PROGRAM/CODING SPECIFICATION FOR THE AFICCS REPORT GENERATION (REPGEN) CAPABILITY

JANUARY 1969

N. B. Sutherland

Prepared for

DIRECTOR OF PLANNING AND TECHNOLOGY
ELECTRONIC SYSTEMS DIVISION
AIR FORCE SYSTEMS COMMAND
UNITED STATES AIR FORCE
L. G. Hanscom Field, Bedford, Massachusetts

This document has been approved for public release and sale; its distribution is unlimited.
When U.S. Government drawings, specifications, or other data are used for any purpose other than a definitely related government procurement operation, the government thereby incurs no responsibility nor any obligation whatsoever, and the fact that the government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data is not to be regarded by implication or otherwise, as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

Do not return this copy. Retain or destroy.
PROGRAM/CODING SPECIFICATION FOR THE AFICCS REPORT GENERATION (REPGEN) CAPABILITY

JANUARY 1969

N. B. Sutherland

Prepared for

DIRECTOR OF PLANNING AND TECHNOLOGY
ELECTRONIC SYSTEMS DIVISION
AIR FORCE SYSTEMS COMMAND
UNITED STATES AIR FORCE.
L. G. Hanscom Field, Bedford, Massachusetts

This document has been approved for public release and sale; its distribution is unlimited.

Project 512V
Prepared by
THE MITRE CORPORATION
Bedford, Massachusetts
Contract AF19(625)-5165
FOREWORD

The work reported in this document was performed by The MITRE Corporation, Bedford, Massachusetts, for the Director of Planning and Technology, Electronic Systems Division, of the Air Force Systems Command under Contract AF 19(628)-5165.

REVIEW AND APPROVAL

This technical report has been reviewed and is approved.

WILLIAM F. HEISLER, COL, USAF
Chief, Command Systems Division
Directorate of Planning & Technology
ABSTRACT

The Report Generation (REPGEN) capability comprises a set of programs designed to run on an IBM 1410 computer under the Air Force Integrated Command and Control System (AFICCS). The primary purpose of REPGEN is to provide a capability for producing a printed report from a file of input data contained on punched cards, magnetic tape or on the system disk. This document contains detailed information regarding the structure and operation of the REPGEN programs, including table descriptions, core maps, program flow-charts and abstracts.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>LIST OF ILLUSTRATIONS</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>v</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>LIST OF TABLES</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>vi</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>INTRODUCTION</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>vi</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>OVERVIEW</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>

- **Table Generation**
  - LID (Load Input Descriptors) 2
  - SAVER 2
  - LEGVAL (Legitimate Value Loader) 4
  - LOD (Load Output Descriptors) 4

- **File Generation**
  - Report Generation 4
  - Report Production 5

<table>
<thead>
<tr>
<th>TABLES</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td></td>
</tr>
</tbody>
</table>

- Report Descriptor Table 6
- Record Descriptor Table 7
- Attribute Descriptor Table 8
- Selected Output Table (SOT) 9
- Output Entry Descriptor Table (OEDT) 10
- Legitimate Value Attribute Name Table 13
- Legitimate Value Table 14
- Condensed Output Program 16
- Saved Statement Table Directory 18
- Saved Statement Table 19
- I/O Communication Record 21

<table>
<thead>
<tr>
<th>OUTPUT PHASE COMMUNICATIONS ZONE</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>22</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SYMBOLIC DISK ADDRESSES</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>26</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>REPGEN FLOWCHARTS AND ABSTRACTS</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>28</td>
<td></td>
</tr>
</tbody>
</table>

- LID 28
- LEGVAL 43
- SAVER 50
- LOD 53
- REPGEN 89
- RGPBYP 103
- RGPOUT 107
- LNINT 110
## TABLE OF CONTENTS (CONCLUDED)

<table>
<thead>
<tr>
<th>Command</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>UPSTAT</td>
<td>113</td>
</tr>
<tr>
<td>PRINTO</td>
<td>116</td>
</tr>
<tr>
<td>CARDO</td>
<td>118</td>
</tr>
<tr>
<td>TAPEO</td>
<td>121</td>
</tr>
<tr>
<td>REALO</td>
<td>125</td>
</tr>
<tr>
<td>REMARK</td>
<td>146</td>
</tr>
<tr>
<td>REPEET</td>
<td>149</td>
</tr>
<tr>
<td>REKALL</td>
<td>151</td>
</tr>
<tr>
<td>DELEET</td>
<td>153</td>
</tr>
<tr>
<td>PRTC RD</td>
<td>155</td>
</tr>
<tr>
<td>PRTTAP</td>
<td>157</td>
</tr>
<tr>
<td>RPTCRD</td>
<td>159</td>
</tr>
</tbody>
</table>
## LIST OF ILLUSTRATIONS

<table>
<thead>
<tr>
<th>Figure Number</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Overview of the Report Generation Process</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>Tables Associated With One Report</td>
<td>11</td>
</tr>
<tr>
<td>3</td>
<td>Descriptor Table Entry</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>LLD Input and Output, Mode 1</td>
<td>12</td>
</tr>
<tr>
<td>5</td>
<td>LEGVAL Input and Output</td>
<td>15</td>
</tr>
<tr>
<td>6</td>
<td>LOD Input and Output</td>
<td>17</td>
</tr>
<tr>
<td>7</td>
<td>SAVER Input and Output</td>
<td>20</td>
</tr>
<tr>
<td>8</td>
<td>REPGEN Output Phase Core Map</td>
<td>25</td>
</tr>
</tbody>
</table>
SECTION I

INTRODUCTION

The purpose of this document is to present detailed information regarding the structure and operation of the various programs which comprise the REPGEN capability. It is assumed that the reader is familiar with the contents of the REPGEN User's Guide.
The Report Generation (REPGEN) capability comprises a set of programs designed to run on an IBM 1410 computer under the standard AFICCS system. The primary purpose of REPGEN is to provide a capability for producing a printed report from a file of input data contained on punched cards, on magnetic tape or on the system disk. The output may be optionally on magnetic tape or cards. The output format is controlled by use of a quasi-English language of limited vocabulary.

There are three major activities required in order to move from source data to finished report. These are:

a. Table generation
b. File generation
c. Report production

Figure 1 shows the basic elements involved in each of these activities.

TABLE GENERATION

Before a report can be generated, certain basic information must be loaded into tables and placed on the system disk, to be available for File generation and Report production. The total REPGEN capability includes a number of support programs which are used to perform the necessary table construction.

LID (Load Input Descriptors)

The input to LID is primarily a description of the source data which will be used to make up the serial file tape (SF tape) which in turn will be the input to the output phase of REPGEN. The various source data record types must be identified, and the data attributes associated with each record type must be defined in terms of length, position within the record, data type, etc.

SAVER

In some cases, the source data may reside on the AFICCS system disk in parallel file form. In order to produce a serial file tape (SF Tape) which can be used by REPGEN output programs, a Query Language retrieval must be performed against the parallel file. The Query Language retrieval statement must specify GETTAC as the process director. GETTAC (a variant of GETB) outputs the retrieved data on an SF Tape.
Figure 1. Overview of the Report Generation Process
SAVER's primary function is to save a Query Language retrieval statement so that it may be called up for execution when required. (SAVER is a variant of SAVE.)

LEGVAL (Legitimate Value Loader)

As information is transferred from source data to the serial file tape, it may be desirable to check certain attributes for legality. This best applies to attributes which can legally take on one of a limited number of values.

LEGVAL produces a table which contains all of the allowable values for each attribute which is to be legality checked.

LOD (Load Output Descriptors)

A quasi-English language of limited vocabulary has been implemented to give the user the capability to structure a report. Using the language provided, data can be selected from the serial file tape, arithmetically manipulated, and arranged for presentation on a printed page. (Output may optionally be to magnetic tape or punched cards.) The set of statements which direct this activity may be called an Output Program.

An output program is initially processed by LOD, which performs syntax checking, and then eliminates all redundant characters in each statement. LOD also replaces attribute names with their length and position on the serial file tape. The resulting condensed output program is placed in a table on the system disk, to be available for later execution.

FILE GENERATION

The purpose of the file generation activity is to produce a serial file tape containing the data from which the desired report(s) will be generated. The source data from which the serial file tape is made may be in punched cards, on magnetic tape, or may reside on the system disk in parallel file form.

When the source data is in cards or on tape, we are said to be operating in Mode 1; when the source data is a parallel file on disk, we are operating in Mode 2. There are internal differences in the operation of the file generation program (REPGEN), depending upon the mode in which it runs. The end result, however, is essentially the same; a serial file tape is produced which contains the data to be used in generating the desired report(s).
REPORT PRODUCTION

Once the required tables have been generated and a serial file tape has been prepared, we are ready to produce a report. This is accomplished by operating what is called the Output Phase.

The output phase can proceed as a continuation of a file generation run; at the completion of file generation the computer operator is given the option to terminate or to continue with the output phase.

A serial file tape which was produced by a previous run can be used to generate a report by operating the output phase via the Bypass Mode. In this mode, file generation is bypassed and the output phase is immediately started.

During the output phase, the condensed output program is read from disk and the statements are interpreted and executed to produce the desired output. Up to nine additional copies of a printed report may be produced automatically.
REPORT DESCRIPTOR TABLE

Created By: LID
Purpose: Each entry in this table identifies by name and number an active REPGEN report.
Table Size: 1 record (expandable) of 2800 characters.
Entry Size: 100 characters per entry, recorded in Move mode.
End of Table: A record mark in the first position of an entry indicates the end of the table.

Entry Description:

<table>
<thead>
<tr>
<th>Positions</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-3</td>
<td>Report Number</td>
</tr>
<tr>
<td>4-15</td>
<td>Report Title</td>
</tr>
<tr>
<td>16</td>
<td>Input Media</td>
</tr>
<tr>
<td>17</td>
<td>Classification</td>
</tr>
<tr>
<td>18-47</td>
<td>Header Label</td>
</tr>
<tr>
<td>48-49</td>
<td>Size of Header Label</td>
</tr>
<tr>
<td>50-52</td>
<td>Low Order Position of Header Label</td>
</tr>
<tr>
<td>53-68</td>
<td>Blank</td>
</tr>
<tr>
<td>69</td>
<td>Card Indicator</td>
</tr>
<tr>
<td>70</td>
<td>Tape Indicator</td>
</tr>
<tr>
<td>71</td>
<td>Printer Indicator</td>
</tr>
<tr>
<td>72</td>
<td>Not Used</td>
</tr>
<tr>
<td>73</td>
<td>History Indicator</td>
</tr>
<tr>
<td>74-76</td>
<td>Save Statement Number</td>
</tr>
<tr>
<td>77-80</td>
<td>Low Order Position of OEDT</td>
</tr>
<tr>
<td>81-84</td>
<td>SDA of Selected Output Table</td>
</tr>
<tr>
<td>85-88</td>
<td>SDA of Record Descriptor Table</td>
</tr>
<tr>
<td>89-92</td>
<td>Number Records for Record Desc. Table</td>
</tr>
<tr>
<td>93-96</td>
<td>SDA of Output Entry Desc. Table</td>
</tr>
<tr>
<td>97-100</td>
<td>Number Records for OEDT</td>
</tr>
</tbody>
</table>
RECORD DESCRIPTOR TABLE

Created By: LID

Purpose: Each entry describes a particular type of input record which may be encountered during generation of a Serial File Tape.

Table Size: 1 record of 2800 characters allocated for each entry in the Report Descriptor Table.

Entry Size: 40 characters per entry, recorded in Move mode.

End of Table: A record mark in the first position of an entry indicates the end of the table.

Entry Description:

<table>
<thead>
<tr>
<th>Positions</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-5</td>
<td>Record Type</td>
</tr>
<tr>
<td>6-8</td>
<td>Low Order Position of Record Type</td>
</tr>
<tr>
<td>9</td>
<td>Length of Record Type</td>
</tr>
<tr>
<td>10</td>
<td>Input Media</td>
</tr>
<tr>
<td>11-12</td>
<td>Maximum Number Records per Entry</td>
</tr>
<tr>
<td>13</td>
<td>Number of Control Attributes</td>
</tr>
<tr>
<td>14</td>
<td>Number of Sub-Control Attributes</td>
</tr>
<tr>
<td>15-17</td>
<td>High Order Position of Remark Text</td>
</tr>
<tr>
<td>18-20</td>
<td>Low Order Position of Remark Text</td>
</tr>
<tr>
<td>21-32</td>
<td>Blank</td>
</tr>
<tr>
<td>33-36</td>
<td>SDA of Attribute Desc Table</td>
</tr>
<tr>
<td>37-40</td>
<td>Number Records for Attribute Desc Table</td>
</tr>
</tbody>
</table>
ATTRIBUTE DESCRIPTOR TABLE

Created By: LID

Purpose: Each Attribute Descriptor Table describes all of the attributes associated with a particular record type.

Table Size: 1 or more 2800 character records (sufficient to describe all attributes pertaining to a given record type).

Entry Size: 40 characters per entry, recorded in Move mode.

End of Table: A record mark in the first position of an entry indicates the end of the table.

Entry Description:

<table>
<thead>
<tr>
<th>Positions</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-17</td>
<td>Attribute Name</td>
</tr>
<tr>
<td>18-19</td>
<td>Length of Attribute</td>
</tr>
<tr>
<td>20-22</td>
<td>Low Order Position of Attribute</td>
</tr>
<tr>
<td>23</td>
<td>Data Class</td>
</tr>
<tr>
<td>24</td>
<td>Audit Check Indicator</td>
</tr>
<tr>
<td>25</td>
<td>Mandatory Value Indicator</td>
</tr>
<tr>
<td>26-30</td>
<td>Record Type</td>
</tr>
<tr>
<td>31</td>
<td>Type of Attribute</td>
</tr>
<tr>
<td>32-36</td>
<td>Blank</td>
</tr>
<tr>
<td>37-39</td>
<td>Low Order Position of the Attribute Value in the Output Entry File</td>
</tr>
<tr>
<td>40</td>
<td>Not Used</td>
</tr>
</tbody>
</table>
SELECTED OUTPUT TABLE (SOT)

Created By: LID. The space is allocated by LID, but table entries are actually loaded by LOD.

Purpose: Each SOT table acts as a directory for up to seven output programs which may be associated with a given report number.

Table Size: 1 record of 245 characters (only the first 210 characters are actually used) is allocated for each entry in the Report Descriptor Table.

Entry Size: 30 characters per entry, recorded in Move mode.

End of Table: A record mark in the first position of an entry indicates the end of the table.

Entry Description:

<table>
<thead>
<tr>
<th>Positions</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Entry Indicator</td>
</tr>
<tr>
<td>2-5</td>
<td>SDA of Condensed Output Format Statements</td>
</tr>
<tr>
<td>6</td>
<td>Output Media</td>
</tr>
<tr>
<td>7-10</td>
<td>Output Entry Size for this Report</td>
</tr>
<tr>
<td>11</td>
<td>History Tape Indicator</td>
</tr>
<tr>
<td>12-14</td>
<td>Highest Statement Number for this Output Program</td>
</tr>
<tr>
<td>15-16</td>
<td>Total Output Statements Fitting into Core with this Format</td>
</tr>
<tr>
<td>17-21</td>
<td>Core Origin of Output Statements for this Output Format</td>
</tr>
<tr>
<td>22-26</td>
<td>Core Origin of Input Area for this Output Format</td>
</tr>
<tr>
<td>27</td>
<td>Usually Blank. Set to 1 by UPSTAT to indicate that the condensed statements to which this SOT entry points have been fully condensed by UPSTAT.</td>
</tr>
<tr>
<td>28-30</td>
<td>Blank</td>
</tr>
</tbody>
</table>
OUTPUT ENTRY DESCRIPTOR TABLE (OEDT)

Created By:  LID

Purpose:     This table describes the format of the records on the Serial File Tape.

Table Size:  1 record of 2800 characters.

Entry Size:  25 characters per entry, written in Move mode.

End of Table: A record mark in the first position of an entry indicates the end of the table.

Entry Description:

<table>
<thead>
<tr>
<th>Positions</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-17</td>
<td>Attribute Name</td>
</tr>
<tr>
<td>18</td>
<td>&quot;Zero&quot; inserted by LID</td>
</tr>
<tr>
<td>19-20</td>
<td>Length of Attribute</td>
</tr>
<tr>
<td>21-23</td>
<td>Low Order Position of Attribute</td>
</tr>
<tr>
<td>24-25</td>
<td>Blank - Not Used</td>
</tr>
</tbody>
</table>
Figure 2. Tables Associated with One Report Descriptor Table Entry
Figure 3. LID Input and Output, Mode 1
LEGITIMATE VALUE ATTRIBUTE NAME TABLE

Created By:  LEGVAL

Purpose:  This table contains the name of every attribute to be checked for legality during file generation, and acts as a directory for the lists of legal values.

Table Size:  10 records of 245 characters each.

Entry Size:  24 characters per entry, recorded in Move mode.

End of Table: A record mark in the first position of an entry indicates the end of the table.

Entry Description:

<table>
<thead>
<tr>
<th>Positions</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-17</td>
<td>Attribute Name</td>
</tr>
<tr>
<td>18-21</td>
<td>SDA of Legitimate Values for this Attribute.</td>
</tr>
<tr>
<td>22</td>
<td>Number of records assigned to above SDA.</td>
</tr>
<tr>
<td>23-24</td>
<td>Maximum length of attribute value.</td>
</tr>
</tbody>
</table>
LEGITIMATE VALUE TABLE

Created By: LEGVAL

Purpose: Each Legitimate Value Table contains the list of all allowable values for one particular attribute.

Table Size: 4 records of 245 characters each.

Entry Size: Variable. The entries in a given Legitimate Value Table are as long as the maximum length of that particular attribute value.

End of Table: A record mark in the first position of an entry indicates the end of the table.
Figure 4. LEGVAL Input and Output
CONDENSED OUTPUT PROGRAM

Created By: LOD

Purpose: Each table contains one Output Program, processed and condensed by LOD and available for execution.

Table Size: The table contains one 245 character record for each statement of the Output Program (only the first 200 characters of each record are actually available for use).

Entry Size: Each entry (condensed statement) is contained in a 245 character record written in Move mode.

End of Table: Each condensed statement is assigned within an SDA to the relative record corresponding to the statement number. Thus, the last entry in the table is contained in the relative record corresponding to the highest statement number used in the Output Program.

Entry Description:

See program description of LOD.
Figure 5. LOD Input and Output
SAVED STATEMENT TABLE DIRECTORY

Created By:   SAVER

Purpose:     This table serves as a directory for locating the Query Language statements which have been stored on the system disk by SAVER.

Table Size:   1 record of 2800 characters (only the first 2000 are used).

Entry Size:   Each entry is 4 characters in length.

End of Table: The end of the table is indicated by the 500th entry or a blank entry, whichever occurs first.

Entry Description:

Each entry is an SDA of a saved Query Language statement. The relative position of the entry in the table corresponds to the statement number previously assigned to the saved statement.
SAVED STATEMENT TABLE

Created By:  SAVER

Purpose: Query Language retrieval statements are stored on the system disk for subsequent recall and execution.

Table Size: A maximum of 500 2800 character records.

Entry Size: Each saved statement is contained in one 2800 character record.

Entry Description:

See program description for SAVER.
Figure 6. SAVER Input and Output
I/O COMMUNICATION RECORD

Created By: REPGEN (or by RGPBYP when using bypass mode).

Purpose: This record contains information which must be passed from the input phase to the output phase.

Table Size: 1 record of 245 characters.

Entry Size: 1 entry of 126 characters.

Entry Description:

<table>
<thead>
<tr>
<th>Positions</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-100</td>
<td>Report Descriptor Table entry for report being processed.</td>
</tr>
<tr>
<td>101-117</td>
<td>AS OF TIME entered on console</td>
</tr>
<tr>
<td>118-120</td>
<td>High order position of Remarks text</td>
</tr>
<tr>
<td>121-123</td>
<td>Low order position of Remarks text</td>
</tr>
<tr>
<td>124</td>
<td>Format number of requested Output Program</td>
</tr>
<tr>
<td>125</td>
<td>Classification code of requested report</td>
</tr>
<tr>
<td>126</td>
<td>Number of additional copies requested</td>
</tr>
</tbody>
</table>
## SECTION IV

### OUTPUT PHASE COMMUNICATION ZONE

<table>
<thead>
<tr>
<th>Name</th>
<th>Loc</th>
<th>Length</th>
<th>Tag</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>WAREA</td>
<td>2500</td>
<td>200</td>
<td>High</td>
<td>Area from which a condensed statement is executed.</td>
</tr>
<tr>
<td>ENTIND</td>
<td>2700</td>
<td>1</td>
<td>Low</td>
<td>Indicator for entry from SOT to be used. A or C = active; D or E or Ń = inactive.</td>
</tr>
<tr>
<td>ENDSDA</td>
<td>2704</td>
<td>4</td>
<td>Low</td>
<td>SDA of condensed statements for this Output Program.</td>
</tr>
<tr>
<td>ENTMED</td>
<td>2705</td>
<td>1</td>
<td>Low</td>
<td>Output device for this Output Program. C = card punch, T = tape, P = printer.</td>
</tr>
<tr>
<td>ENTSIZ</td>
<td>2709</td>
<td>4</td>
<td>Low</td>
<td>Size of the output image for this Output Program.</td>
</tr>
<tr>
<td>ENTHIS</td>
<td>2710</td>
<td>1</td>
<td>Low</td>
<td>History tape indicator. Y = yes, Ń = no.</td>
</tr>
<tr>
<td>ENTHIH</td>
<td>2713</td>
<td>3</td>
<td>Low</td>
<td>Highest statement number for this condensed Output Program.</td>
</tr>
<tr>
<td>ENTAMT</td>
<td>2715</td>
<td>2</td>
<td>Low</td>
<td>Number of condensed statements that can fit in core for this Output Program.</td>
</tr>
<tr>
<td>ENTSTA</td>
<td>2720</td>
<td>5</td>
<td>Low</td>
<td>Core Origin of the condensed statements for this Output Program.</td>
</tr>
<tr>
<td>ENTPIN</td>
<td>2725</td>
<td>5</td>
<td>Low</td>
<td>Core origin of the input area for this Output Program.</td>
</tr>
<tr>
<td>ENTRY</td>
<td>2729</td>
<td>4</td>
<td>Low</td>
<td>The low-order character of the SOT entry in use.</td>
</tr>
<tr>
<td>SOT</td>
<td>2730</td>
<td>245</td>
<td>High</td>
<td>Selected Output Table for this report.</td>
</tr>
<tr>
<td>OVFLOW</td>
<td>2976</td>
<td>2</td>
<td>Low</td>
<td>The line after which overflow is to occur.</td>
</tr>
<tr>
<td>COUNTR</td>
<td>2977</td>
<td>51 x 17</td>
<td>High</td>
<td>The 50 Counters available to an Output Program, plus the REMAINDER Counter (Counter 0).</td>
</tr>
<tr>
<td>Name</td>
<td>Loc</td>
<td>Length</td>
<td>Tag</td>
<td>Meaning</td>
</tr>
<tr>
<td>-------------</td>
<td>-----</td>
<td>--------</td>
<td>-----</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>ASFDAT</td>
<td>3860</td>
<td>17</td>
<td>Low</td>
<td>Location where AS OF TIME from console is stored.</td>
</tr>
<tr>
<td>DATE</td>
<td>3877</td>
<td>17</td>
<td>Low</td>
<td>Location where system date is stored (referred to by system attribute name TODAY).</td>
</tr>
<tr>
<td>CLASS</td>
<td>3894</td>
<td>17</td>
<td>Low</td>
<td>Report classification.</td>
</tr>
<tr>
<td>SWITCH</td>
<td>3895</td>
<td>10</td>
<td>High</td>
<td>The 9 Switches available to an Output Program.</td>
</tr>
<tr>
<td>LNCTER</td>
<td>3906</td>
<td>2</td>
<td>Low</td>
<td>Line counter.</td>
</tr>
<tr>
<td>COMINO</td>
<td>3909</td>
<td>3</td>
<td>Low</td>
<td>The statement number of the first level statement in operation.</td>
</tr>
<tr>
<td>COM1PO</td>
<td>3914</td>
<td>5</td>
<td>Low</td>
<td>The position in the first level statement in operation.</td>
</tr>
<tr>
<td>COM1AC</td>
<td>3915</td>
<td>1</td>
<td>Low</td>
<td>First level active indicator; 1 = active; Ø = inactive.</td>
</tr>
<tr>
<td>COM2NO</td>
<td>3918</td>
<td>3</td>
<td>Low</td>
<td>The statement number of the second level statement in operation.</td>
</tr>
<tr>
<td>COM2PO</td>
<td>3923</td>
<td>5</td>
<td>Low</td>
<td>The position in the second level phrase in operation.</td>
</tr>
<tr>
<td>COM2AC</td>
<td>3924</td>
<td>1</td>
<td>Low</td>
<td>Second level active indicator; 1 = active; Ø = inactive.</td>
</tr>
<tr>
<td>COMREP</td>
<td>3927</td>
<td>3</td>
<td>Low</td>
<td>The statement number of the REPEAT statement in operation.</td>
</tr>
<tr>
<td>COMSOR</td>
<td>3932</td>
<td>5</td>
<td>Low</td>
<td>The core origin of the condensed statements.</td>
</tr>
<tr>
<td>COMSNO</td>
<td>3934</td>
<td>2</td>
<td>Low</td>
<td>The number of statements in core.</td>
</tr>
<tr>
<td>COMIOR</td>
<td>3939</td>
<td>5</td>
<td>Low</td>
<td>The core origin of the input area.</td>
</tr>
<tr>
<td>COMIEN</td>
<td>3944</td>
<td>5</td>
<td>Low</td>
<td>The end of the input area in core.</td>
</tr>
<tr>
<td>COMOOR</td>
<td>3949</td>
<td>5</td>
<td>Low</td>
<td>The core origin of the output area.</td>
</tr>
<tr>
<td>Name</td>
<td>Loc</td>
<td>Length</td>
<td>Tag</td>
<td>Meaning</td>
</tr>
<tr>
<td>-----------</td>
<td>-----</td>
<td>--------</td>
<td>-----</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>COMOEN</td>
<td>3954</td>
<td>5</td>
<td>Low</td>
<td>The end of the output area in core.</td>
</tr>
<tr>
<td>COMTES</td>
<td>3955</td>
<td>1</td>
<td>Low</td>
<td>The result of the last IF test; 1 = true; 0 = false.</td>
</tr>
<tr>
<td>COMPUN</td>
<td>3956</td>
<td>1</td>
<td>Low</td>
<td>The punctuation from the last phrase passed.</td>
</tr>
<tr>
<td>COMDON</td>
<td>3957</td>
<td>1</td>
<td>Low</td>
<td>Finished indicator; 1 = no more input data.</td>
</tr>
<tr>
<td>COMSTK</td>
<td>3958</td>
<td>1</td>
<td>Low</td>
<td>The stacker into which the next card to be punched will be placed.</td>
</tr>
<tr>
<td>COMQL</td>
<td>3959</td>
<td>1</td>
<td>Low</td>
<td>Query Language indicator. Non blank = Query Language.</td>
</tr>
<tr>
<td>COMTYP</td>
<td>3960</td>
<td>1</td>
<td>Low</td>
<td>The statement type of the last statement operated.</td>
</tr>
<tr>
<td>COMDR</td>
<td>3961</td>
<td>1</td>
<td>Low</td>
<td>Type of first-level statement last executed; D = DO; R = REPEAT.</td>
</tr>
<tr>
<td>COMOVF</td>
<td>3962</td>
<td>1</td>
<td>Low</td>
<td>Overflow indicator.</td>
</tr>
<tr>
<td>HILO</td>
<td>3968</td>
<td>6</td>
<td>Low</td>
<td>The high and low order addresses of remarks data on remarks card.</td>
</tr>
<tr>
<td>COMNEW</td>
<td>3969</td>
<td>1</td>
<td>Low</td>
<td>Indicator that a new statement has become active.</td>
</tr>
<tr>
<td>COPIES</td>
<td>3970</td>
<td>1</td>
<td>Low</td>
<td>Number of additional copies of the report requested.</td>
</tr>
<tr>
<td>NEWPAG</td>
<td>3971</td>
<td>1</td>
<td>Low</td>
<td>Indicator that a SKIP TO NEW PAGE was executed.</td>
</tr>
<tr>
<td>SPACES</td>
<td>3972</td>
<td>2</td>
<td>Low</td>
<td>Number of blank lines printed by the last SKIP statement.</td>
</tr>
<tr>
<td>WRTSCT</td>
<td>3975</td>
<td>2</td>
<td>Low</td>
<td>Not used.</td>
</tr>
</tbody>
</table>
Figure 7. REPGEN Output Phase Core Map

(MODE 2 ONLY)

<table>
<thead>
<tr>
<th>5000</th>
<th>5000</th>
<th>5000</th>
<th>5000</th>
</tr>
</thead>
<tbody>
<tr>
<td>RGOUT</td>
<td>RGOUT</td>
<td>RGOUT</td>
<td>RGOUT</td>
</tr>
<tr>
<td>1500</td>
<td>1500</td>
<td>1500</td>
<td>1500</td>
</tr>
<tr>
<td>RWTAPE</td>
<td>RWTAPE</td>
<td>RWTAPE</td>
<td>RWTAPE</td>
</tr>
<tr>
<td>2500</td>
<td>2500</td>
<td>2500</td>
<td>2500</td>
</tr>
<tr>
<td>COMM ZONE</td>
<td>COMM ZONE</td>
<td>COMM ZONE</td>
<td>COMM ZONE</td>
</tr>
<tr>
<td>4100</td>
<td>4100</td>
<td>4100</td>
<td>4100</td>
</tr>
<tr>
<td>CONSOLE</td>
<td>CONSOLE</td>
<td>CONSOLE</td>
<td>CONSOLE</td>
</tr>
<tr>
<td>4300</td>
<td>4300</td>
<td>4300</td>
<td>4300</td>
</tr>
<tr>
<td>INIT</td>
<td>UPSTAT</td>
<td>TAPEO, CARDO OR PRINTO</td>
<td></td>
</tr>
<tr>
<td>6000</td>
<td>6000</td>
<td>6000</td>
<td>6000</td>
</tr>
<tr>
<td>BLANK</td>
<td>BLANK</td>
<td>BLANK</td>
<td>BLANK</td>
</tr>
<tr>
<td>6600</td>
<td>6600</td>
<td>6600</td>
<td>6600</td>
</tr>
<tr>
<td>REALO (CTL)</td>
<td>REALO (CTL)</td>
<td>REALO (CTL)</td>
<td>REALO (CTL)</td>
</tr>
<tr>
<td>15000</td>
<td>15000</td>
<td>15000</td>
<td>15000</td>
</tr>
<tr>
<td>BLANK</td>
<td>BLANK</td>
<td>BLANK</td>
<td>BLANK</td>
</tr>
<tr>
<td>32500</td>
<td>32500</td>
<td>32500</td>
<td>32500</td>
</tr>
<tr>
<td>SYSTEM</td>
<td>SYSTEM</td>
<td>SYSTEM</td>
<td>SYSTEM</td>
</tr>
<tr>
<td>40000</td>
<td>40000</td>
<td>40000</td>
<td>40000</td>
</tr>
</tbody>
</table>
SECTION V
SYMBOLIC DISK ADDRESSES

The REPGEN programs make frequent use of the system disk. All disk references are made through the use of DSKACC, using Symbolic Disk Addresses (SDA's). Several programs in REPGEN use the system allocation routine DSKALL to dynamically obtain needed SDA's. However, there are a number of SDA's which must be obtained prior to installing the REPGEN capability. These SDA's are assembled directly into some of the REPGEN programs.

Table 1 which follows describes the characteristics of the disk areas for which SDA's must be obtained prior to installing REPGEN.
<table>
<thead>
<tr>
<th>SDA Explanation</th>
<th>Mode Code</th>
<th># of Records</th>
<th># Char. Per Rec.</th>
<th>LID</th>
<th>LOD</th>
<th>LVAL</th>
<th>LIDT</th>
<th>TAREP</th>
<th>REPGEN</th>
<th>RGIP</th>
<th>RGIPV</th>
<th>RGIPU</th>
<th>REALO</th>
<th>CARDO</th>
<th>TAPED</th>
<th>PRINTO</th>
<th>INIT</th>
<th>UPSTAT</th>
<th>REMARK</th>
<th>CHKLOG</th>
<th>SAVX</th>
<th>REKALL</th>
<th>DELET</th>
<th>REPOST</th>
</tr>
</thead>
<tbody>
<tr>
<td>General Tape Descriptor Table</td>
<td>3/01</td>
<td>2165</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Typewriter Input Area</td>
<td>4/1440</td>
<td>2800</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Current Date</td>
<td>8/01</td>
<td>245</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scratch Area</td>
<td>4/8429</td>
<td>2800</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>I/O Communication Record</td>
<td>8/01</td>
<td>245</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LEGVAL Attribute Name Table</td>
<td>8/10</td>
<td>245</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Report Descriptor Table</td>
<td>4/01</td>
<td>2800</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Communications Zone</td>
<td>5/04</td>
<td>400</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CARDO Routine</td>
<td>3/01</td>
<td>2165</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TAPEO Routine</td>
<td>5/04</td>
<td>400</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PRINTO Routine</td>
<td>5/05</td>
<td>400</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>REALO Routine</td>
<td>3/04</td>
<td>2165</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INIT Routine</td>
<td>5/04</td>
<td>400</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UPSTAT Routine</td>
<td>5/06</td>
<td>400</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>REMARK Routine</td>
<td>3/02</td>
<td>2165</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Temporary Header Storage</td>
<td>6/02</td>
<td>529</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>History Tape Overlay Area</td>
<td>5/05</td>
<td>400</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Table of Saved Statement SDA's</td>
<td>4/01</td>
<td>2800</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**TABLE 1**

**SDA'S REQUIRED BY REPGEN PROGRAMS**

**USED BY**
SECTION VI
REPGEN FLOWCHARTS AND ABSTRACTS

6.1  **LID**

**Purpose**

LID reads a deck of input descriptor cards and generates a set of tables which are required by subsequent REPGEN programs.

**Calling Sequence**

Enter LID on the console typewriter. Return will always be to IDLOOP.

**Description**

There are four acceptable input card types to LID. They are the Report Descriptor Card, the Record Descriptor Card, the Attribute Descriptor Card and the END Card.

**Report Descriptor Card** -- When LID reads a Report Descriptor Card it does the following:

a. Checks the Report Descriptor Table to make sure there is an open slot or that this is not a duplicate of another report number.

b. If the input medium declared on this card is not disk, an SDA is allocated for a Record Descriptor Table.

c. An SDA is allocated for an SOT.

d. An SDA is allocated for an OEDT.

**Record Descriptor Card** -- When LID encounters a Record Descriptor Card it:

a. Allocates an SDA for an Attribute Descriptor Table.

**Attribute Descriptor Card** -- An Attribute Descriptor Card causes LID to:

a. Make an entry in the current Attribute Descriptor Table.

b. Read the next Attribute Descriptor Card and repeat the process until all attributes for this record type have been treated.

c. Make an entry in the Record Descriptor Table.
END Card (or next Report Descriptor Card)-- When an END Card is read, LID will:

a. Construct an Output Entry Descriptor Table (OEDT) to describe the format of each record as it will later appear on the Serial File Tape.

b. Modify each Attribute Descriptor Table entry to include data identifying the attribute's assigned position in the Serial File Tape records.

c. The modified Report Descriptor Table (and Record Descriptor Table if input is not disk) is written onto the system disk.
PRINT MESSAGE!
"REPORT CARD MISSING"

PRINT MESSAGE:
"RPT OR END CARD MUST FOLLOW RPT CARD WITH D IN COL. 18"
SET REPORT INDICATOR SWITCH ON

DELETE CARD
YES MOVE
NO

DISK INPUT
YES MOVE
NO

ADD CARD
YES MOVEA
NO

PRTCD
PRINT CARD

PRINT MESSAGE "ILLEGAL PROGRAM ACTION CODE"

FCIC

MOVEA

SET NOMIND ON

SAVE REPORT NUMBER

FIRTSW
YES FTSSET
NO

ERRSWT
YES FIXIT
NO

ORLIK

RPTDL2

FCIC

FCIC

MOVE

FIXIT

DEALLO
DE-ALLOCATE

TURN ON FIRTSW
PRINT MESSAGE
"RECORD DESCRIPTOR CARD MISSING"

TURN OFF FIRTSW ERSSWT

PRINT MESSAGE ABOUT BAD REPORT NUMBER

PRINT MESSAGE "RECORD DESCRIPTOR CARD MISSING"

PRINT CARD

PRINT MESSAGE "RECORD DESCRIPTOR CARD MISSING"

PRINT CARD

PRINT MESSAGE ABOUT BAD REPORT NUMBER
ON

(attribute descriptor card)

NO

FC4A

YES

RMKSW ON

YES

REMARK ERR

NO

ECHKAT

FCID

ATTDL

FCID

(ERROE RETURN)

(ERROE RETURN)

TURN OFF RECORD SWITCH

TURN ON ATTRIBUTE SWITCH

FCIB

NO

END CARD

YES

YENDCD

FC4A

END CARD

YES

YENDCD

NO

PRTCD

PRINT CARD

PRINT MESSAGE "INVALID CARD"

FCIA
LPRINT MESSAGE
REMARKS RECORD MAY HAVE NO ATTRIBUTES

TURN ON ERRSWT

EX Lid

RETURN TO CALLER
ATTDL

SCAN TO FIRST AVAILABLE SLOT IN ATTRIBUTE TABLE

MAKE ENTRY IN ATTRIBUTE TABLE

CONTROL ATTRIBUTE?

YES -> ADD ONE TO KCTL

NO -> SUB-CONTROL ATTRIBUTE?

YES -> ADD ONE TO KSUB

NO -> READ CARD

READ CARD

ATTRIBUTE CARD?

YES -> ECHKAT

ERROR CHECK ATTR. CARD

ERROR RETURN

NO -> WRITE ATTRIBUTE DESC. TABLE ONTO DISK

RCDDL

ADD AN ENTRY TO RECORD DESC. TABLE

CLEAR ATTRIBUTE BUFFER

PUT RECORD MARK AT FRONT OF ATTRIBUTE BUFFER

TURN ON ERROR SWITCH

KERX

TURN OFF ERROR SWITCH

ERROR SWITCH ON?

YES -> ERROR EXIT TO CALLER

NO -> NORMAL EXIT TO CALLER
READ ATTRIBUTE DESC. TABLE

END OF ATTR. DESC. TBL?

YES

KBNXTB

NO

STEP TO NEXT ENTRY

2ND PASS?

YES

DATA ATTRIBUTE?

NO

2ND PASS DONE?

NO

KBLI

YES

ARE WE AT END OF REC. DESC. TBL?

NO

STEP TO NEXT ENTRY

SUB CONTROL ATTRIBUTE?

NO

WRITE OUT EDT

CONTROL ATTRIBUTE?

YES

KBXTRK

NO

RETURN TO CALLER

STEP TO NEXT ENTRY

IS ATTRIBUTE IN EDT?

YES

ENTER ATTRIBUTE IN EDT

NO

MODIFY ATTRIBUTE TABLE ENTRY TO SHOW POSITION IN OUTPUT ENTRY

TURN ON 2ND PASS INDICATOR

KBLI
HLDSYM

ENTER SDA AT HLDSDA INTO SAVED SDA TABLE

RETURN TO CALLER

ORLIK

NOMIND SWITCH ON

ATTIND SWITCH ON

TURN ON FIRTSW

TURN OFF RECORD SWITCH

PRINT MESSAGE "ATTRIBUTE DESCRIPTOR CARDS MISSING"

DEALLO DEALLOCATE SDA'S

TURN OFF NOMIND

TURN OFF ATTIND

RETURN TO CALLER

KBPUTO

SUFFICIENT SPACE ON DISK FOR OEDT

WRITE OEDT TO DISK

CLEAR OEDT BUFFER

RETURN TO CALLER

DSKALL EXPAND OEDT AREA

NO
LEGVAL

Purpose

To generate a disk table containing all allowable values for one or more attributes. This table is to be used during the file generation process which creates a Serial File Tape from source input data.

Calling Sequence

Enter LEGVAL on the console typewriter.

Description

Each input card to LEGVAL contains the name of an attribute and one of the allowable values for that attribute. LEGVAL checks to see if that attribute name has been previously encountered. If not, the name is added to a list of attributes which are to be legality checked. After the name has been added to the name list, an SDA is allocated to be used for storing the list of legal values pertaining to the attribute. When all allowable values for a given attribute have been read, the value table is written on the system disk. When an END card is read, the name list is written on the system disk.
LEGVAL
LSTART

CLEAR TABLE AREAS

SET "NAME TABLE NOT IN CORE" SWITCH

SET WORD MARKS IN ATTRIBUTE DESC. CARD AREA

SAVE PREVIOUS ATTRIBUTE NAME

READ A CARD

IS THIS END CARD?

YES

NO

LLAST

END OF FILE

NO

LMCM2

NO

MESSAGE: MORE CARDS OR END CARD

YES

LREAD

DELETE

SET "NAME TABLE NOT IN CORE" SWITCH

CALL IN ATTRIBUTE NAME TABLE

IS "NAME TABLE NOT IN CORE" SWITCH SET?

YES

NO

IS NEW ATTRIBUTE NAME SAME AS OLD NAME?

YES

RESET "PRESENT ATTRIBUTE VALUE TABLE IN CORE" SWITCH

PUT OLD VALUE TABLE ONTO DISK

LMCM2

NO

MESSAGE: MORE CARDS OR END CARD

YES

LMCM2
SET TABLE POINTERS TO ZERO

HAS ATTRIBUTE NAME BEEN FOUND?

IS VALUE TABLE FOR THIS ATTRIBUTE ALREADY IN CORE?

CALL IN CORRECT ATTRIBUTE VALUE TABLE

WAS ATTRIBUTE VALUE FOUND?

INDICATE THAT THIS ATTRIBUTE VALUE DOESN'T EXIST

PUT BLANKS IN PLACE OF ATTRIBUTE VALUE

LREAD

LADD

IS PRESENT ATTRIBUTE VALUE TABLE IN CORE?

IS VALUE TABLE POINTER AT END OF TABLE?

LLOOP

IS ATTRIBUTE NAME IN TABLE?

IS THERE A SPACE FOR NEW ATTRIBUTE NAME?

SET "END OF NAME TABLE" SWITCH

INDICATE ATTRIBUTE NAME TABLE FULL

LINSRI

IDLOOP
ENTER NEW ATTRIBUTE NAME PARAMETERS INTO TABLE

ALLOCATE AN ATTRIBUTE VALUE TABLE FOR THIS NAME

SET "NEW VALUE TABLE" SWITCH

PUT VALUE SDA INTO NAME TABLE

IS THIS VALUE TABLE NEW?

SET "END OF ATTRIBUTE NAMES" INDICATOR INTO "NAME TABLE"

RESET "END OF ATTRIBUTE VALUE TABLE" SWITCH

CALL ATTRIBUTE VALUE TABLE

INITIALIZE VALUE TABLE
CLEAR "NEW VALUE TABLE" SWITCH

IS "NEW VALUE TABLE" SWITCH SET?

NO

LUPDAT

YES

LLOOI1

47
LLOOP!

IS THIS END OF VALUE TABLE?
YES

IS THIS A BLANK SPACE IN TABLE?
YES LMCWI
NO

SEE IF VALUE ALREADY IN TABLE?
YES LREAD
NO

ERROR MESSAGE "VALUE ALREADY IN TABLE"

IS VALUE TABLE FULL?
YES LMCWI

ERROR MESSAGE "VALUE TABLE FULL"

LREAD
PUT ATTRIBUTE VALUE INTO TABLE

LREAD

NO

IS END OF VALUE TABLE SWITCH SET?

YES

PUT RECORD MARK AT END OF TABLE

LREAD

LMCWI

WRITE OUT ON PRINTER AND CONSOLE "END OF UPDATE"

LLAST

PUT ATTRIBUTE NAME TABLE BACK ON DISK

WAS THIS AN ELIMINATE ACTION?

YES

PUT CURRENT ATTRIBUTE VALUE TABLE ONTO DISK

NO

IDLOOP

18-26,192
SAVER

Purpose

A query language statement containing the GETTAC process director and report number(s) is stored on disk for utilization by the query portion of the report generation program.

Calling Sequence

From I/O typewriter, or cards with sense switch B on, starting in column 1.

SAVER&GETTAC&AAAA''AA[BBB/BBB'']
XXX+XXX+''+XXX

AA'''' = qualifying attributes
BB'''' = output selector
XXX = report numbers (maximum of 10)

Description

The saved statement SDA table is read from disk. The first blank entry is located and noted. The statement to be saved is accessed from the buffer SDA 'BBBB'. The statement is scanned for correctness, such as appearance of a left and right bracket and report numbers. The report numbers are stored in a small table area. The output selectors are stored in a table area 50 characters to each entry. The report descriptor file is now accessed. Report numbers are now matched from the report descriptor file with those of the saved statement. If a match is found the saved statement number is inserted in the appropriate slot of the report descriptor file. The output entry descriptor table SDA and number of records referenced by the report descriptor is saved in a table area. The new report descriptor is written back on system disk. The process is continued until all report numbers from the saved statement are exhausted.

The output selector table is now written on disk at the SDA's indicated by the matching reports discovered. Disk space is allocated for the saved statement and the statement is saved at the SDA allocated. The actual SDA for the statement is inserted in the SDA statement buffer and rewritten on the system disk. The program will terminate with a printout of the statement number assigned to the 'SAVER' statement.
START

READ IN STATEMENT SDA TABLE

SCAN FOR BLANK ENTRY IN TABLE

FIND A BLANK ENTRY?

YES

SAVE ASSIGNED STATEMENT NUMBER

READ IN "SAVER" STATEMENT

SCAN FOR LEFT BRACKET

FIND IT?

YES

SAVE LOCATION OF LEFT BRACKET

ERROR: TERMINATE

NO

PROCESS DIRECTOR GETTAC?

NO

ERROR: TERMINATE

YES

SAVE REPORT NUMBER (s) IN TABLE

SCAN 31

ERROR: TERMINATE

ANY FOUND?

NO

ERROR: TERMINATE

YES

LOOK FOR REPORT NUMBER (s)

FIND IT?

YES

ERROR: TERMINATE

NO
SET UP OUTPUT SELECTORS IN SELECTOR TABLE

READ IN REPORT DESC. TABLE

COMPARE REPORT #'s AGAINST FIRST (NEXT) ENTRY

MATCH ?

YES INSERT SAVED STATEMENT IN REPORT DESC. TABLE ENTRY

NO END OF REPORT DESC. TABLE ?

NO

PICK-UP OEDT SDA AND SAVE IT

YES

ANY REPORT #'s MATCHED ?

WRITE REPORT DESC. TABLE BACK ON DISK

NO

ALL REPORT #'s MATCHED ?

ERROR? TERMINATE

NO

WRITE OUTPUT SELECTORS IN OEDT

ALLOCATE SPACE FOR SAVER STATEMENT

WRITE STATEMENT TO DISK AREA JUST ALLOCATED

ADD SDA TO SAVED STATEMENT DIRECTORY

EXIT TO IDLOOP

TYPE MESSAGE "STATEMENT NNN HAS BEEN SAVED"
LOD

Purpose

LOD processes an Output Program, checking the output format statements syntactically, and condensing them into a convenient form for interpretation and execution by the output phase of REPGEN.

Calling Sequence

Enter LOD on the console typewriter. Return will be to IDLOOP.

Description

The Output Descriptor Card is read and Report Descriptor Table is called from disk. A search is instituted for the report indicated on the Output Descriptor Card.

When the report is found, the SOT Table for this report is called in from disk and updated with the current information from the Output Descriptor Card. An SDA is now allocated for the condensed output format statements.

The SOT Table is now put back on disk with the present run recorded as in error status. The condensed output statement SDA is put into the appropriate DSKACC call.

The LOD statement cards are read in next and are condensed by LOD. Each numbered statement is allotted a complete 245 character record within the assigned SDA. When an attribute name is encountered during processing of the statement cards, an Output Entry Descriptor Table (OEDT -- an expandable 2800 character record) for this report is brought into core from disk. The attribute name is replaced by a six digit value from the OEDT. This value specifies the attribute value length and its relative position in the intermediate tape created by the first phase of REPGEN.

The OEDT usually will remain in core throughout the processing of the present statement cards except when an additional segment of the OEDT has to be called in from disk. On the next reference to the OEDT the first segment of the table will be brought back into core.

When the final statement has been processed, the report descriptor table will be written back onto disk, after being submitted to a minor update. This update involves putting an indicator for output media into the table.

53
A second wind-up operation puts the updated SOT Table onto disk. A successful set of statements get an "A" or "C" in position 0 of the table. Otherwise an "E" is left in the table.

**Phrase Types and Condensed Form**

<table>
<thead>
<tr>
<th>Type 1</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>DO XXX</td>
<td>1DXXX</td>
</tr>
<tr>
<td>REPEAT XXX</td>
<td>1RXXX</td>
</tr>
<tr>
<td>GO TO XXX</td>
<td>1GXXX</td>
</tr>
<tr>
<td>CONTINUE XXX</td>
<td>1CXXX</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type 2</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>SET OVERFLOW AFTER NN</td>
<td>2SNN</td>
</tr>
<tr>
<td>SET SWITCH N</td>
<td>2SSN</td>
</tr>
<tr>
<td>SET COUNTER NN LENGTH=PP</td>
<td>2SCNPP</td>
</tr>
<tr>
<td>SET REMAINDER LENGTH=NN</td>
<td>2SCN0NN</td>
</tr>
<tr>
<td>CLEAR SWITCH N</td>
<td>2CSN</td>
</tr>
<tr>
<td>CLEAR SWITCHES</td>
<td>2CS</td>
</tr>
<tr>
<td>CLEAR COUNTER NN</td>
<td>2CCN</td>
</tr>
<tr>
<td>CLEAR COUNTERS</td>
<td>2CC</td>
</tr>
<tr>
<td>CLEAR</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type 3</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>SKIP TO A NEW PAGE</td>
<td>3E1</td>
</tr>
<tr>
<td>SKIP NN</td>
<td>3CNN</td>
</tr>
<tr>
<td>STACKER N</td>
<td>3SN</td>
</tr>
<tr>
<td>WRITE TAPE MARK</td>
<td>3T</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type 4</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>END LINE</td>
<td>4Ø</td>
</tr>
<tr>
<td>END CARD</td>
<td>4Ø</td>
</tr>
<tr>
<td>END HEADER XXX</td>
<td>4HXXX</td>
</tr>
<tr>
<td>END TRAILER XXX</td>
<td>4TXXX</td>
</tr>
<tr>
<td>END ENTRY</td>
<td>4Ø</td>
</tr>
<tr>
<td>NEXT ENTRY</td>
<td>4</td>
</tr>
<tr>
<td>END</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type 5</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>IF OVERFLOW</td>
<td>5Ø</td>
</tr>
<tr>
<td>IF FINISHED</td>
<td>5F</td>
</tr>
</tbody>
</table>
Type 5 (continued)

IF SWITCH X

N <
IF b (....) = N U D (....)/
D >

Type 6

N
S

COL XXX L = D (....)
R C b

COL XXX = N S (....)

D

COL XXX = C (....) b

COL XXX L = N \(N \frac{N}{b (....)} - \frac{N}{b (....)} \) + x

COL XXX = N \(N \frac{N}{b (....)} - \frac{N}{b (....)} \) + x

COUNTER XX = D (....)
b

COUNTER XX = N \(N \frac{N}{b (....)} - \frac{N}{b (....)} \) + x

58X

N
U
N
S
b
D
T

6XXX

N
S
D
C
b

6XXXR

N
S
....

6XXXL

D
C
b

6XXXR

N
S
....

6CXX

D
b

6CXX

N
S
b
....

A

N
b
....

A

S
b
....

A

S
b
....
GIVE MESSAGE "FIRST STATEMENT MUST BE 000"
N0G0
INACTIVATE DISK UPDATES
PACKER PACK CONTINUATION STATEMENTS
SENSES PRINT OUT CARD
READ READ CARD

CONDEN PROCESS A PHRASE
RGPACK
BUMP CONTINUATION CARD COUNTER
RGPGOT
BYPASS STATEMENT CARDS
GIVE ERROR "TOO MANY CONTINUATION CARDS"

ARE THERE MORE THAN 2 CONTINUATIONS?

ARE THERE CONTINUATION CARDS?

IS NEXT CARD AN OUTPUT DESCRIPTOR?

IS NEXT CARD AN END CARD?
LOCATE

CLEAR SOT TABLE

PUT SOT SDA FROM REPORT DESCRIPTOR TABLE INTO DSKACC

GET RELATIVE POSITION OF SOT FOR THIS OUTPUT FORMAT

GET SOT TABLE FROM DISK

SET "END OF TABLE" INDICATOR

EXIT FROM LOCATE
BLANK PHRASE IDENTIFICATION & UNCONDENSED STATEMENT COUNTER

BLANK PACK AND UNPACK POINTER STORAGE AREAS

UNPACK LOOK FOR SPECIAL STATEMENT CHAR

IS THERE A TERMINATING PERIOD?

YES

GIVE ERROR "NO PERIOD"

NO

NOGO INACTIVATE DISK UPDATES

SETUP PHRASE IDENTIFICATION ROUTINE

IS PHRASE TYPE IN TABLE?

IS THIS END OF TABLE?

INCREASEMENT FOR NEXT LOOK

YES

DENOUT

OUTERR

NO

YES

IDENTIFIED PHRASE TYPE

PROCESS PHRASE

PACK CONDENSED PHRASE

UNPPHRR

429x276 INCREMENT FOR NEXT LOOK

429x267 FOR NEXT LOOK

441x258 LOOK

294x80 64
OUTERR

GIVE MESSAGE "PHRASE TYPE NOT IDENTIFIED"

NOGO

INACTIVATE DISK UPDATES

UNPPHR

UNPACK

BLANK PHRASE WORK AREAS

SET STATEMENT POINTER AT BEGINNING OF PHRASE

IS THERE A "/" ?

YES

BRCHECK

NO

UNPNEX

YES

EXIT FROM UNPACK

NO

EXIT FROM UNPACK

SET UP ERROR

UNPACK

UPDNOP
INCREMENT POINTER

IS THERE A ")")?

HAS STATEMENT LENGTH BEEN EXCEEDED?

ERROR, "RIGHT PARENTHESES MISSING"

NOGO

INACTIVATE DISK UPDATES

EXIT FROM CONDEN

CALCULATE PHRASE LENGTH

ISOLATE PHRASE FROM STATEMENT

EXIT FROM UNPACK
ERROR MESSAGE
"ACTION CODE INCORRECT"

PUT NEW ACTION CODE INTO SOT TABLE
BLANK QL SOT UPDATE SWITCH

READ

IS NEXT CARD AN OUTPUT DESCRIPTOR?

IS NEXT CARD END?

ERROR "DELETE NOT FOLLOWED BY END OR OUTPUT DESCRIPTOR"

SKIPAL
IS THERE AN UNNECESSARY RECORD MARK AT START OF SOT TABLE?

YES

ELIMINATE RECORD MARK

NO

IS PREVIOUS ACTION CODE BLANK, DELETE, ERROR OR ≠?

NO

LOCCEQ

YES

PUT PRESENT ACTION CODE INTO SOT TABLE

CALCULATE OUTPUT MEDIA STORAGE BIN SIZE. PUT INTO SOT TABLE

IS THERE AN INVALID OUTPUT MEDIA CHARACTER?

NO

UPDSER

YES

MESSAGE "INVALID OUTPUT MEDIA CHARACTER"

SKIPAL
SET LOWER LIMIT OF I/O BUFFERS

MESSAGE "INVALID HISTORY TAPE CODE"

DETERMINE OUTPUT MEDIA, AND ALLOT BUFFER SPACE FOR EACH TYPE.

ADD OUTPUT MEDIA BUFFER SPACE TO LOWER LIMIT BASE.

SET TOP OF AVAILABLE CORE

INPUT IN QUERY LANGUAGE FORM

DOLLY

CATOUT
SET AN INPUT BIN OF 1000 FOR QUERY LANGUAGE (HIGH CORE MINUS 1000)

CALCULATE INPUT SPACE REQUIRED FOR ENTRY FROM INTERMEDIATE TAPE

SUBTRACT REQUIRED ENTRY SPACE FROM HIGH CORE VALUE

PUT CALCULATED VALUE INTO SOT TABLE. DENOTES CORE ORIGIN OF INPUT AREA

SPACE BETWEEN INPUT AND OUTPUT BUFFER ALLOTTED FOR CONDENSED STATEMENTS

CALCULATE NUMBER OF CONDENSED STATEMENTS FITTING IN CORE

PUT NUMBER OF OUTPUT STATEMENTS INTO SOT

PUT OUTPUT STATEMENTS' SDA INTO WRITE DISK CALL

CLEANBINS

IS THIS STATEMENT NUMBER GREATER THAN PREVIOUSLY STORED NUMBER?

YES

LOCAL
SAVE SOT POINTER

DOES THIS REVISED FORMAT CONTAIN THE SAME STATEMENTS AS THE OLD FORMAT?

NO

ALLOCATE CONDENSED STATEMENT SDA
NUMBER OF RECORDS EQUAL HIGHEST STATEMENT NUMBER

PUT CONDENSED STATEMENT SDA INTO SOT TABLE

PUT CONDENSED STATEMENT SDA INTO DISK WRITE CALL

SAVE SOT ACTION CODE, PUT ERROR ACTION CODE IN SOT WRITE SOT TO DISK

RESTORE SOT ACTION CODE

CLEAN BINS

SET UP DISK WRITE CALL FOR CONDENSED STATEMENT BIN.

BLANK OUT A RECORD OF CONDENSED STATEMENT BIN ON DISK

BUMP RECORD COUNTER

HAVE ALL RECORDS BEEN BLANKED?

NO

YES

UPD SOT

CLEAN BINS
SIX

BLANK PHRASE AND CONDENSED PHRASE POINTERS

IS THIS A COLUMN PHRASE?

- YES: COL
  - RETURN TO CALLER

- NO: IS IT A COUNTER PHRASE?
  - NO: ERROR MESSAGE "PHRASE TYPE NOT IDENTIFIED"
    - RETURN TO CALLER
  - YES: PUT A '6C' INTO CONDENSED AREA

MODIFY POINTERS

ISOLATE COUNTER VALUE. PUT INTO CONDENSED AREA

COUNT

CHECK FOR LEGAL COUNTER

CHECK FOR RIGHT OR LEFT JUSTIFICATION

SET ERROR MESSAGE FOR MISSING OR OUT OF PLACE "=" SIGN

PUT JUSTIFICATION CHARACTER INTO CONDENSED PHRASE

CONNEX

IS NEXT CHARACTER AN "=" SIGN?

- NO: RETURN TO CALLER
  - YES: INSERT A RIGHT JUSTIFICATION CHARACTER INTO CONDENSED PHRASE

SET UP INCORRECT INDICATOR MESSAGE

IS CHARACTER AN "EQUALS" SIGN?

- NO: RETURN TO CALLER
  - YES:
ADVANCE POINTER TO START OF PAREN CONTENTS

DOES NEXT INDICATOR CHARACTER SPECIFY "N"?

YES

SIXBRB

NO

DOES IT SPECIFY "D"?

YES

SIXBRK

NO

DOES IT SPECIFY A "("?

YES

SET UP IMPROPER INDICATOR CHARACTER MESSAGE

RETURN TO CALLER

NO

ADVANCE POINTER TO START OF PAREN CONTENTS

BRACT

EVALUATE PARENTHESES CONTENTS

SIXBAK

REPV

CHECK PHRASE SYNTAX

EXIT FROM SIX

IS NEXT CHARACTER A "("

YES

NO

CONNNT

SIXEXT

9-26-220

73
IS NEXT CHARACTER A " "?

NO

SET UP INCORRECT FORMAT MESSAGE

RETURN TO CALLER

YES

IS NEXT CHARACTER A "N" OR "?"

NO

IS NEXT CHARACTER A "("?

NO

SIXEXT

YES

PUT FIRST NUMERIC CHARACTER INTO CONDENSING AREA

BUMP PHRASE POINTERS

IS THIS CHARACTER "N" OR "(?"

NO

SIXJMF

YES

SIXXAB

NO

SET UP INCORRECT FORMATTING MESSAGE

RETURN TO CALLER

SIXERY
PUT SUBTRACT CHARACTER INTO CONDENSED AREA

PUT DIVIDE CHARACTER "(" INTO CONDENSED AREA

PUT MULTIPLY "*" CHARACTER INTO CONDENSED AREA

SET INCORRECT OPERATION SIGN MESSAGE.

RETURN TO CALLER

BUMP UNCONDENSED PHRASE COUNTER
SET UP IMPROPER JUSTIFICATION CHARACTER MESSAGE

RETURN TO CALLER

ADJUST CONDENSED POINTER

CHECK FOR "D" "C" OR "I"

YES COLLEF

NO

CHECK FOR "N" OR "S" CHARACTER

NO COLLEF

YES COLERR

PUT "R" JUSTIFICATION CHARACTER INTO CONDENSED AREA

BUMP CONDENSED COUNTER

SIXBRB

PUT "L" JUSTIFICATION CHARACTER INTO CONDENSED AREA

BUMP CONDENSED COUNTER

SIXEXT
PUT JUSTIFICATION CHARACTER INTO CONDENSED AREA

BUMP STATEMENT COUNTERS

IS INDICATOR CHARACTER "N" OR "S"?

IS INDICATOR CHARACTER "D" OR "L"?

IS INDICATOR CHARACTER "("?

SET UP IMPROPER INDICATOR CHARACTER MESSAGE

RETURN TO CALLER

BLANK CONDENSED PHRASE POINTER

IS THIS "IF FINISHED" PHRASE?

IS THIS "IF OVERFLOW" PHRASE?

IS THIS "IF SWITCH" PHRASE?

SET UP CONDENSED AREA WITH IF CONDITIONS PHRASE

BUMP POINTERS

IS CHARACTER A", "N" OR "D"?

RETURN TO CALLER

FIVEA

CONEXP

78
SET UP ERROR ENTRY INTO TYPE SIX SUBROUTINE

CONNNT

BRACT EVALUATE PARENTHESES CONTENTS

BUMP UNCONDENSED POINTER

IS OPERATION SIGN "UNEQUAL"?

NO

IS OPERATION SIGN "="?

NO

IS OPERATION SIGN "<"?

NO

INDICATE BAD OPERATION SIGN

YES

RETURN TO CALLER

YES

GREATT

YES

LESTHN

NOTEQL

EQUALV
BRACKET
EVALUATE PARENTHESES CONTENTS

PUT 50 IN CONDENSING BIN AND BUMP POINTERS!

IS NEXT CHARACTER A "#"?

SET UP 'SLASH MISSING AFTER IF PHRASE' MESSAGE

RETURN TO CALLER

PUT "5F" IN CONDENSING BIN AND BUMP POINTERS

FIVABC

FIVENE

FIVABC

IFOVER

PUT "50" IN CONDENSING BIN AND BUMP POINTERS

FIVABC

IFSWIT

PUT "5S" IN CONDENSING BIN AND BUMP POINTERS

SEND SWITCH NUMBER TO CONDENSING AREA-
BUMP COUNTER

FIVABC

FIVABC

IFFIV

FIVABC

8-26-76

81
BRACT

IS UNCONDENSED PHRASE CHARACTER A "("?

YES → BRACON

NO → BLANK OUT RELATIVE POINTER FOR Q.L. OEDT ENTRY

BLANK RELATIVE RECORD IN DSKACC CALL FOR OEDT

PUT TYPE OF DATA INDICATOR INTO CONDENSED AREA

BUMP COUNTERS

IS THIS A "COUNTER" PHRASE?

YES → BRACOU

NO → IS THIS "AS OF TIME" PHRASE?

YES → ASOFMV

NO → IS THIS "TODAY" PHRASE?

YES → TODAYS

NO → IS THIS "REMAINDER" PHRASE?

YES → BRARER

NO → IS THIS "CLASSIFICATION" PHRASE?

YES → CLASS

NO → NO

SET UP DSKACC CALL FOR OEDT

IS OEDT TABLE IN CORE?

YES → NO

SET UP BIN FOR STORING AN ATTRIBUTE NAME

BRAREP

18-26,177

82
BUMP POINTERS

SET UP MESSAGE RIGHT PAREN MISSING

RETURN TO CALLER

PUT ATTRIBUTE NAME INTO COMPARING BIN

ARE POINTERS AT LO. END OF ATTRIBUTE?

NO

BUMP POINTERS

CHECK FOR MISSING ")"

YES

SET IN CORE SWITCH

CALL IN OEDT

BRACOM

IS OEDT TABLE IN CORE?

YES

BRACOM

NO

NO

ID
BUMP Q.L.
RELATIVE
POINTER

IS ATTRIBUTE NAME IN OEDT?
YES: BRAOUT
NO:

IS THIS END OF TABLE (CHARACTER)?
YES: SET UP MESSAGE "ATTRIBUTE NAME NOT IN TABLE"
NO:

IS THIS END OF OEDT SECTOR?
YES: BUMP TO NEXT ATTRIBUTE IN OEDT
NO: BRACOM

BUMP RELATIVE RECORD

NULL OUT OEDT IN CORE SWITCH

BRAQQQ
PUT RELATIVE POSITION OF ATTRIBUTE IN OUTPUT CONDENSING BIN

BUMP COUNTERS

IS NEXT CHARACTER A " ) " ?

YES

EXIT FROM BRACHT

BRAERI

SET UP MESSAGE "RIGHT PARENTHESES MISSING"

NOGO

BRANEX
PUT COUNTER IDENTIFIER (COO) INTO CONDENSING BIN.

BUMP POINTERS

FILL IN COUNTER IDENTIFIER WITH PROPER COUNTER NUMBER

BUMP CONDENSED POINTER

COUNT MAKE SURE COUNTER IS LEGAL

BUMP UNCONDENSED POINTER

IS NEXT CHARACTER A "0"?

SET UP INCORRECT FORMAT MESSAGE.

NO

YES

INACTIVATE DISK UPDATES

UNPPHR
PUT A BLANK INTO CONDENSING BIN

BUMP POINTERS

IS NEXT CHARACTER A?

YES
BRAOFF

NO

IS THIS CHARACTER A''?

YES
BRACUP

NO

BUMP POINTERS

HAS BIN LIMIT BEEN EXCEEDED?

YES
BRAERI

NO

BRAPOT
REPGEN

Purpose

REPGEN is operated to produce a Serial File Tape from a source data file contained on cards, magnetic tape or on the system disk. The Serial File Tape is then used as input to the output phase of the Report Generation capability.

Calling Sequence

Enter on the console typewriter

REPGENrrrfcon

where  rrr = 3-digit report number (no blanks)

f = Format number of the Output Program to be executed. Must be 1-7 or blank.

c = Classification code: U, C, S or T for Unclassified, Confidential, Secret or Top Secret.

n = Number of copies of printed output. Must be 1-9 or blank.

Final return will be made to IDLOOP.

Description

The Report Descriptor Table is read from disk and the data from the typewriter call is stored away. If no report number was given with the call, the PRTIND routine is operated, printing an index of REPGEN reports followed by a return to IDLOOP.

If the report number was not blank, the requested report is located in the Report Descriptor Table, mount messages are issued and the input medium is checked.

If the input medium is disk the saved Query Language statement number is taken from the Report Descriptor Table entry and used to construct a dummy call to REPEET. REPEET is called via COPS; it fetches the Query Language statement, types it on the console and passes control to GETTAC. GETTAC executes the Query Language statement, placing the retrieved data on a Serial File Tape. The option is now given to the operator to enter the output phase or to terminate.

If the input medium is not disk, the Record Descriptor Table is brought in from disk and a source data record (either card or tape) is read. Entries in the Record Descriptor Table are examined
against the input record until the record type is identified, at which
time the Attribute Descriptor Table for this record type is read into
core.

All control attributes are collected into a control
field for this input record. Sub-control attributes are collected
into a sub-control field.

The control field is compared with the previous control
field. If they differ, a control break is noted. If the sub-control
field differs from the previous sub-control field, a sub-control break
is noted. If the current record caused no control or sub-control break,
the record is given at once to the audit routine.

The audit routine processes each attribute of the input
record separately, performing left or right justification and legality
checks as required or specified. Each attribute, after it is audited,
results in an entry in the DAWORK Table. The entry in DAWORK specifies
the location of the attribute in the input area, its destination in
the output area, and its length. If the input record contained sub-
control attributes, a similar entry is made in the DAWOR2 table. After
all input attributes for the current record have been audited, the
DAWORK table is used to move the audited fields from the input area
to the output buffer. The next input record is then read.

At the point where control and sub-control fields were
checked with previous fields, if a control break or sub-control break
was noted the current contents of the output buffer will be written
on the Serial File Tape. If a control break occurred, the entire
output image is then cleared. If a sub-control break occurred, the
DAWOR2 table is used to determine those fields to selectively clear
in the output image. After the output image is cleared, the current
input record is given to the audit routine, and processing continues
as described before.

When the input data file has been fully processed and
the Serial File Tape is complete, a disk record called the I/O Communi-
cation Record is written, containing the current Report Descriptor
Table entry and various typewriter inputs (e.g., Report Classification
Code, Format Number, AS OF TIME) to be passed along to the output phase.
The operator is given a typewriter option to cancel or to continue into
the output phase. If the output phase is requested, a dummy call is
set up to RGPBYP. Control will eventually be returned from the output
phase to REPCGEN which will exit to IDLOOP.
IRDT
READ REPORT DESCRIPTOR TABLE
TURN ON FIRST AND PFIRST
READ TYPE-WRITER INPUT AREA
SAVE REPORT NUMBER, FORMAT COPIES
MOVE REPORT DESC. TABLE ENTRY TO CURREP
READ "AS OF TIME" INTO MONTH
READ "AS OF TIME" FROM CONSOLE
GIVE TAPE MOUNTING MESSAGE
PRINT MESSAGE "INPUT MEDIUM IN ERROR"
EXIT FROM REPGEN
EXIT FROM REPGEN
END OF REPORT DESC. TABLE
STEP TO NEXT ENTRY
SCAN FOR A MATCHING REPORT#
FIND IT?
YES
NO
PRINT MESSAGE "REPORT # NOT IN FILE"
PG
READ REPORT DESCRIPTOR TABLE
TURN ON FIRST AND PFIRST
READ TYPE-WRITER INPUT AREA
SAVE REPORT NUMBER, FORMAT COPIES
MOVE REPORT DESC. TABLE ENTRY TO CURREP
READ "AS OF TIME" INTO MONTH
READ "AS OF TIME" FROM CONSOLE
GIVE TAPE MOUNTING MESSAGE
PRINT MESSAGE "INPUT MEDIUM IN ERROR"
EXIT FROM REPGEN
EXIT FROM REPGEN
SET UP I/O COMMUNICATIONS RECORD

WRITE IT ON DISK

TYPE OUT "END OF INPUT PHASE"

GIVE OPTION TO CONTINUE OR CANCEL

CANCEL?

SET UP TYPEWRITER AREA ON DISK WITH DUMMY CALL TO RGPBYP

CALL RGPBYP VIA COPS

RGPBYP ENTER OUTPUT PHASE

EXIT FROM REPGEN

EXIT FROM REPGEN
INSURE THAT ANY MANDATORY INPUT RECORDS HAVE BEEN READ

ERROR MESSAGE "MANDATORY INPUT RECORD MISSING"

WRITE A RECORD ON OUTPUT TAPE

TURN ON PCLRSB

PWRTAI

MASC TL OR MASSUB OR PFIRST ON?

PATT3

MASC TL OR MASSUB ON?

PBEGF

PWR TAP

RSW TAPE

PWR TAI

PSET3

ANY MISSING?

PSET

PC LRSB ON?

PSET3

PATT3

PSET
SELECT NEXT ATTRIBUTE

ARE WE AT END OF ATTR TBL?

NO

SELECT NEXT ATTRIBUTE

RGPAUD
AUDIT ATTRIBUTE

MASSER ON?

NO

MAKE ENTRY IN DAWORK TABLE, GIVING ATTRIBUTE POSITION IN INPUT AREA, ATTR.POS. IN OUTPUT IMAGE & ATTRIBUTE LENGTH

NO

MAKE EQUIVALENT ENTRY IN DAWORD2 TABLE

YES

WHICH ON?

NO

TURN ON MERSWF

TURN ON MASCTL & MERSWF

YES

TURN ON MASSUB & MERSWF

CLEAR INPUT AREA

TURN OFF PCLRSB, WHICH CTFIN, PFIRST

EXIT TO PDSLK CALLER
PSET

CLEAR ENTIRE OUTPUT AREA

SAVE LAST CONTROL FIELD; CLEAR SUB-CONTROL FIELD

TURN OFF MASCTL, MASSUB

CLEAR DAWOR2 TABLE

TURN ON PSUBFS

TURN OFF PFIRST

PATT3

PBEGF

PSET3

USING FIELDS DEFINED IN DAWOR2, SELECTIVELY CLEAR OUTPUT IMAGE

CLEAR DAWOR2 TABLE

TURN OFF PSUBFS

SAVE LAST SUB-CONTROL FIELD

PATT3
ISOLATE THE FIELD

LAT/LONG CHECK? YES
ALLCK PERFORM LAT/LONG CHECK

NO
ALPHA NUMERIC FIELD? YES
ALNI LEFT JUSTIFY ONLY

NO

ALPHA FIELD? YES
TURN ON SW1

NO
NUMERIC FIELD? YES
ANIN RIGHT JUSTIFY AND CHECK NUMERIC

NO

LEGITIMATE VALUE CHECK? YES
ALVAL PERFORM LEGITIMATE VALUE CHECK

NO

MANDATORY VALUE CHECK? YES
TURN OFF SW1, SW5

NO

MASERR ON? YES
PRINT ATTRIBUTE DESC. ENTRY

NO
EXIT TO CALLER

AERR4 NO

SW5 ON? YES
ALLCK

LAT >90°?

YES AERR

NO

IS LAT NUMERIC?

NO AERR

YES

EXIT TO CALLER

LAT MINUTES >60°?

NO

YES AERR

NO

LONG CHARACTER ERROR?

YES

NO

LONG MINUTES NUMERIC?

YES

NO

LONG MINUTES >60°?

YES

IS LONG NUMERIC?

NO

YES

AERR

AERR
CHECK FIELD FOR ALPHA DATA

LEFT JUSTIFY FIELD

ALCIR
CHECK FIELD FOR ALPHA DATA

EXIT TO CALLER

NO

YES

ALCHK
CHECK FIELD FOR ALPHA DATA

EXIT TO CALLER

READ IN LEGITIMATE VALUE ATTRIBUTE NAME TABLE

READ IN LEGITIMATE VALUE TABLE

TRY TO MATCH CURRENT ATTRIBUTE WITH A TABLE ENTRY

FIND A MATCH?

NO

AERR6

YES

EXIT TO CALLER

FIND IT?

YES

NO

AERR5

EXIT TO CALLER

RIGHT JUSTIFY NUMERIC DATA

RGTJUS

RIGHT JUSTIFY NUMERIC DATA

ANUCK
CHECK FIELD FOR NUMERIC DATA

INSERT LEADING ZEROES

EXIT TO CALLER

ANIN

ANIN

ALVAL

READ IN LEGITIMATE VALUE ATTRIBUTE NAME TABLE

SCAN FOR ATTRIBUTE NAME

EXIT TO CALLER

EXIT TO CALLER
SET UP A DUMMY CALL TO REPEAT

CALL REPEAT VIA COPS

REPEAT FETCH AND EXECUTE SAVED Q.L. STATEMENT

ANY ERRORS?

PRINT Q.L. ERROR MESSAGE

EXIT FROM REPGEN

TYPE

FETCH MESSAGE AND PUT GMWM AT END

TYPE MESSAGE ON CONSOLE

EXIT TO CALLER

RETURN TO CALLER
RGPBYP

Purpose

This routine issues mount messages for the output phase, completes the I/O Communication Record and calls in the output phase.

Calling Sequence

If called internally from REPGEN, the dummy typewriter call which is constructed is

RGPBYP***

If called externally, enter on the console typewriter

RGPBYPrrrfcn

where   rrr = 3-digit report number (no blanks)

f = Format number of the Output Program to be executed. Must be 1-7 or blank.

c = Classification code: U, C, S or T for Unclassified, Confidential, Secret or Top Secret.

n = Number of copies of printed output. Must be 1-9 or blank.

Final return is to calling program.

Description

If RGPBYP was called from REPGEN, the I/O Communication Record is read back from disk. If RGPBYP was externally called, the AS OF TIME is requested, and an I/O Communication Record is constructed.

The SOT Table for the requested report is accessed and used to determine the output devices required and the mount messages to be issued. Requests for multiple copies are ignored if the output device is not the printer.

After the mount messages are issued, the I/O Communication Record is completed and written on the system disk. The output program RGPOUT is then called via COPS. After return is made from RGPOUT, RGPBYP will exit to its calling program.
RGPOUT

Purpose

RGPOUT acts as the output phase supervisor, controlling the loading and execution of the various output phase programs.

Calling Sequence

RGPOUT is called internally from RGPBYP as follows:

B COPS
DCW 'RGPOUT'

Description

The Read/Write Tape routine RWTAPE, the CONSOLE routine and the initial Communication Zone are all assembled with RGPOUT, and are loaded into core when RGPOUT is loaded. They remain in core throughout the output phase.

RGPOUT calls the output phase initialization routine ININT into core and executes it. Next the SOT Table entry for the output program to be executed is examined. If input was via Query Language, the SOT entry is checked to see if UPSTAT has been executed for this output program.

If input was via Query Language and UPSTAT has not been executed, UPSTAT is loaded and executed to complete the condensing of the output program.

The appropriate output routine, PRINTO, CARDO or TAPEO, is loaded depending upon whether the output device is the printer, the card punch or magnetic tape. The main output program REALO is loaded and executed to produce the desired output.

After REALO executes the output program, RGPOUT checks to see if any more formats were requested for this report. If so, the appropriate output program is loaded, REALO is loaded, and the next output program is executed.

When all requested output programs have been executed, RGPOUT checks to see if extra copies of printed reports were requested. If so, the extra copies tape is rewound and an internal call to PRTTAP is set up and executed to yield the necessary printed copies.
READ IN REMARKS PROCESSOR

PRINT MESSAGE "END OF OUTPUT PHASE"

ANY EXTRA COPIES REQUESTED?

CONSOLE GIVE SENSE SWITCH D OPTION FOR REPEAT PRINT

WRITE TAPE MARK ANDREWIND EXTRA COPIES TAPE

SET UP DUMMY CALL TO PRTTAP

CALL PRTTAP VIA COPS

PRTTAP PRINT ADDITIONAL COPIES

EXIT FROM RGPOUT

ANY PRINT REMARKS

REMARK PRINT REMARKS

REMARK PRINT REMARKS

EXTRA COPIES REQUESTED AT CONSOLE

GIVE SENSE SWITCH D OPTION FOR REPEAT PRINT

REPEAT PRINTOUT ELECTED?

REWRITE INPUT TAPE

QUERY LANGUAGE INPUT?

M102

QUERY LANGUAGE INPUT?

M102

M11

NO

CONSOLE GIVE SENSE SWITCH D OPTION FOR REPEAT PRINT

REPEAT PRINTOUT ELECTED?

REWRITE INPUT TAPE

QUERY LANGUAGE INPUT?

M102

QUERY LANGUAGE INPUT?

M102

M11

NO

CONSOLE GIVE SENSE SWITCH D OPTION FOR REPEAT PRINT

REPEAT PRINTOUT ELECTED?

REWRITE INPUT TAPE

QUERY LANGUAGE INPUT?

M102

QUERY LANGUAGE INPUT?

M102

M11

NO

CONSOLE GIVE SENSE SWITCH D OPTION FOR REPEAT PRINT

REPEAT PRINTOUT ELECTED?

REWRITE INPUT TAPE

QUERY LANGUAGE INPUT?

M102

QUERY LANGUAGE INPUT?

M102

M11

NO

CONSOLE GIVE SENSE SWITCH D OPTION FOR REPEAT PRINT

REPEAT PRINTOUT ELECTED?

REWRITE INPUT TAPE

QUERY LANGUAGE INPUT?

M102

QUERY LANGUAGE INPUT?

M102

M11

NO

CONSOLE GIVE SENSE SWITCH D OPTION FOR REPEAT PRINT

REPEAT PRINTOUT ELECTED?

REWRITE INPUT TAPE

QUERY LANGUAGE INPUT?

M102

QUERY LANGUAGE INPUT?

M102

M11

NO

CONSOLE GIVE SENSE SWITCH D OPTION FOR REPEAT PRINT

REPEAT PRINTOUT ELECTED?

REWRITE INPUT TAPE

QUERY LANGUAGE INPUT?

M102

QUERY LANGUAGE INPUT?

M102

M11

NO

CONSOLE GIVE SENSE SWITCH D OPTION FOR REPEAT PRINT

REPEAT PRINTOUT ELECTED?

REWRITE INPUT TAPE

QUERY LANGUAGE INPUT?

M102

QUERY LANGUAGE INPUT?

M102

M11

NO

CONSOLE GIVE SENSE SWITCH D OPTION FOR REPEAT PRINT

REPEAT PRINTOUT ELECTED?

REWRITE INPUT TAPE

QUERY LANGUAGE INPUT?

M102

QUERY LANGUAGE INPUT?

M102

M11

NO

CONSOLE GIVE SENSE SWITCH D OPTION FOR REPEAT PRINT

REPEAT PRINTOUT ELECTED?

REWRITE INPUT TAPE

QUERY LANGUAGE INPUT?

M102

QUERY LANGUAGE INPUT?

M102

M11

NO

CONSOLE GIVE SENSE SWITCH D OPTION FOR REPEAT PRINT

REPEAT PRINTOUT ELECTED?

REWRITE INPUT TAPE

QUERY LANGUAGE INPUT?

M102

QUERY LANGUAGE INPUT?
ININT

Purpose

Performs initialization for the output phase of REPGEN.

Calling Sequence

ININT is called from RGPOUT by

B  ININT

(Normal Return)

Description

ININT reads the I/O Communication Record from disk. The classification code is checked for U, C, S or T and causes the system attribute CLASS to be set to Unclassified, Confidential, Secret or Top Secret. Other data is taken from the I/O Communication Record and entered into the Communication Zone. The SOT Table is read into the Communication Zone. If only one specific format was requested, all other entries in the SOT Table are set to blanks. The Communication Zone is written on the disk from where it may be read back to re-initialize it between execution of output programs.

Return is made to RGPOUT.
START

READ I/O COMMUNICATIONS RECORD

CHECK CLASSIFICATION CHARACTER FROM CONSOLE

BLANK?

YES

CHECK CLASSIFICATION CHARACTER IN SOT ENTRY

NO

S?

YES

SET CLASS = SECRET

NO

T?

YES

SET CLASS = TOP SECRET

NO

U?

YES

SET CLASS = UNCLASSIFIED

NO

C?

YES

SET CLASS = CONFIDENTIAL

NO

MESSAGE: CLASSIFICATION ERROR

EXIT FROM RGPOUT

SET CLASS = SECRET

SET CLASS = TOP SECRET

SET CLASS = CONFIDENTIAL

SET CLASS = UNCLASSIFIED

X

111
ENTR INTO COMM ZONE: AS OF TIME:
Q.L. INDICATOR:
HILO FOR REMARKS
# OF EXTRA COPIES

READ SYSTEM DATE AND PUT INTO COMM ZONE (DATE)

READ SOT

ALL FORMATS REQUESTED?
YES

FORMAT # > 7?
YES
PRINT ERROR MESSAGE
NO
EXIT FROM RSPOUT

IS SELECTED FORMAT ACTIVE?
YES
BLANK OUT ALL ENTRIES IN SOT IN CORE EXCEPT FOR REQUESTED FORMAT
NO
EXIT TO CALLER

RIGHT JUSTIFY AS OF TIME
WRITE COMM ZONE ON DISK

112
UPSTAT

Purpose

UPSTAT is called by RGPOUT to complete the condensing of output statements which LOD could not completely condense. This only occurs in the case where the Serial File Tape is the product of a Query Language retrieval from a parallel file stored on system disk.

Calling Sequence

UPSTAT is called from RGPOUT by

B UPSTAT

(Normal Return)

Description

When LOD condenses the statements of an output program which uses a Query Language generated Serial File Tape as input, the statements cannot be completely condensed. In particular, LOD does not know the length or position in the Serial File Tape records of each attribute. LOD does know the order in which the attributes will appear on the Serial File Tape. In the case in point, LOD assigns each attribute a number corresponding to its relative position in the Serial File Tape records, and substitutes this number for each occurrence of an attribute name in an output statement.

When UPSTAT is executed, it reads the first record of the Serial File Tape and uses it to construct a table. The table so constructed contains the length and relative position on the Serial File Tape of each attribute. The condensed output program is read from disk, and the attribute numbers are now replaced with length and position data from the table. The fully condensed statements are written back onto disk, the "UPSTAT executed" indicator is set in the appropriate SOT entry, and return is made to RGPOUT.
START

UPSTAT

READ FIRST RECORD FROM TAPE 3

BUILD A TABLE IN UPAREA GIVING LENGTH AND POSITION OF EACH ATTRIBUTE WITHIN A RECORD ON TAPE 3

UPDATE

UPDSOT

EXITS TO CALLER

UPDSOT

READ I/O COMMUNICATIONS RECORD

GET SDA OF SOT AND READ SOT

GET OUTPUT PROGRAM FORMAT # FROM XR7

SET "UPSTAT EXECUTED" CHARACTER IN PROPER SOT ENTRY

WRITE SOT BACK ON DISK

RETURN TO CALLER
USE HIGHEST STATEMENT* TO DETERMINE NUMBER TO PROCESS

ANY STATEMENTS LEFT?

RETURN TO CALLER

MORE THAN 40 LEFT?

SET UP TO READ 40; DECREASE COUNT BY 40

SET UP TO READ REMAINING STATEMENTS

READ A SET OF OUTPUT STATEMENTS

SELECT FIRST (NEXT) STATEMENT

WRITE STATEMENT BACK ONTO DISK

ANY MORE STATEMENTS IN CORE?

YES

END OF STATEMENT

NO

STEP TO NEXT PHRASE

TYPE 5 PHRASE?

YES

FINISHED OVERFLOW OR SWITCH TYPE?

NO

ANY OF OTHER VARIOUS TYPES?

NO

NO

MOVE IN NUMBERS FROM UPAREA TABLE

NO

NO

NO

NO

YES

YES

YES

YES
PRINTO

Purpose

Print a line on the printer or read a record from the Serial File Tape.

Calling Sequence

PRINTO is entered from the REALO program. If a record is to be read from the Serial File Tape, the call is

B INPUT (Normal return)

If the output image is to be printed, the call is

B OUTPUT (Normal return)

Description

If entry is made at INPUT, one record is read from the Serial File Tape. If an end-of-file is encountered, the indicator COMDON is turned on in the Communications Zone.

If entry is made at OUTPUT, the first 132 characters of the output image are printed. The line counter is incremented, and if it exceeds the value in OVFLOW, the overflow indicator is turned on.

If additional copies of the printed report were requested, the print image is also written onto the extra copies tape. Three characters are added to the record on tape to indicate any blank lines or page ejects which should be accomplished before this line is printed. (This spacing information will be interpreted by the PRTTAP program which generates additional copies of the printed report from the extra copies tape.)
PRINTTO
OUTPUT

MOVE FIRST
132 CHARACTERS
FROM OUTPUT
IMAGE TO
PRINT AREA

INCREMENT
LINE COUNTER
BY ONE

LNCTR > OFLOW
?

PRINT A LINE

ADDITIONAL
COPIES
REQUESTED
?

ADD CARRIAGE
CONTROL
CHARACTERS
TO PRINT IMAGE

EXIT TO
CALLER

TURN ON
OVERFLOW
INDICATOR

TURN ON
COMMON
INDICATOR

READ A
RECORD FROM
TAPE 3

END-OF-FILE
?

EXIT TO
CALLER

WRITE PRINT
IMAGE ONTO
EXTRA COPIES
TAPE
CARDO

Purpose

Punch a card from the output image or read a record from the Serial File Tape.

Calling Sequence

CARDO is entered from the REALO program. If a record is to be read from the Serial File Tape, the call is

B INPUT
(normal return)

If a card is to be punched, the call is

B OUTPUT
(normal return)

Description

If entry is made at INPUT, one record is read from the Serial File Tape. If an end-of-file is encountered, the indicator COMDON is turned on in the Communications Zone.

If entry is made at OUTPUT, a check is made to see if a history tape was requested. If a history tape was requested, the first entry at OUTPUT causes the history tape to be spaced until an end-of-file is detected, and then positioned so that records may be added to the file. The output image is then written on the history tape.

If Sense Switch C is on, the output image is printed.

Next the stacker select character is acquired from the Communication Zone, and the first 80 characters of the output image are punched in a card.
CARDO
INPUT

READ A
RECORD FROM
TAPE 3

END-OF-
FILE?

YES
TURN ON
COMMON
INDICATOR

NO
EXIT
TO
CALLER
TAPEO

Purpose

Write an output image onto magnetic tape, write a Header record on magnetic tape, write a Trailer record on magnetic tape or read a record from the Serial File Tape.

Calling Sequence

TAPEO is entered from the REALO program. If a record is to be read from the Serial File Tape, the call is

B INPUT
(normal return)

If the output image is to be written on tape, the call is

B OUTPUT
(normal return)

If a Header is to be written on tape, the call is

B HEADER
DCW 'NNN'

where NNN is the number of characters of the output image to be used as a Header record. If a Trailer record is to be written on tape, the call is

B TRAILER
DCW 'NNN'

where NNN is the number of characters of the output image to be used as a Trailer record.

Description

If entry is made at INPUT, one record is read from the Serial File Tape. If an end-of-file is encountered, the indicator COMDON is turned on in the Communication Zone.

If entry is made at OUTPUT, a check is made to see if a history tape was requested. If a history tape was requested, the first entry at OUTPUT causes the history tape to be spaced until an end-of-file is detected, and then positioned so that records may be added to the file. The output image (up to 1000 characters) is
then written on the history tape. If Sense Switch C is on, the first 132 characters of the output image are printed. The output image is then written on the output tape.

If entry is made at HEADER, the Header indicator is turned on, the first NNN characters of the output image are written on the output tape, and are also saved on the system disk (if a subsequent entry at OUTPUT results in an end-of-reel condition on the output tape, the saved Header will be written as the first record of the next reel).

If entry is made at TRAILER, the first NNN characters of the output image are written on the output tape.
TAPED

OUTPUT

HISTORY TAPE TO BE MADE

YES

FIRST CALL FOR HISTORY TAPE

NO

SET CALL TO NOT FIRST TIME

YES

SAVE A PIECE OF CORE ON DISK

READ HISTORY TAPE

EOF?

YES

READ IN SAVED HEADER FROM DISK (IF ANY)

WRITE HEADER ON NEW TAPE

WRITE OUTPUT IMAGE ONTO TAPE

WRITE OUTPUT IMAGE ONTO HISTORY TAPE

END OF REEL?

NO

PRINT END-OF-REEL MESSAGE AND WAIT FOR OPERATOR ACTION

EXIT TO CALLER

END OF REEL?

YES

WRITE THE OUTPUT IMAGE ONTO TAPE

PRINT THE OUTPUT IMAGE

WAIT FOR OPERATOR ACTION

REWIND HISTORY TAPE

RESTORE SAVED CORE

END OF REEL?

NO

SENSING SWITCH C ON?

YES

BACKSPACE TWICE, READ ONCE, BACKSPACE ONCE

WRITE BACK LAST RECORD OF HISTORY TAPE

RESTORE SAVED CORE

EOF?

NO

WAS FIRST RECORD EOF?

YES

WAS FIRST RECORD EOF?

NO
TAPEO
INPUT

READ A RECORD FROM TAPE 3

END-OF-FILE?

YES

TURN ON COMMON INDICATOR

NO

EXIT TO CALLER

TAPEO
HEADER

TURN ON TAPE HEADER INDICATOR

WRITE HEADER ON OUTPUT TAPE

SAVE HEADER ON DISK

EXIT TO CALLER

TAPEO
TRAILER

WRITE TRAILER RECORD ON OUTPUT TAPE

EXIT TO CALLER
REALO

Purpose

Interpretively execute an output program to produce the desired report.

Calling Sequence

REALO is called internally from RGPOUT by

B   CTL
(normal return)

Description

An initialize routine is first executed to read into core as much of the condensed output program as will fit. Statements are kept in core just as they appeared on disk, in 245 character records. Each statement, as it is required, is moved to a buffer at WAREA from where it is actually executed. The current statement is examined one phrase at a time. Each phrase begins with a number from 1 to 6 indicating the phrase type. Recognition of the type causes a transfer to the appropriate TYPE routine, where the phrase is completely identified and executed. Section 6.4.4 under LOD shows the phrases grouped by type, and the format of the condensed phrases. Execution of an END phrase causes control to be passed from REALO back to RGPOUT.
TURN 2ND LEVEL ACTIVE INDICATOR OFF (COM2AC = 0)

SAVE CURRENT PUNCTUATION IN COMPUN

KILLCD SWITCH ON?

SET COMDON = \emptyset

CBUNP

COMTYPE = 1?

CBUNP

COMPUN = .?

CDO

COMDR = D?

CDO

COMDON = 1?

CLEAR WAREA THRU WAREA + 199

USING STATEMENT NUMBER IN COMER BUILD A PHONY REPEAT STATEMENT IN WAREA

TURN ON KILLCD SWITCH

SET COMDR = D

CNEXT

CCHK

18-26, 202

127
INIT

PUT TODAY'S DATE INTO DATE

SET COMSOR = CORE ORIGIN OF CONDENSED STATEMENTS

SET COMSNO = # OF CONDENSED STATEMENTS THAT WILL FIT IN CORE

SET COMIOR = CORE ORIGIN OF INPUT AREA

SET COMOEN = CORE ADDRESS OF END OF OUTPUT AREA

SET UP TO READ MAX. NUMBER OF STATEMENTS

READ A LOAD OF CONDENSED STATEMENTS

RETURN TO CALLER

STEP COMIPO = 3 POSITIONS RIGHT, PAST STATEMENT #

SET 2nd LEVEL ACTIVE INDICATOR OFF

SET 1st LEVEL ACTIVE INDICATOR ON

SET COMIPO = COMSOR

SET 1st LEVEL STATEMENT NUMBER COMINO = 000
EXAMINE FIRST CHARACTER OF CURRENT PHRASE IN WAREA TO DETERMINE THE TYPE

WAREA = 6?
YES
TYPE 6
PROCESS TYPE 6 PHRASE

NO
WAREA = 5?
YES
TYPE 5
PROCESS TYPE 5 PHRASE

NO
WAREA = 4?
YES
TYPE 4
PROCESS TYPE 4 PHRASE

NO
WAREA = 3?
YES
TYPE 3
PROCESS TYPE 3 PHRASE

NO
WAREA = 2?
YES
TYPE 2
PROCESS TYPE 2 PHRASE

NO
WAREA = 1?
YES
TYPE 1
PROCESS TYPE 1 PHRASE

NO
WAREA = 0?
STOP

RESULT OF LAST IF TEST FALSE?
NO
WARE THIS IF FINISHED STATEMENT?
YES
CANYWH

NO
CNEXT

(NORMAL RETURN)

CNEXT

(END PHRASE RETURN)

CNEXT
SET TINUM = STATEMENT* OF CURRENT 2ND LEVEL STATEMENT

MOVE TINUM TO COMREP

'READ RE0D^STATEMENT ONTO CORE/

SET 2ND LEVEL ACTIVE INDICATOR ON

SET TINUM = STATEMENT# OF CURRENT 2ND LEVEL STATEMENT

SET 2ND LEVEL ACTIVE INDICATOR ON

SET 1ST LEVEL ACTIVE INDICATOR OFF

ADJUST COM2PO POINT

CONTINUE ?

YES

NO

END ?

YES

NO

GO TO ?

YES

TIGO

IS STATEMENT IN CORE?

NO

YES

TISRCH

MOVE TINUM TO COMREP

SET COMDR=R

REPEAT ?

NO

MUST BE DO. SET COMDR=D

YES

CONTINUE ?

YES

IST RETURN TO CALLER

IST RETURN TO CALLER
TISRCH

IS STATEMENT IN CORE?

NO

YES

MAKE REQUIRED STATEMENT THE CURRENT 1ST LEVEL STATEMENT

READ IN REQUIRED STATEMENT

SET COMINO = STATEMENT # OF CURRENT 1ST LEVEL STATEMENT

SET COMIPO = INITIAL POSITION IN CURRENT STATEMENT

SET 1ST LEVEL ACTIVE INDICATOR ON

STEP COMIPO OVER STATEMENT #

1ST RETURN TO TYPE 1 CALLER

TISRCH

SET A POINTER TO ORIGIN OF CONDENSED STATEMENTS IN CORE

COMPARE A STATEMENT NUMBER WITH REQUIRED STATEMENT NUMBER

MATCH?

YES

2ND RETURN TO CALLER

NO

ANY MORE STATEMENTS IN CORE?

1ST RETURN TO CALLER

YES

BUMP POINTER TO NEXT STATEMENT IN CORE
CLEAR FIRST 02 CHARACTERS OF OUTPUT AREA

OUTPUT MEDIUM PRINTER?

CLEAR FROM 15000 TO END OF OUTPUT AREA

RETURN TO TYPE 2 CALLER

T2CSW

MUST BE SET SWITCH SET REQUIRED SWITCH = 1

RETURN TO TYPE 2 CALLER

T2CCS

BLANK THE SPECIFIED COUNTER

RETURN TO CALLER

T2SET

SET LENGTH?

SET OVERFLOW?

MOVE NUMBER TO OVFLOW AND ADD ONE TO IT

T2SETL

YES

NO

YES

NO

YES

NO
SET SWITCHES

I - 9
*   0

SET POINTER
TO SPECIFIED
COUNTER

SET SWITCH N
= 0

RETURN TO TYPE 2 CALLER

CLEAR ALL
?

YES

NO

ERASE WORD MARK FOR THIS COUNTER

SET HIGH-ORDER WORD MARK TO GIVE REQUIRED LENGTH

RETURN TO TYPE 2 CALLER

CLEAR ALL 50 COUNTERS

CLEARING IS DONE BY RIPPLING A BLANK THROUGH EACH COUNTER

RETURN TO TYPE 2 CALLER

18 - 26, 271
SET WORD MARK UNDER HIGH ORDER POSITION OF A-FIELD DATA ATTRIBUTE

SET UP A-FIELD OF A COMPARE INSTRUCTION

T5TYPEO

PUT LOW-ORDER POSITION OF CONSTANT IN A-FIELD OF COMPARE INSTR.

PUT Q CHARACTER INTO D-MODIFIER OF BRANCH INSTR. WHICH TESTS THE RESULT OF COMPARE (Q=S,T,U OR \).

SET POINTER TO START OF B-FIELD IN CURRENT PHRASE

T5CON

T5ACOU

FETCH COUNTER # AND CALCULATE RELATIVE POSITION OF COUNTER

PUT LOW-ORDER OF COUNTER INTO A-FIELD OF COMPARE INSTR.
SET T6RES TO SPECIFIED POSITION IN OUTPUT AREA (NNN)

SET T6RES TO LOW ORDER POSITION OF SPECIFIED COUNTER (NN)

SAVE JUSTIFICATION CHARACTER IN T6JUS

SET T6 AFLD TO LOW ORDER POSITION OF CONSTANT

SET T6 AFLD TO LOW ORDER POSITION OF COUNTER

SET T6 AFLD TO LOW ORDER POSITION OF DATA ATTRIBUTE

SET WORD MARK UNDER HIGH ORDER POSITION OF CONSTANT

GET PREFIX CHARACTER N, S, C OR D AND SAVE IN T6NSCD

SET T6ANUM TO LENGTH OF A-FIELD

T62B

ANY MORE FIELDS

T6BFD

140
SET T6BFLD TO LOW ORDER POSITION OF CONSTANT
SET T6BFLD TO LOW ORDER POSITION OF COUNTER
SET T6BFLD TO LOW ORDER POSITION OF DATA ATTRIBUTE
SET WORD MARK UNDER HIGH-ORDER POSITION OF CONSTANT
MOVE ARITHMETIC OPERATOR TO ASMD
SET T6BNUM TO LENGTH OF B-FIELD

(ADD) T62C
(ADD) T62C
(ADD) T63A

YES WAS OPERATOR = A ?
YES WAS OPERATOR = S ?
YES WAS OPERATOR = > ?

NO
NO
NO

POINT T6MVEI TO LOW ORDER OF RESULT FIELD
SAVE REMAINDER IN COUNTER @@ SET UP DIVIDE AREA EQUAL IN LENGTH TO T6ANUM + T6BNUM
DIVIDE A-FIELD BY B-FIELD
MULTIPLY A-FIELD BY B-FIELD

POINT T6MVEI TO LOW ORDER OF RESULT FIELD

MOVE T6RES TO T6LOW (RELATIVE LOW ORDER POS. OF FIELD IN OUTPUT AREA)

POINT T6OHIH TO HIGH ORDER POS. OF FIELD IN OUTPUT AREA

POINT T6MHIH TO HIGH ORDER POS. OF RESULT FIELD

MOVE T6MVEI TO T6ML0W (LOW-ORDER POS. OF RESULT FIELD)

T6NSCD = C ?

T6CTRL

T6OK

T6MCS
T60K

T6JUS = R?

T6JUS

T6JUS

T6NSCD

T6NSCD

T6NSCD

T6NSCD

T6END

T6END

T6END

T6END

T6END

T6END
POINT XRI TO LOW ORDER POSITION OF SAME FIELD IN PREVIOUS PRINT IMAGE

COMPARE CURRENT RESULT FIELD WITH SAME FIELD IN LAST PRINT IMAGE

ARE THEY THE SAME?

MOVE CURRENT FIELD TO EQUIVALENT POSITION IN PREVIOUS IMAGE AREA

MOVE RESULT FIELD TO OUTPUT AREA

T6JUS = R?

T6RJUS RIGHT JUSTIFY OUTPUT FIELD

T6LJUS LEFT JUSTIFY OUTPUT FIELD

T6END
T6MCS

MOVE RESULT
FIELD TO OUTPUT
AREA AND
SUPPRESS LEAD
ZEROES

MOVE
DATA TO
COUNTER

YES

T6END

NO

T6JUS

LEFT JUSTIFY
OUTPUT FIELD

YES

NEGATIVE
RESULT

NO

T6JUS

NEGATIVE
RESULT?

T6END

T6END

T6RJUS

RIGHT JUSTIFY
OUTPUT FIELD

NO

NEGATIVE
RESULT?

YES

PUT MINUS SIGN
TO RIGHT OF LOW
ORDER POSITION
OF OUTPUT FIELD

T6END
REMARK

Purpose

Print or punch remarks which were stored on disk during the input phase of a REPGEN run.

Calling Sequence

REMARK is loaded and called from RGPOUT by

B   REMARK
   (normal return)

Description

During the input or file generation phase of REPGEN, any input records defined as remarks are put into a remarks area on the system disk. Remarks normally would be punched into cards, and a given remark may not exceed five cards in length.

If the output medium is punched cards, the REMARK processor simply reads back the remarks stored on disk and punches them into cards.

If the output medium is the printer, REMARK prints a heading and reads a remark from the disk. The remark is packed into a print line and printed. If it overflows to more than one line, subsequent lines are indented ten spaces until the entire remark is printed. The printer is spaced once, the next remark is read and the process is repeated until all remarks have been printed.

The REMARK routine is called and executed after REALO has finished executing an output program.
READ REMARKS FROM DISK

END OF ALL REMARKS ON DISK?
YES RETURN TO CALLER
NO

PUNCH A CARD

DONE WITH THIS REMARK?
YES
NO

PRINT HEADING

SET UP HEADER AND TRAILER LINES

PRINT ON PRINTER OR PUNCH?

OUTPUT ON PRINTER OR PUNCH?

READ REMARKS FROM DISK

SKIP ONE LINE ON PRINTER

END OF ALL REMARKS ON DISK?
YES WRITE TRAILER AT BOTTOM OF PAGE
NO

PACK REMARKS IN OUTPUT AREA

END OF THIS REMARK?
YES

RETURN TO CALLER

A
YES

SET LINE LENGTH TO 100 CHAR.

SET LINE LENGTH TO 90 CHAR.

YES

NO

REWRIT

PRINT A LINE

MOVE REMAINING CHAR. TO PRINT AREA

REWRIT

PRINT A LINE

B
REPEET

Purpose

A stored query GETTAC statement is retrieved, expanded and executed.

Calling Sequence

REPEET is called internally by REPGEN. The call is placed in SDA 0003 as

\[ \text{REPEET} \text{NNN} \]

where NNN is the saved statement number assigned by SAVER. REPGEN then issues a

\[ \text{B COPS} \]
\[ \text{DCW 'REPEET'} \]

REPEET may be called externally by typing

\[ \text{REPEET} \text{NNN} \]

on the console typewriter.

Description

The statement number is validity checked. The saved statement SDA table is read in from disk and the SDA corresponding to the given statement number is extracted. The saved statement is read into core from disk. The statement is scanned for record marks. Each record mark is labeled by a record mark number. The statement is then typed out and data insertions are made via typewriter for each record mark in the statement. The edited statement is written to the system communication area and the GETTAC processor is called via COPS.
REKALL

Purpose

Print out all statements previously stored on disk by the program SAVER.

Calling Sequence

Enter REKALL on the console typewriter. Return will be to IDLOOP.

Description

REKALL reads the saved statement SDA table into core. Each non-blank entry points to a saved statement. Each saved statement is read in turn from the disk and is printed along with its assigned statement number. A total of 500 statements may be saved.
DELEET

Purpose

Delete from the system disk a Query Language statement previously stored there by the SAVER program.

Calling Sequence

Enter on the console typewriter

DELEETNNN

where NNN is the saved statement number assigned by SAVER.

Description

The DELEETNNN message is retrieved from SDA 0003. The 3 character number referencing the statement to be deleted is stored. The table of 4 character SDA's which references where each saved statement is stored is read into core. The first entry in the table references statement number 001, the second entry references statement number 002, and so on. Blank entries indicate that statement number has not been saved and is available. The statement SDA to be deleted is then located. If it is blank, a message will be printed on the console indicating that this statement cannot be deleted because it has not been saved. If an SDA is found in the table entry corresponding to the statement number to be deleted, the saved statement is typed out on the I/O typewriter for the operator's approval. If okay, the SDA is deleted from the statement table and also deallocated so that the statement itself is effectively deleted. The new SDA statement table is now rewritten on the disk and the program returns to IDLOOP.
START
READ DELETE #NNN MESSAGE FROM COMMUNICATIONS BUFFER
STORE STATE- NUMBER NNN
READ STATE- MENT SDA TABLE
IS THERE AN ENTRY FOR NNN ?
YES
FETCH THE STATEMENT
TYPE IT AND ASK OPERATOR IF DELETE
NO
PRINT MESSAGE "STATEMENT #NNN NOT SAVED"
RETURN TO CALLER
OPERATOR ACTION
OKAY TO DELETE ?
YES
REMOVE SDA FROM TABLE OF SDA'S
DEALLOCATE THE STATEMENT
WRITE AMMENDED SDA TABLE BACK ON DISK
RETURN TO CALLER
NO
154
PRTCRD

Purpose

PRTCRD is used to print a report from cards which were generated by RPTCRD.

Calling Sequence

Enter PRTCRD on the console typewriter. Return will be to IDLOOP.

Description

The print area is cleared, the printer is spaced to a new page, and a card is read.

Each input card contains half of a print image in columns 1-66. Columns 77-80 contain information identifying which half of the print image the card represents and any page or line spacing to be done before printing. Columns 77-80 are interpreted as follows:

Column 77 = L or $ ; This is left half of print image.
R ; This is right half of print image.

Columns 78-79 = nn ; Number of times to space before printing.
Blank specifies single spacing.

Column 80 = $ ; Print after putting this card in buffer and performing required spacing.
1 ; Same as $, except skip to new page first.
* ; Put this card in buffer but do not print or space.

As each card is read, columns 1-66 are placed into the left or right half of a print image. If column 80 is a blank or 1, required spacing is performed and a line is printed. If column 80 is *, no printing or spacing is done, and the next card is immediately read.
PRTTAP

Purpose

PRTTAP is used to print additional copies of a report from a REPGEN extra-copies tape.

Calling Sequence

PRTTAP is called internally from RGPOUT. RGPOUT puts

PRTTAPn

where n is the number of copies desired, into SDA 0003 and then calls

B COPS
DCW 'PRTTAP'

The program can also be called externally by entering

PRTTAPn

on the console typewriter. Return will be to IDLOOP.

Description

The input to PRTTAP is tape 4, with one file of 135 character records written in even parity. The first 132 characters of each record represent a print image. The last three characters of each record contain information regarding page and line spacing to be accomplished before printing the image.

The format of each record is as follows:

Character  1-132 = Print image.
133-134 = Number of times to space before printing.
Blank specifies single spacing
135 = Page control.
1 means skip to new page before printing; blank means do not skip.

PRTTAP reads a record and first examines the page control character. If it is a 1, a skip to channel 1 is executed. Next, if any spacing is indicated it is accomplished by printing the appropriate number of blank lines. The line is then printed. When an end-of-file is encountered, the tape is rewound, and the number of copies is decremented by one. If it is not blank, another run is made through the tape.
RPTCRD

Purpose

RPTCRD is used to punch cards from a REPGEN extra-copies tape in a format which will allow the cards to be processed by PRTCRD.

Calling Sequence

Enter RPTCRD on the console typewriter. Return will be to IDLOOP.

Description

Input to RPTCRD is tape 4, with one file of 135 character records written in even parity.

Each input record from tape 4 is treated as a print line, where the first 132 characters constitute a print image, and the last three characters are for carriage control. (See write-up of PRTTAP for discussion of the carriage control characters.)

RPTCRD will divide each print image into a left and right half, and punch each half into the first 66 columns of a card using columns 77-80 of the cards for carriage control information. These cards are in proper format to be processed by PRTCRD, which will use these cards to generate a report on the printer. If Sense Switch C is on, each card will be printed as well as punched.

The format of the output cards and the meaning of the data entered in columns 77-80 is the same as that which was presented in section 6.18.3 to describe the input cards to PRTCRD.

When an end-of-file is encountered on tape 4, two blank cards are punched and control returns to IDLOOP.
The Report Generation (REPGEN) capability comprises a set of programs designed to run on an IBM 1410 computer under the Air Force Integrated Command and Control System (AFICCS). The primary purpose of REPGEN is to provide a capability for producing a printed report from a file of input data contained on punched cards, magnetic tape or on the system disk. This document contains detailed information regarding the structure and operation of the REPGEN programs, including table descriptions, core maps, program flow-charts and abstracts.
<table>
<thead>
<tr>
<th>KEY WORDS</th>
<th>LINK A</th>
<th>LINK B</th>
<th>LINK C</th>
</tr>
</thead>
<tbody>
<tr>
<td>REPORT GENERATOR</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AFICCS REPORT GENERATION CAPABILITY</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>REPORT GENERATOR PROGRAM SPECIFICATIONS - (1410)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>