II.4-RES-SNGL–B-RCL SINGLE RESERVOIR REGULATION OPERATION RESERVOIR COMMAND LANGUAGE (RCL)

Introduction

The Reservoir Command Language (RCL) is the mechanism allowing the flexibility to accommodate different operating plans for each reservoir.

Operating plans are instructions linked and triggered by logical checks on hydrologic and other conditions. Within the model these instructions are carried out by the computational routines. The links are provided by the RCL.

There are three statement types used in RCL.

The SET statement is used more as a convenience to the user and is not critical in duplicating operating rules of a reservoir.

The other two statement types are:

- IF-ENDIF - used for checking current conditions
- DO - used for executing Schemes and Utilities

Details on the use of these statement types follow.

A detailed description of all RCL statements, the syntax rules, formats and input requirements is presented in Section V.3.3-RES-SNGL-RCL.

IF-ENDIF Statement Block

Logical checks on existing hydrologic or timing conditions are achieved with the IF-ENDIF statement block.

Basic Block Structure

The basic form of an IF-ENDIF block is:

```
IF (<expression>) THEN <statement_group>
ENDIF
```

where <expression> is a logical expression that has a result of either 'true' or 'false'
<statement_group> is a DO statement followed by one or more DO statements and/or IF-ENDIF blocks

An evaluation of 'true' for the <expression> results in the statements between the THEN and ENDIF to be executed. A 'false' result ignores those statements and executes the statement following the ENDIF statement.

Logical Expressions

The checks that can be made in an IF-ENDIF block use two types of comparisons:

1. Relational expressions
2. Logical variables

A relational expression compares two quantities. A logical variable by itself has a 'true' or 'false' value.

Relational Expressions

A relational expression takes the form

<system_variable><relational_operator><user_variable>

The <system_variable> is one of 7 comparators:

1. QO - instantaneous discharge
2. QOM - mean discharge
3. QI - instantaneous inflow
4. QIM - mean inflow
5. POOL - pool elevation
6. STORAGE - storage contents
7. DAY - relative Julian date

The <relational_operator> is one of:

1. .EQ. - left and right sides are equal
2. .NE. - left and right sides are not equal
3. .LT. - left side is less than the right side
4. .GT. - left side is greater than the right side
5. .LE. - left side is less than or equal to the right side
6. .GE. - left side is greater than or equal to the right side

The <user_variable> can be one of 3 types:

1. SET variable name defined prior to the IF-ENDIF block in a SET statement
2. Numerical value

3. System function name

SET variables are described in Section V.3.3-RES-SNGL-RCL. Numerical values can be either integer or real. There are two system functions:

1. RULE - current rule curve elevation

2. MAXQ - maximum discharge that can be released at current elevation

These functions can be used as their actual values or modified by adding, subtracting, multiplying by or dividing by a factor. For example if the pool elevation is to be checked against the rule curve elevation plus an additional amount the relational expression would look like:

(POOL.LE.RULE+2.0)

Examples of relational expressions are:

1. Comparing the instantaneous inflow against exceeding the constant value 50000:

   (QO.GT.50000.0)

2. Comparing the mean discharge for being less than a SET variable named MAXMEAN:

   (QOM.LT.MAXMEAN)

3. Comparing the relative Julian day to the first of February:

   (DAY.GT.31)

Whenever a comparison is made the system variable quantity is taken as the last value computed for that variable. So if an IF-ENDIF block is entered before any Scheme is executed the value used is the model result from the previous period. Since the IF-ENDIF block uses the most recent model results for testing it will pick up the last Scheme's output in the case of stacked Scheme executions preceding an IF-ENDIF block.

Logical Variables

A logical variable represents the state of either 'on' ('true') or 'off' ('false') of various hydrologic and timing conditions. The variable by itself constitutes a valid logical expression. The variables available and their 'on' conditions are:

Variable Name	'On' Condition
FORECAST	Run is preceding or on last period of observed data

Variable Name	'On' Condition
OBSERVED	Run is past last period of observed data
FLOOD	Utility SUMINF has been executed and 'flood' criteria have been exceeded (is 'false' if no SUMINF has been executed prior to test)
SURCHARGE	Utility ENTERISC has been executed and indicates that induced surcharge is needed ('false' if ENTERISC not executed)
GOFLASH	Utility GOFLASH has been executed and indicates that flashboard Scheme is needed ('false' if GOFLASH not executed)
RISING	Pool elevation is rising from previous period to this period
FALLING	Pool elevation is falling from previous period to this period
NFLOOD	Opposite of FLOOD
NSURCHARGE	Opposite of SURCHARGE
NGOFLASH	Opposite of GOFLASH

Expression Combinations

Multiple relational expressions and/or logical variables can be combined in one logical expression by the logical operations .AND. and .OR. .
For example,

(QO.GT.MAXQ.AND.RISING)

is a valid expression.

Sub-expressions (less than the entire logical expression) can be isolated by parentheses to ensure proper order of evaluation. For example:

((QO.GT.MAXQ.OR.QI.GT.10000.).AND.FALLING)

In this example the combination surrounding the .OR. will be evaluated and that result used with the .AND. operator and the FALLING variable to get the final expression result.

The hierarchy of evaluation (in the order of done first to done last) is:

- expression in parentheses
- .AND. separated expressions

o .OR. separated expressions

All operators of equal hierarchy are evaluated from left to right.

Statement Group

The first RCL in the <statement_group> of an IF-ENDIF block must be a DO statement. Optionally additional RCL statements (either DO statement or IF-ENDIF groups) can be entered after the first DO statement). Any IF-ENDIF blocks imbedded within the first IF-ENDIF block are evaluated independently of the first block. There is no limit to the number of imbedded IF-ENDIF blocks.

An example of the basic IF-END block is:

```
DO ENTERISC
IF (SURCHARGE) THEN DO INDSRCHGE
   ENDIF
```

ELSE Feature

Another feature of the IF-ENDIF block is an ELSE to which processing control is passed if the comparison of the IF turns out 'false.' Its use is optional but provides a clear way to specify alternative actions in a comparison.

The form of the IF-ENDIF block using the ELSE feature is

```
IF <expression> THEN
   <statement_group_1>
   ELSE
      <statement_group_2>
   ENDIF
```

If the <expression> is 'true' then statement_group_1 is executed and control is passed to the statement following ENDIF. If the expression is 'false' statement then group 2 is executed and control goes to the statement following the ENDIF. Only one ELSE can appear in an IF-ENDIF block.

An example of RCL employing the ELSE feature is:

```
   DO ENTERISC
   IF (SURCHARGE) THEN DO INDSRCHGE
      ELSE DO PASSFLOW
      ENDIF
```

ELSEIF Feature

In the case where a selection is needed between more than two possible paths of execution within an IF-ENDIF block then the ELSEIF feature can be used. The form of an IF-ENDIF block using the ELSEIF feature is:

```
IF <expression_1> THEN
   <statement_group_1>
   ELSEIF <expression_2> THEN
      <statement_group_2>
         ELSE <statement_group_3>
   ENDIF
```

If <expression_1> is 'true' then statement_group_1 is executed and control goes to the statement after the ENDIF. If <expression_1> is 'false' then control is passed to the ELSEIF where <expression_2> is evaluated. If this proves 'true' then statement_group_2 is executed and control goes to the statement after the ENDIF. If <expression_2> is 'false' then statement_group_3 is executed.

There is no limit to the number of ELSEIF options that can appear in an IF-ENDIF block. The ELSE is optional and if used must appear after the last ELSEIF. Only one ELSE can appear in the IF-ENDIF block.

An example of the use of the ELSEIF feature (using only one ELSEIF and an ELSE) in an IF-ENDIF block is:

```
DO ENTERISC
DO GOFLASH
IF (SURCHARGE) THEN
   DO INDSRCHGE
   ELSEIF (GOFLASH) THEN
      DO PASSFLOW
   ENDIF
```

DO Statements

The DO statement is used to execute a Scheme or Utility defined in the SPECIFIC input section. The form of a DO statement is:

```
DO name
```

where name is the Scheme or Utility identifier (see Table 1)

All Schemes need a DO statement for execution but not all Utilities do. Table 1 also lists those Utilities requiring a DO statement for execution.

An example of a DO statement (executing the prescribed discharge Scheme) is:

```
DO SETQ
```

RCL Examples

Example 1:

The reservoir is to be operated by passing inflow.

```
   RCL
      DO PASSFLOW
      ENDRCL
```

Example 2:

Conditions are to be checked for going into induced surcharge and if not met the prescribed discharge Scheme is to be used.

```
   RCL
      DO ENTERISC
      IF (SURCHARGE) THEN DO INDSRCHGE
         ELSE DO SETQ
         ENDIF
      ENDRCL
```

Example 3:

Conditions are checked for going into induced surcharge and if not met the prescribed discharge and the downstream stage and pool control Schemes are executed and the minimum of those two outputs (for mean discharge) is taken as the model output.

```
   RCL
      DO ENTERISC
      IF (SURCHARGE) THEN DO INDSRCHGE
         ELSE
            DO SETQ
            DO STPOOLQ
            DO SETMIN
         ENDIF
      ENDRCL
```

Example 4:

Inflow is normally passed unless the pool reaches a certain elevation or the instantaneous discharge exceeds the allowed maximum. If either of these conditions are met then the spillway Scheme is used.

```
   RCL
      DO PASSFLOW
      IF (POOL.GT.109.OR.QO.GT.MAXQ) THEN
         DO SPILLWAY
         ENDIF
      ENDRCL
```

Table 1. Scheme/Utility Identifiers

Identifier	Need RCL Statement	Description
ADJUST	no	Adjustment of model outputs using observed values Utility
BACKFLOW	no	Adjustment of inflow values using observed mean discharge and pool elevation Utility
ENTERISC	yes	Determine need for induced surcharge Scheme Utility
FILLSPILL	yes	Fill and spill Scheme
FLASHBDS	yes	Flashboard Scheme
GOFLASH	yes	Determine need for flash boards Scheme Utility
INDSRCHGE	yes	Induced surcharge Scheme
MINQ	yes	Minimized discharge Scheme
MAXQ	no 1/	Compute maximum discharge at given elevation Utility
PASSFLOW	yes	Pass inflow Scheme
POOLQ	yes	Elevation versus discharge Scheme
POWERGEN	yes	Power generation Scheme
RAINEVAP	no	Rainfall and evaporation on reservoir surface Utility
RULEADJ	no	Rule curve adjustment Utility
RULECURVE	yes	Rulecurve Scheme
SETDH	yes	Daily rate of change of pool elevation Scheme
SETDQ	yes	Daily rate of change of reservoir release Scheme
SETQ	yes	Prescribed discharge Scheme
SETH	yes	Prescribed elevation Scheme
SETMIN	yes	Select minimum quantity of model outputs Utility
SETMAX	yes	Select maximum quantity of model outputs

Identifier	Need RCL Statement	Description
		Utility
SPILLWAY	yes	Spillway Scheme
STPOOLQ	yes	Downstream state and pool elevation controlled discharge Scheme
SUMINF	no	Inflow summation Utility

Note:

1/ MAXQ is the only Scheme/Utility activated by an RCL statement other than a DO statement. It is used for comparisons in an IF statement.