USER, INFONAME)
C**********************************************************************C
C           VERTIFYING EXISTENCE AND OPENING OF INFONAME INFO FILE
C**********************************************************************C
CALL INFOPN (INFONAME, DIRECT, USER, ACCESS, 
&                                FNUM, NUMREC, RECLEN, IERROR)
C**************************************************C
C******************** ERRORS **********************C
C**************************************************C
IF (IERROR.LE.-4) THEN
CALL MESCHR (INFONAME, 0)
CALL INFORM
& ('Unable to open external Info file %1% (INFO_OPENS).',-1)
CALL INFORM
& ('Abnormal Termination of Info_Utility_Subroutines',-1)
RETURN 1
ELSE IF (IERROR.EQ.-3) THEN
CALL MESCHR (INFONAME, 0)
CALL INFORM
ENDIF
CALL INFORM
& ('\Unable to open Info file %1% (INFO_OPENS).',-1)
RETURN 1
ELSE IF(IERROR.EQ.-1) THEN
CALL MESCHR (INFONAME,0)
CALL INFORM ('\Unable to open Info file %1% , ' //
& 'does not exist (INFO_OPENS).',-1)
CALL INFORM
& ('\Abnormal Termination of Info_Utility_Subroutines',-1)
RETURN 1
ELSE
RETURN
ENDIF
C**********************************************************************C
RETURN
END
C**********************************************************************C
C<><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><>C
SUBROUTINE INFO_READINT ( 
  I                         FNUM,NUMREC,ITEM,
  O                         IOUTVAL,
  E                         *)
C-----------------------------------------------------------------------
C
C---VERSION 2.0 INFO_READINT 18JUNE1990
C
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
C .. Purpose:
C
C       Reads the values from one integer item in an open Arc/Info
C        data file.
C
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
C .. Language: FORTRAN, with ARC/INFO subroutine call version 5.0
C
C Inputs:
C
FNUM    : FILE TABLE NUMBER (OPEN)
ITEM    : NAME OF ITEM
NUMREC  : NUMBER OF RECORDS IN DATA FILE

C Outputs:

IOUTVAL : INTEGER ARRAY VALUES (NUMREC)

C .. History:
C
Leonard L. Orzol     06/18/90     Modified Coding.
USGS WRD Portland, Or
Fts: 429-2256

C<><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><>C
C-----------------------------------------------------------------------
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
C
REAL*8 INFOVAL
INTEGER IOUTVAL(NUMREC)
CHARACTER*32 STR
CHARACTER*16 ITEM
INTEGER FNUM,IERROR,INREC(2000)
INTEGER ITEMAR4(4), RECLEN, RECNUM
C*******************************************************************************
C LOOP TO READ ITEM IN INFONAME INFO FILE
C*******************************************************************************
CALL INFEXI (FNUM,ITEM,ITEMAR4,IERROR)
C*******************************************************************************
C ERRORS*************************************************************************
C*******************************************************************************
IF (IERROR.LT.1) THEN
   CALL MESCHR (ITEM,0)
   CALL INFORM
   & ('\Info item %1% does not exist (INFO_READINT).',-1)
   CALL INFCLS (FNUM)
   CALL INFORM
   & ('\Abnormal Termination of Info Utility Subroutines',-1)
   RETURN 1
ENDIF
C*******************************************************************************
C CLEAR INTEGER BUFFER FOR ITEM VALUE STORAGE
C*******************************************************************************
CALL INFIBF (FNUM,INREC,2000,IERROR)
C*******************************************************************************
C SELECTING RECORDS FROM INFONAME INFO FILE
C*******************************************************************************
DO 200 RECNUM=1,NUMREC
C*******************************************************************************
C GETTING RECORDS FROM INFONAME INFO FILE
C*******************************************************************************
CALL INFGET (FNUM,RECNUM,INREC,IERROR)
C*******************************************************************************
C ERRORS*************************************************************************
C*******************************************************************************
IF (IERROR.NE.0) THEN
   CALL MESCHR (ITEM,0)
   CALL MESINT (RECNUM,0)
   CALL INFORM ('\Unable to read Info item %1% for record %2%' //
   & 'during INFO_READINT.',-1)
   CALL INFCLS (FNUM)
   CALL INFORM
   & ('\Abnormal Termination of Info Utility Subroutines',-1)
   RETURN 1
ENDIF
C*******************************************************************************
C READING TRANSLATED VALUES
C*******************************************************************************
CALL INFDEC (INREC,8000,ITEMAR4,INFOVAL,STR,IERROR)
C*******************************************************************************
C ERRORS*************************************************************************
C*******************************************************************************
IF (IERROR.EQ.-1) THEN
   CALL MESCHR (ITEM,0)
   CALL MESINT (RECNUM,0)
   CALL INFORM ('\Unable to decode Info item %1% for record %2% '  //
   & '(INFO_READINT).',-1)
   CALL INFCLS (FNUM)
   CALL INFORM
   & ('\Abnormal Termination of Info Utility Subroutines',-1)
   RETURN 1
ELSE IF (IERROR.LT.-1) THEN
   CALL MESCHR (ITEM,0)
   CALL MESINT (RECNUM,0)
   CALL INFORM
   & ('\Bad value in Info item %1% for record %2% '  //
   & '(INFO_READINT).',-1)
   CALL INFCLS (FNUM)
   CALL INFORM
   & ('\Abnormal Termination of Info Utility Subroutines',-1)
   RETURN 1
ELSE
   IOUTVAL(RECNUM) = INFOVAL
ENDIF
SUBROUTINE INFO_READMULT (FNUM, NUMREC, NUMVAL, ITEMS, NITEMS, VAL, *)

C-----------------------------------------------------------------------
C
C---VERSION 2.0 INFO_READREAL 18JUNE1990
C
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
C .. Purpose:
C
C       Reads the values from n real items in an open Arc/Info data file.
C
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
C .. Language: FORTRAN, with ARC/INFO subroutine call version 5.0
C
C Inputs:
C
FNUM : FILE TABLE NUMBER (OPEN)
ITEMS : NAME OF ITEM
NITEMS : NUMBER OF ITEMS TO READ
NUMREC : NUMBER OF RECORDS IN DATA FILE
NUMVAL : NUMBER OF VALUES TO READ (NUMREC*NITEMS) IN DATA FILE

C Outputs:
C
VAL : INTEGER ARRAY VALUES (NUMVAL)

C History:
C
Leonard L. Orzol 06/18/90  Modified Coding.
USGS WRD Portland, Or
Fts: 429-2256

CSPECIFICATIONS:
---------------------------------------------------------------
C
REAL*8 INFOVAL
REAL VAL(NUMVAL)
CHARACTER*32 STR
CHARACTER*16 ITEMS(NITEMS)
INTEGER FNUM, IERROR, INREC(2000), ITEMAR4(4)
INTEGER NUMREC, RECLEN, RECNUM
C
C-----------------------------------------------------------------------
C
IRECORD=0
C
CALL INFGET (FNUM, 1, RECNUM, INREC, IERROR)
C
IF (IERROR .NE. 0) THEN
  CALL MESINT (RECNUM, 0)
  CALL INFORM ('\Unable to read record %1% during INFO_READMULT.', -1)
  CALL INFCLS (FNUM)
ENDIF
CALL INFORM
& ('\Abnormal Termination of Info_Utility_Subroutines', -1)
RETURN 1
ENDIF
C******************************************************************************C
C           LOOP TO READ ITEM IN INFONAME INFO FILE          C
C******************************************************************************C
DO 110 I=1,NITEMS
CALL INFEXI (FNUM, ITEMS(I), ITEMAR4, IERROR)
C******************************************************************************C
C******************** ERRORS **********************C
C******************************************************************************C
IF (IERROR.LT.1) THEN
CALL MESCHR (ITEMS(I), 0)
CALL INFORM
& ('\Info item %1% does not exist (INFO_READMULT).', -1)
CALL INFCLS (FNUM)
CALL INFORM
& ('\Abnormal Termination of Info_Utility_Subroutines', -1)
RETURN 1
ENDIF
C******************************************************************************C
C************ READING TRANSLATED VALUES ***********C
C******************************************************************************C
CALL INFDEC (INREC, 8000, ITEMAR4, INFOVAL, STR, IERROR)
C******************************************************************************C
C******************** ERRORS **********************C
C******************************************************************************C
IF (IERROR.EQ.-1) THEN
CALL MESCHR (ITEMS(I), 0)
CALL MESINT (RECNUM, 0)
CALL INFORM
& ('\Unable to decode Info item %1% for record %2% ' //
& '\(INFO_READMULT).', -1)
CALL INFCLS (FNUM)
CALL INFORM
& ('\Abnormal Termination of Info_Utility_Subroutines', -1)
RETURN 1
ELSE IF (IERROR.LT.-1) THEN
CALL MESCHR (ITEMS(I), 0)
CALL MESINT (RECNUM, 0)
CALL INFORM
& ('\Bad value in Info item %1% for record %2% ' //
& '\(INFO_READMULT).', -1)
CALL INFCLS (FNUM)
CALL INFORM
& ('\Abnormal Termination of Info_Utility_Subroutines', -1)
RETURN 1
ELSE
IRECORD=IRECORD+1
VAL(IRECORD)=INFOVAL
ENDIF
C******************************************************************************C
110 CONTINUE
100 CONTINUE
RETURN
END
C******************************************************************************C
C<><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><>C
SUBROUTINE INFO_READREAL ( 
    I                          FNUM, NUMREC, ITEM, 
    O                          VAL, 
    E                          *)
C-----------------------------------------------------------------------
C
C---VERSION 2.0 INFO_READREAL 18JUNE1990
C
C::: Purpose:
C
C    Reads the values from one real item in an open Arc/Info
data file.

.. Language: FORTRAN, with ARC/INFO subroutine call version 5.0

Inputs:

FNUM : FILE TABLE NUMBER (OPEN)
ITEM : NAME OF ITEM
NUMREC : NUMBER OF RECORDS IN DATA FILE

Outputs:

VAL : INTEGER ARRAY VALUES (NUMREC)

.. History:

Leonard L. Orzol 06/18/90 Modified Coding.
USGS WRD Portland, Or
Fts: 429-2256

SPECIFICATIONS:

REAL*8 INFOVAL
REAL VAL(NUMREC)
CHARACTER*32 STR
CHARACTER*16 ITEM
CHARACTER*8 USER
INTEGER FNUM,IERROR,INREC(2000),ITEMAR4(4),RECLEN,RECNUM

LOOP TO READ ITEM IN INFONAME INFO FILE
CALL INFEXI (FNUM,ITEM,ITEMAR4,IERROR)

******************** ERRORS **********************
IF(IERROR.LT.1) THEN
  CALL MESCHR (ITEM,0)
  CALL INFORM
  & ('\Info item %1% does not exist (INFO_READREAL).',-1)
  CALL INFCLS (FNUM)
  CALL INFORM
  & ('\Abnormal Termination of Info_Utility_Subroutines',-1)
RETURN 1
ENDIF

SELECTING RECORDS FROM INFONAME INFO FILE
DO 100 RECNUM=1,NUMREC
  GETTING RECORDS FROM INFONAME INFO FILE
  CALL INFGET (FNUM,RECNUM,INREC,IERROR)
  ERRORS ************
  IF(IERROR.NE.0) THEN
    CALL MESCHR (ITEM,0)
    CALL MEINT (RECNUM,0)
    CALL INFORM ('\Unable to read Info item %1% for record %2% ' //
    \'during INFO_READREAL\'.',-1)
    CALL INFCLS (FNUM)
    CALL INFORM
    & ('\Abnormal Termination of Info_Utility_Subroutines',-1)
RETURN 1
ENDIF
  READING TRANSLATED VALUES

C**************************************************C
CALL INFDEC (INREC,8000,ITEMAR4,INFOVAL,STR,IERROR)
C**************************************************C
C**************** ERRORS **********************C
C**************************************************C
IF (IERROR.EQ.-1) THEN
   CALL MESCHR (ITEM,0)
   CALL MESINT (RECNUM,0)
   CALL INFORM &
   & \"Unable to decode Info item %1% for record %2% \" //
   & \"(INFO_READREAL),\",-1)
   CALL INFCLS (FNUM)
   CALL INFORM &
   & \"Abnormal Termination of Info_Utility_Subroutines\",-1)
   RETURN 1
ELSE IF (IERROR.LT.-1) THEN
   CALL MESCHR (ITEM,0)
   CALL MESINT (RECNUM,0)
   CALL INFORM &
   & \"Bad value in Info item %1% for record %2% \" //
   & \"(INFO_READREAL),\",-1)
   CALL INFCLS (FNUM)
   CALL INFORM &
   & \"Abnormal Termination of Info_Utility_Subroutines\",-1)
   RETURN 1
ELSE
   VAL(RECNUM)=SNGL(INFOVAL)
ENDIF
C**********************************************************************C
100 CONTINUE
RETURN
END
C**********************************************************************C
C<><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><>C
SUBROUTINE INFO_ROWCOLUMN (I,FNUM,NROWS,NCOLS,E)
C-----------------------------------------------------------------------
C---VERSION 2.0 INFO_ROWCOLUMN 18JUNE1990
C
C::: Purpose:
C
C   Loads values in the item named ROWCOLUMN (used by
C   Fortran_Data_Interface, Orzol and McGrath, 1990).
C
C::: Language: FORTRAN, with ARC/INFO subroutine call version 5.0
C
C::: Inputs:
C
C   FNUM : FILE TABLE NUMBER
C   NCOLS : NUMBER OF COLUMNS
C   NROWS : NUMBER OF ROWS
C
C::: Outputs:
C
C   FNUM : FILE TABLE NUMBER (OPEN)
C
C::: History:
C
C   Leonard L. Orzol 06/18/90 Modified Coding.
C   USGS WRD Portland, Or
C   Fts: 429-2256
C
C-------------------------------------------------------------------------
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
REAL*8 ROWCOLUMNS
CHARACTER*32 STR
CHARACTER*16 ITEM
INTEGER COLUMNS,FNUM,IERROR,INREC(2000),ITEMAR4(4),
&    NROWS,NCOLS,RECNUM,ROWS
C**********************************************************************C
NUMREC=NROWS*NCOLS
ITEM=’ROWCOLUMN’
COLUMNS=0
ROWS=1
CALL POWER (NCOLS,MULT2)
C************************************************************C
C                   Retrieving item format                   C
C************************************************************C
CALL INFEXI (FNUM,ITEM,ITEMAR4,IERROR)
C**************************************************C
C******************** ERRORS **********************C
C**************************************************C
IF(IERROR.LT.1) THEN
   CALL MESCHR (ITEM,0)
   CALL INFORM
   & (‘\Info item %1% does not exist (INFO_ROWCOLUMN)’)
   CALL INFCLS (FNUM)
   CALL INFORM
   & (‘\Abnormal Termination of Info_Utility_Subroutines’,-1)
   RETURN 1
ENDIF
C**********************************************************************C
C                 LOOP TO WRITE VALUES INTO ITEM ROWCOLUMN             C
C**********************************************************************C
DO 100 RECNUM=1,NUMREC
   COLUMNS=COLUMNS+1
   IF(COLUMNS.GT.NCOLS) THEN
      ROWS=ROWS+1
      COLUMNS=1
   ENDIF
   ROWCOLUMNS=FLOAT(ROWS*(10**MULT2)+COLUMNS)
C************************************************************C
C          CLEAR INTEGER BUFFER FOR ITEM VALUE STORAGE
C************************************************************C
C      CALL INFIBF (FNUM,INREC,2000,IERROR)
C************************************************************C
C                       Encoding items                       C
C************************************************************C
CALL INFENC (ITEMAR4,ROWCOLUMNS,STR,8000,INREC,IERROR)
C**************************************************C
C******************** ERRORS **********************C
C**************************************************C
IF(IERROR.LT.0) THEN
   CALL MESCHR (ITEM,0)
   CALL MESINT (RECNUM,0)
   CALL INFORM
   & (‘\Illegal item type for Info item %1% ‘ //
   & ‘when encoding for record %2% (INFO_ROWCOLUMN)’)
   CALL INFCLS (FNUM)
   CALL INFORM
   & (‘\Abnormal Termination of Info_Utility_Subroutines’,-1)
   RETURN 1
ENDIF
C**********************************************************************C
C               Write records to INFO file                   C
C**********************************************************************C
CALL INFPUT (FNUM,RECNUM,INREC,IERROR)
C**************************************************C
C******************** ERRORS **********************C
C**************************************************C
IF(IERROR.NE.0) THEN
   CALL MESCHR (ITEM,0)
   CALL MESINT (RECNUM,0)
   CALL INFORM
   & (‘\Unable to write to Info item %1% for record number %2% ’ //
& '('INFO_ROWCOLUMN')
CALL INFCLS (FNUM)
CALL INFORM
& ('Abnormal Termination of Info_Utility_Subroutines',-1)
RETURN 1
ENDIF
C**********************************************************************C
100 CONTINUE
RETURN
END
C**********************************************************************C
C<><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><>C
SUBROUTINE INFO_WRITE
  (  FNUM,NUMREC,INFOITEM,VAL,
    *)
C-----------------------------------------------------------------------
C---VERSION 2.0 INFO_WRITE 18JUNE1990
C
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
C .. Purpose:
C
Writes item values into an open Arc/Info data file.
C
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
C .. Language: FORTRAN, with ARC/INFO subroutine call version 5.0
C
Inputs:
C
FNUM     : FILE TABLE NUMBER (OPEN)
ITEM     : NAME OF ITEM
NUMREC   : NUMBER OF RECORDS IN DATA FILE
VAL      : INTEGER ARRAY VALUES (NUMREC)

Outputs:
C
C .. History:
C
Leonard L. Orzol 06/18/90 Modified Coding.
USGS WRD Portland, Or
Fts: 429-2256
C
C-------------------------------------------------------------------------
C        SPECIFICATIONS:
C     ------------------------------------------------------------------
C
REAL*8 INFOVAL
REAL VAL(NUMREC)
CHARACTER*(*) INFOITEM
CHARACTER*32 STR
CHARACTER*16 ITEM
INTEGER FNUM,IERROR,INREC(2000),ITEMAR4(4),RECNUM
C**********************************************************************C
ITEM=INFOITEM
C Loop to read item format in INFONAME info file
C
CALL INFEXI (FNUM,ITEM,ITEMAR4,IERROR)
C**************************************************
C******************** ERRORS **********************
C**************************************************
IF(IERROR.LT.1) THEN
  CALL MESCHR (ITEM,0)
  CALL INFORM
  & ('Unable to write Info item %1% (INFO_WRITE).',',-1)
  CALL INFCLS (FNUM)
  CALL INFORM
  & ('Abnormal Termination of Info_Utility_Subroutines',-1)
  RETURN 1
END IF
C***************************************************************C
C           LOOP TO READ ITEM IN INFONAME INFO FILE          C
C***************************************************************C
DO 100 RECNUM=1,NUMREC
INFOVAL=VAL(RECNUM)
C***************************************************************C
C            Getting Records From INFONAME Info File         C
C***************************************************************C
CALL INFGET (FNUM,RECNUM,INREC,IERROR)
C**************************************************C
C******************** ERRORS **********************C
C**************************************************C
IF(IERROR.NE.0) THEN
   CALL MESCHR (ITEM,0)
   CALL MESINT (RECNUM,0)
   CALL INFCLS (FNUM)
   CALL INFORM ('\Unable to get record %2% for Info item %1%' //
    &  ' during INFO_WRITE.',-1)
   RETURN 1
ENDIF
C***************************************************************C
C                       Encoding items                       C
C***************************************************************C
CALL INFENC (ITEMAR4,INFOVAL,STR,8000,INREC,IERROR)
C**************************************************C
C******************** ERRORS **********************C
C**************************************************C
IF(IERROR.LT.0) THEN
   CALL MESCHR (ITEM,0)
   CALL MESINT (RECNUM,0)
   CALL INFORM  
    & ('\Encoding for record %2%, Illegal item type for Info' //
    &  ' item %1% (INFO_WRITE).',-1)
   CALL INFCLS (FNUM)
   CALL INFORM 
    & ('\Abnormal Termination of Info_Utility_Subroutines',-1)
   RETURN 1
ENDIF
C**********************************************************************C
100 CONTINUE
RETURN
END
C**********************************************************************C
C<><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><>C
SUBROUTINE INFO_WRITES (FNUM,NUMREC,INFOITEM,VAL,E)
C-----------------------------------------------------------------------
C
C---VERSION 2.0 INFO_WRITES 18JUNE1990
C
C:::----------------------------------------------:
C
END
Purpose:

Writes the first item values into an open Arc/Info data file.

Language: FORTRAN, with ARC/INFO subroutine call version 5.0

Inputs:

FNUM : FILE TABLE NUMBER (OPEN)
ITEM : NAME OF ITEM
NUMREC : NUMBER OF RECORDS IN DATA FILE
VAL : INTEGER ARRAY VALUES (NUMREC)

Outputs:

History:

Leonard L. Orzol 06/18/90 Modified Coding.
USGS WRD Portland, Or
Fts: 429-2256

---

REAL*8 INFOVAL
REAL VAL(NUMREC)
CHARACTER*(*) INFOITEM
CHARACTER*32 STR
CHARACTER*16 ITEM
INTEGER FNUM,IERROR,INREC(2000),ITEMAR4(4),RECNUM

ITEM=INFOITEM

CALL INFEXI (FNUM,ITEM,ITEMAR4,IERROR)

IF(IERROR.LT.1) THEN
    CALL MESCHR (ITEM,0)
    CALL INFORM ('\Info item %1% does not exist ' //
        'Unable to write Info item %1% (INFO_WRITES).',-1)
    CALL INFCLS (FNUM)
    CALL INFORM
        ('\Abnormal Termination of Info_Utility_Subroutines',-1)
    RETURN 1
ENDIF

CALL INFENC (ITEMAR4,INFOVAL,STR,8000,INREC,IERROR)

IF(IERROR.LT.0) THEN
    CALL MESCHR (ITEM,0)
    CALL MESINT (RECNUM,0)
    CALL INFORM
        ('\Encoding for record %2%, Illegal item type for Info' //
        ' item %1% (INFO_WRITES).',-1)
    CALL INFCLS (FNUM)
    CALL INFORM
ENDIF

DO 100 RECNUM=1,NUMREC
    INFOVAL=VAL(RECNUM)

    CALL INFENC (ITEMAR4,INFOVAL,STR,8000,INREC,IERROR)

    IF(IERROR.LT.0) THEN
        CALL MESCHR (ITEM,0)
        CALL MESINT (RECNUM,0)
        CALL INFORM
            ('\Encoding for record %2%, Illegal item type for Info' //
            ' item %1% (INFO_WRITES).',-1)
        CALL INFCLS (FNUM)
        CALL INFORM
    ENDIF

100 CONTINUE
& ('\Abnormal Termination of Info_Utility_Subroutines',-1)
RETURN 1
ENDIF
C******************************************************************************C
C               Write records to INFO file                                    C
C******************************************************************************C
CALL INFPUT (FNUM,RECNUM,INREC,IERROR)
C**************************************************C
C******************** ERRORS **********************C
C**************************************************C
IF(IERROR.NE.0) THEN
CALL MESCHR (ITEM,0)
CALL MESINT (RECNUM,0)
CALL INFORM
& ('\Unable to write record %2% for Info item %1% ' //
& '(INFO_WRITES).',-1)
CALL INFCLS (FNUM)
CALL INFORM
& ('\Abnormal Termination of Info_Utility_Subroutines',-1)
RETURN 1
ENDIF
C******************************************************************************C
100 CONTINUE
RETURN
END
C******************************************************************************C
C<><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><>C
C                  ENDING OF ARC/INFO SESSION                                C
C<><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><>C
C******************************************************************************C
SUBROUTINE INFO_ENDING
C******************************************************************************C
1000 CALL INFEND
RETURN
END
C<><><><><><><><><><><> SUBROUTINE INFO_NAMING <><><><><><><><><><><>C
C                                                                      C
C         PROGRAM WRITTEN BY LLORZOL                                   C
C                              USGS PORTLAND, OR                       C
C                              FTS 429-2025 OR 429-2014                C
C                              DECEMBER 1988                           C
C                                                                      C
C<><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><>C
C----------------------------------------------------------------------C
C*************************  PROGRAM VARIABLES  ************************C
C----------------------------------------------------------------------C
C**********************************************************************C
C-------------------------  CHARACTER VARIABLES  ----------------------C
C**********************************************************************C
C FORM         Character*80 Format for variable                        C
C NAME         Character*80 Dummy character holding name               C
C**********************************************************************C
C--------------------------  INTEGER VARIABLES  -----------------------C
C**********************************************************************C
C KPER         Integer Dummy value holding stress period               C
C KSTP         Integer Dummy value holding step within stress period   C
C ILAY         Integer Dummy value holding layer number                C
C MULT         Integer Dummy value                                     C
C NCHAR        Integer Dummy value length of name                      C
C**********************************************************************C
SUBROUTINE INFO_NAMING (NAME,KPER,KSTP,ILAY,NCHAR)
CHARACTER*(*) NAME
CHARACTER*80 FRMT
CHARACTER*3 PERIOD,STEP
CHARACTER*2 LAYER
CHARACTER*1 UNDER
INTEGER KPER,KSTP,ILAY,NCHAR,STRDEX
DATA UNDER /'_'/,FRMT /''/
INTRINSIC INDEX
EXTERNAL FORMATING,POWER
C**********************************************************************C
IF(ILAY.GT.0) THEN
STRDEX=INDEX(NAME,’ ’)-1
NAME=NAME (1:STRDEX)//UNDER
STRDEX=STRDEX+1
NAME=NAME (1:STRDEX)
CALL POWER (ILAY,MULT)
CALL FORMATING (MULT,FRMT)
WRITE(LAYER,FRMT) ILAY
NAME=NAME(1:STRDEX)//LAYER (1:MULT)
NAME=NAME (1:STRDEX+MULT)
ENDIF

C*******************************************************************************C

IF(KPER.GT.0) THEN
    STRDEX=INDEX(NAME,’ ’)-1
    NAME=NAME (1:STRDEX)//UNDER
    STRDEX=STRDEX+1
    NAME=NAME (1:STRDEX)
    CALL POWER (KPER,MULT)
    CALL FORMATING (MULT,FRMT)
    WRITE(PERIOD,FRMT) KPER
    NAME=NAME (1:STRDEX)//PERIOD (1:MULT)
    NAME=NAME (1:STRDEX+MULT)
ENDIF

C*******************************************************************************C

IF(KSTP.GT.0) THEN
    STRDEX=INDEX(NAME,’ ’)-1
    NAME=NAME (1:STRDEX)//UNDER
    STRDEX=STRDEX+1
    NAME=NAME (1:STRDEX)
    CALL POWER (KSTP,MULT)
    CALL FORMATING (MULT,FRMT)
    WRITE(STEP,FRMT) KSTP
    NAME=NAME (1:STRDEX)//STEP (1:MULT)
    NAME=NAME (1:STRDEX+MULT)
ENDIF

C*******************************************************************************C

RETURN
END

C<><><><><><><><><><><><><> POWER SUBROUTINE <><><><><><><><><><><><><>C

C PROGRAM WRITTEN BY LLORZOL
C USGS PORTLAND, OR
C FTS 429-2025 OR 429-2014
C DECEMBER 1988
C
C<><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><>C

C----------------------------------------------------------------------C

C*************************  PROGRAM VARIABLES  ************************C

C----------------------------------------------------------------------C

C--------------------------  INTEGER VARIABLES  -----------------------C

SUBROUTINE POWER (NUMBER,MULT)

ONE=1.0

DO 10 I=1,20
    MULT=I
    DIVIDORS=FLOAT(NUMBER)/10.0**I
    IF(DIVIDORS.LT.ONE) GO TO 100
10 CONTINUE

RETURN
END

C<><><><><><><><><><><><><> SUBROUTINE FORMATING <><><><><><><><><><><><><>C

C PROGRAM WRITTEN BY LLORZOL
C USGS PORTLAND, OR
C FTS 429-2025 OR 429-2014
C DECEMBER 1988
C
C**********************************************************************C
C--------------------------  INTEGER VARIABLES  -----------------------C
C**********************************************************************C
C MULT        Integer Dummy value                                     C
C**********************************************************************C
C-------------------------  CHARACTER VARIABLES  ----------------------C
C**********************************************************************C
C FORM         Character*80 Format for variable                        C
C NUMBER       Character*1 Dummy variable integer length of MULT       C
C**********************************************************************C
SUBROUTINE FORMATING (MULT,FORM)
    CHARACTER*80 FORM
    CHARACTER*1 NUMBER

    FORM=''
    NUMBER=''

    WRITE(NUMBER,'(I1)') MULT
    FORM='(I'//NUMBER//')'

   100 RETURN
END

Variables that are passed values to the INFO Utility Subroutines

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCESS</td>
<td>Program</td>
<td>Flag indicating whether access to the ARC/INFO file is read or write.</td>
</tr>
<tr>
<td>FNUM</td>
<td>Program</td>
<td>Integer Info variable holding ISP channel number.</td>
</tr>
<tr>
<td>INFOPATH</td>
<td>Program</td>
<td>Character*128 Full treename to ARC/INFO files.</td>
</tr>
<tr>
<td>IOUTVAL</td>
<td>Program</td>
<td>Integer Dummy array (nx1) holding integer values.</td>
</tr>
<tr>
<td>ITEM</td>
<td>Program</td>
<td>Character*16 An Dummy variable holding item name</td>
</tr>
<tr>
<td>ITEMS</td>
<td>Program</td>
<td>Character*16 An Dummy array (nx1) holding items’ names.</td>
</tr>
<tr>
<td>NCOLUMNS</td>
<td>Program</td>
<td>Integer holding the number of model columns.</td>
</tr>
<tr>
<td>NROWS</td>
<td>Program</td>
<td>Integer holding the number of model rows.</td>
</tr>
<tr>
<td>NUMREC</td>
<td>Program</td>
<td>Integer value for the number of records within the file specified by</td>
</tr>
<tr>
<td>NUMRECORDS</td>
<td>Program</td>
<td>INFOPATH.</td>
</tr>
<tr>
<td>VAL</td>
<td>Program</td>
<td>Real*8 Dummy array (nx1) holding integer values.</td>
</tr>
</tbody>
</table>

Variables for Info Utility Subroutines

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALTERN</td>
<td>Module</td>
<td>Character*16 Info variable holding alternate item name.</td>
</tr>
<tr>
<td>AFTER</td>
<td>Module</td>
<td>Character*16 Info item name where to add new item after.</td>
</tr>
<tr>
<td>COPY</td>
<td>Module</td>
<td>Integer Info variable switches on item file procedure.</td>
</tr>
<tr>
<td>DECIML</td>
<td>Module</td>
<td>Integer Info variable holding number of decimal places.</td>
</tr>
<tr>
<td>DIRECT</td>
<td>Module</td>
<td>Character*128 Path to ARC/INFO files.</td>
</tr>
<tr>
<td>INFONAME</td>
<td>Module</td>
<td>Character*32 ARC/INFO internal filenames.</td>
</tr>
<tr>
<td>INFOVAL</td>
<td>Module</td>
<td>Real*8 values for Info variables (real,integer:date).</td>
</tr>
<tr>
<td>EXISTS</td>
<td>Module</td>
<td>Integer Info variable holding error value.</td>
</tr>
<tr>
<td>IERR</td>
<td>Module</td>
<td>Integer Info variable holding error value.</td>
</tr>
</tbody>
</table>
Variables for Info Utility Subroutines

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>INDEX</td>
<td>Module</td>
<td>Integer Info variable holding index number of items.</td>
</tr>
<tr>
<td>INREC</td>
<td>Module</td>
<td>Integer Info array (2000x1) byte-encoded record.</td>
</tr>
<tr>
<td>IRECORD</td>
<td>Module</td>
<td>Integer Info variable holding subscript number of VAL.</td>
</tr>
<tr>
<td>ITEMAR4</td>
<td>Module</td>
<td>Integer Info array (4x1) holding Info item formats.</td>
</tr>
<tr>
<td>KEYLEV</td>
<td>Module</td>
<td>Integer Info variable holding key level of items.</td>
</tr>
<tr>
<td>KEYTYP</td>
<td>Module</td>
<td>Integer Info variable holding key type of items.</td>
</tr>
<tr>
<td>NITEMS</td>
<td>Module</td>
<td>Integer Info variable holding number of items.</td>
</tr>
<tr>
<td>NITEM</td>
<td>Module</td>
<td>Integer Info holding the number of these items.</td>
</tr>
<tr>
<td>NUMREC</td>
<td>Module</td>
<td>Integer Info holding record number of ARC/INFO files.</td>
</tr>
<tr>
<td>OCCUR</td>
<td>Module</td>
<td>Integer Info variable holding occurrence count of item.</td>
</tr>
<tr>
<td>OUTPUT</td>
<td>Module</td>
<td>Integer Info variable holding output width of item.</td>
</tr>
<tr>
<td>POSIT</td>
<td>Module</td>
<td>Integer Info variable starting column redefined items.</td>
</tr>
<tr>
<td>PROTCT</td>
<td>Module</td>
<td>Integer Info variable holding protection level of item.</td>
</tr>
<tr>
<td>READON</td>
<td>Module</td>
<td>Integer Info variable holds read access level of item.</td>
</tr>
<tr>
<td>RECLEN</td>
<td>Module</td>
<td>Integer Info variable record length of ARC/INFO files.</td>
</tr>
<tr>
<td>REDEF</td>
<td>Module</td>
<td>Integer Info switch for normal(0) or redefine(1) item.</td>
</tr>
<tr>
<td>RECNUM</td>
<td>Module</td>
<td>Integer value for the number of the record within the file specified by INFOPATH.</td>
</tr>
<tr>
<td>ROWCOLUMNS</td>
<td>Module</td>
<td>Real*8 Info variable holding combined rowcolumn value.</td>
</tr>
<tr>
<td>STR</td>
<td>Module</td>
<td>Character*32 Info variables (character).</td>
</tr>
<tr>
<td>USER</td>
<td>Module</td>
<td>Character*3 Info user name.</td>
</tr>
<tr>
<td>TYPE</td>
<td>Module</td>
<td>Integer Info variable holds item type (integer,...).</td>
</tr>
<tr>
<td>WIDTH</td>
<td>Module</td>
<td>Integer Info variable holding input width of item.</td>
</tr>
</tbody>
</table>

File Utility Module

The File utility module is a collection of routines that is used repeatedly to open, close or delete either ASCII or unformatted files. MODFLOWARC uses this module to open, close, or delete files that must be used during a model simulation. Each of the File routines is described below along with a list of routines and modules, and documentation for program code, and a list of variables used in the routines.

File utility module routines

OPEN_FILE: Opens ASCII or unformatted files and checks the existence of the file using the path supplied by the user.

CLOSE_FILE: Closes ASCII or unformatted files and checks the existence of the file using the path supplied by the user.

DELETE_FILE: Deletes ASCII or unformatted files and checks the existence of the file using the path supplied by the user.

<table>
<thead>
<tr>
<th>Modules</th>
<th>Routines</th>
</tr>
</thead>
<tbody>
<tr>
<td>MODFIL</td>
<td>OPEN_FILE</td>
</tr>
<tr>
<td>MODFLOWARC</td>
<td>CLOSE_FILE</td>
</tr>
<tr>
<td>MODFIL</td>
<td>DELETE_FILE</td>
</tr>
</tbody>
</table>
Purpose: Although this program has been used by the U.S. Geological Survey, no warranty, expressed or implied, is made by the USGS as to the accuracy and functioning of the program and related program material nor shall the fact of distribution constitute any such warranty, and no responsibility is assumed by the USGS in connection therewith.

Purpose: A set of subroutines that deal with files input.

Subroutines:
- OPEN_FILE: Opens files or checks file status for either direct or sequential access, formatted or unformatted, and old or new.
- CLOSE_FILE: Closes files.
- DELETE_FILE: Closes and deletes files.

Secondary Subroutines:
- NONE

Language: FORTRAN, with ARC/INFO subroutine call version 5.01

History:
USGS WRD Portland, Or
Fts: 429-2256

Purpose:
This routine either opens, creates, or inquires files. The user can...
specified the function of open_file as described below.

.. Language: Fortran77

Inputs:

IU = FORTRAN UNIT NUMBER OF FILE
< 0 => UNFORMATTED FILE
0 => RETURN FORMAT STATUS (WHEN NSTAT=0 ONLY)
> 0 => FORMATTED FILE

FNAME = FILE NAME

NSTAT = STATUS OF FILE
DIRECT ACCESS ONLY
-4 => UNDETERMINED STATUS (MAY OR MAY NOT EXIST)
IF IT DOES NOT EXIST, IT IS CREATED BY OPEN STATEMENT
IF IT DOES EXIST, IT IS OPENED AS 'OLD' FILE
-3 => SCRATCH FILE (DELETED AUTOMATICALLY WHEN RUN ENDS)
-2 => NEW FILE
-1 => OLD FILE

FILE STATUS
0 => RETURN FILE STATUS FOR OPEN, FORMAT, FILE ACCESS
SEQUENTIAL ACCESS ONLY
1 => OLD FILE
2 => NEW FILE
3 => SCRATCH FILE (DELETED AUTOMATICALLY WHEN RUN ENDS)
4 => UNDETERMINED STATUS (MAY OR MAY NOT EXIST)
IF IT DOES NOT EXIST, IT IS CREATED BY OPEN STATEMENT
IF IT DOES EXIST, IT IS OPENED AS 'OLD' FILE

ITALK = FLAG INDICATING IF THERE IS INTERACTIVE DIALOGUE AT TERMINAL
0 => THERE IS INTERACTIVE DIALOGUE
1 => THERE IS NOT INTERACTIVE DIALOGUE (BATCH MODE)

Outputs: WHEN INPUT
NSTAT => 0
IU => 0

IU = FORMAT STATUS
 99 => FORMATTED FILE
 0 => UNDETERMINED STATUS
-99 => UNFORMATTED FILE
-999 => FILES DOES NOT EXIST

NSTAT = ACCESS STATUS OF FILE
 99 => SEQUENTIAL ACCESS
 0 => UNDETERMINED STATUS
-99 => DIRECT ACCESS
-999 => FILE DOES NOT EXIST

ITALK = OPEN STATUS OF FILE
 99 => FILE IS OPEN
-99 => FILE IS CLOSED
-999 => FILE DOES NOT EXIST

.. History:

Dave Pollack? ??/??/?? Original Concept.
U.S. Geological Survey, Reston, Virginia

Leonard L. Orzol 03/13/90 Original Coding.
12/18/90 Modified for Prime,Suns or DisII

-------------------------------------------------------------------------

CHARACTER*(*) FNAME
CHARACTER*80 ACCSS,DRECT,FMT,SEQUENT,UNFMT
INTEGER NSTAT,IU,ITALK,IERR
LOGICAL*4 EXISTS, OPN

************************************************************************

C----Inquire on file

2 INQUIRE ( 
  F   FILE=FNAME,
  I   IOSTAT=IERR,
  L   ERR=7777,
  E   EXIST=EXISTS,
  S   OPENED=OPN,
  T   ACCESS=ACCSS,
  A   SEQUENTIAL=SEQUENT,
  T   DIRECT=DIRECT,
  U   FORMATTED=FMT,
  S   UNFORMATTED=UNFMT
)

C0-----Return file parameters

C    IF(NSTAT.EQ.0) THEN

C0A-----File existence status

C    IF(EXISTS) THEN
    NSTAT=0
    IU=0
    ITALK=0
    ELSE
    NSTAT=-999
    IU=-999
    ITALK=-999
    RETURN
    ENDIF

C0B-----File open status

C    IF(OPN) THEN
    ITALK=99
    ELSE
    ITALK=-99
    ENDIF

C0C-----File access status

C    IF(SEQUENT.EQ.’YES’) THEN
    NSTAT=99
    ELSE IF(SEQUENT.EQ.’NO’) THEN
      IF(DRECT.EQ.’YES’) THEN
        NSTAT=-99
      ELSE
        NSTAT=0
      ENDIF
    ELSE
      NSTAT=0
    ENDIF
    ELSE
      IF(DRECT.EQ.’YES’) THEN
        NSTAT=-99
      ELSE
        NSTAT=0
      ENDIF
    ENDIF

C0C-----File record format status

C    IF(FMT.EQ.’YES’) THEN
      IU=99
    ELSE IF(FMT.EQ.’NO’) THEN
        IU=-99
      ELSE
        IU=0
      ENDIF
    ENDIF
IF(UNFMT.EQ.'YES') THEN
    IU=-99
ELSE
    IU=0
ENDIF
ENDIF

C0D-----Return file parameters
C
RETURN
C
C1----OPEN AN EXISTING FILE
C
ELSE IF(NSTAT.EQ.1 .OR. NSTAT.EQ.-1) THEN
    C1A-----File open status
    IF(OPN) THEN
        GO TO 6666
    C1B-----File exist status
    ELSE IF(.NOT.EXISTS) THEN
        IF(ITALK.GT.0) THEN
            PRINT *,'Does Not Exists ',FNAME (1:INDEX(FNAME,' ')-1)
            PRINT *,'Enter The Name Of An Existing File (<Cr>=Quit):'
            READ (*,'(A)',ERR=104) FNAME
            IF(FNAME.EQ.' ') GO TO 9999
            GO TO 2
        ELSE
            PRINT *,'Does Not Exists ',FNAME (1:INDEX(FNAME,' ')-1)
            GO TO 9999
        ENDIF
        ENDIF
        ENDIF
    ENDIF
C1C-----File access status
C
C1CA-----Sequential access
    IF(SEQUENT.EQ.'YES') THEN
        ACCSS='SEQUENTIAL'
        IF(NSTAT.LT.0) THEN
            PRINT *,'File is direct access not sequential'
            NSTAT=-99
            GO TO 9999
        ENDIF
    ELSE
        IF(DRECT.EQ.'YES') THEN
            ACCSS='DIRECT'
            IF(NSTAT.GT.0) THEN
                PRINT *,'File is sequential access not direct'
                NSTAT=99
                GO TO 9999
            ENDIF
            ENDIF
        ENDIF
    ENDIF
C1CB-----Direct access
C
ELSE
    IF(DIRECT.EQ.'YES') THEN
        ACCSS='DIRECT'
        IF(NSTAT.GT.0) THEN
            PRINT *,'File is sequential access not direct'
            NSTAT=99
            GO TO 9999
        ENDIF
        ENDIF
        ENDIF
    ENDIF
C1D-----File format status
C
IF(IU.LT.0) THEN
    IF(FMT.EQ.'YES') THEN
        PRINT *,'File is formatted not unformatted'
        IU=99
        GO TO 9999
        ENDIF
        FMT='UNFORMATTED'
IU=-IU
ELSE
   IF (UNFMT.EQ.'YES') THEN
      PRINT *, 'File is unformatted not formatted'
      IU=-99
      GO TO 9999
   ENDIF
   FMT='FORMATTED'
ENDIF

C1E-----Open file
C
OPEN (F, IU, L, E, O, P, N)
RETURN

C2-----Open An New File
C
ELSE IF (NSTAT.EQ.2 .OR. NSTAT.EQ.-2) THEN

C2A-----File exists already
C
   IF (EXISTS) THEN
      IF (ITALK.GT.0) THEN
         PRINT *, 'Already Exists ', FNAME (1:INDEX(FNAME,' ')-1)
         PRINT *, 'Enter The Name Of An New File (<Cr>=Quit):'
         READ (*, '(A)', ERR=202) FNAME
         IF (FNAME.EQ. ' ') GO TO 9999
         GO TO 2
      ELSE
         PRINT *, 'Already Exists ', FNAME (1:INDEX(FNAME,' ')-1)
         GO TO 9999
      ENDIF
   C
   C2B-----File access status
C
   ELSE

   C2BA-----Sequential access
C
      IF (NSTAT.GT.0) THEN
         ACCSS='SEQUENTIAL'
      C
      C2BB-----Direct access
C
         ELSE
         ACCSS='DIRECT'
      ENDF
C
   C2C-----File format status
C
      IF (IU.LT.0) THEN
         FMT='UNFORMATTED'
         IU=-IU
      ELSE
         FMT='FORMATTED'
      ENDF
C
   C2D-----Open file
C
OPEN (F, IU, L, E, O, P, N)
E          ACCESS=ACCSS,
O          FORM=FMT,
P          IOSTAT=IERR,
E          ERR=8888
N         )
RETURN
C
C3-----Open An Scratch File
C
ELSE IF(NSTAT.EQ.3 .OR. NSTAT.EQ.-3) THEN
C
C3A-----File exists already
C
IF(EXISTS) THEN
  IF(ITALK.GT.0) THEN
    302      PRINT *, 'Already Exists ',FNAME (1:INDEX(FNAME,' ')-1)
    PRINT *, 'Enter The Name Of An Scratch File (<Cr>=Quit):'
    READ (*,'(A)',ERR=302) FNAME
    IF(FNAME.EQ.' ') GO TO 9999
    GO TO 2
  ENDIF
C
C3B-----File access status
C
ELSE
C
C3BA-----Sequential access
C
  IF(NSTAT.GT.0) THEN
    ACCSS='SEQUENTIAL'
  ELSE
    ACCSS='DIRECT'
  ENDIF
C
C3C-----File format status
C
  IF(IU.LT.0) THEN
    FMT='UNFORMATTED'
    IU=-IU
  ELSE
    FMT='FORMATTED'
  ENDIF
ENDIF
C
C3D-----Open file
C
OPEN (
  F          IU,
  I          FILE=FNAME,
  L          STATUS='SCRATCH',
  E          ACCESS=ACCSS,
  O          FORM=FMT,
  P          IOSTAT=IERR,
  E          ERR=8888
N         )
RETURN
C
C4-----Open An Unknown Status File
C
ELSE IF(NSTAT.EQ.4 .OR. NSTAT.EQ.-4) THEN
C
C4A-----File exists
C
IF(EXISTS) THEN
C
C4AA-----Already open
C
IF(OPN) THEN
  ITALK=99
C C4AAA-----File sequential access
C IF(NSTAT.GT.0) THEN
NSTAT=99
IF(ACCSS.EQ.'DIRECT') THEN
PRINT *, 'File is direct access not sequential'
NSTAT=-99
ENDIF
C C4AAB-----File direct access
C ELSE
NSTAT=-99
IF(ACCSS.EQ.'SEQUENTIAL') THEN
NSTAT=99
PRINT *, 'File is sequential access not direct'
ENDIF
C C4AAC-----File format status
C IF(IU.LT.0) THEN
IU=-99
IF(FMT.EQ.'YES') THEN
PRINT *, 'File is formatted not unformatted'
IU=99
ENDIF
ELSE
IU=99
IF(FMT.EQ.'UNFORMATTED') THEN
PRINT *, 'File is unformatted not formatted'
IU=-99
ENDIF
ENDIF
GO TO 6666
C C4AB-----Unopen
C ELSE
C C4ABA-----File sequential access
C IF(NSTAT.GT.0) THEN
IF(ACCSS.EQ.'DIRECT') THEN
PRINT *, 'File is direct access not sequential'
GO TO 9999
ENDIF
C C4ABB-----Direct access
C ELSE
IF(ACCSS.EQ.'SEQUENTIAL') THEN
PRINT *, 'File is sequential access not direct'
GO TO 9999
ENDIF
C C4ABC-----File format status
C IF(IU.LT.0) THEN
IF(FMT.EQ.'FORMATTED') THEN
PRINT *, 'File is formatted not unformatted'
GO TO 9999
ENDIF
IU=-IU
ELSE
IF(FMT.EQ.'UNFORMATTED') THEN
PRINT *, 'File is unformatted not formatted'
GO TO 9999
ENDIF
ENDIF
C4ABD-----Open file
C
OPEN (IU, FILE=FNAME, STATUS='OLD',
      & ACCESS=ACCSS, FORM=FMT, IOSTAT=IERR, ERR=8888)
RETURN
ENDIF
C4B-----File exist statuuss
C
ELSE
  GO TO 2
ENDIF
C
C5-----Unknow Option
C
ELSE
  PRINT *, 'Poor choice for Type of File'
  GO TO 9999
END IF
C**********************************************************************C
C                       Report errors
C**********************************************************************C
6666 PRINT *, ' *********************__ERROR__*************************
PRINT *, ' Already open for file ',
& FNAME (1:INDEX(FNAME,' ')-1), ' for Unit', IU
PRINT *, ' IOSTAT=', IERR
PRINT *, ' Abnormal Termination of Open_File_Subroutine'
RETURN 1
C
7777 PRINT *, ' *********************__ERROR__*************************
PRINT *, ' Can not inquire on file ',
& FNAME (1:INDEX(FNAME,' ')-1), ' for Unit', IU
PRINT *, ' IOSTAT=', IERR
PRINT *, ' Abnormal Termination of Open_File_Subroutine'
RETURN 1
C
8888 PRINT *, ' *********************__ERROR__*************************
PRINT *, ' Can not open file ',
& FNAME (1:INDEX(FNAME,' ')-1), ' on Unit', IU
PRINT *, ' IOSTAT=', IERR
PRINT *, ' Abnormal Termination of Open_File_Subroutine'
RETURN 1
C
9999 Print *, ' Abnormal Termination of Open_File_Subroutine'
RETURN 1
C**********************************************************************C
1000 RETURN
END
C
SUBROUTINE DELETE_FILE (FNAME,*)
C-----------------------------------------------------------------------
C
C---VERSION 2.0 29MARCH1990
C
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
C .. Purpose:
C
C  THIS ROUTINE DELETES A SINGLE FILE.
C
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
C .. Language: Fortran77
C
C
C Inputs:
C
FNAME = File Name
CHARACTER(*) FNAME
INTEGER IERR
INTEGER IU
LOGICAL*4 EXISTS, OPN
C************************************************************************C
C1-----Inquire About Existing File
C
INQUIRE (FILE=FNAME, IOSTAT=IERR, ERR=7777,
& EXIST=EXISTS, OPENED=OPN, NUMBER=IU)
C1-----Existence of File
C
IF (.NOT. EXISTS) THEN
  PRINT *, FNAME (1:INDEX(FNAME, ' ') - 1), ' File exist status'
  RETURN
ELSE
  IF (OPENED) THEN
    CLOSE (IU, STATUS='DELETE', ERR=8888, IOSTAT=IERR)
  ELSE
    IU=10
  ENDIF
  INQUIRE (UNIT=IU, IOSTAT=IERR, ERR=7777,
& EXIST=EXISTS, OPENED=OPN)
  IF (OPENED) THEN
    IU=IU+1
  GO TO 2
ENDIF
  OPEN (IU, FILE=FNAME, ERR=9999, IOSTAT=IERR)
  CLOSE (IU, STATUS='DELETE', ERR=8888, IOSTAT=IERR)
ENDIF
RETURN
C**********************************************************************C
C                       Report errors
C**********************************************************************C
7777 PRINT *, ' *********************__ERROR__*************************' PRINT *, ' Can not inquire on file ', & FNAME (1:INDEX(FNAME, ' ') - 1), ' for Unit', IU
PRINT *, ' IOSTAT=', IERR
PRINT *, ' Abnormal Termination of Delete_File_Subroutine'
RETURN 1
C
8888 PRINT *, ' *********************__ERROR__*************************' PRINT *, ' Can not close file ', & FNAME (1:INDEX(FNAME, ' ') - 1), ' on Unit', IU
PRINT *, ' IOSTAT=', IERR
PRINT *, ' Abnormal Termination of Delete_File_Subroutine'
RETURN 1
C
9999 PRINT *, ' *********************__ERROR__*************************' PRINT *, ' Can not open file ', & FNAME (1:INDEX(FNAME, ' ') - 1), ' on Unit', IU
PRINT *, ' IOSTAT=', IERR
PRINT *, ' Abnormal Termination of Delete_File_Subroutine'
RETURN 1
C**********************************************************************C
1000 RETURN
END
C
**********************************************************************C
SUBROUTINE CLOSE_FILE (FNAME,*)
**********************************************************************C
C---VERSION 2.0 12MARCH1990
C
THIS ROUTINE CLOSES A SINGLE FILE.

Language: Fortran77

Inputs:
- FNAME = FILE NAME

History:
- Leonard L. Orzol 03/13/90 Original Coding
USGS WRD Portland, Or Fts: 429-2256

CHARACTER*80 FNAME
INTEGER IERR
LOGICAL EXISTS,OPN

--- Inquire About Existing File
INQUIRE (FILE=FNAME, IOSTAT=IERR, ERR=9999,
& EXIST=EXISTS, OPENED=OPN, NUMBER=IU)

--- Existence of File
IF(.NOT.EXISTS) THEN
  PRINT *,'System File ',FNAME (1:INDEX(FNAME,' ')-1),', does not exist'
  RETURN
ELSE
  IF(OPN) THEN
    CLOSE (IU, STATUS='KEEP', ERR=8888, IOSTAT=IERR)
  ENDIF
ENDIF
RETURN
END
Variables that are passed values to the File Utility Subroutines

<table>
<thead>
<tr>
<th>Variable</th>
<th>Range</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>FNAME</td>
<td>Program</td>
<td>Character*(*) Full treename to file.</td>
</tr>
<tr>
<td>ITALK</td>
<td>Program</td>
<td>Integer variable holding interactive dialog flag.</td>
</tr>
<tr>
<td>IU</td>
<td>Program</td>
<td>Integer variable holding unit number.</td>
</tr>
<tr>
<td>NSTAT</td>
<td>Program</td>
<td>Integer variable holding status of file.</td>
</tr>
</tbody>
</table>
REFERENCES CITED


APPENDIX A

Sample Problem

The sample problem that follows is from the manual of the McDonald and Harbaugh (1988) three-dimensional ground-water flow model. The description of this sample problem is not repeated here. The user is referred to the sample problem in the ground-water flow model manual. All input values of the data arrays are unchanged, but the storage location of the input and output arrays values is different. Output control for this sample problem was reset for testing purposes and directs output arrays to be recorded in ARC/INFO files for heads, drawdown, and cell-by-cell flow terms.

Opening Data and Control Files for Sample Problem

The user builds an ASCII file containing the unit numbers and filenames that must be opened for a model simulation. Each line consists of a unit number followed by the associated filename in free format with the filenames surrounded by single quotes such as ‘modflow.list’. The first entry in this file must be the unit number and filename of the input control record for the Basic (BAS) package. The last entry in this file must be the unit number and filename of the file where all printer output is directed. If input data are stored in or output data are record to unformatted files, the user enters the unit number as negative. An example for the sample problem is listed below.

7 'bas.arc'
8 'bcf.arc'
9 'wel.arc'
10 'drn.arc'
11 'riv.arc'
15 'rch.arc'
16 'sip.arc'
19 'out.arc'
20 'str.arc'
99 'modflow.list'

Input Data and Control for Sample Problem

The input files for the sample problem used to run the model uses both input from ASCII and ARC/INFO files, several of which are listed below.

**ARC/INFO Directory containing input array files**

<table>
<thead>
<tr>
<th>TYPE</th>
<th>NAME</th>
<th>INTERNAL NAME</th>
<th>NO. RECS</th>
<th>LENGTH</th>
<th>EXTERNL</th>
</tr>
</thead>
<tbody>
<tr>
<td>DF</td>
<td>IBOUND</td>
<td>ARCO00DAT</td>
<td>225</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>DF</td>
<td>DRAINS</td>
<td>ARCO01DAT</td>
<td>9</td>
<td>14</td>
<td></td>
</tr>
<tr>
<td>DF</td>
<td>WELLS</td>
<td>ARCO02DAT</td>
<td>15</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>DF</td>
<td>TRANS</td>
<td>ARCO03DAT</td>
<td>225</td>
<td>16</td>
<td></td>
</tr>
<tr>
<td>DF</td>
<td>HEADS</td>
<td>ARCO04DAT</td>
<td>225</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>DF</td>
<td>HYCOND</td>
<td>ARCO05DAT</td>
<td>225</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>DF</td>
<td>ANISOTROPY</td>
<td>ARCO07DAT</td>
<td>3</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>DF</td>
<td>DELR</td>
<td>ARCO08DAT</td>
<td>15</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>DF</td>
<td>DELC</td>
<td>ARCO09DAT</td>
<td>15</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>DF</td>
<td>VTHYCOND</td>
<td>ARCO10DAT</td>
<td>225</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>DF</td>
<td>BOTTOMS</td>
<td>ARCO13DAT</td>
<td>225</td>
<td>20</td>
<td></td>
</tr>
</tbody>
</table>

For the Basic package, the model input arrays, IBOUND and HEADS, were read from either ARC/INFO or ASCII files. The IBOUND arrays, IBOUND_1 and IBOUND_2, are read from ARC/INFO file GWINF>SAMPLE>INFO!ARC!IBOUND for layers 1 and 2 (fig. 1A). Starting HEADS arrays are read from ARC/INFO file GWINF>SAMPLE>INFO!ARC!HEADS for layer 1 (SHEAD_1). Item names are shown to the right for each input array. Two-dimensional arrays, integer or real, have a layer suffix (such as _1) attached to the root name. The root name for the boundary array is IBOUND and each layer is appended as a suffix to the root. For one-dimensional arrays such as the anisotropy factor, the root name is sufficient
and an example is TRPY (fig. 2A). The root names originate from the text of the ground-water flow manual (McDonald and Harbaugh, 1988). From page 4-9 of the ground-water flow manual and under the topic "for each simulation", a list of arrays for the Basic package shows their names associated with each array and their dimension. The root name has been taken from this section for Basic package and likewise for all other packages within the ground-water flow manual.

<table>
<thead>
<tr>
<th>Package</th>
<th>File type</th>
<th>IBOUND</th>
<th>IBOUND</th>
<th>ITEM</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBOUND LAYER 1</td>
<td>ARC/INFO</td>
<td></td>
<td>IBOUND</td>
<td>IBOUND_1</td>
</tr>
<tr>
<td></td>
<td>ASCII</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ITEM</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IBOUND LAYER 2</td>
<td>ARC/INFO</td>
<td></td>
<td>IBOUND</td>
<td>IBOUND_2</td>
</tr>
<tr>
<td></td>
<td>ASCII</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ITEM</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IBOUND LAYER 3</td>
<td>ARC/INFO</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>ASCII</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ITEM</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HEADS LAYER 1</td>
<td>ARC/INFO</td>
<td>HEADS</td>
<td></td>
<td>SHEAD_1</td>
</tr>
<tr>
<td></td>
<td>ASCII</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ITEM</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HEADS LAYER 2</td>
<td>ARC/INFO</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>ASCII</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ITEM</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HEADS LAYER 3</td>
<td>ARC/INFO</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>ASCII</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ITEM</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 1A.-- Relation between ARC/INFO and ASCII files for input arrays to the BAS package.

For the BCF package, the model input arrays: anisotropy, cell width along rows, cell width along columns, hydraulic conductivity, elevation of aquifer bottom, vertical hydraulic conductivity, and transmissivity, were read from ARC/INFO files as shown in figure 2A. Item names are shown to the right for each input array.
For packages which use modules from the utility package (UTLARC) to read input data such as the Basic package, the input arrays are read by U2DINTARC and U2DRELARC modules. The following figures show examples of the ARC/INFO files listing both the item names within these files as well the array values that will be read in a model run (fig. 3A-11A).
**Description of items**

ARC/INFO file NAME: IBOUND

4 ITEMS: STARTING IN POSITION 1

<table>
<thead>
<tr>
<th>COL</th>
<th>ITEM NAME</th>
<th>WDTH</th>
<th>OPUT</th>
<th>TYP</th>
<th>N.DEC</th>
<th>ALTERNATE NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ROW</td>
<td>2</td>
<td>2</td>
<td>I</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>COLUMN</td>
<td>2</td>
<td>2</td>
<td>I</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>IBOUND_1</td>
<td>2</td>
<td>2</td>
<td>I</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>IBOUND_2</td>
<td>2</td>
<td>2</td>
<td>I</td>
<td>-</td>
<td></td>
</tr>
</tbody>
</table>

**REDEFINED ITEMS**

1 ROWCOLUMN 4 4 I -

**Example of item values**

$RECNO   ROW COLUMN IBOUND_1 IBOUND_2
1 1 1 -1 -1
2 1 2 1 1
3 1 3 1 1
4 1 4 1 1
5 1 5 1 1
6 1 6 1 1
7 1 7 1 1
8 1 8 1 1
9 1 9 1 1
10 1 10 1 1
11 1 11 1 1
12 1 12 1 1
13 1 13 1 1
14 1 14 1 1
15 1 15 1 1
16 2 1 -1 -1

Figure 3A.—Location of the IBOUND arrays for the first two layers within the ARC/INFO datafile, IBOUND.
Description of items

ARC/INFO file NAME: HEADS
4 ITEMS: STARTING IN POSITION 1

<table>
<thead>
<tr>
<th>COL</th>
<th>ITEM NAME</th>
<th>WDTH</th>
<th>OPUT</th>
<th>TYP</th>
<th>N.DEC</th>
<th>ALTERNATE NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ROW</td>
<td>2</td>
<td>2</td>
<td>I</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>COLUMN</td>
<td>2</td>
<td>2</td>
<td>I</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>** REDefined ITEMS **</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>ROWCOLUMN</td>
<td>4</td>
<td>4</td>
<td>I</td>
<td>-</td>
<td></td>
</tr>
</tbody>
</table>

Example of item values

$RECNO ROW COLUMN SHEAD_1
1 1 01 0.000
2 1 02 0.000
3 1 03 0.000
4 1 04 0.000
5 1 05 0.000
6 1 06 0.000
7 1 07 0.000
8 1 08 0.000
9 1 09 0.000
10 1 10 0.000
11 1 11 0.000
12 1 12 0.000
13 1 13 0.000
14 1 14 0.000
15 1 15 0.000
16 2 01 0.000

Figure 4A: Location of the SHEAD arrays for layer one within the ARC/INFO datafile, HEADS.

Description of items

ARC/INFO file NAME: ANISOTROPY
2 ITEMS: STARTING IN POSITION 1

<table>
<thead>
<tr>
<th>COL</th>
<th>ITEM NAME</th>
<th>WDTH</th>
<th>OPUT</th>
<th>TYP</th>
<th>N.DEC</th>
<th>ALTERNATE NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>LAYER</td>
<td>2</td>
<td>2</td>
<td>I</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>TRPY</td>
<td>4</td>
<td>12</td>
<td>F</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

Example of item values

$RECNO LAYER TRPY
1 1 1.000
2 2 1.000
3 3 1.000

Figure 5A: Location of the TRPY array for all three model layers within the ARC/INFO datafile, ANISOTROPY.
### Description of items

**ARC/INFO file NAME: DELR**

<table>
<thead>
<tr>
<th>COL</th>
<th>ITEM NAME</th>
<th>WDTH</th>
<th>OPUT</th>
<th>TYP</th>
<th>N.DEC</th>
<th>ALTERNATE NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>NCOL</td>
<td>2</td>
<td>2</td>
<td>I</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>DELR</td>
<td>4</td>
<td>12</td>
<td>F</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

**Example of item values**

<table>
<thead>
<tr>
<th>$RECNO</th>
<th>NCOL</th>
<th>DELR</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>5,000.000</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>5,000.000</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>5,000.000</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>5,000.000</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>5,000.000</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>5,000.000</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>5,000.000</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>5,000.000</td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>5,000.000</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>5,000.000</td>
</tr>
<tr>
<td>11</td>
<td>11</td>
<td>5,000.000</td>
</tr>
<tr>
<td>12</td>
<td>12</td>
<td>5,000.000</td>
</tr>
<tr>
<td>13</td>
<td>13</td>
<td>5,000.000</td>
</tr>
<tr>
<td>14</td>
<td>14</td>
<td>5,000.000</td>
</tr>
<tr>
<td>15</td>
<td>15</td>
<td>5,000.000</td>
</tr>
</tbody>
</table>

Figure 6A.---Location of the DELR arrays within the ARC/INFO datafile, DELR.

---

### Description of items

**ARC/INFO file NAME: DELC**

<table>
<thead>
<tr>
<th>COL</th>
<th>ITEM NAME</th>
<th>WDTH</th>
<th>OPUT</th>
<th>TYP</th>
<th>N.DEC</th>
<th>ALTERNATE NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>NROW</td>
<td>2</td>
<td>2</td>
<td>I</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>DELC</td>
<td>4</td>
<td>12</td>
<td>F</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

**Example of item values**

<table>
<thead>
<tr>
<th>$RECNO</th>
<th>NROW</th>
<th>DELC</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>5,000.000</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>5,000.000</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>5,000.000</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>5,000.000</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>5,000.000</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>5,000.000</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>5,000.000</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>5,000.000</td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>5,000.000</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>5,000.000</td>
</tr>
<tr>
<td>11</td>
<td>11</td>
<td>5,000.000</td>
</tr>
<tr>
<td>12</td>
<td>12</td>
<td>5,000.000</td>
</tr>
<tr>
<td>13</td>
<td>13</td>
<td>5,000.000</td>
</tr>
<tr>
<td>14</td>
<td>14</td>
<td>5,000.000</td>
</tr>
<tr>
<td>15</td>
<td>15</td>
<td>5,000.000</td>
</tr>
</tbody>
</table>

Figure 7A.---Location of the DELC arrays within the ARC/INFO datafile, DELC.
Description of items

ARC/INFO file NAME: HYCOND

3 ITEMS: STARTING IN POSITION 1

<table>
<thead>
<tr>
<th>COL</th>
<th>ITEM NAME</th>
<th>WIDTH</th>
<th>OUTPUT</th>
<th>TYPE</th>
<th>N. DEC</th>
<th>ALTERNATE NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ROW</td>
<td>2</td>
<td>2</td>
<td>I</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>COLUMN</td>
<td>2</td>
<td>2</td>
<td>I</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>HY_1</td>
<td>4</td>
<td>12</td>
<td>F</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

** REDEFINED ITEMS **

<table>
<thead>
<tr>
<th>COL</th>
<th>ITEM NAME</th>
<th>WIDTH</th>
<th>OUTPUT</th>
<th>TYPE</th>
<th>N. DEC</th>
<th>ALTERNATE NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ROWCOLUMN</td>
<td>4</td>
<td>4</td>
<td>C</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>HY_2</td>
<td>4</td>
<td>12</td>
<td>F</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

Example of item values

$RECNO | ROW COLUMN | HY_1
1  | 1 01 | 0.001
2  | 1 02 | 0.001
3  | 1 03 | 0.001
4  | 1 04 | 0.001
5  | 1 05 | 0.001
6  | 1 06 | 0.001
7  | 1 07 | 0.001
8  | 1 08 | 0.001
9  | 1 09 | 0.001
10 | 1 10 | 0.001
11 | 1 11 | 0.001
12 | 1 12 | 0.001
13 | 1 13 | 0.001
14 | 1 14 | 0.001
15 | 1 15 | 0.001
16 | 2 01 | 0.001
17 | 2 02 | 0.001
18 | 2 03 | 0.001
19 | 2 04 | 0.001
20 | 2 05 | 0.001
21 | 2 06 | 0.001
22 | 2 07 | 0.001

Figure 8A.---Location of the HY arrays for layer one within the ARC/INFO datafile, HYCOND.
Description of items

ARC/INFO file NAME: BOTTOMS
6 ITEMS: STARTING IN POSITION 1

<table>
<thead>
<tr>
<th>COL</th>
<th>ITEM NAME</th>
<th>WIDTH</th>
<th>OUTPUT</th>
<th>TYPE</th>
<th>N.DEC</th>
<th>ALTERNATE NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ROW</td>
<td>2</td>
<td>2</td>
<td>I</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>COLUMN</td>
<td>2</td>
<td>2</td>
<td>I</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>BOT_1</td>
<td>4</td>
<td>12</td>
<td>F</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>BOT_2</td>
<td>4</td>
<td>12</td>
<td>F</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>BOT_3</td>
<td>4</td>
<td>12</td>
<td>F</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

** REDEFINED ITEMS **

<table>
<thead>
<tr>
<th>COL</th>
<th>ITEM NAME</th>
<th>WIDTH</th>
<th>OUTPUT</th>
<th>TYPE</th>
<th>N.DEC</th>
<th>ALTERNATE NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ROWCOLUMN</td>
<td>4</td>
<td>4</td>
<td>I</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>TOP_2</td>
<td>4</td>
<td>12</td>
<td>F</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>TOP_3</td>
<td>4</td>
<td>12</td>
<td>F</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

Example of item values

<table>
<thead>
<tr>
<th>$RECNO$</th>
<th>$ROW$</th>
<th>$COLUMN$</th>
<th>$BOT_1$</th>
<th>$BOT_2$</th>
<th>$BOT_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>01</td>
<td>-150.000</td>
<td>-300.000</td>
<td>-600.000</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>01</td>
<td>-150.000</td>
<td>-300.000</td>
<td>-600.000</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>01</td>
<td>-150.000</td>
<td>-300.000</td>
<td>-600.000</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>01</td>
<td>-150.000</td>
<td>-300.000</td>
<td>-600.000</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>01</td>
<td>-150.000</td>
<td>-300.000</td>
<td>-600.000</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>01</td>
<td>-150.000</td>
<td>-300.000</td>
<td>-600.000</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>01</td>
<td>-150.000</td>
<td>-300.000</td>
<td>-600.000</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>01</td>
<td>-150.000</td>
<td>-300.000</td>
<td>-600.000</td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>01</td>
<td>-150.000</td>
<td>-300.000</td>
<td>-600.000</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>01</td>
<td>-150.000</td>
<td>-300.000</td>
<td>-600.000</td>
</tr>
<tr>
<td>11</td>
<td>11</td>
<td>01</td>
<td>-150.000</td>
<td>-300.000</td>
<td>-600.000</td>
</tr>
<tr>
<td>12</td>
<td>12</td>
<td>01</td>
<td>-150.000</td>
<td>-300.000</td>
<td>-600.000</td>
</tr>
<tr>
<td>13</td>
<td>13</td>
<td>01</td>
<td>-150.000</td>
<td>-300.000</td>
<td>-600.000</td>
</tr>
<tr>
<td>14</td>
<td>14</td>
<td>01</td>
<td>-150.000</td>
<td>-300.000</td>
<td>-600.000</td>
</tr>
<tr>
<td>15</td>
<td>15</td>
<td>01</td>
<td>-150.000</td>
<td>-300.000</td>
<td>-600.000</td>
</tr>
<tr>
<td>16</td>
<td>1</td>
<td>02</td>
<td>-150.000</td>
<td>-300.000</td>
<td>-600.000</td>
</tr>
<tr>
<td>17</td>
<td>2</td>
<td>02</td>
<td>-150.000</td>
<td>-300.000</td>
<td>-600.000</td>
</tr>
<tr>
<td>18</td>
<td>3</td>
<td>02</td>
<td>-150.000</td>
<td>-300.000</td>
<td>-600.000</td>
</tr>
<tr>
<td>19</td>
<td>4</td>
<td>02</td>
<td>-150.000</td>
<td>-300.000</td>
<td>-600.000</td>
</tr>
<tr>
<td>20</td>
<td>5</td>
<td>02</td>
<td>-150.000</td>
<td>-300.000</td>
<td>-600.000</td>
</tr>
<tr>
<td>21</td>
<td>6</td>
<td>02</td>
<td>-150.000</td>
<td>-300.000</td>
<td>-600.000</td>
</tr>
<tr>
<td>22</td>
<td>7</td>
<td>02</td>
<td>-150.000</td>
<td>-300.000</td>
<td>-600.000</td>
</tr>
</tbody>
</table>

Figure 9A.---Location of the BOT arrays for all model layers within the ARC/INFO datafile, BOTTOMS.
Description of items

ARC/INFO file NAME: VTHYCOND
4 ITEMS: STARTING IN POSITION 1

COL ITEM NAME WIDTH OPUT TYP N.DEC ALTERNATE NAME
1 ROW                 2    2 I    -
3 COLUMN              2    2 I    -
5 VCONT_1             4   12 F    3
9 VCONT_2             4   12 F    3
** REDEFINED ITEMS **
1 ROWCOLUMN           4    4 I    -

Example of item values

$RECNO ROW COLUMN VCONT_1 VCONT_2
1  1 01 2.000 1.000
2  1 02 2.000 1.000
3  1 03 2.000 1.000
4  1 04 2.000 1.000
5  1 05 2.000 1.000
6  1 06 2.000 1.000
7  1 07 2.000 1.000
8  1 08 2.000 1.000
9  1 09 2.000 1.000
10 1 10 2.000 1.000
11 1 11 2.000 1.000
12 1 12 2.000 1.000
13 1 13 2.000 1.000
14 1 14 2.000 1.000
15 1 15 2.000 1.000
16 2 01 2.000 1.000
17 2 02 2.000 1.000
18 2 03 2.000 1.000
19 2 04 2.000 1.000
20 2 05 2.000 1.000
21 2 06 2.000 1.000
22 2 07 2.000 1.000

Figure 10A.---Location of the VCONT arrays for first two model layers within the ARC/INFO datafile, VTHYCOND.
For packages which do not use a module from the utility package (UTLARC) to read input data such as the well package, the input arrays are read from the "read and prepare" module WEL1RP (in the case of MODFLOWARC, the module is WEL1RPARC). The root name for each item to be read is derived from the names for each of these package under the topic "for each stress period". For the well package, the root names originate from variables: layer, row, column (some packages such as the river package did not truncate this word and therefore the entire word column is used as a root name). A listing of the ARC/INFO files, WELLS and DRAINS, containing the sample input array for well and drain packages are shown in figures 12A and 13A.

<table>
<thead>
<tr>
<th>Item</th>
<th>Name</th>
<th>Width</th>
<th>Output Type</th>
<th>Number of Decimals</th>
<th>Alternate Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ROW</td>
<td>2</td>
<td>I</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>COLUMN</td>
<td>2</td>
<td>I</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>TRAN_1</td>
<td>4</td>
<td>F</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>TRAN_2</td>
<td>4</td>
<td>F</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>TRAN_3</td>
<td>4</td>
<td>F</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

**REDEFINED ITEMS**

<table>
<thead>
<tr>
<th>Item</th>
<th>Name</th>
<th>Width</th>
<th>Output Type</th>
<th>Number of Decimals</th>
<th>Alternate Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ROWCOLUMN</td>
<td>4</td>
<td>I</td>
<td>-</td>
<td></td>
</tr>
</tbody>
</table>

**Example of item values**

<table>
<thead>
<tr>
<th>$RECNO</th>
<th>ROW COLUMN</th>
<th>TRAN_1</th>
<th>TRAN_2</th>
<th>TRAN_3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1 01</td>
<td>1.000</td>
<td>0.010</td>
<td>0.020</td>
</tr>
<tr>
<td>2</td>
<td>1 02</td>
<td>1.000</td>
<td>0.010</td>
<td>0.020</td>
</tr>
<tr>
<td>3</td>
<td>1 03</td>
<td>1.000</td>
<td>0.010</td>
<td>0.020</td>
</tr>
<tr>
<td>5</td>
<td>1 04</td>
<td>1.000</td>
<td>0.010</td>
<td>0.020</td>
</tr>
<tr>
<td>6</td>
<td>1 05</td>
<td>1.000</td>
<td>0.010</td>
<td>0.020</td>
</tr>
<tr>
<td>7</td>
<td>1 06</td>
<td>1.000</td>
<td>0.010</td>
<td>0.020</td>
</tr>
<tr>
<td>8</td>
<td>1 07</td>
<td>1.000</td>
<td>0.010</td>
<td>0.020</td>
</tr>
<tr>
<td>9</td>
<td>1 08</td>
<td>1.000</td>
<td>0.010</td>
<td>0.020</td>
</tr>
<tr>
<td>10</td>
<td>1 09</td>
<td>1.000</td>
<td>0.010</td>
<td>0.020</td>
</tr>
<tr>
<td>11</td>
<td>1 10</td>
<td>1.000</td>
<td>0.010</td>
<td>0.020</td>
</tr>
<tr>
<td>12</td>
<td>1 11</td>
<td>1.000</td>
<td>0.010</td>
<td>0.020</td>
</tr>
<tr>
<td>13</td>
<td>1 12</td>
<td>1.000</td>
<td>0.010</td>
<td>0.020</td>
</tr>
<tr>
<td>14</td>
<td>1 13</td>
<td>1.000</td>
<td>0.010</td>
<td>0.020</td>
</tr>
<tr>
<td>15</td>
<td>1 14</td>
<td>1.000</td>
<td>0.010</td>
<td>0.020</td>
</tr>
<tr>
<td>16</td>
<td>1 15</td>
<td>1.000</td>
<td>0.010</td>
<td>0.020</td>
</tr>
<tr>
<td>17</td>
<td>2 01</td>
<td>1.000</td>
<td>0.010</td>
<td>0.020</td>
</tr>
<tr>
<td>18</td>
<td>2 02</td>
<td>1.000</td>
<td>0.010</td>
<td>0.020</td>
</tr>
<tr>
<td>19</td>
<td>2 03</td>
<td>1.000</td>
<td>0.010</td>
<td>0.020</td>
</tr>
<tr>
<td>20</td>
<td>2 04</td>
<td>1.000</td>
<td>0.010</td>
<td>0.020</td>
</tr>
<tr>
<td>21</td>
<td>2 05</td>
<td>1.000</td>
<td>0.010</td>
<td>0.020</td>
</tr>
<tr>
<td>22</td>
<td>2 06</td>
<td>1.000</td>
<td>0.010</td>
<td>0.020</td>
</tr>
</tbody>
</table>

Figure 11A.—Location of the TRAN arrays for last two model layers within the ARC/INFO datafile, TRANS.
Description of items

ARC/INFO file NAME: WELLS
4 ITEMS: STARTING IN POSITION 1
COL ITEM NAME WDTH OPUT TYP N.DEC ALTERNATE NAME
1 LAYER 2 2 I -
3 ROW 2 2 I -
5 COLUMN 2 2 I -
7 Q 4 12 F 3
** REDEFINED ITEMS **
7 STRESSRATE 4 12 F 3

Example of item values

$RECNO LAYER ROW COLUMN Q
1 3 5 11 -5.000
2 2 4 6 -5.000
3 2 6 12 -5.000
4 1 9 8 -5.000
5 1 9 10 -5.000
6 1 9 12 -5.000
7 1 9 14 -5.000
8 1 11 8 -5.000
9 1 11 10 -5.000
10 1 11 12 -5.000
11 1 11 14 -5.000
12 1 13 8 -5.000
13 1 13 10 -5.000
14 1 13 12 -5.000
15 1 13 14 -5.000

Figure 12A---Location of the well array within the ARC/INFO datafile WELLS.

Description of items

ARC/INFO file NAME: DRAINS
5 ITEMS: STARTING IN POSITION 1
COL ITEM NAME WDTH OPUT TYP N.DEC ALTERNATE NAME
1 LAYER 2 2 I -
3 ROW 2 2 I -
5 COLUMN 2 2 I -
7 ELEVATION 4 12 F 3
11 COND 4 12 F 3

Example of item values

$RECNO LAYER ROW COLUMN ELEVATION COND
1 1 8 2 0.000 1.000
2 1 8 3 0.000 1.000
3 1 8 4 10.000 1.000
4 1 8 5 20.000 1.000
5 1 8 6 30.000 1.000
6 1 8 7 50.000 1.000
7 1 8 8 70.000 1.000
8 1 8 9 90.000 1.000
9 1 8 10 100.000 1.000

Figure 13A---Location of the DRAIN array within the ARC/INFO datafile DRAINS.
A listing of control files for each package used for input to the sample problem show below the listing indicates where model input arrays were stored (the ARC/INFO path to the files containing the input arrays in bold type). These control files are identifiable to those files used in the original sample problem in MODFLOW, except an ARC/INFO path has been included for those input arrays that are read from ARC/INFO files.

**Input for FORTRAN unit 7 -- Basic Package:**

123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890
SAMPLE--3 LAYERS, 15 ROWS, 15 COLUMNS; STEADY STATE; CONSTANT HEADS COLUMN 1, LAYERS 1 AND 2;
RECHARGE, WELLS AND DRAINS
3 15 15 1 1
1234567890123456789012345678901234567890123456789012345678901234567890
8 9 10 0 0 0 15 16 0 0 19
0 1 IAPART, ISTRT
5 1(15I3) 3GWINF>SAMPLE>INFO!ARC!IBOUND
5 1(15I3) 3GWINF>SAMPLE>INFO!ARC!IBOUND
0 1 IBOUND=3
999.99
5 1(15I3) 3GWINF>SAMPLE>INFO!ARC!HEADS
0 0. HEAD=2
0 0. HEAD=3
86400. 1 1. PERLEN, NSTP, TSMULT

**Input for FORTRAN unit 8 -- Block-Centered Flow Package:**

123456789012345678901234567890123456789012345678901234567890123456789012345678901234567
1 1 ISS, IBCFB
1 0 0
5 1(15I3) 3GWINF>SAMPLE>INFO!ARC!ANISOTROPY
5 0.0(15I3) 3GWINF>SAMPLE>INFO!ARC!DELR
5 0.0(15I3) 3GWINF>SAMPLE>INFO!ARC!DELC
5 0.0(15I3) 3GWINF>SAMPLE>INFO!ARC!HYCOND
5 1.E-8(15I3) 12GWINF>SAMPLE>INFO!ARC!VTHYCOND
5 1.0(15I3) 12GWINF>SAMPLE>INFO!ARC!TRANS
5 1.E-8(15I3) 12GWINF>SAMPLE>INFO!ARC!VTHYCOND
5 1.0(15I3) 3GWINF>SAMPLE>INFO!ARC!TRANS

**Input for FORTRAN unit 15 -- Recharge Package:**

123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890
1 0 NRCHOP, IRCHBD
1 INRECH
0 3.E-8 RECH=1

**Input for FORTRAN unit 16 -- Strongly Implicit Procedure Package:**

123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890
50 5 MXITER, NPARM
1 .001 0 .001 1 ACCL, ERR, IPCALC, WSEED

**Input for FORTRAN unit 10 -- Drain Package:**

123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890
9 77 MXDRAI, IDRNB
9GWINF>SAMPLE>INFO!ARC!DRAINS

**Input for FORTRAN unit 9 -- Well Package:**

123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890
15 77 MXWELL, IWELLB
15GWINF>SAMPLE>INFO!ARC!WELLS
Output Data for Sample Problem

Originally, model output control was set at the default level and for testing purposes has been reset to direct output arrays to be recorded on disk. Output arrays can be recorded in three types of data files: (1) ASCII files, (2) unformatted files, and (3) ARC/INFO files. Combinations of ASCII and unformatted files is possible as well as ASCII and ARC/INFO files. However, the combination of unformatted and ARC/INFO files is not possible. MODFLOWARC simply allows the user to redirect output to ARC/INFO files instead of unformatted files. Furthermore, the heads and drawdown output arrays can be directed to a different workspace than where the budget output arrays are recorded. The output ARC/INFO file names also specify the stress period and the time step within the stress period. For the sample problem, heads and drawdowns for each layer are recorded in two ARC/INFO files named HEDBUD_1_1 and DDNBUD_1_1 as shown in the following ARC/INFO directory listing (fig. 14A).

<table>
<thead>
<tr>
<th>TYPE</th>
<th>NAME</th>
<th>INTERNAL NAME</th>
<th>NO. RECS</th>
<th>LENGTH</th>
<th>EXTERNL</th>
</tr>
</thead>
<tbody>
<tr>
<td>DF</td>
<td>DNBUD_1_1</td>
<td>ARC002DAT</td>
<td>225</td>
<td>16</td>
<td></td>
</tr>
<tr>
<td>DF</td>
<td>HEDBUD_1_1</td>
<td>ARC003DAT</td>
<td>225</td>
<td>16</td>
<td></td>
</tr>
<tr>
<td>DF</td>
<td>WELBUD_1_1</td>
<td>ARC004DAT</td>
<td>225</td>
<td>16</td>
<td></td>
</tr>
<tr>
<td>DF</td>
<td>DRNBUD_1_1</td>
<td>ARC005DAT</td>
<td>225</td>
<td>16</td>
<td></td>
</tr>
<tr>
<td>DF</td>
<td>FRFBUD_1_1</td>
<td>ARC010DAT</td>
<td>225</td>
<td>16</td>
<td></td>
</tr>
<tr>
<td>DF</td>
<td>FFFBUD_1_1</td>
<td>ARC007DAT</td>
<td>225</td>
<td>16</td>
<td></td>
</tr>
<tr>
<td>DF</td>
<td>CHDBUD_1_1</td>
<td>ARC011DAT</td>
<td>225</td>
<td>16</td>
<td></td>
</tr>
<tr>
<td>DF</td>
<td>RCHBUD_1_1</td>
<td>ARC012DAT</td>
<td>225</td>
<td>16</td>
<td></td>
</tr>
</tbody>
</table>

Figure 14A.---Location of the output arrays within the ARC/INFO directory.

Output Control for Sample Problem

Shown below is a listing of the control file for the BAS package used for output from the sample problem showing where model output arrays are recorded (the ARC/INFO path to the files where output arrays are in bold type).

```
Input for FORTRAN unit 7 -- Basic Package:
123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890
4   8    76    77GWINF>SAMPLE>INFO!ARC!
1   -1    1     -1GWINF>BUDGET>INFO!ARC!
1   1    1      1
1   1    1      1
1   1    1      1
```

The programs of MODFLOWARC are activated that during the output phase of a model run, and directs output arrays to ARC/INFO files. The user includes a path to the ARC/INFO directory where the array values for model output would be sent as well as changing the values of the following variables, IHDDFL and ICBCFL, to negative one. Output arrays from a model run as seen below were directed to a workspace called SAMPLE (GWINF>SAMPLE>INFO!ARC!), but the names of each of the ARC/INFO files that contain the output values were omitted. The MODFLOWARC modules formulate these output file names. For this sample problem, the output heads and drawdown for the three layers was directed to SAMPLE, while the budget output could be directed to another workspace. The output ARC/INFO file names also specify the stress period and the time step within the stress period. For the sample problem, heads, drawdowns, and budget values for cell-by-cell, wells, and drains for each layer were recorded in
ARC/INFO files named HEDBUD_1_1 (heads), DDNBUD_1_1 (drawdown), CHDBUD_1_1 (constant head cells for cell-by-cell), FRFBUD_1_1 (flow right face for cell-by-cell), FFFBUD_1_1 (flow front face for cell-by-cell), FLFBUD_1_1 (flow lower face for cell-by-cell), WELBUD_1_1 (wells), RCHBUD_1_1 (recharge), and DRNBUD_1_1 (drains) (fig. 15A-23A).

Furthermore, the item names in which the output array values were recorded were also formulated by the modules of MODFLOWARC. The layer that is being recorded within ARC/INFO files is appended to the root name of LAYER such as for the heads array for layer 2. The output values are recorded in the item named LAYER_2 (fig. 15A).

<table>
<thead>
<tr>
<th>Description of items</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATAFILE NAME: HEDBUD_1_1</td>
</tr>
<tr>
<td>5 ITEMS: STARTING IN POSITION 1</td>
</tr>
<tr>
<td>COL ITEM NAME WDTH OPUT TYP N.DEC ALTERNATE NAME</td>
</tr>
<tr>
<td>1 ROW 2 2 I -</td>
</tr>
<tr>
<td>3 COLUMN 2 2 I -</td>
</tr>
<tr>
<td>5 LAYER_1 4 12 F 3</td>
</tr>
<tr>
<td>9 LAYER_2 4 12 F 3</td>
</tr>
<tr>
<td>13 LAYER_3 4 12 F 3</td>
</tr>
<tr>
<td>** REDEFINED ITEMS **</td>
</tr>
<tr>
<td>1 ROWCOLUMN 4 4 I -</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Example of item values</th>
</tr>
</thead>
<tbody>
<tr>
<td>$RECNO ROW COLUMN LAYER_1 LAYER_2 LAYER_3</td>
</tr>
<tr>
<td>1 1 1 0.000 0.000 1.800</td>
</tr>
<tr>
<td>2 1 2 24.945 24.663 24.342</td>
</tr>
<tr>
<td>3 1 3 44.007 43.730 43.363</td>
</tr>
<tr>
<td>4 1 4 59.257 59.018 58.699</td>
</tr>
<tr>
<td>5 1 5 71.824 71.611 71.326</td>
</tr>
<tr>
<td>6 1 6 82.518 82.321 82.057</td>
</tr>
<tr>
<td>7 1 7 91.907 91.723 91.478</td>
</tr>
<tr>
<td>8 1 8 100.035 99.862 99.630</td>
</tr>
<tr>
<td>9 1 9 106.915 106.749 106.528</td>
</tr>
<tr>
<td>10 1 10 112.646 112.486 112.272</td>
</tr>
<tr>
<td>11 1 11 117.385 117.229 117.021</td>
</tr>
<tr>
<td>12 1 12 121.270 121.117 120.914</td>
</tr>
<tr>
<td>13 1 13 124.295 124.145 123.946</td>
</tr>
<tr>
<td>14 1 14 126.381 126.233 126.036</td>
</tr>
<tr>
<td>15 1 15 127.447 127.300 127.104</td>
</tr>
</tbody>
</table>

Figure 15A---Location of the head array for all layers within the ARC/INFO datafile HEDBUD_1_1.
### Description of items

**DATAFILE NAME:** DDNBUD_1_1  
**5 ITEMS: STARTING IN POSITION 1**  

<table>
<thead>
<tr>
<th>COL</th>
<th>ITEM NAME</th>
<th>WDTH</th>
<th>OPUT</th>
<th>TYP</th>
<th>N.DEC</th>
<th>ALTERNATE NAME</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ROW</td>
<td>2</td>
<td>2</td>
<td>I</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>COLUMN</td>
<td>2</td>
<td>2</td>
<td>I</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>LAYER_1</td>
<td>4</td>
<td>12</td>
<td>F</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>LAYER_2</td>
<td>4</td>
<td>12</td>
<td>F</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>LAYER_3</td>
<td>4</td>
<td>12</td>
<td>F</td>
<td>3</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**REDEFINED ITEMS**

<table>
<thead>
<tr>
<th>COL</th>
<th>ITEM NAME</th>
<th>WDTH</th>
<th>OPUT</th>
<th>TYP</th>
<th>N.DEC</th>
<th>ALTERNATE NAME</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ROWCOLUMN</td>
<td>4</td>
<td>4</td>
<td>I</td>
<td>-</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Example of item values

<table>
<thead>
<tr>
<th>$RECNO</th>
<th>ROW</th>
<th>COLUMN</th>
<th>LAYER_1</th>
<th>LAYER_2</th>
<th>LAYER_3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>01</td>
<td>0.000</td>
<td>0.000</td>
<td>-1.799</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>02</td>
<td>-24.928</td>
<td>-24.646</td>
<td>-24.325</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>03</td>
<td>-43.976</td>
<td>-43.699</td>
<td>-43.332</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>04</td>
<td>-59.212</td>
<td>-58.973</td>
<td>-58.654</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>05</td>
<td>-71.765</td>
<td>-71.552</td>
<td>-71.268</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>06</td>
<td>-82.445</td>
<td>-82.248</td>
<td>-81.985</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>07</td>
<td>-91.821</td>
<td>-91.637</td>
<td>-91.392</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>08</td>
<td>-99.936</td>
<td>-99.763</td>
<td>-99.531</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>09</td>
<td>-106.805</td>
<td>-106.639</td>
<td>-106.417</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>10</td>
<td>-112.525</td>
<td>-112.365</td>
<td>-112.151</td>
</tr>
<tr>
<td>11</td>
<td>1</td>
<td>11</td>
<td>-117.257</td>
<td>-117.101</td>
<td>-116.892</td>
</tr>
<tr>
<td>12</td>
<td>1</td>
<td>12</td>
<td>-121.136</td>
<td>-120.984</td>
<td>-120.780</td>
</tr>
<tr>
<td>13</td>
<td>1</td>
<td>13</td>
<td>-124.159</td>
<td>-124.009</td>
<td>-123.809</td>
</tr>
<tr>
<td>14</td>
<td>1</td>
<td>14</td>
<td>-126.243</td>
<td>-126.095</td>
<td>-125.898</td>
</tr>
<tr>
<td>15</td>
<td>1</td>
<td>15</td>
<td>-127.308</td>
<td>-127.161</td>
<td>-126.965</td>
</tr>
</tbody>
</table>

Figure 16A.---Location of the drawdown array for all layers within the ARC/INFO datafile DDNBUD_1_1.
Description of items

DATAFILE NAME: CHDBUD_1_1
5 ITEMS: STARTING IN POSITION 1

<table>
<thead>
<tr>
<th>COL</th>
<th>ITEM NAME</th>
<th>WDH</th>
<th>OUT</th>
<th>TYP</th>
<th>N.DEC</th>
<th>ALTERNATE NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ROW</td>
<td>2</td>
<td>2</td>
<td>I</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>COLUMN</td>
<td>2</td>
<td>2</td>
<td>I</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>LAYER_1</td>
<td>4</td>
<td>12</td>
<td>F</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>LAYER_2</td>
<td>4</td>
<td>12</td>
<td>F</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>LAYER_3</td>
<td>4</td>
<td>12</td>
<td>F</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

** REDEFINED ITEMS **
1 ROWCOLUMN 4 2 I -

Example of item values

<table>
<thead>
<tr>
<th>SRECNO</th>
<th>ROW</th>
<th>COLUMN</th>
<th>LAYER_1</th>
<th>LAYER_2</th>
<th>LAYER_3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>01</td>
<td>-4.029</td>
<td>-0.697</td>
<td>0.000</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>02</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>03</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>04</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>05</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>06</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>07</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>08</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>09</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>10</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>11</td>
<td>1</td>
<td>11</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>12</td>
<td>1</td>
<td>12</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>13</td>
<td>1</td>
<td>13</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>14</td>
<td>1</td>
<td>14</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>15</td>
<td>1</td>
<td>15</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>16</td>
<td>2</td>
<td>01</td>
<td>-3.943</td>
<td>-0.683</td>
<td>0.000</td>
</tr>
<tr>
<td>17</td>
<td>2</td>
<td>02</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>18</td>
<td>2</td>
<td>03</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>19</td>
<td>2</td>
<td>04</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>20</td>
<td>2</td>
<td>05</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>21</td>
<td>2</td>
<td>06</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>22</td>
<td>2</td>
<td>07</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
</tbody>
</table>

Figure 17A.—Location of the constant head array for all layers within the ARC/INFO datafile CHDBUD_1_1.
Description of items

DATAFILE NAME: FRFBUD_1_1
5 ITEMS: STARTING IN POSITION 1

<table>
<thead>
<tr>
<th>COL</th>
<th>ITEM NAME</th>
<th>WDTH</th>
<th>OPUT</th>
<th>TYP</th>
<th>N.DEC</th>
<th>ALTERNATE NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ROW</td>
<td>2</td>
<td>2</td>
<td>I</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>COLUMN</td>
<td>2</td>
<td>2</td>
<td>I</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>LAYER_1</td>
<td>4</td>
<td>12</td>
<td>F</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>LAYER_2</td>
<td>4</td>
<td>12</td>
<td>F</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>LAYER_3</td>
<td>4</td>
<td>12</td>
<td>F</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

** REDEFINED ITEMS **

1 ROWCOLUMN 4 2 I -

Example of item values

<table>
<thead>
<tr>
<th>$RECNO</th>
<th>ROW COLUMN</th>
<th>LAYER_1</th>
<th>LAYER_2</th>
<th>LAYER_3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1 01</td>
<td>-4.029</td>
<td>-0.247</td>
<td>-0.451</td>
</tr>
<tr>
<td>2</td>
<td>1 02</td>
<td>-3.507</td>
<td>-0.191</td>
<td>-0.380</td>
</tr>
<tr>
<td>3</td>
<td>1 03</td>
<td>-3.070</td>
<td>-0.153</td>
<td>-0.307</td>
</tr>
<tr>
<td>4</td>
<td>1 04</td>
<td>-2.706</td>
<td>-0.126</td>
<td>-0.253</td>
</tr>
<tr>
<td>5</td>
<td>1 05</td>
<td>-2.428</td>
<td>-0.107</td>
<td>-0.215</td>
</tr>
<tr>
<td>6</td>
<td>1 06</td>
<td>-2.226</td>
<td>-0.094</td>
<td>-0.188</td>
</tr>
<tr>
<td>7</td>
<td>1 07</td>
<td>-1.999</td>
<td>-0.081</td>
<td>-0.163</td>
</tr>
<tr>
<td>8</td>
<td>1 08</td>
<td>-1.744</td>
<td>-0.069</td>
<td>-0.138</td>
</tr>
<tr>
<td>9</td>
<td>1 09</td>
<td>-1.488</td>
<td>-0.057</td>
<td>-0.115</td>
</tr>
<tr>
<td>10</td>
<td>1 10</td>
<td>-1.256</td>
<td>-0.047</td>
<td>-0.095</td>
</tr>
<tr>
<td>11</td>
<td>1 11</td>
<td>-1.046</td>
<td>-0.039</td>
<td>-0.078</td>
</tr>
<tr>
<td>12</td>
<td>1 12</td>
<td>-0.825</td>
<td>-0.030</td>
<td>-0.061</td>
</tr>
<tr>
<td>13</td>
<td>1 13</td>
<td>-0.574</td>
<td>-0.021</td>
<td>-0.042</td>
</tr>
<tr>
<td>14</td>
<td>1 14</td>
<td>-0.295</td>
<td>-0.011</td>
<td>-0.021</td>
</tr>
<tr>
<td>15</td>
<td>1 15</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>16</td>
<td>2 01</td>
<td>-3.943</td>
<td>-0.242</td>
<td>-0.442</td>
</tr>
<tr>
<td>17</td>
<td>2 02</td>
<td>-3.420</td>
<td>-0.187</td>
<td>-0.372</td>
</tr>
<tr>
<td>18</td>
<td>2 03</td>
<td>-2.979</td>
<td>-0.149</td>
<td>-0.299</td>
</tr>
<tr>
<td>19</td>
<td>2 04</td>
<td>-2.608</td>
<td>-0.122</td>
<td>-0.245</td>
</tr>
<tr>
<td>20</td>
<td>2 05</td>
<td>-2.343</td>
<td>-0.104</td>
<td>-0.208</td>
</tr>
<tr>
<td>21</td>
<td>2 06</td>
<td>-2.246</td>
<td>-0.096</td>
<td>-0.192</td>
</tr>
<tr>
<td>22</td>
<td>2 07</td>
<td>-2.021</td>
<td>-0.083</td>
<td>-0.166</td>
</tr>
</tbody>
</table>

Figure 18A.—Location of the flow right face array for all layers within the ARC/INFO datafile FRFBUD_1_1.
### Description of items

**DATAFILE NAME: FFFBUD_1_1**

5 ITEMS: STARTING IN POSITION 1

<table>
<thead>
<tr>
<th>COL</th>
<th>ITEM NAME</th>
<th>WDTH</th>
<th>OPUT</th>
<th>TYP</th>
<th>N.DEC</th>
<th>ALTERNATE NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ROW</td>
<td>2</td>
<td>2</td>
<td>I</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>COLUMN</td>
<td>2</td>
<td>2</td>
<td>I</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>LAYER_1</td>
<td>4</td>
<td>12</td>
<td>F</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>LAYER_2</td>
<td>4</td>
<td>12</td>
<td>F</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>LAYER_3</td>
<td>4</td>
<td>12</td>
<td>F</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

**REDEFINED ITEMS**

| 1   | ROWCOLUMN | 4    | 2    | I   | -      |                |

### Example of item values

<table>
<thead>
<tr>
<th>$RECNO</th>
<th>ROW COLUMN</th>
<th>LAYER_1</th>
<th>LAYER_2</th>
<th>LAYER_3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1 01</td>
<td>0.000</td>
<td>0.000</td>
<td>0.001</td>
</tr>
<tr>
<td>2</td>
<td>1 02</td>
<td>0.087</td>
<td>0.005</td>
<td>0.010</td>
</tr>
<tr>
<td>3</td>
<td>1 03</td>
<td>0.175</td>
<td>0.009</td>
<td>0.018</td>
</tr>
<tr>
<td>4</td>
<td>1 04</td>
<td>0.267</td>
<td>0.013</td>
<td>0.026</td>
</tr>
<tr>
<td>5</td>
<td>1 05</td>
<td>0.365</td>
<td>0.017</td>
<td>0.033</td>
</tr>
<tr>
<td>6</td>
<td>1 06</td>
<td>0.450</td>
<td>0.020</td>
<td>0.040</td>
</tr>
<tr>
<td>7</td>
<td>1 07</td>
<td>0.430</td>
<td>0.018</td>
<td>0.036</td>
</tr>
<tr>
<td>8</td>
<td>1 08</td>
<td>0.408</td>
<td>0.016</td>
<td>0.033</td>
</tr>
<tr>
<td>9</td>
<td>1 09</td>
<td>0.412</td>
<td>0.016</td>
<td>0.032</td>
</tr>
<tr>
<td>10</td>
<td>1 10</td>
<td>0.437</td>
<td>0.017</td>
<td>0.034</td>
</tr>
<tr>
<td>11</td>
<td>1 11</td>
<td>0.462</td>
<td>0.017</td>
<td>0.035</td>
</tr>
<tr>
<td>12</td>
<td>1 12</td>
<td>0.453</td>
<td>0.017</td>
<td>0.034</td>
</tr>
<tr>
<td>13</td>
<td>1 13</td>
<td>0.424</td>
<td>0.016</td>
<td>0.031</td>
</tr>
<tr>
<td>14</td>
<td>1 14</td>
<td>0.397</td>
<td>0.014</td>
<td>0.029</td>
</tr>
<tr>
<td>15</td>
<td>1 15</td>
<td>0.381</td>
<td>0.014</td>
<td>0.028</td>
</tr>
<tr>
<td>16</td>
<td>2 01</td>
<td>0.000</td>
<td>0.000</td>
<td>0.001</td>
</tr>
<tr>
<td>17</td>
<td>2 02</td>
<td>0.174</td>
<td>0.010</td>
<td>0.020</td>
</tr>
<tr>
<td>18</td>
<td>2 03</td>
<td>0.346</td>
<td>0.018</td>
<td>0.036</td>
</tr>
<tr>
<td>19</td>
<td>2 04</td>
<td>0.526</td>
<td>0.025</td>
<td>0.051</td>
</tr>
<tr>
<td>20</td>
<td>2 05</td>
<td>0.742</td>
<td>0.034</td>
<td>0.069</td>
</tr>
<tr>
<td>21</td>
<td>2 06</td>
<td>0.996</td>
<td>0.046</td>
<td>0.096</td>
</tr>
<tr>
<td>22</td>
<td>2 07</td>
<td>0.860</td>
<td>0.036</td>
<td>0.074</td>
</tr>
</tbody>
</table>

Figure 19A.—Location of the flow front face array for all layers within the ARC/INFO datafile FFFBUD_1_1.
**Description of items**

DATAFILE NAME: FLFBUD_1_1  
5 ITEMS: STARTING IN POSITION 1  
COL ITEM NAME WDTH OPUT TYP N_DEC ALTERNATE NAME  
1 ROW 2 2 I -  
3 COLUMN 2 2 I -  
5 LAYER_1 4 12 F 3  
9 LAYER_2 4 12 F 3  
13 LAYER_3 4 12 F 3  
** REDEFINED ITEMS **  
1 ROWCOLUMN 4 2 I -

**Example of item values**

<table>
<thead>
<tr>
<th>$RECNO</th>
<th>ROW</th>
<th>COLUMN</th>
<th>LAYER_1</th>
<th>LAYER_2</th>
<th>LAYER_3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>01</td>
<td>0.000</td>
<td>-0.450</td>
<td>0.000</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>02</td>
<td>0.141</td>
<td>0.080</td>
<td>0.000</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>03</td>
<td>0.138</td>
<td>0.092</td>
<td>0.000</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>04</td>
<td>0.119</td>
<td>0.080</td>
<td>0.000</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>05</td>
<td>0.107</td>
<td>0.071</td>
<td>0.000</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>06</td>
<td>0.099</td>
<td>0.066</td>
<td>0.000</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>07</td>
<td>0.092</td>
<td>0.061</td>
<td>0.000</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>08</td>
<td>0.087</td>
<td>0.058</td>
<td>0.000</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>09</td>
<td>0.083</td>
<td>0.055</td>
<td>0.000</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>10</td>
<td>0.080</td>
<td>0.053</td>
<td>0.000</td>
</tr>
<tr>
<td>11</td>
<td>1</td>
<td>11</td>
<td>0.078</td>
<td>0.052</td>
<td>0.000</td>
</tr>
<tr>
<td>12</td>
<td>1</td>
<td>12</td>
<td>0.076</td>
<td>-0.051</td>
<td>0.000</td>
</tr>
<tr>
<td>13</td>
<td>1</td>
<td>13</td>
<td>0.075</td>
<td>0.050</td>
<td>0.000</td>
</tr>
<tr>
<td>14</td>
<td>1</td>
<td>14</td>
<td>0.074</td>
<td>0.049</td>
<td>0.000</td>
</tr>
<tr>
<td>15</td>
<td>1</td>
<td>15</td>
<td>0.073</td>
<td>0.049</td>
<td>0.000</td>
</tr>
<tr>
<td>16</td>
<td>2</td>
<td>01</td>
<td>0.000</td>
<td>-0.441</td>
<td>0.000</td>
</tr>
<tr>
<td>17</td>
<td>2</td>
<td>02</td>
<td>0.139</td>
<td>0.079</td>
<td>0.000</td>
</tr>
<tr>
<td>18</td>
<td>2</td>
<td>03</td>
<td>0.138</td>
<td>0.091</td>
<td>0.000</td>
</tr>
<tr>
<td>19</td>
<td>2</td>
<td>04</td>
<td>0.119</td>
<td>0.080</td>
<td>0.000</td>
</tr>
<tr>
<td>20</td>
<td>2</td>
<td>05</td>
<td>0.108</td>
<td>0.073</td>
<td>0.000</td>
</tr>
<tr>
<td>21</td>
<td>2</td>
<td>06</td>
<td>0.107</td>
<td>0.072</td>
<td>0.000</td>
</tr>
<tr>
<td>22</td>
<td>2</td>
<td>07</td>
<td>0.095</td>
<td>0.064</td>
<td>0.000</td>
</tr>
</tbody>
</table>

Figure 20A.---Location of the flow lower face array for all layers within the ARC/INFO datafile FLFBUD_1_1.
Description of items

DATAFILE NAME: WELBUD_1_1
5 ITEMS: STARTING IN POSITION 1

<table>
<thead>
<tr>
<th>COL</th>
<th>ITEM NAME</th>
<th>WDTH</th>
<th>OPT</th>
<th>TYP</th>
<th>N.DEC</th>
<th>ALTERNATE NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ROW</td>
<td>2</td>
<td>2</td>
<td>I</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>COLUMN</td>
<td>2</td>
<td>2</td>
<td>I</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>LAYER_1</td>
<td>4</td>
<td>12</td>
<td>F</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>LAYER_2</td>
<td>4</td>
<td>12</td>
<td>F</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>LAYER_3</td>
<td>4</td>
<td>12</td>
<td>F</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

** REDEFINED ITEMS **
1 ROWCOLUMN 4 2 I -

Example of item values

<table>
<thead>
<tr>
<th>$RECNO</th>
<th>ROW</th>
<th>COLUMN</th>
<th>LAYER_1</th>
<th>LAYER_2</th>
<th>LAYER_3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>01</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>02</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>03</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>04</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>05</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>06</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>07</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>08</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>09</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>10</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>11</td>
<td>1</td>
<td>11</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>12</td>
<td>1</td>
<td>12</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>13</td>
<td>1</td>
<td>13</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>14</td>
<td>1</td>
<td>14</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>15</td>
<td>1</td>
<td>15</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>16</td>
<td>2</td>
<td>01</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>17</td>
<td>2</td>
<td>02</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>18</td>
<td>2</td>
<td>03</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>19</td>
<td>2</td>
<td>04</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>20</td>
<td>2</td>
<td>05</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>21</td>
<td>2</td>
<td>06</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>22</td>
<td>2</td>
<td>07</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
</tbody>
</table>

Figure 21A.—Location of the well budget array for all layers within the ARC/INFO datafile WELBUD_1_1 (note only cells that have wells located in them will have non-zero values).
Description of items

DATAFILE NAME: RCHBUD_1_1
5 ITEMS: STARTING IN POSITION 1

<table>
<thead>
<tr>
<th>COL</th>
<th>ITEM NAME</th>
<th>WDTH</th>
<th>OPUT</th>
<th>TYP</th>
<th>N.DEC</th>
<th>ALTERNATE NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ROW</td>
<td>2</td>
<td>2</td>
<td>I</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>COLUMN</td>
<td>2</td>
<td>2</td>
<td>I</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>LAYER_1</td>
<td>4</td>
<td>12</td>
<td>F</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>LAYER_2</td>
<td>4</td>
<td>12</td>
<td>F</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>LAYER_3</td>
<td>4</td>
<td>12</td>
<td>F</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

** REDEFINED ITEMS **

<table>
<thead>
<tr>
<th>COL</th>
<th>ITEM NAME</th>
<th>WDTH</th>
<th>OPUT</th>
<th>TYP</th>
<th>N.DEC</th>
<th>ALTERNATE NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ROWCOLUMN</td>
<td>4</td>
<td>2</td>
<td>I</td>
<td>-</td>
<td></td>
</tr>
</tbody>
</table>

Example of item values

<table>
<thead>
<tr>
<th>$RECNO</th>
<th>ROW</th>
<th>COLUMN</th>
<th>LAYER_1</th>
<th>LAYER_2</th>
<th>LAYER_3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>01</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>02</td>
<td>0.750</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>03</td>
<td>0.750</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>04</td>
<td>0.750</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>05</td>
<td>0.750</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>06</td>
<td>0.750</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>07</td>
<td>0.750</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>08</td>
<td>0.750</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>09</td>
<td>0.750</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>10</td>
<td>0.750</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>11</td>
<td>1</td>
<td>11</td>
<td>0.750</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>12</td>
<td>1</td>
<td>12</td>
<td>0.750</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>13</td>
<td>1</td>
<td>13</td>
<td>0.750</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>14</td>
<td>1</td>
<td>14</td>
<td>0.750</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>15</td>
<td>1</td>
<td>15</td>
<td>0.750</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>16</td>
<td>2</td>
<td>01</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>17</td>
<td>2</td>
<td>02</td>
<td>0.750</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>18</td>
<td>2</td>
<td>03</td>
<td>0.750</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>19</td>
<td>2</td>
<td>04</td>
<td>0.750</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>20</td>
<td>2</td>
<td>05</td>
<td>0.750</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>21</td>
<td>2</td>
<td>06</td>
<td>0.750</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>22</td>
<td>2</td>
<td>07</td>
<td>0.750</td>
<td>0.000</td>
<td>0.000</td>
</tr>
</tbody>
</table>

Figure 22A.---Location of the recharge budget array for all layers within the ARC/INFO datafile RCHBUD_1_1.
Results for Sample Problem

The output of both models, MODFLOW and MODFLOWARC, show nearly duplicate results when run on the same computer and some minor differences when compared to the results by McDonald and Harbaugh (1988) due to truncation and rounding performed by different computers. The MODFLOWARC results reported were from a model run done on a Sun 4/110. These differences occur in the sections entitled "maximum head change for each iteration" and "volumetric budget for entire model at the end of time step 1 in stress period 1" of the summary output from the sample problem.

Figure 23A.---Location of the drain budget array for all layers within the ARC/INFO datafile DRNBUD_1_1 (note only cells that have drains located in them will have non-zero values).
31 ITERATIONS FOR TIME STEP 1 IN STRESS PERIOD 1

MAXIMUM HEAD CHANGE FOR EACH ITERATION:

<table>
<thead>
<tr>
<th>Head Change</th>
<th>Layer, Row, Col</th>
<th>Head Change</th>
<th>Layer, Row, Col</th>
<th>Head Change</th>
<th>Layer, Row, Col</th>
<th>Head Change</th>
<th>Layer, Row, Col</th>
<th>Head Change</th>
<th>Layer, Row, Col</th>
</tr>
</thead>
<tbody>
<tr>
<td>-22.41</td>
<td>(3, 5, 11)</td>
<td>12.48</td>
<td>(1, 1, 15)</td>
<td>13.39</td>
<td>(3, 1, 14)</td>
<td>48.21</td>
<td>(1, 1, 15)</td>
<td>35.91</td>
<td>(3, 1, 13)</td>
</tr>
<tr>
<td>2.482</td>
<td>(1, 9, 14)</td>
<td>1.430</td>
<td>(3, 10, 13)</td>
<td>6.214</td>
<td>(1, 12, 14)</td>
<td>7.411</td>
<td>(3, 11, 14)</td>
<td>13.66</td>
<td>(1, 15, 15)</td>
</tr>
<tr>
<td>0.5503</td>
<td>(3, 8, 7)</td>
<td>0.4821</td>
<td>(2, 6, 9)</td>
<td>0.4711</td>
<td>(3, 5, 10)</td>
<td>2.018</td>
<td>(1, 11, 14)</td>
<td>2.302</td>
<td>(3, 5, 13)</td>
</tr>
<tr>
<td>0.1108</td>
<td>(1, 13, 12)</td>
<td>0.7058E-01</td>
<td>(3, 12, 11)</td>
<td>0.2819</td>
<td>(1, 14, 14)</td>
<td>0.3141</td>
<td>(3, 13, 14)</td>
<td>0.3320</td>
<td>(1, 15, 15)</td>
</tr>
<tr>
<td>0.7853E-02</td>
<td>(1, 13, 12)</td>
<td>0.1586E-01</td>
<td>(2, 11, 11)</td>
<td>0.1777E-01</td>
<td>(3, 11, 10)</td>
<td>0.7910E-01</td>
<td>(1, 14, 14)</td>
<td>0.8499E-01</td>
<td>(3, 7, 14)</td>
</tr>
<tr>
<td>0.4168E-02</td>
<td>(1, 13, 14)</td>
<td>0.2555E-02</td>
<td>(3, 14, 15)</td>
<td>0.9768E-02</td>
<td>(1, 14, 14)</td>
<td>0.1082E-01</td>
<td>(3, 13, 14)</td>
<td>0.1030E-01</td>
<td>(1, 15, 15)</td>
</tr>
<tr>
<td>0.2429E-03</td>
<td>(1, 13, 12)</td>
<td>0.1108</td>
<td>(1, 13, 12)</td>
<td>0.7058E-02</td>
<td>(1, 13, 12)</td>
<td>0.2819</td>
<td>(1, 12, 14)</td>
<td>0.3141</td>
<td>(3, 13, 14)</td>
</tr>
</tbody>
</table>

VOLUMETRIC BUDGET FOR ENTIRE MODEL AT END OF TIME STEP 1 IN STRESS PERIOD 1

<table>
<thead>
<tr>
<th>CUMULATIVE VOLUMES</th>
<th>L**3</th>
<th>RATES FOR THIS TIME STEP</th>
<th>L**3/T</th>
</tr>
</thead>
<tbody>
<tr>
<td>IN:</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>STORAGE</td>
<td>0.00000</td>
<td>STORAGE</td>
<td>0.00000</td>
</tr>
<tr>
<td>CONSTANT HEAD</td>
<td>0.00000</td>
<td>CONSTANT HEAD</td>
<td>0.00000</td>
</tr>
<tr>
<td>WELLS</td>
<td>0.00000</td>
<td>WELLS</td>
<td>0.00000</td>
</tr>
<tr>
<td>DRAINS</td>
<td>0.00000</td>
<td>DRAINS</td>
<td>0.00000</td>
</tr>
<tr>
<td>RECHARGE</td>
<td>0.13608E+08</td>
<td>RECHARGE</td>
<td>157.50</td>
</tr>
<tr>
<td>TOTAL IN</td>
<td>0.13608E+08</td>
<td>TOTAL IN</td>
<td>157.50</td>
</tr>
<tr>
<td>OUT:</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>STORAGE</td>
<td>0.00000</td>
<td>STORAGE</td>
<td>0.00000</td>
</tr>
<tr>
<td>CONSTANT HEAD</td>
<td>0.43265E+07</td>
<td>CONSTANT HEAD</td>
<td>50.075</td>
</tr>
<tr>
<td>WELLS</td>
<td>0.64800E+07</td>
<td>WELLS</td>
<td>75.000</td>
</tr>
<tr>
<td>DRAINS</td>
<td>0.28011E+07</td>
<td>DRAINS</td>
<td>32.420</td>
</tr>
<tr>
<td>RECHARGE</td>
<td>0.00000</td>
<td>RECHARGE</td>
<td>0.00000</td>
</tr>
<tr>
<td>TOTAL OUT</td>
<td>0.13608E+08</td>
<td>TOTAL OUT</td>
<td>157.50</td>
</tr>
<tr>
<td>IN - OUT</td>
<td>412.00</td>
<td>IN - OUT</td>
<td>0.47607E-02</td>
</tr>
</tbody>
</table>

PERCENT DISCREPANCY = 0.00