

AD720315

**DISTRIBUTION STATEMENT A**  
Approved for public release;  
Distribution Unlimited

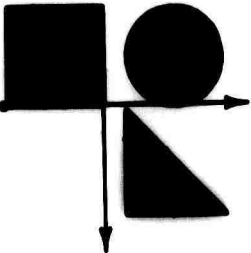
D D C  
**RECEIVED**  
MAR 19 1971  
D. /

Massachusetts

# COMPUTER ASSOCIATES

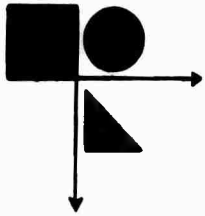
division of

# APPLIED DATA RESEARCH, INC.



Reproduced by  
**NATIONAL TECHNICAL  
INFORMATION SERVICE**  
Springfield, Va. 22151

170



# APPLIED DATA RESEARCH, INC.

LAKESIDE OFFICE PARK, WAKEFIELD, MASSACHUSETTS 01880 • (617) 245-9540

FINAL REPORT - TASK AREA I  
(Volume III)

(21 June 1968 - 31 December 1970)

FOR THE PROJECT  
RESEARCH IN MACHINE-INDEPENDENT  
SOFTWARE PROGRAMMING

Principal Investigators:

Task Area I	Carlos Christensen	(617) 245-9540
Task Area II	Anatol W. Holt	(617) 245-9540

Project Manager:

Robert E. Millstein (617) 245-9540  
ARPA Order Number - ARPA 1228  
Program Code Number - 8D30

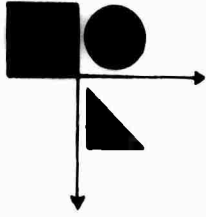
Approved for public release  
Distribution Unlimited

Contractor:	Massachusetts Computer Associates, Inc., Division of ADR
Contract No.:	DAH04-68-C-0043
Effective Date:	21 June 1968
Expiration Date:	30 September 1971
Amount:	\$696,800.00

DRDC  
RECEIVED  
MAR 19 1971  
D.

Sponsored by  
Advanced Research Projects Agency  
ARPA Order Number - 1228

CA-7102-2613



# APPLIED DATA RESEARCH, INC.

LAKESIDE OFFICE PARK, WAKEFIELD, MASSACHUSETTS 01880 • (617) 245-9540

A REPORT ON AMBIT/G  
(Volume III)

by

Carlos Christensen  
Michael S. Wolfberg  
Michael J. Fischer \*

CA-7102-2613  
February 26, 1971

\*Consultant to Applied Data Research, Inc.  
Address: Department of Mathematics, M.I.T.,  
Cambridge, Massachusetts

This is the third of four volumes of the final report on Task Area I of the project "Research in Machine-Independent Software Programming". This research was supported by the Advanced Research Projects Agency of The Department of Defense and was monitored by U.S. Army Research Office-Durham, Box CM, Duke Station, Durham, North Carolina 27706, under Contract DAHC04-68-C-0043.

## CONTENTS

### Volume I

	<b>Abstract</b>	
1.	Summary	1
2.	Fundamentals data graph, constraints, program, general philosophy specific languages.	3
3.	Representation of Programs overview, program syntax, correspondence between program graphs and diagrams.	15
4.	The Interpreter overview, the compiler, interpretation of 'linkrep's, user-defined functions, error messages.	30
5.	The Loader overview, error messages, loader syntax, sample encodement, sample error.	48
6.	Initialization and the Built-in System hints, built-in nodes, built-in links, built-in function definitions, built-in rules, built-in data, built-in functions, sample error.	65
7.	The Debugging Facility lexical conventions, statements, statement forms, sample session.	99
8.	The Implementation credits and acknowledgements, internal view, files, PL/I data formats, PL/i implementation of the inter- preter and loader.	114
9.	Further Work	152
10.	Project Bibliography	157

### Volume II

11.	Examples of AMBIT/G Programs observations, introductory examples: reversing a list, two forms of input, function calling, LISP gar- bage collector, another garbage collector, an inter- active program, sorting, factorial computation and recursion.	
-----	---	--

### Volume III

12.	The AMBIT/G Interpreter as an AMBIT/G Program description, listing.	
-----	--	--

### Volume IV

13.	The AMBIT/G Loader as an AMBIT/G Program description, listing.	
-----	---	--

## CHAPTER 12

### THE AMBIT/G INTERPRETER AS AN AMBIT/G PROGRAM

This volume of "A Report on AMBIT/G" describes the AMBIT/G interpreter as an AMBIT/G program, which is, in fact, how it is implemented. We begin by giving a short verbal description of the interpreter where each function of the interpreter is briefly discussed. The remainder of the volume consists of many pages of diagrams which constitute the listing of the interpreter.

The reader who wants to understand the program (rather than just look at the pretty pictures) is urged to be familiar with the contents of Chapters 3 and 4 of this report in Volume I. Although we have attempted to describe completely what the interpreter does, it is the listing of the interpreter which can be used as a final reference for obscure issues. Furthermore, this listing contributes to a formal definition of AMBIT/G. The listing begins with definitions of characteristic shapes of built-in nodes used by the interpreter and links used by the interpreter.

A table of contents for the listing is given at the beginning of the listing.

#### A DESCRIPTION

The AMBIT/G interpreter consists of a main program and ten major functions. In addition, the interpreter includes calls on functions 'tail\_1', 'tail\_2' and 'head\_1' which are used to transmit arguments and results. The listing of the interpreter does not include the definitions of these functions. Another kind of omission in the listing is the write calls on built-ins 'read\_function' and 'write\_function' to define the functions of the interpreter.

Since Chapter 4 (in Volume I) described the interesting actions of the interpreter, we shall only include here individual descriptions of its ten major functions.

### getnode

This function is used to get a node off of a free list. Its first argument is the head node of some free list, and its second argument is the terminator node for the list. The result of the function is a node off of the front of the list if there is one. Otherwise, 'locate' is used to find a node of the appropriate type which is the result of the function.

### freelist

This function is called as a write function with two tail arguments and a head argument. Its first tail argument is the head node of some free list, and its second tail argument is the terminator node for the list. The head argument is a list of zero or more nodes of the type corresponding to the tail arguments; the effect of this function is to return this list to the free list specified by the tail arguments.

### arglist

This function is used to prepare a 'tails' argument list or a 'heads' argument list for making a call on a user function. Its first argument is a 'diamond' which is the destination of either the 'org' link or 'dest' link of the 'linkrep' which represents the user function call about to be made. The second argument of 'arglist' is either 'flag def' or 'flag undef'. If the second argument is 'flag def', this function forms as its result, a list of 'pipe' nodes. The 'value' link of each 'pipe' node points to a user node which is the destination of the 'rep' link of the 'noderep' pointed to by the corresponding 'diamond' of the list supplied as first argument. If, however, the second argument is 'flag undef', this function forms as its result a list of 'pipe' nodes whose 'value' links point to the node 'undef undef'. The result list then has the same length as the length of the list supplied as first argument.

### typelist

This function is used to prepare a list of 'type' nodes connected by 'cell's for a read call on either the built-in 'read\_function' or 'write\_function'.

Its single argument is a 'diamond' which is the destination of the 'org' link of the 'linkrep' which represents the link being processed. Its result is a list of 'cell's, where each 'cell' corresponds to a 'diamond' in the (possibly null) list given as argument. The 'value' link of each 'cell' points to the 'type' node corresponding to the type of the destination of the 'rep' link of the 'noderep' pointed to by the corresponding 'diamond'.

#### vset

This function is called as a write function to verify or to set (bind) a 'rep' link of a 'noderep' pointed to by a 'diamond' in a list of 'diamond's hanging off the 'dest' link of a 'linkrep' node. Its first tail argument is the 'noderep' in question, and its second tail argument is the 'diamond' in question which points to that 'noderep'. The one head argument is the user node to be considered. If the 'sets' link of the 'noderep' points to the 'diamond', the 'rep' link of the 'noderep' is set to point at the user node. Otherwise, the destination of the 'rep' link is checked to see if it is the user node; if not, 'circle proceed' is made to point at 'flag no'.

#### lvset

This function is called as a write function to apply the 'vset' function individually to several sets of arguments which correspond to all heads of a read call made on a user function. Its one tail argument is the 'diamond' which is the destination of the 'dest' link of the 'linkrep' representing the user function call which just occurred. Its one head argument is the list of 'pipe's used to transmit the (possibly zero) results of the user function. If the two lists do not have the same number of elements, an error condition occurs. The function ends by returning the given 'pipe' list to the 'pipe' free list.

#### compile

This function is called with one argument which is the 'rule' node which is to be compiled. A section in the chapter on the interpreter (in Vol. I) described this function in detail.

### initnode

This function is called by the compiler to initialize the 'noderep' which is its only argument. If the 'variability' link of the 'noderep' points to 'flag fixed', then its 'sets' link is made to point to 'diamond matched'; or if the 'variability' link of the 'noderep' points to 'flag dummy', then its 'sets' link is made to point to 'diamond unmatched'. Otherwise, an error condition occurs.

### initlist

This function is called by the compiler to initialize a list of 'noderep's connected by 'diamond's. The 'diamond' which heads such a (possibly empty) list is the only argument. The 'initnode' function is applied to every 'noderep' in the list.

### allmatch

This function is called by the compiler to check whether all 'noderep's have been matched on a list of 'noderep's connected by 'diamond's. Its one argument is a 'diamond' which heads such a list. If any 'noderep' in the (possibly empty) list has a 'sets' link pointing to 'diamond unmatched' then the one result is 'diamond unmatched'; otherwise, it is 'diamond matched'.

## THE LISTING

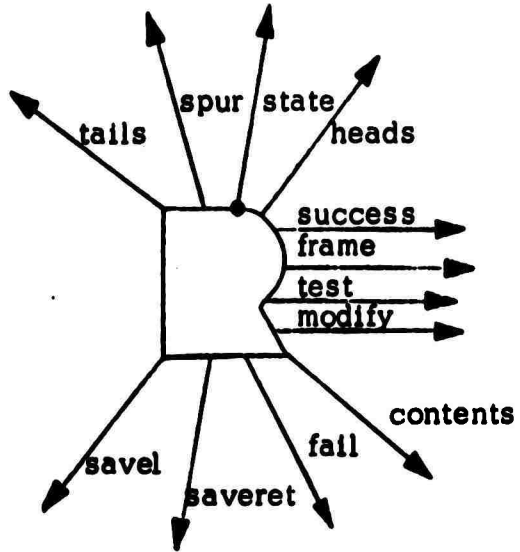
The remainder of this volume consists of the listing of the AMBIT/G interpreter. The following is a table of contents of the listing.

<u>Function</u>	<u>Loader Page Titles *</u>	<u>Starting Page Number</u>
(definitions)	1 - v	6
interpreter	1 - 29	11
getnode	30 - 30a	52
freelist	31 - 31	54
arglist	32 - 32b	55
typelist	33 - 33a	58
vset	34 - 35	60
lvset	36 - 37	62
compile	101 - 109	64
initnode	110 - 110	73
initlist	111 - 111	74
allmatch	112 - 112	75

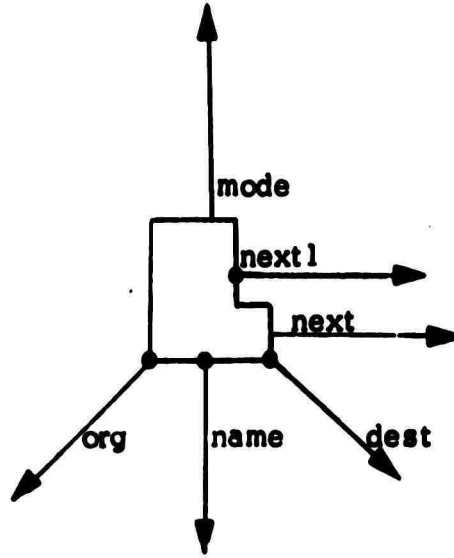
---

\* Note that the loader page titles appear in the upper-right corner of each page.

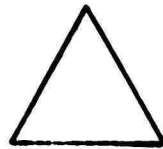
rule :



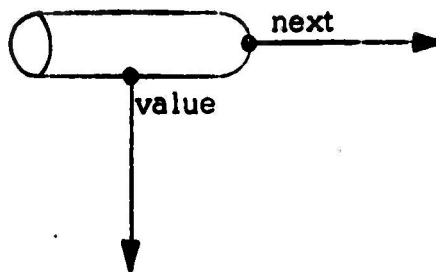
linkrep :



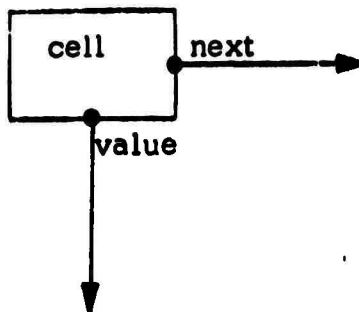
flag:



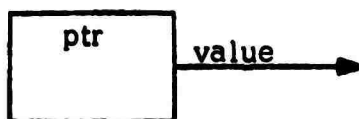
pipe :



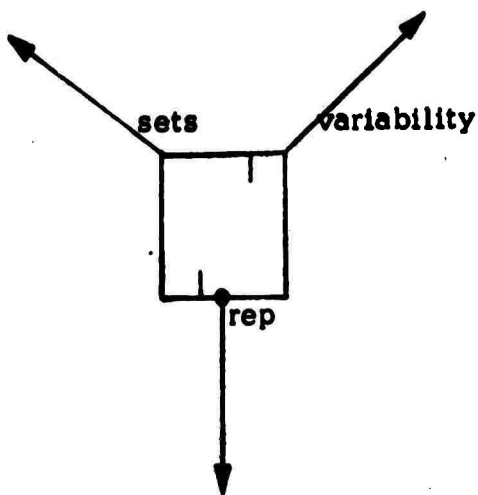
cell :



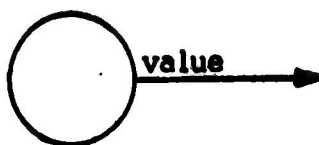
ptr :



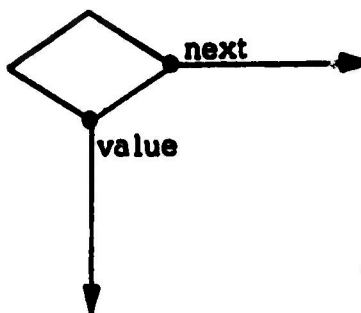
noderep :



circle :



diamond :



**bultin :**

**bultin**

**link:**

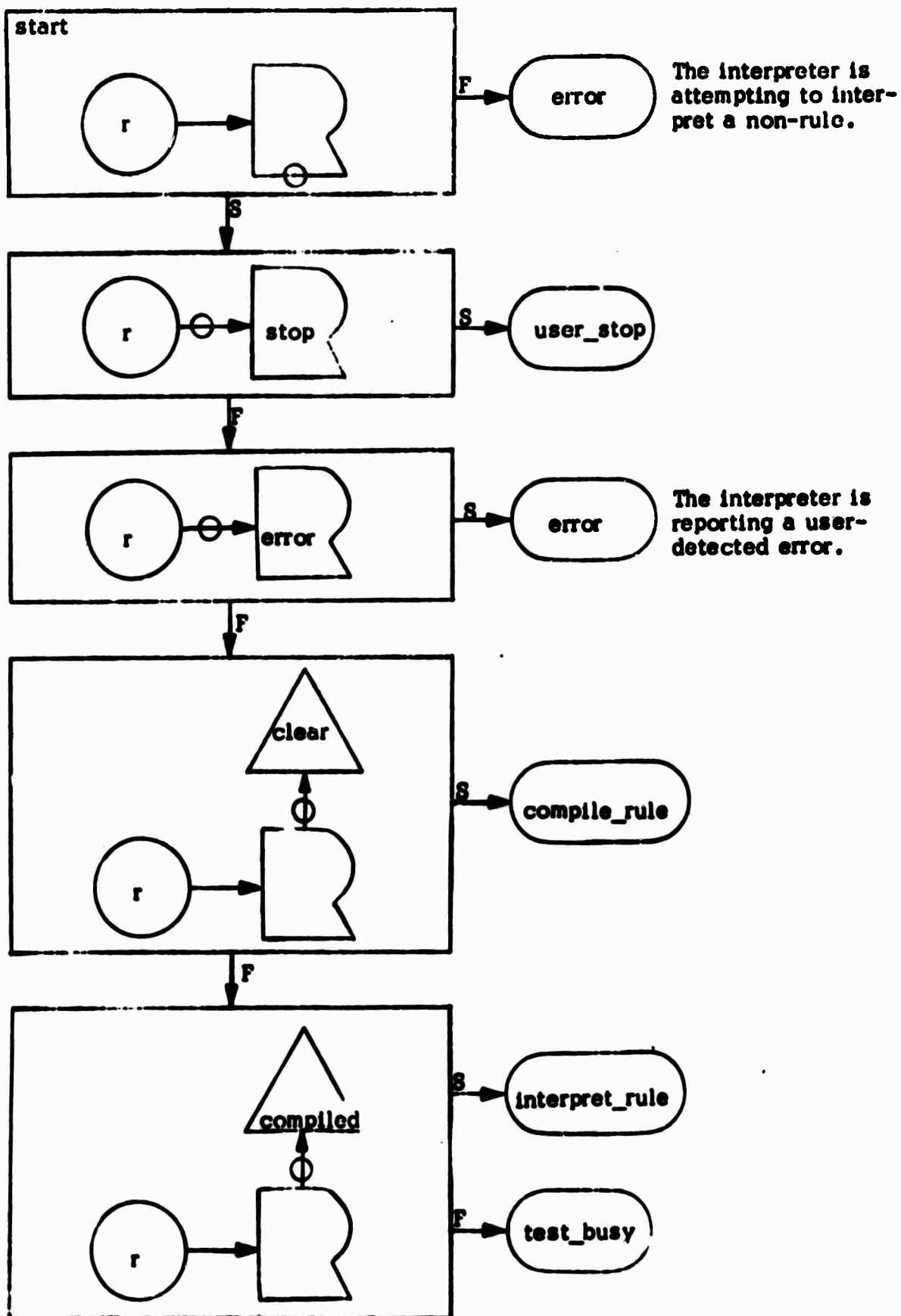
**link**

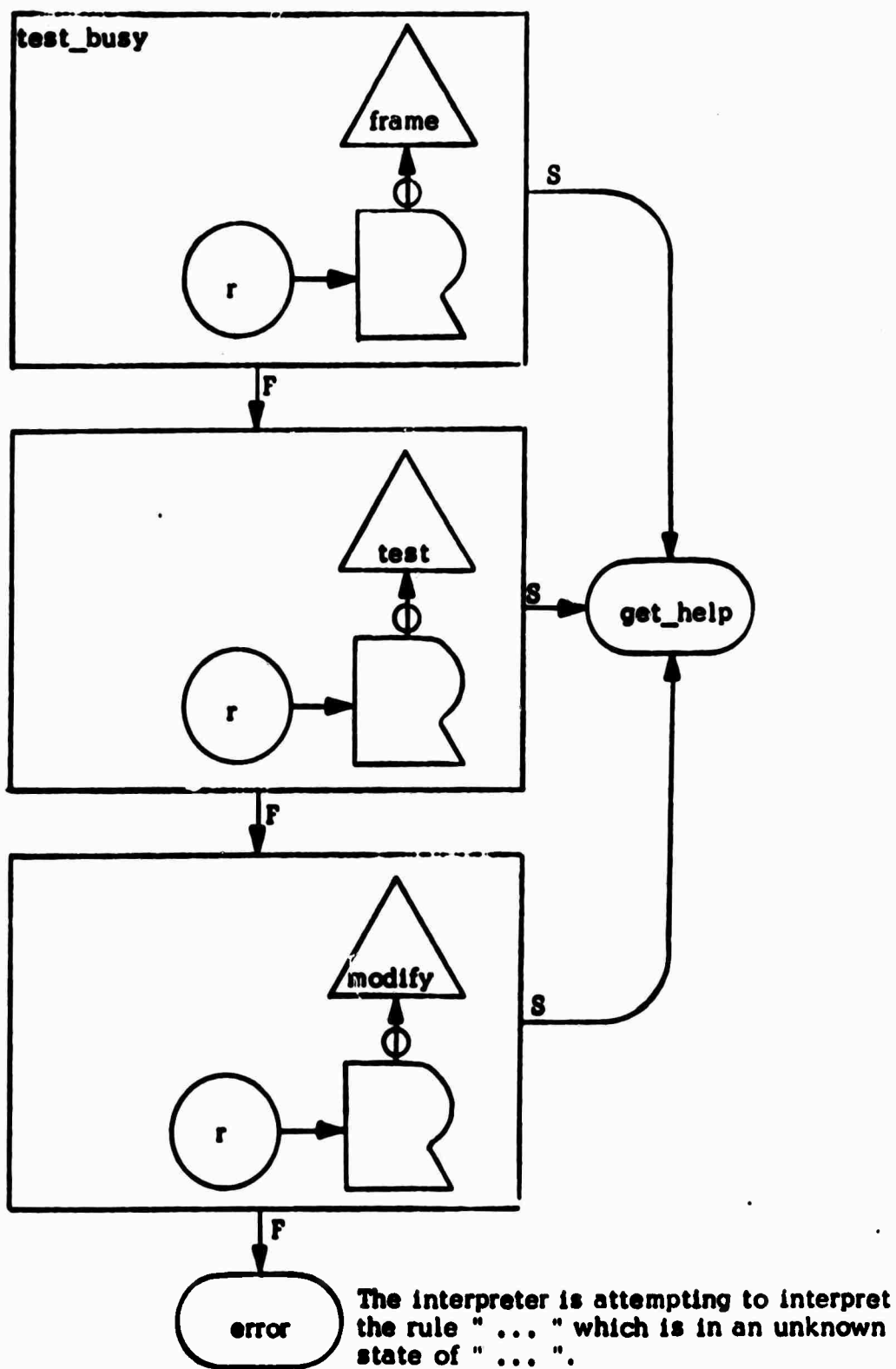
**type :**

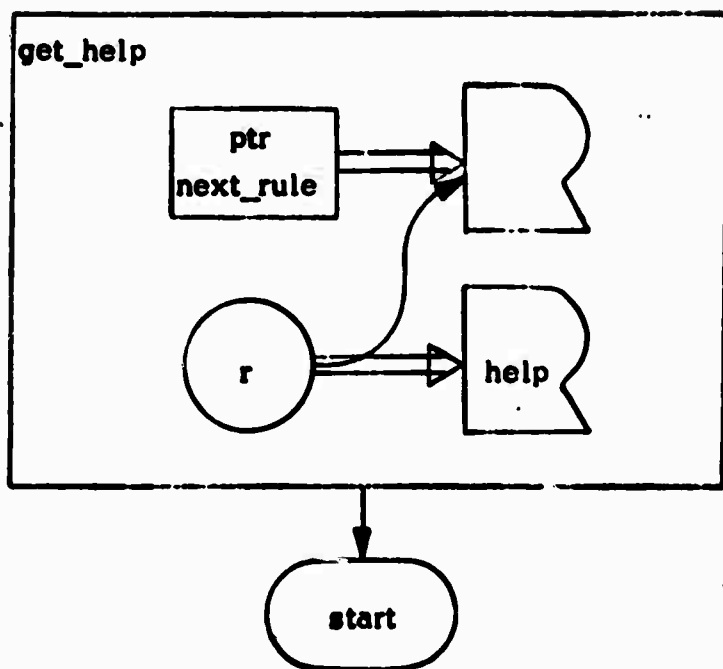


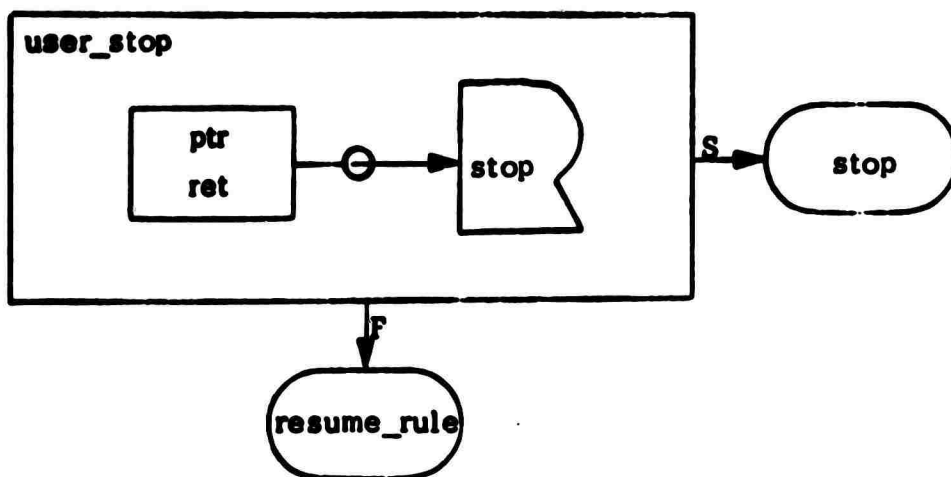
**undef :**

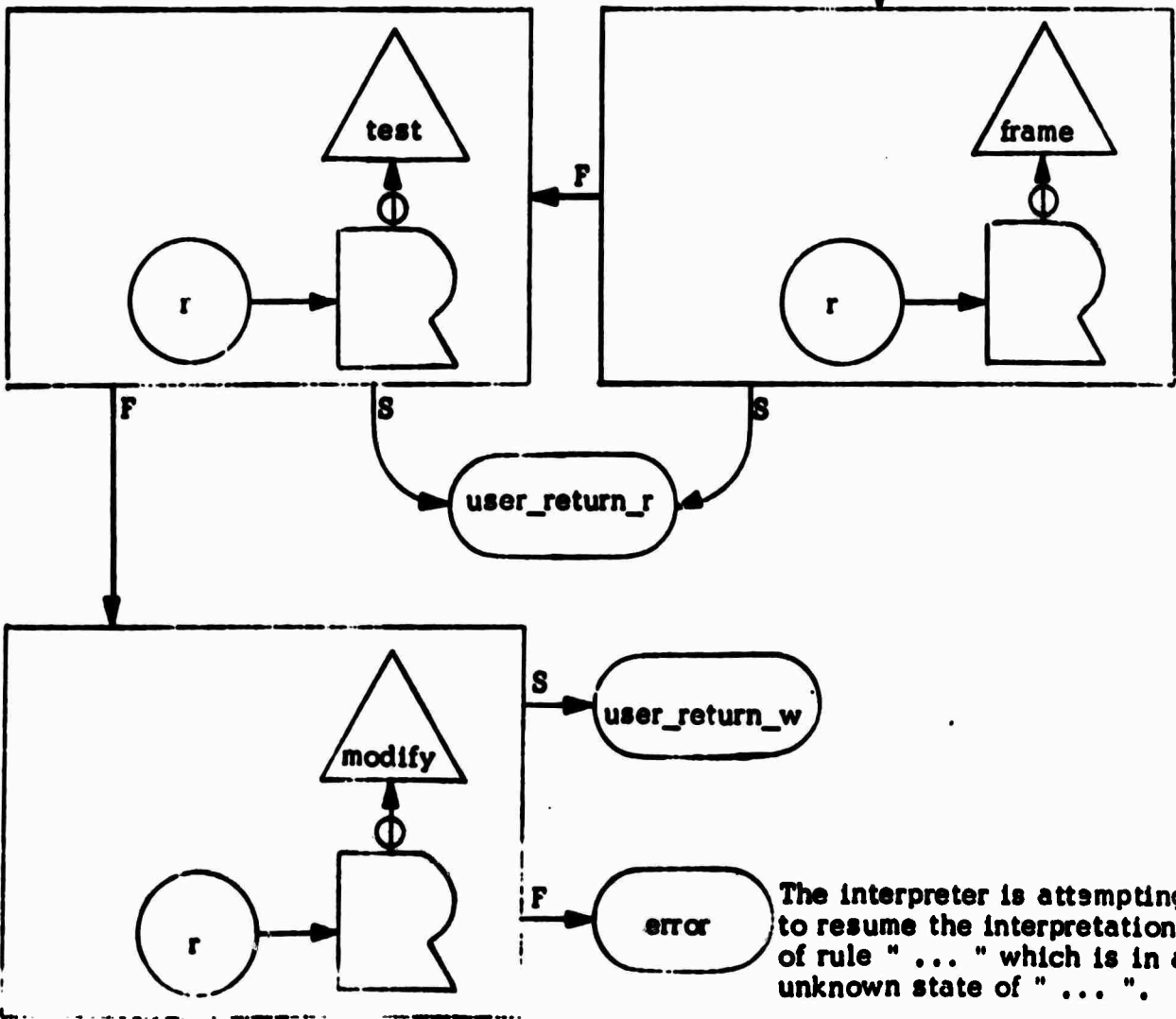
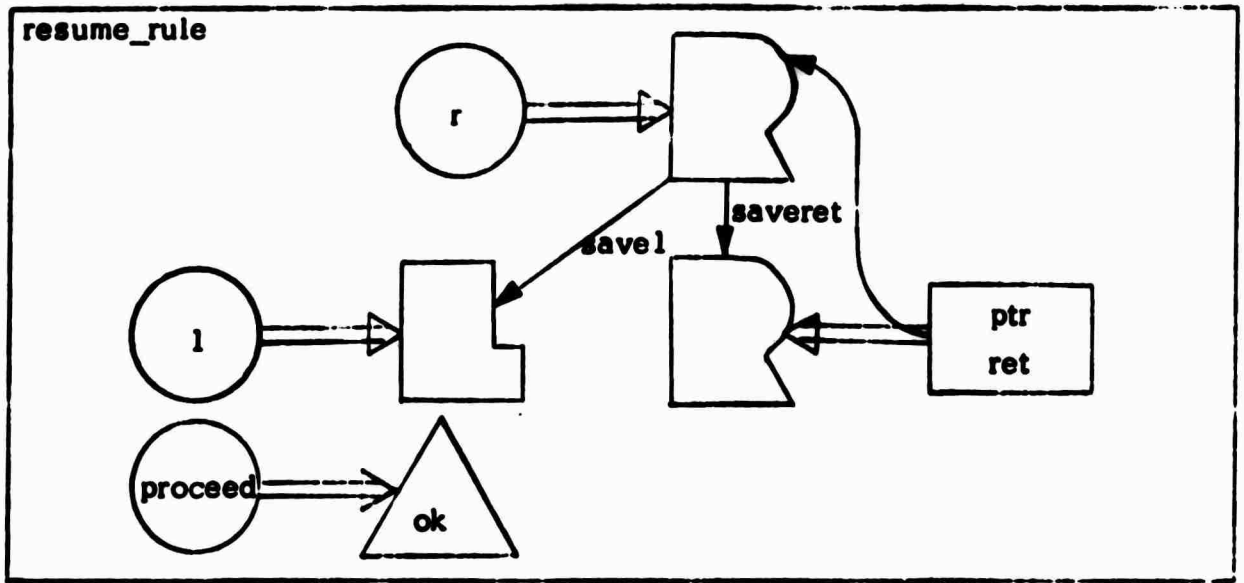
**undef**

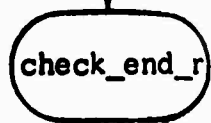
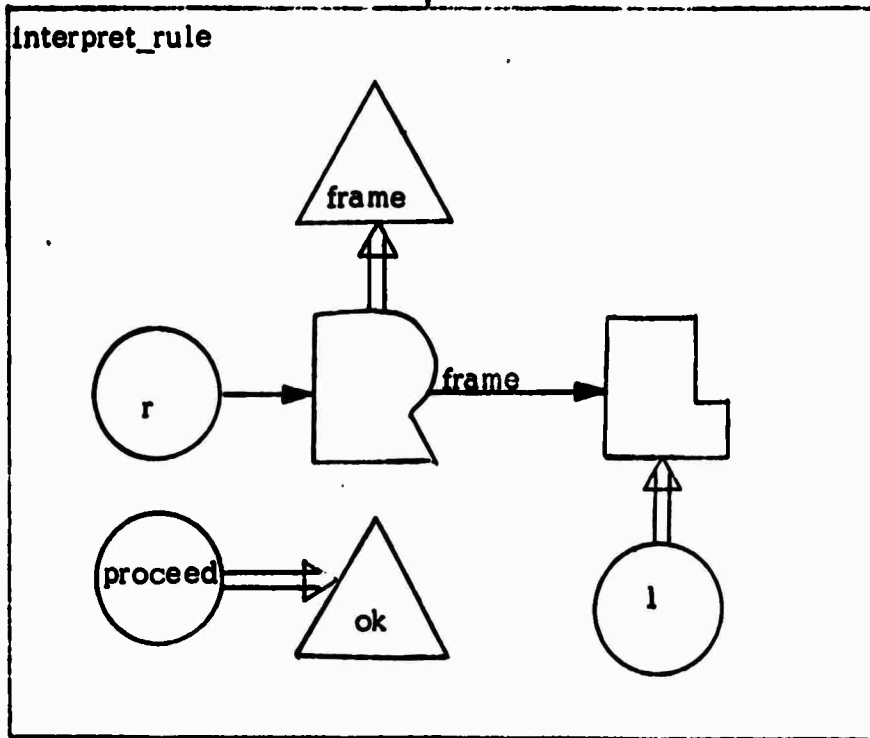
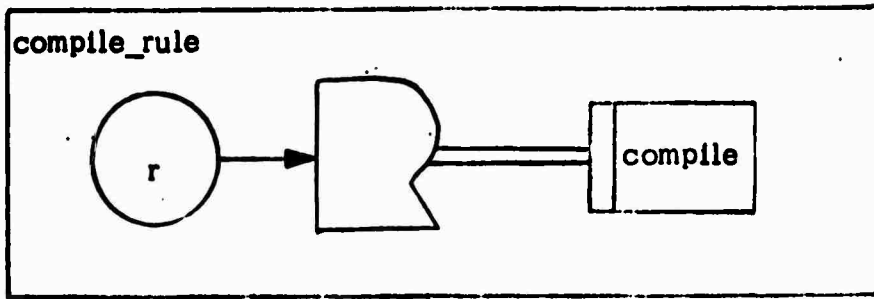


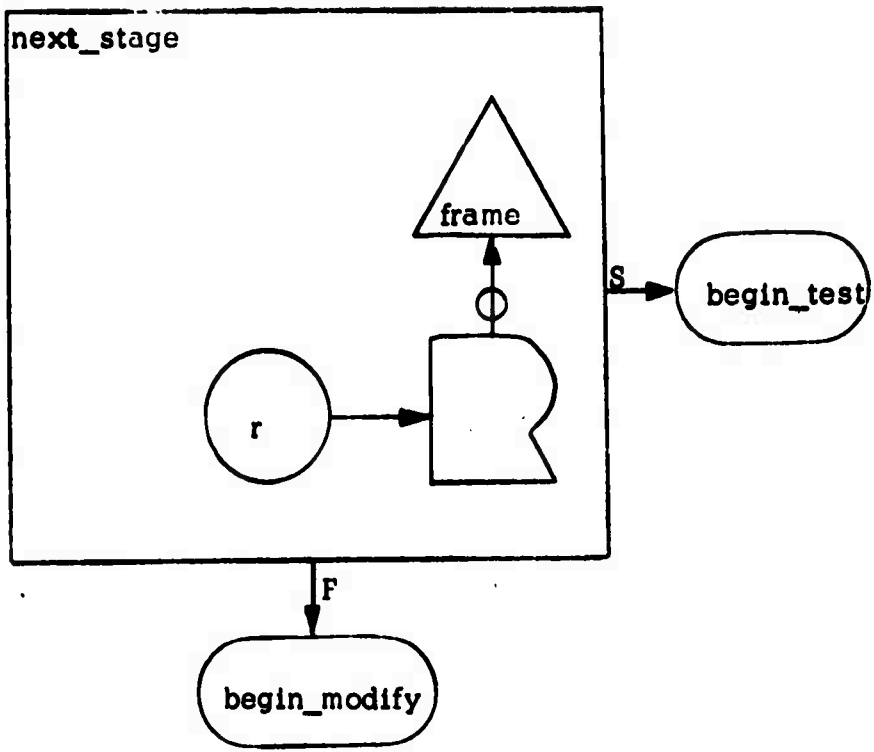


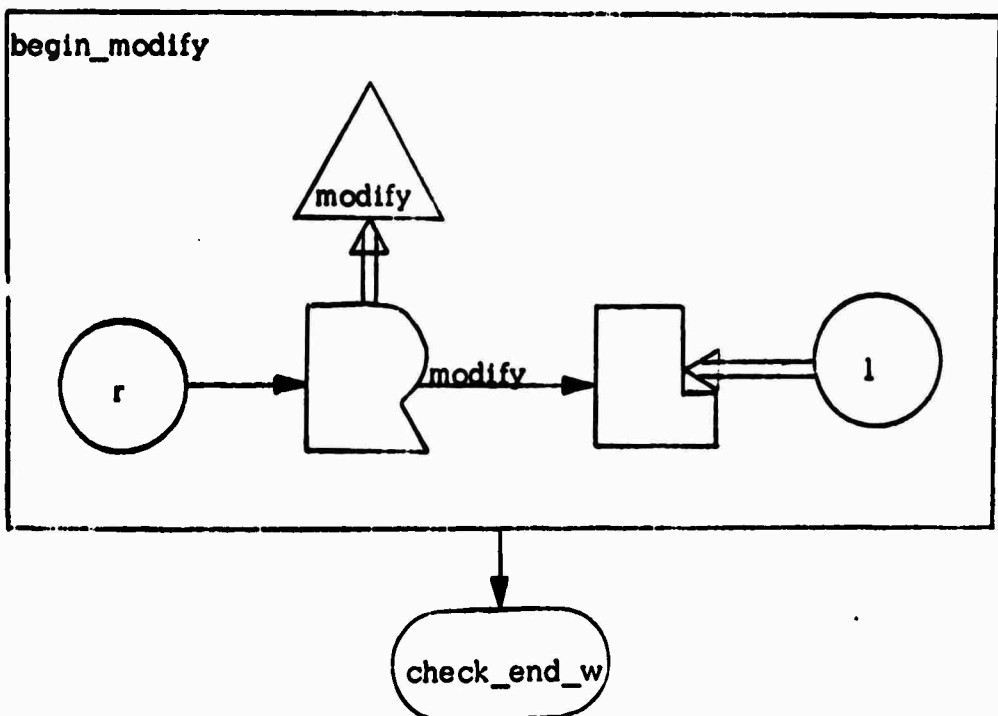
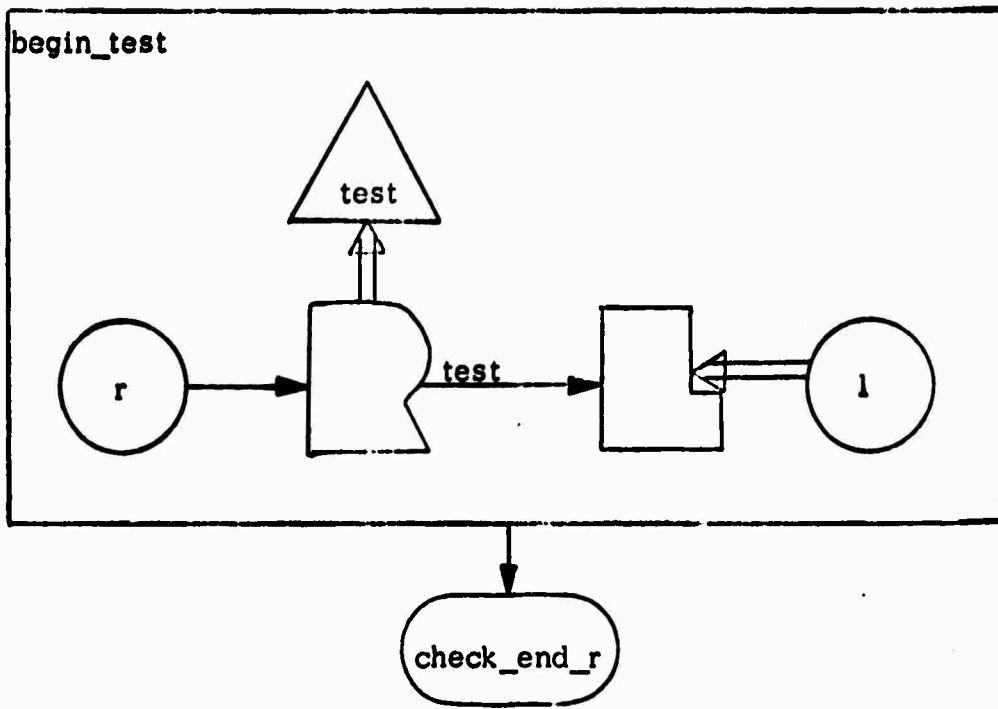


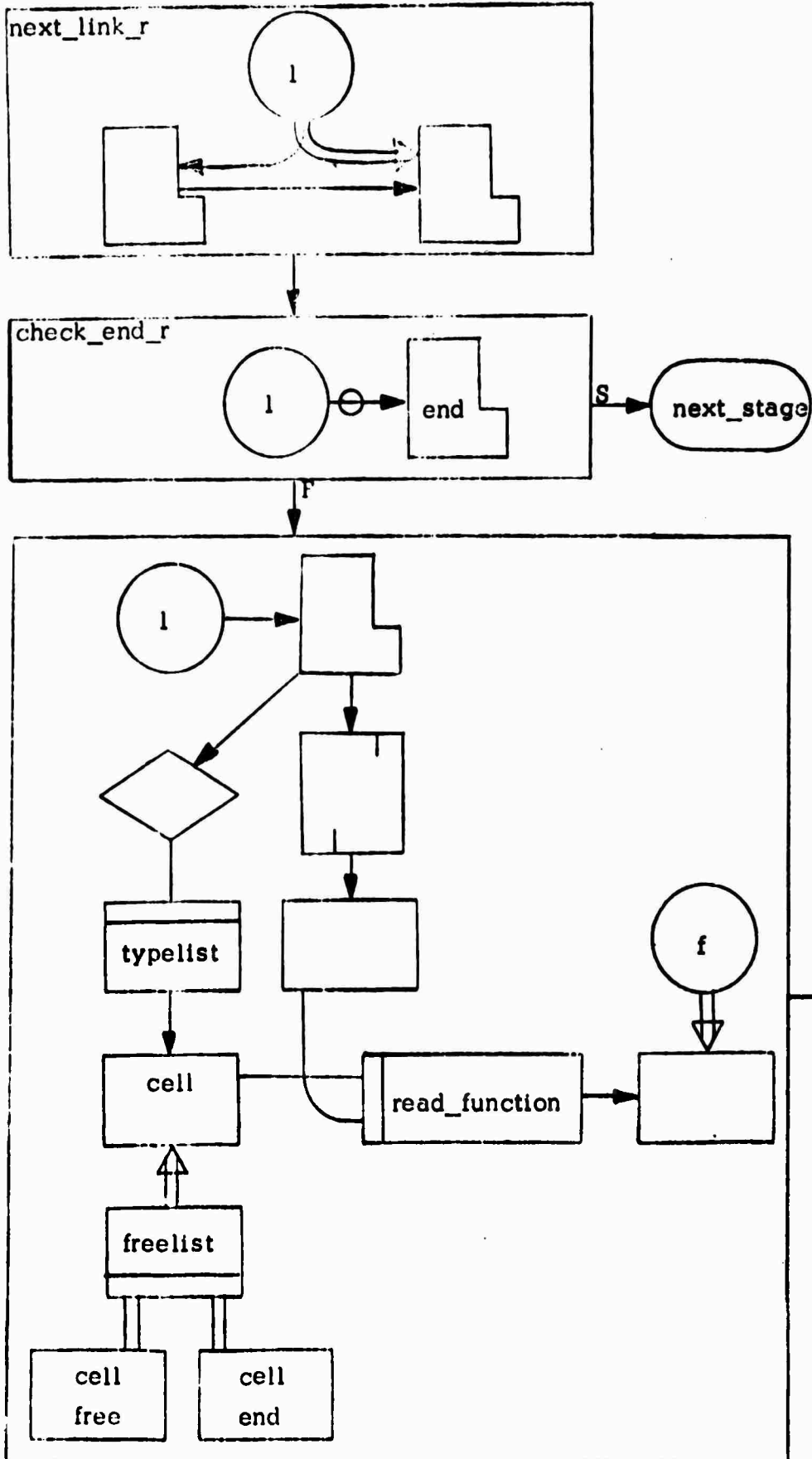


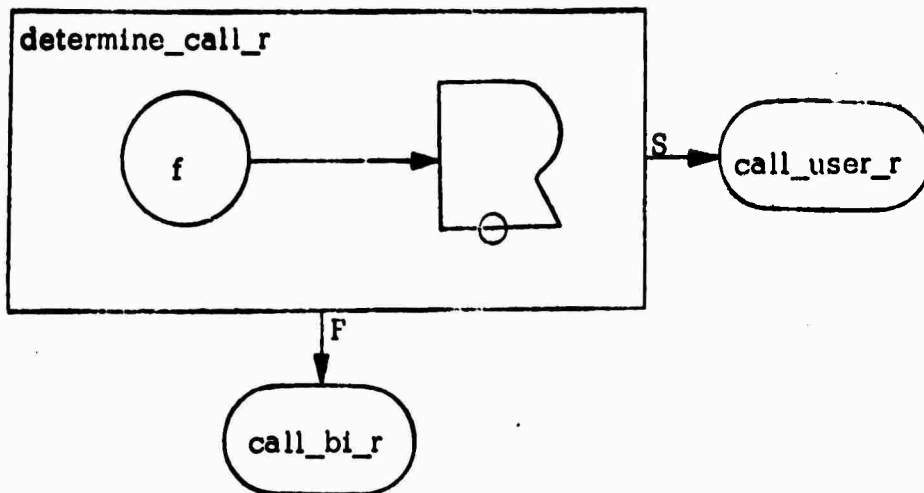


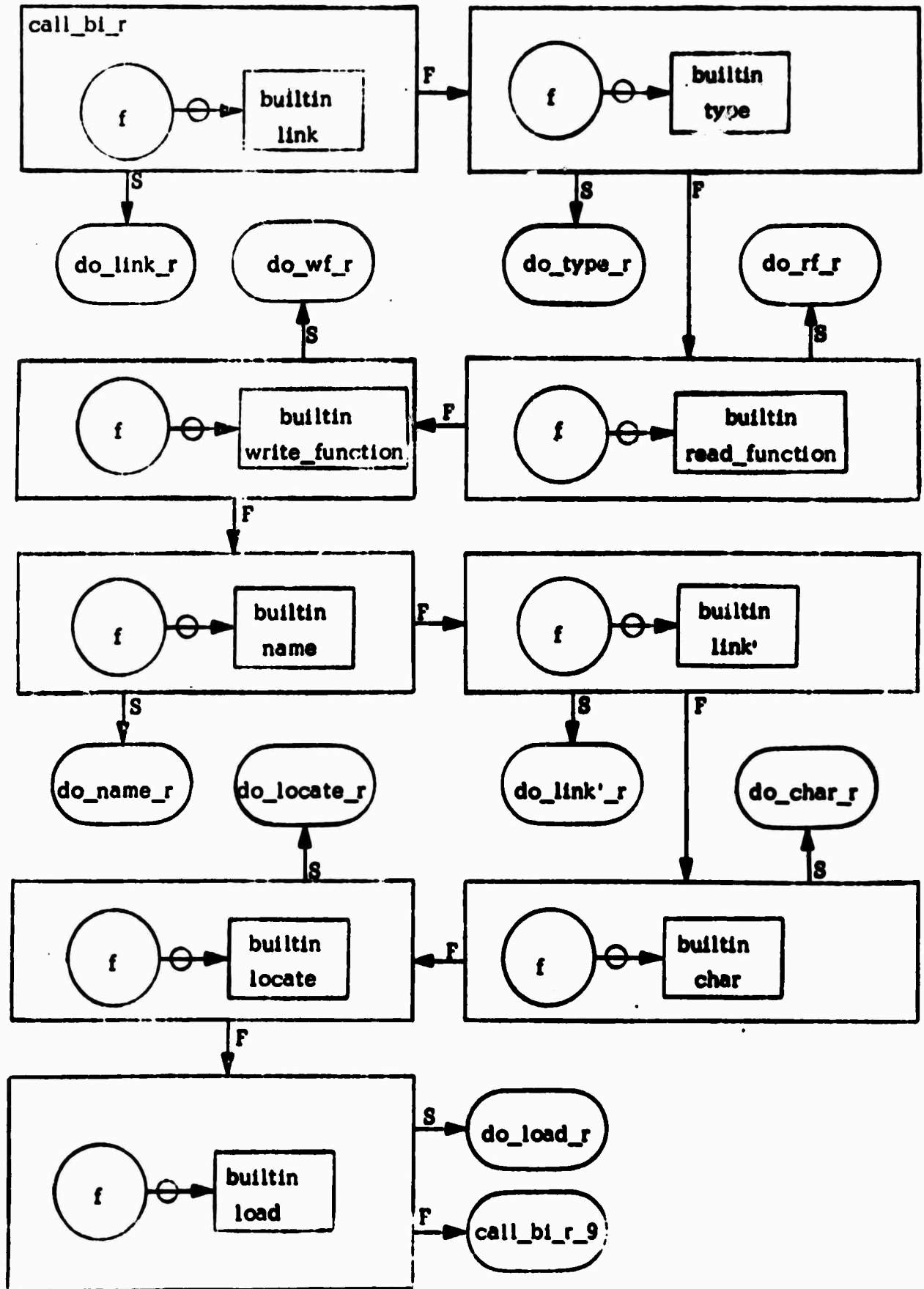


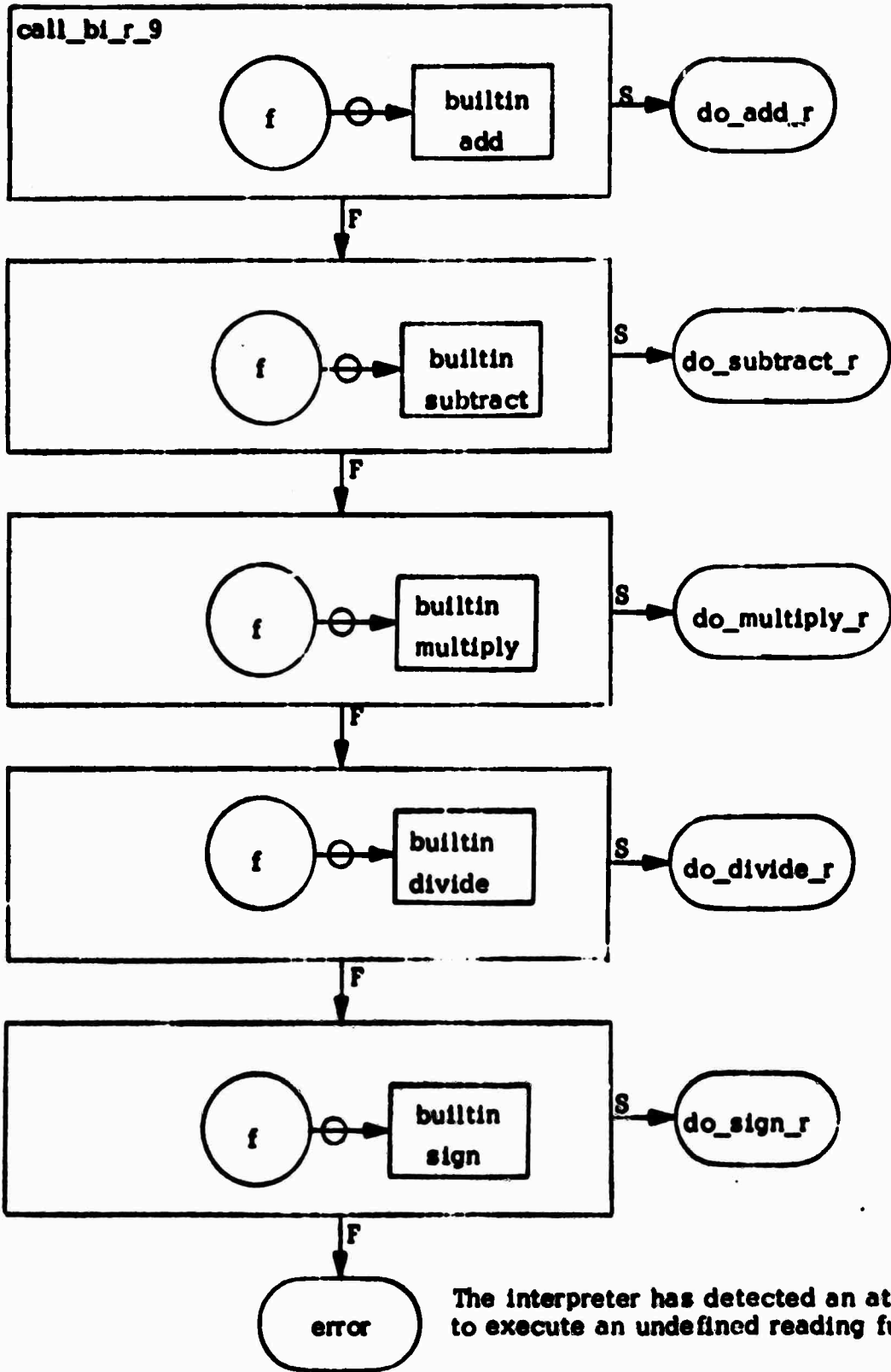


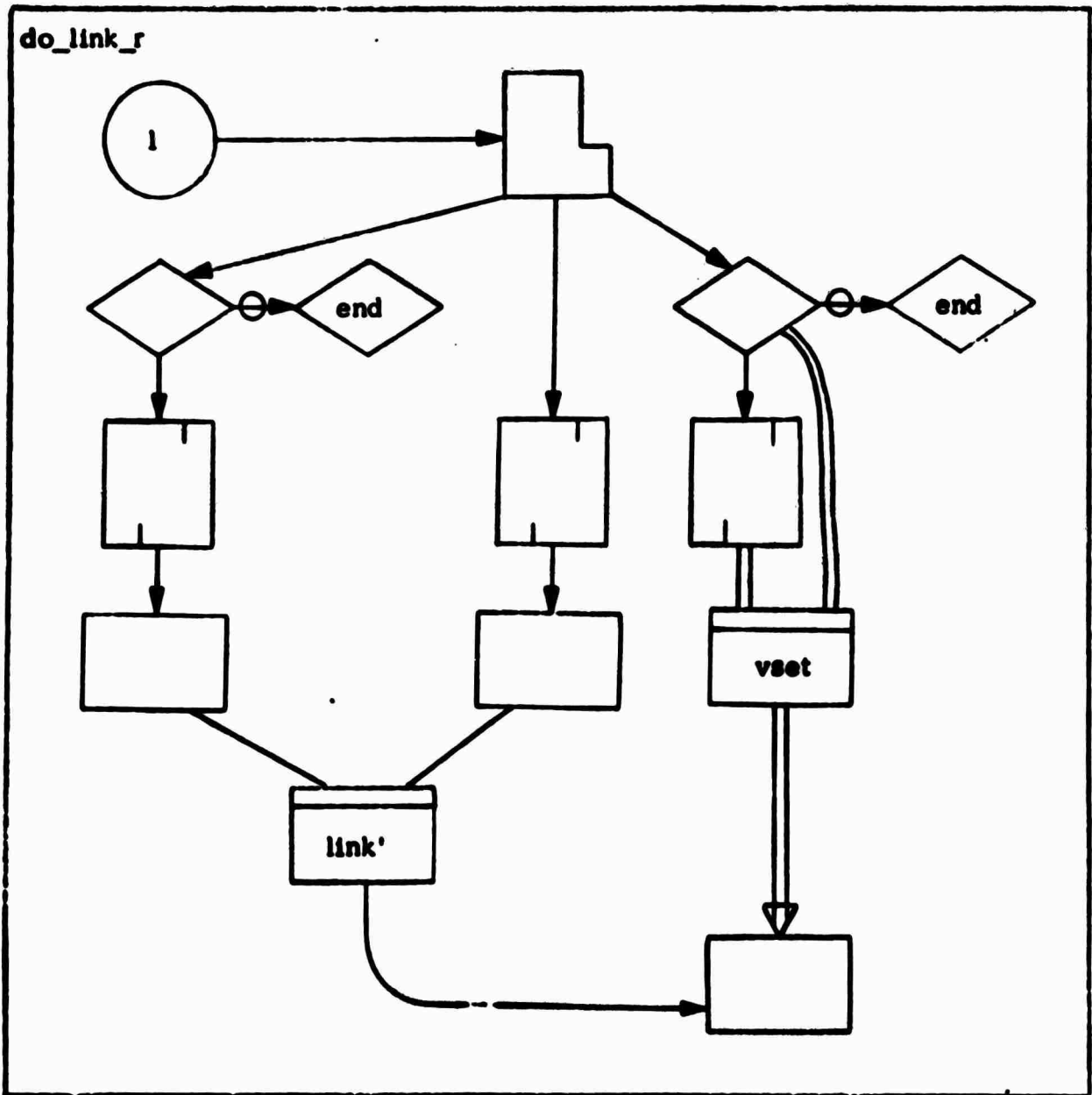








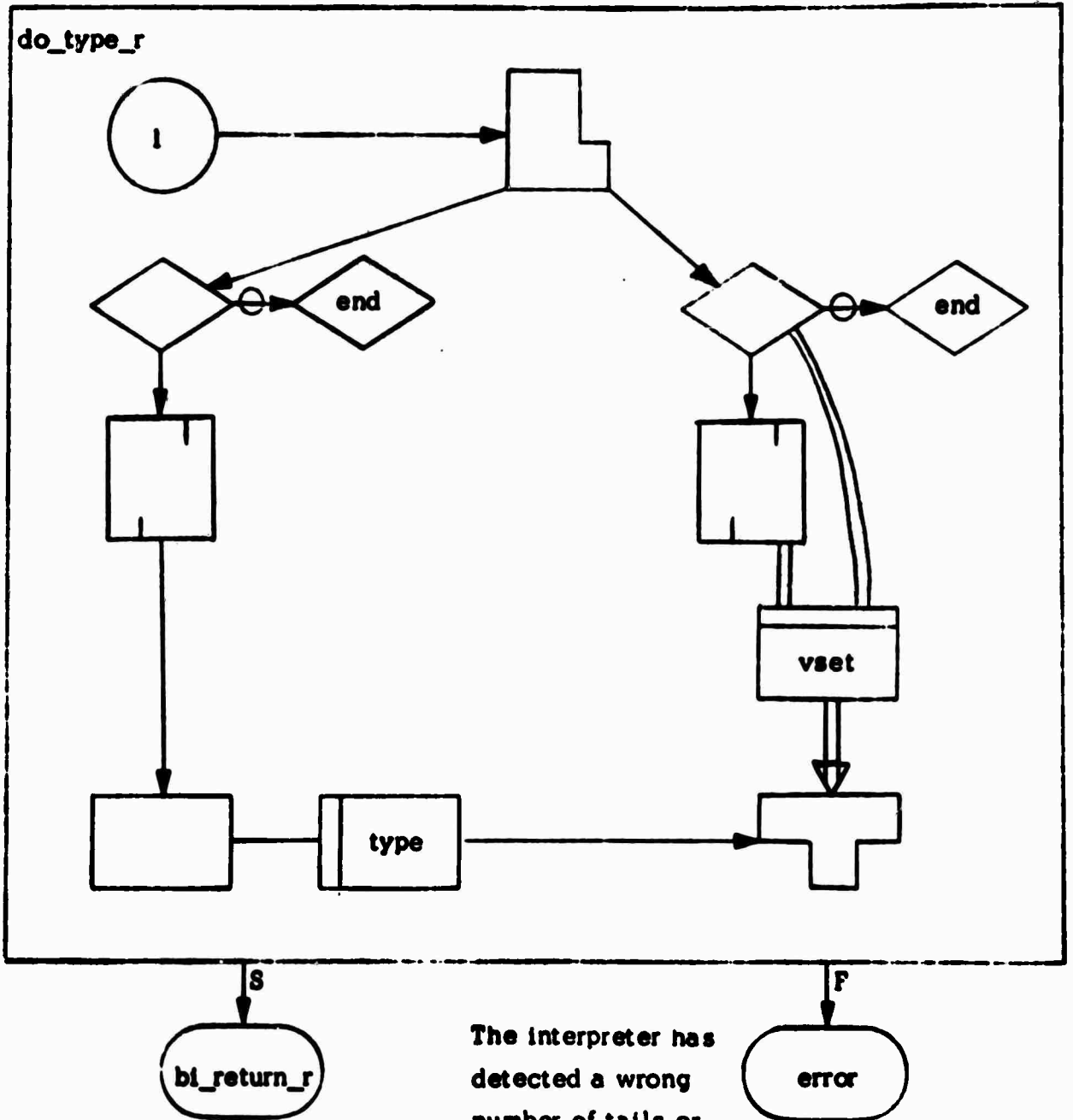




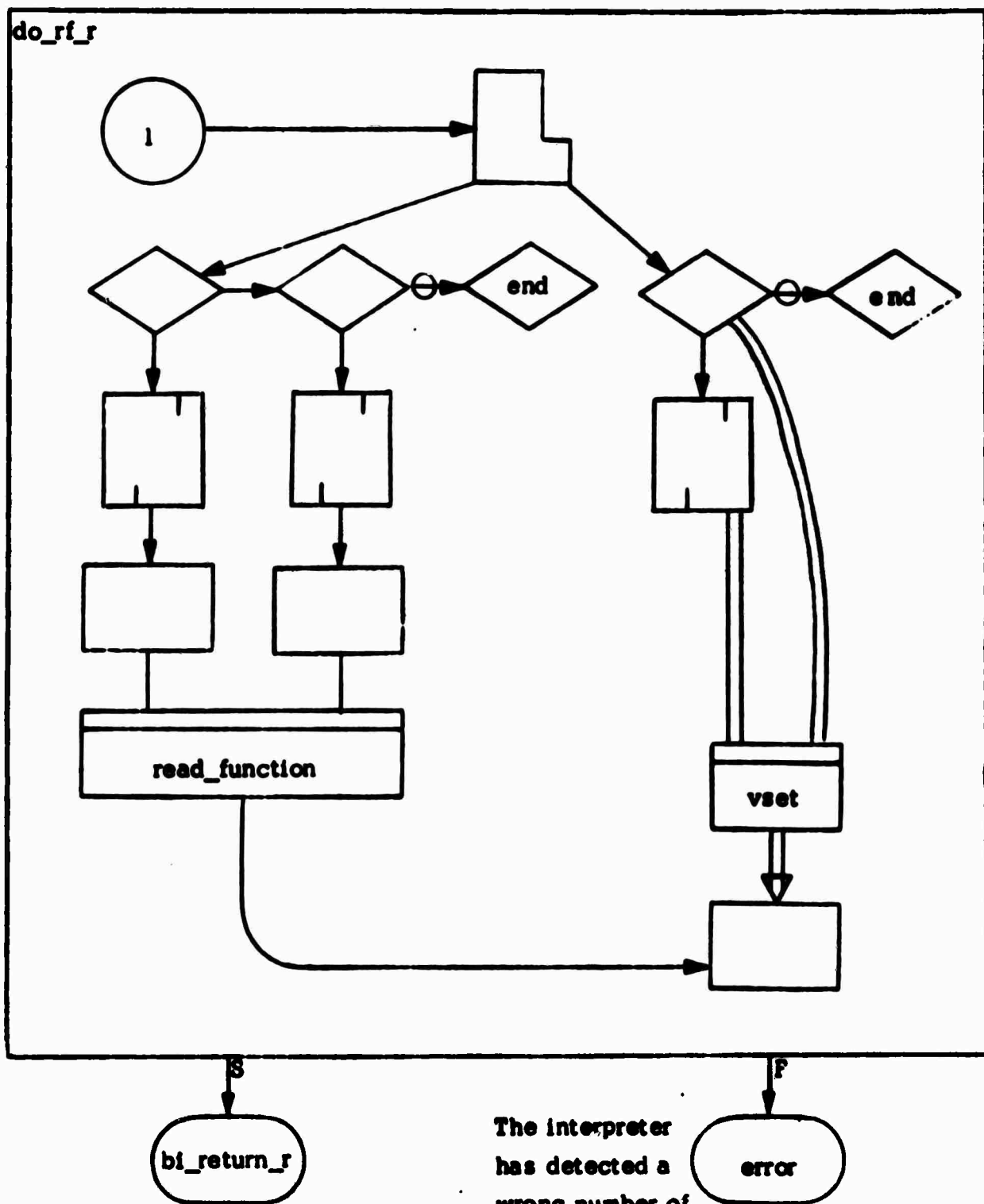
S  
bi\_return\_r

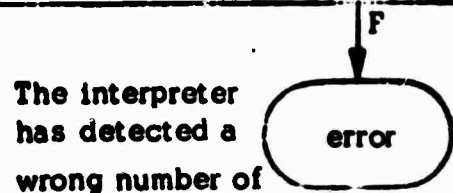
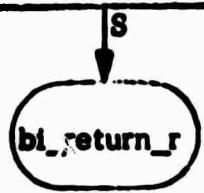
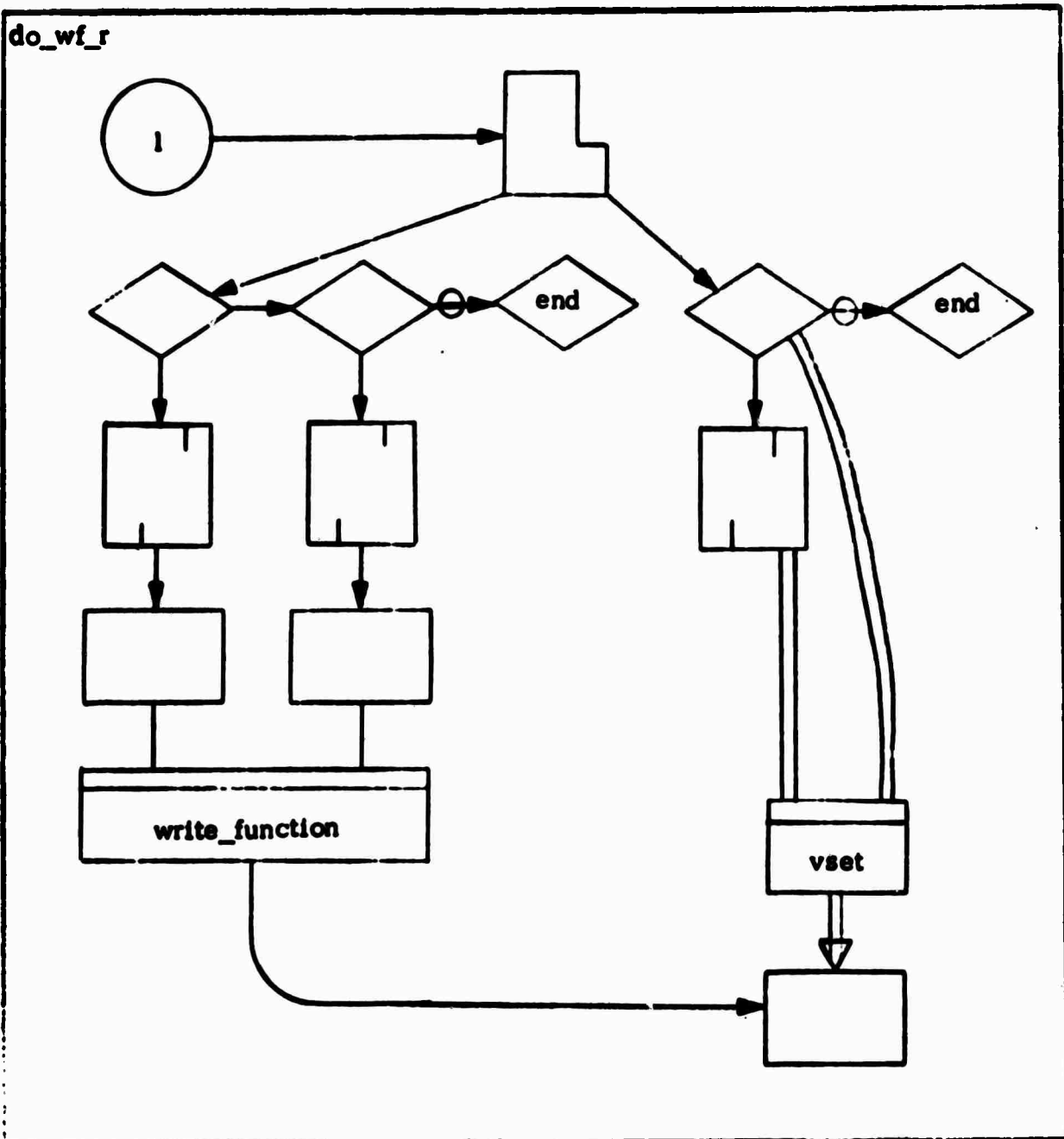
The interpreter has detected a wrong number of tails or heads on a read-call on the builtin "link".

F  
error

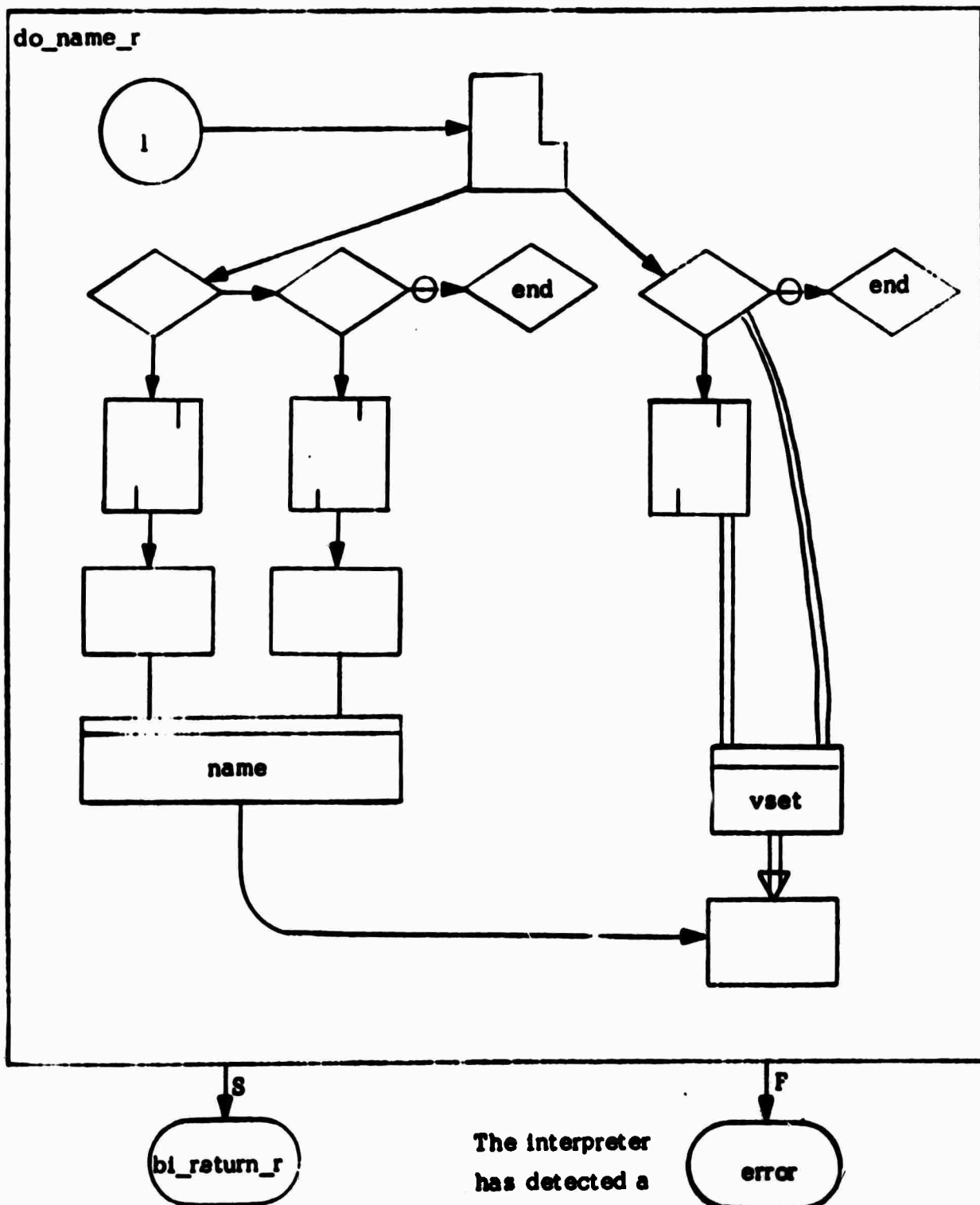


The interpreter has detected a wrong number of tails or heads on a read-call on the builtin "type".

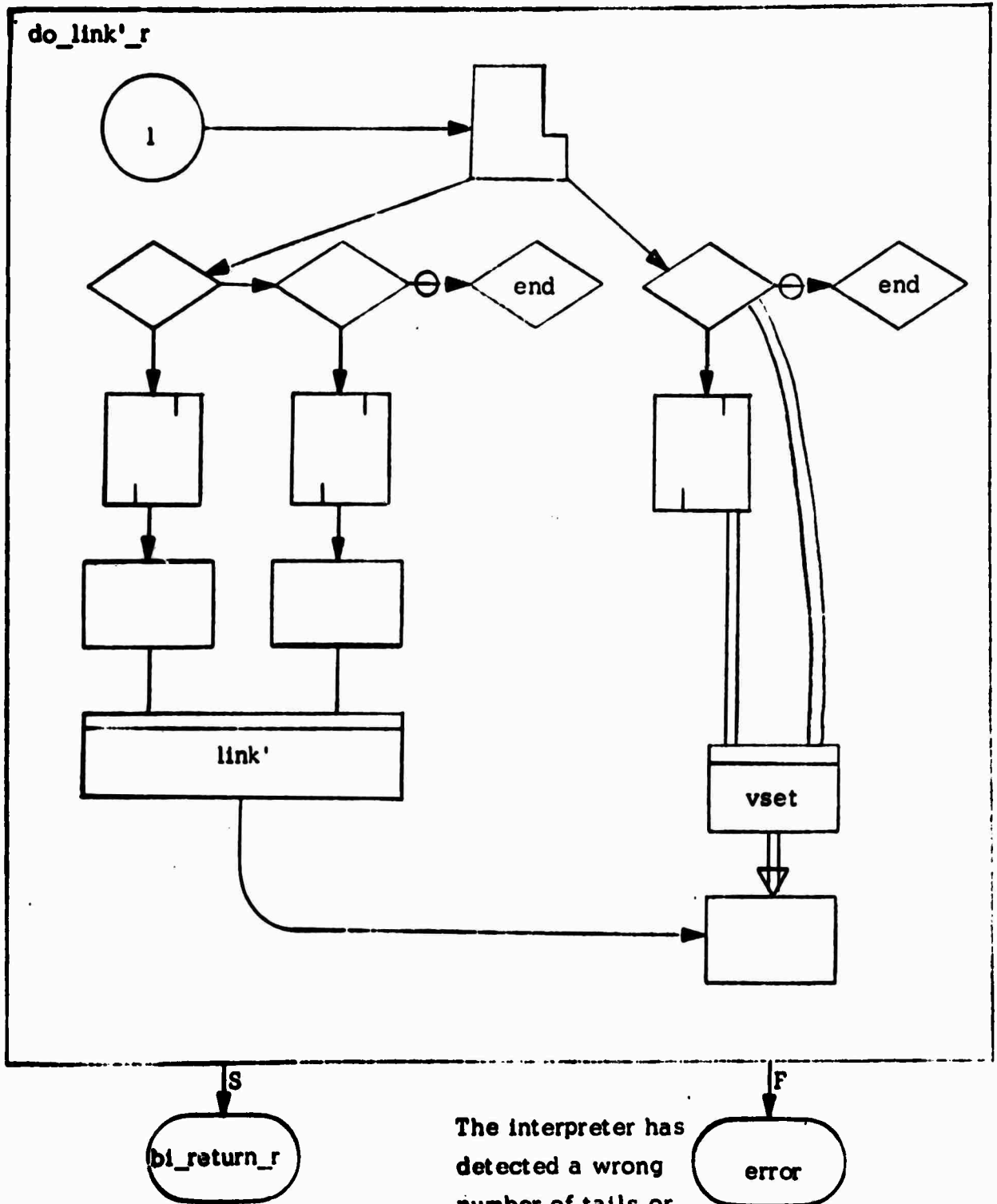


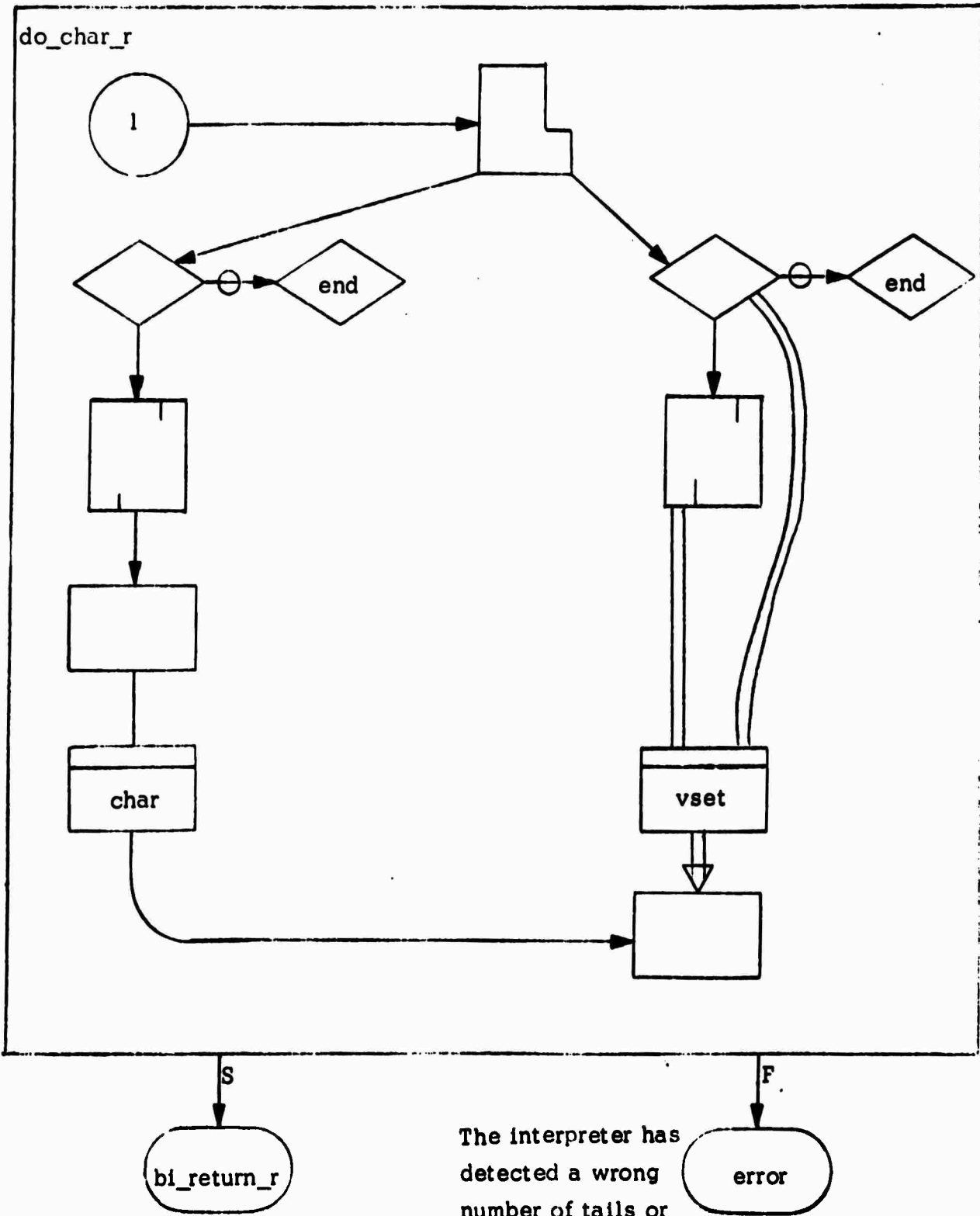


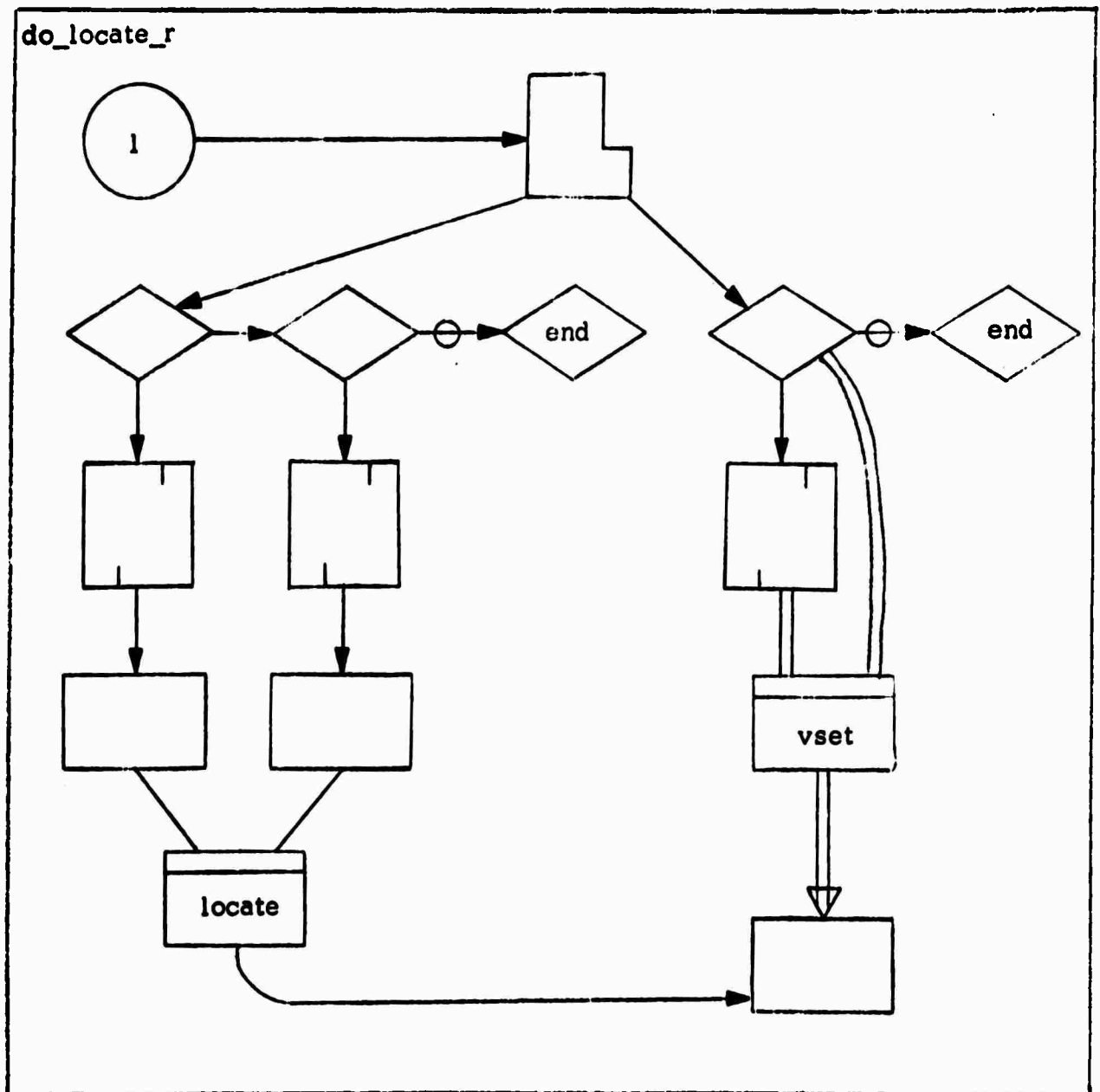
The interpreter has detected a wrong number of tails or heads on a read-call on the builtin "write\_function".



The interpreter has detected a wrong number of tails or heads on a read-call on the builtin "name".



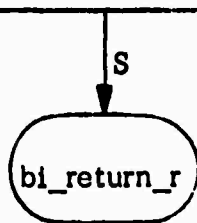
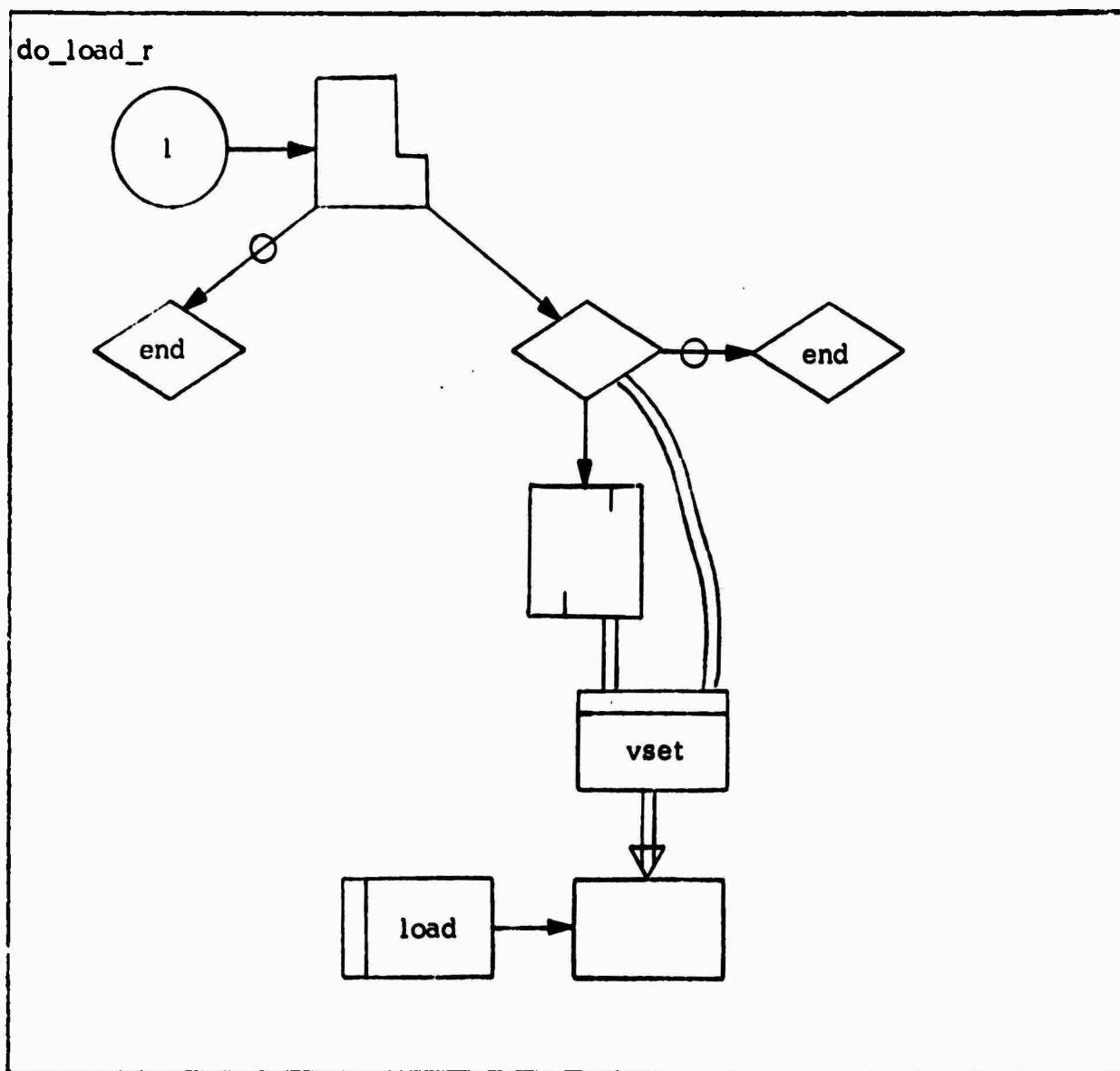




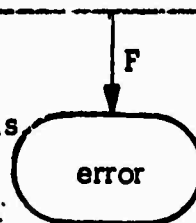
S  
bi\_return\_r

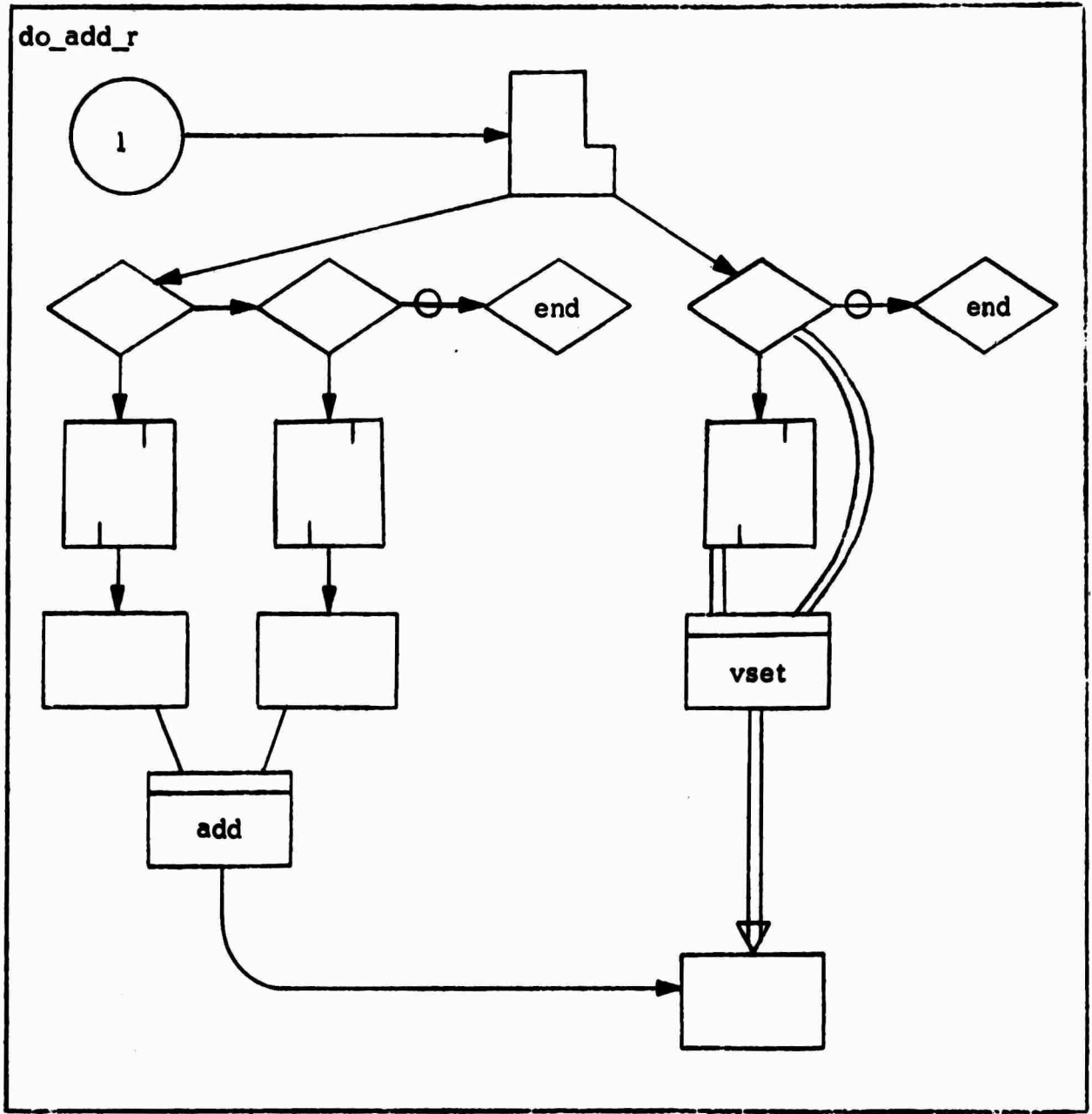
The interpreter has detected a wrong number of tails or heads on a read-call on the builtin "locate".

F  
error



The interpreter has detected a wrong number of tails or heads on a read-call on the builtin "load".

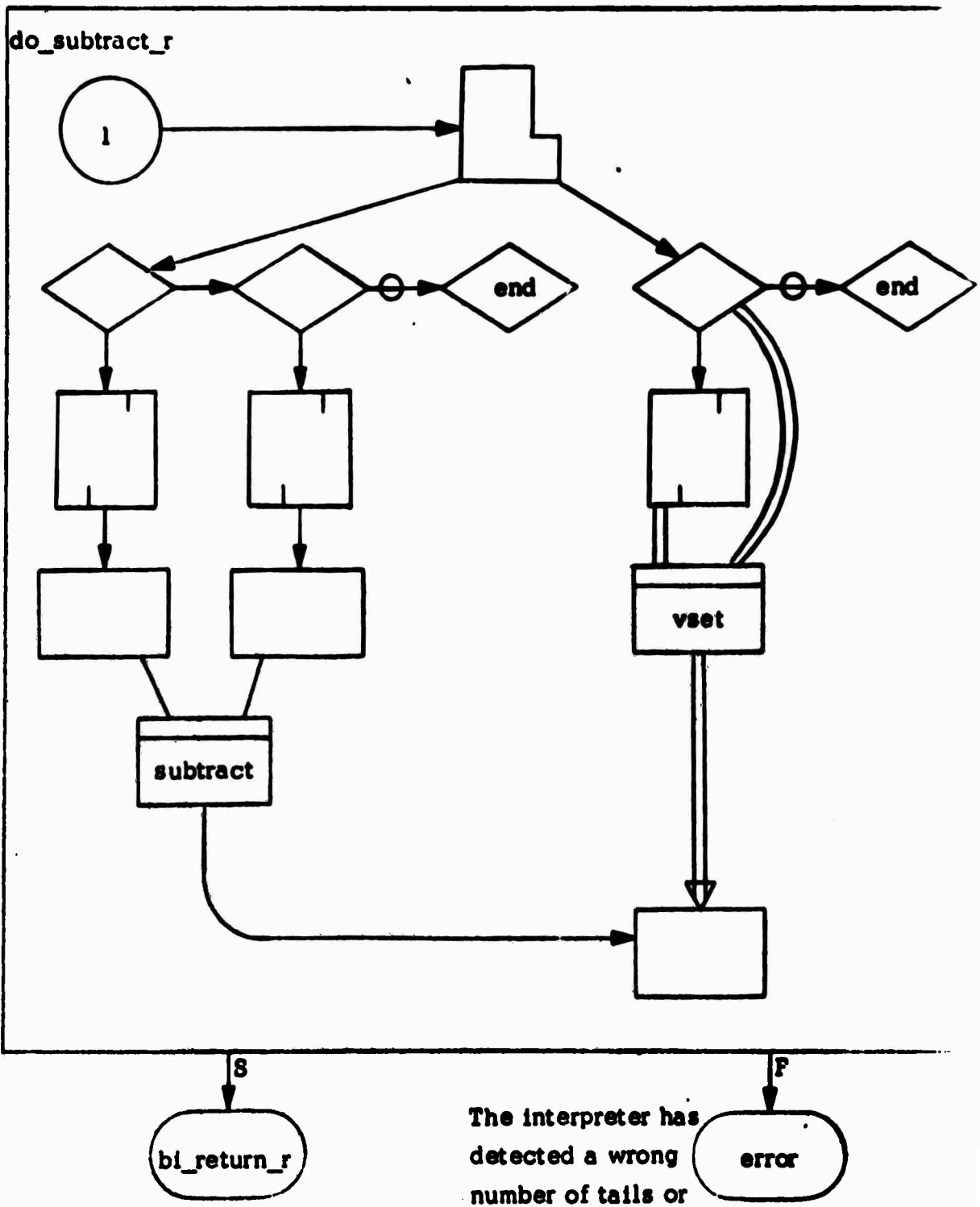




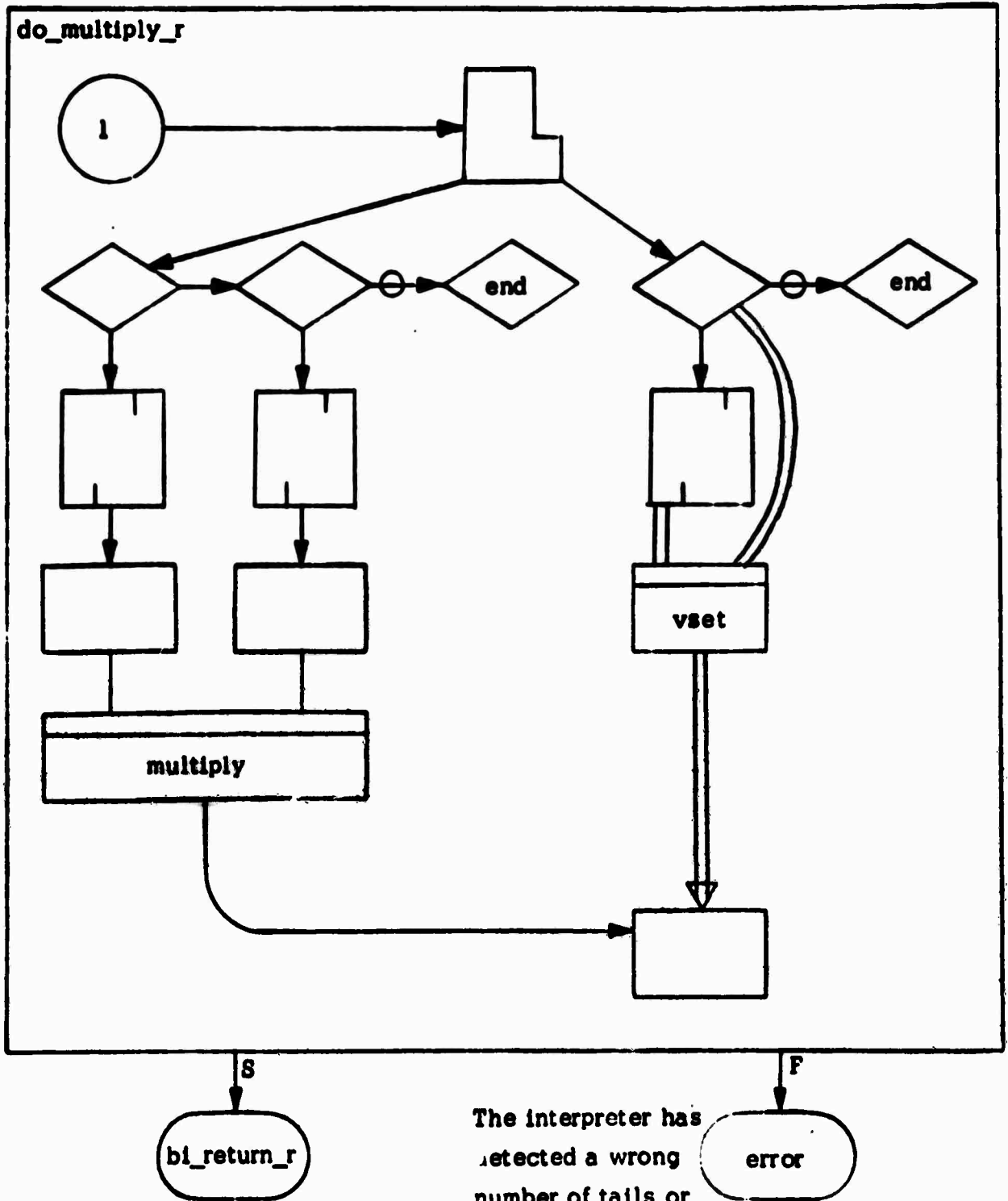
bi\_return\_r

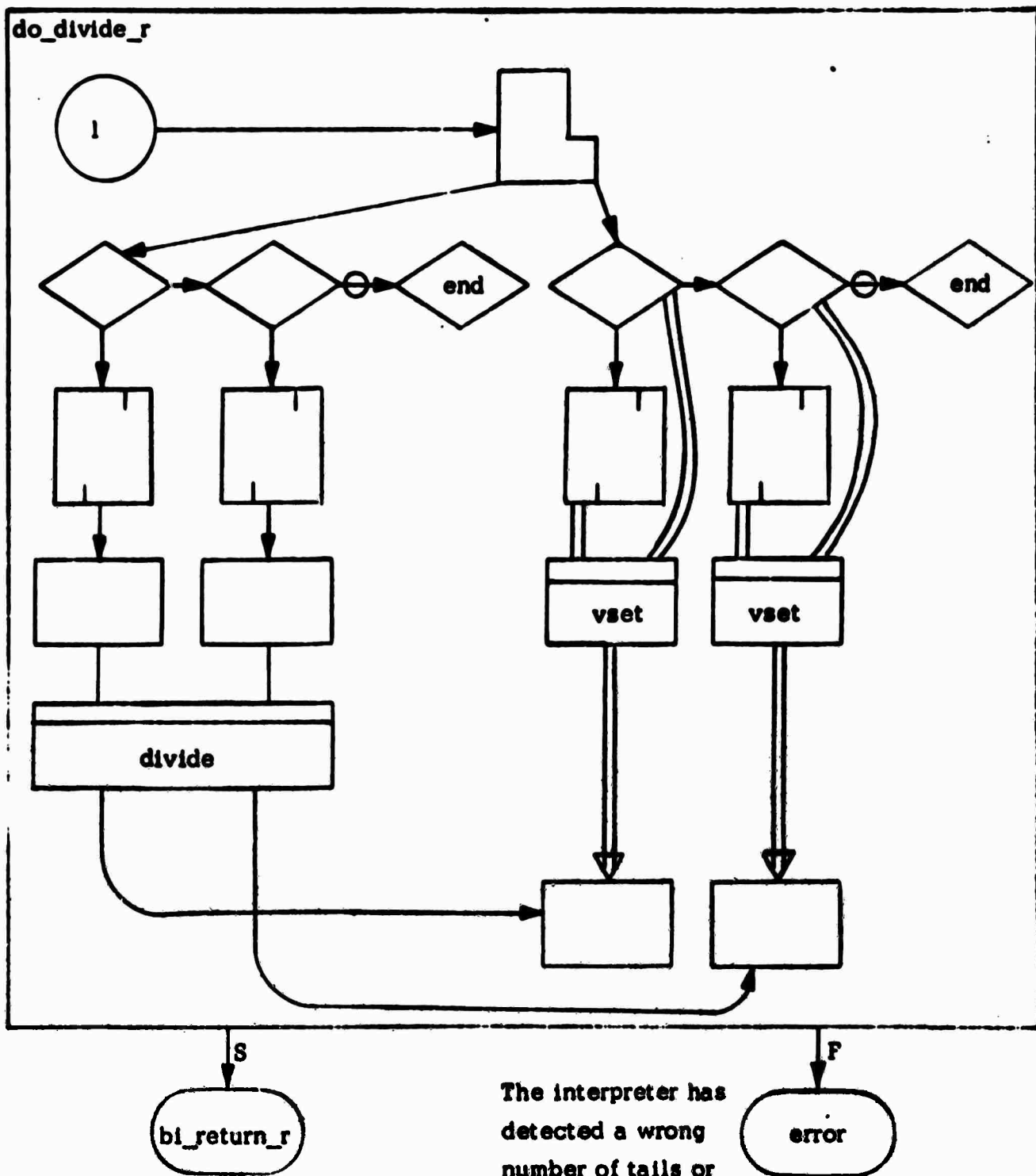
The interpreter has detected a wrong number of tails or heads on a read-call on the builtin "add".

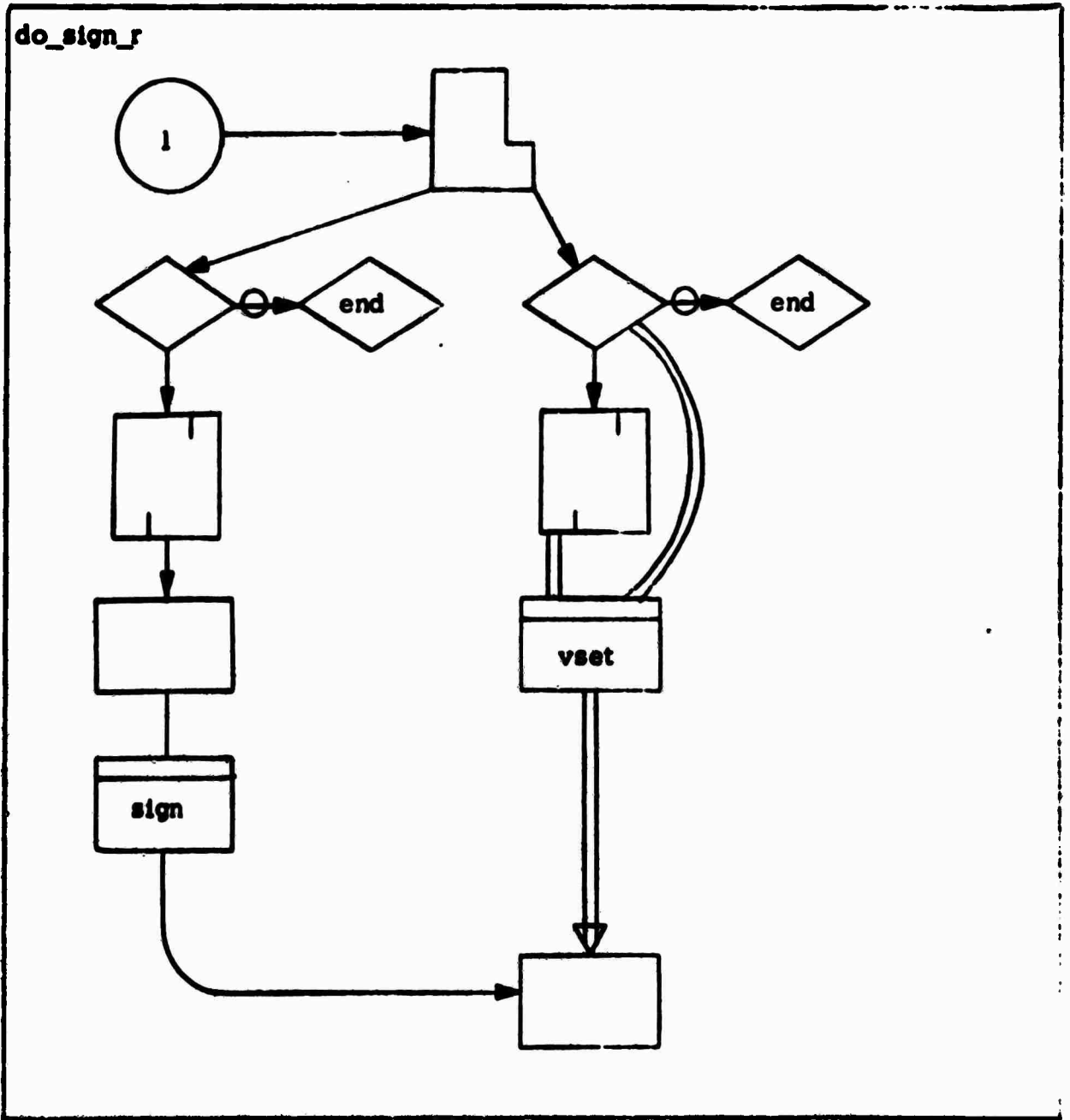
error



The interpreter has detected a wrong number of tails or heads on a read-call on the builtin "subtract".



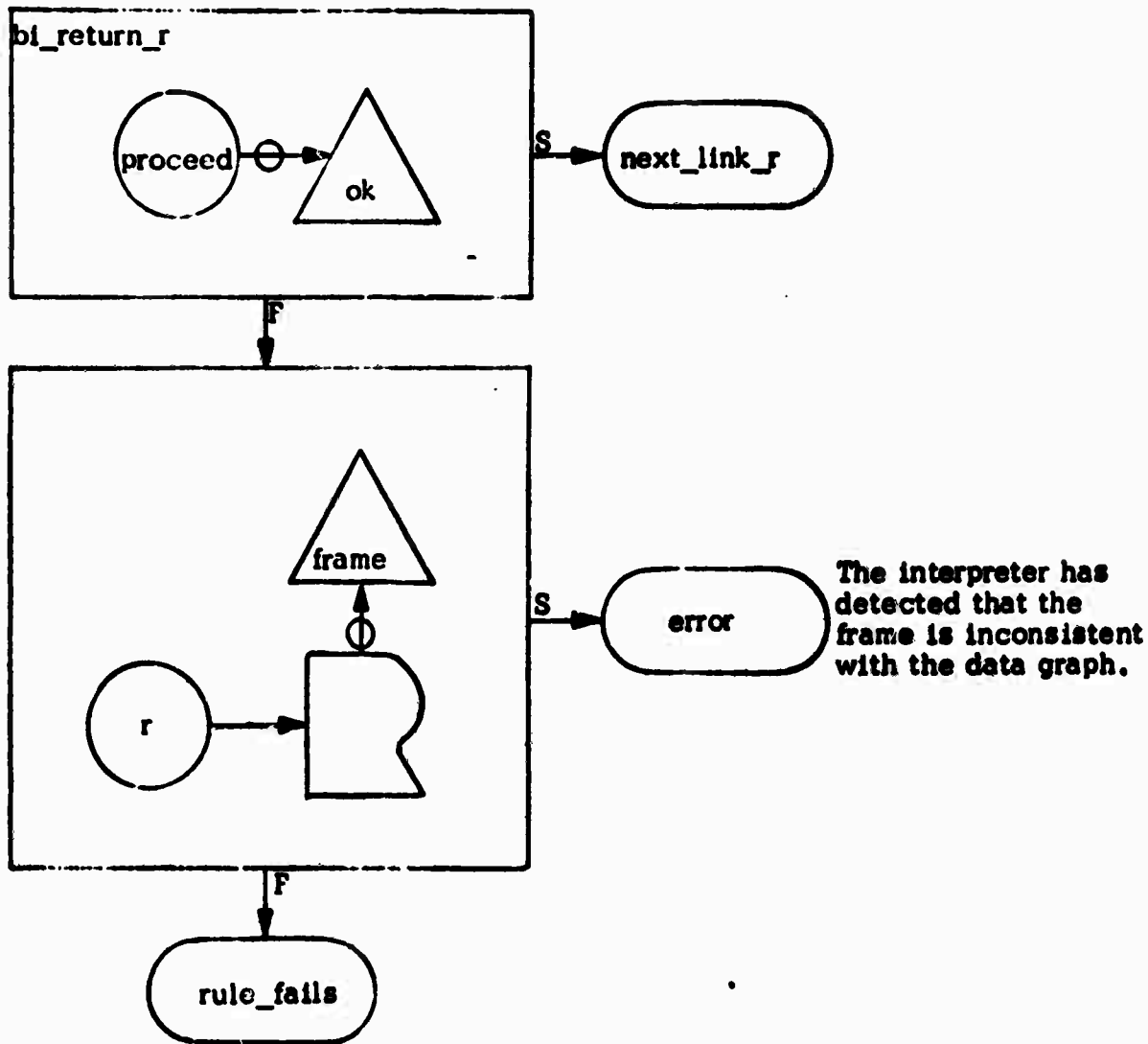


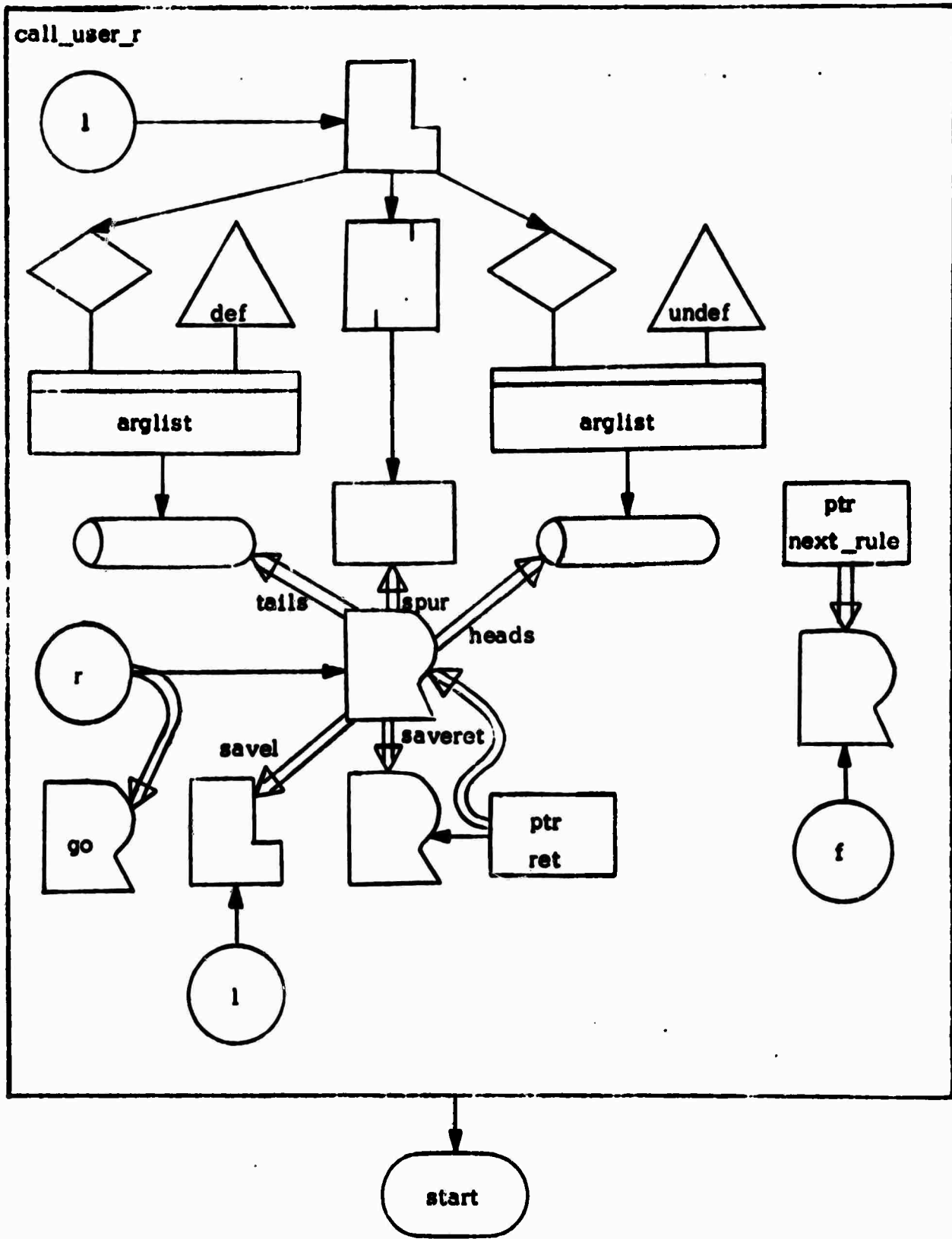


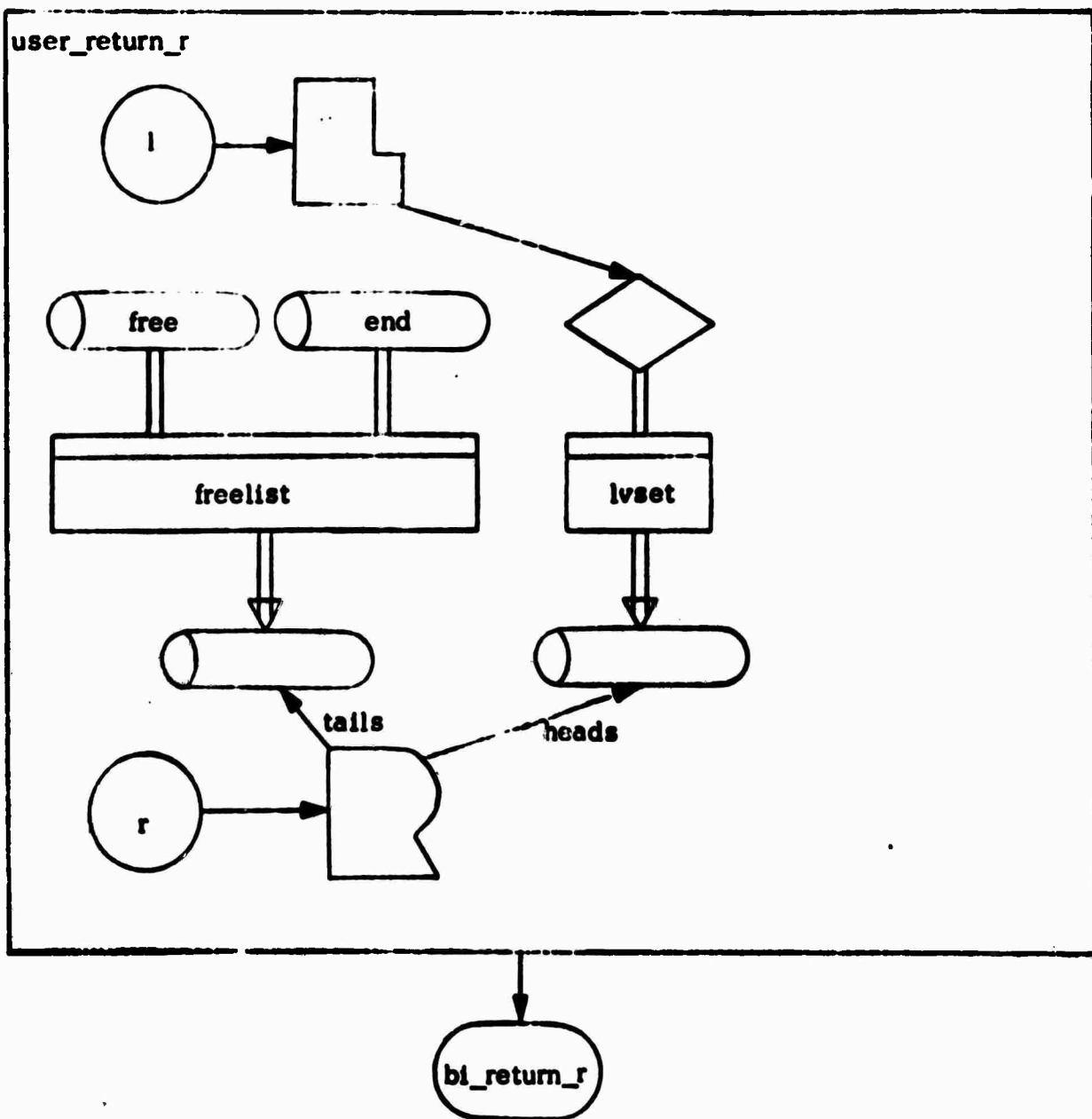
S  
bi\_return\_r

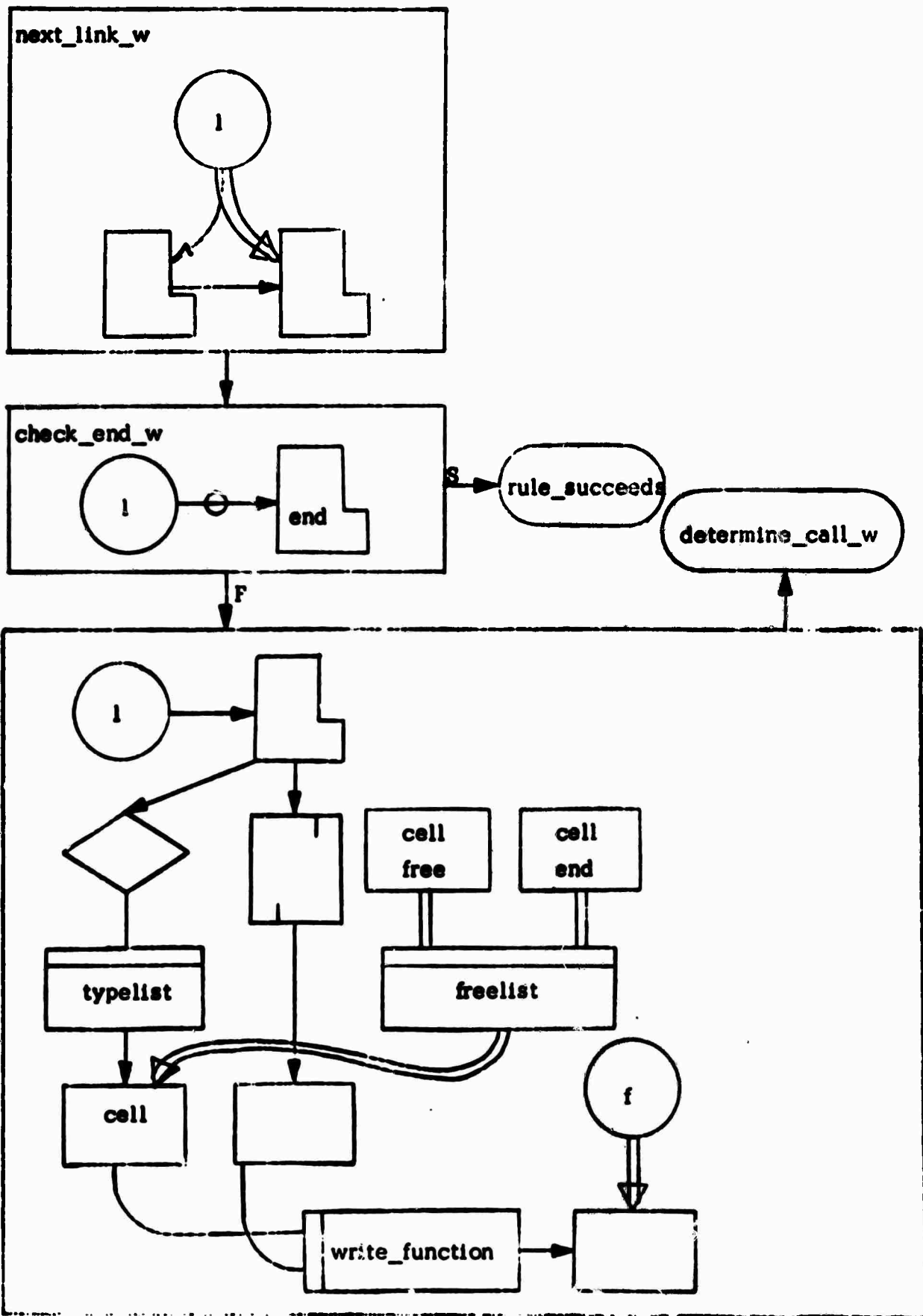
The interpreter has detected a wrong number of tails or heads on a read-call on the builtin "sign".

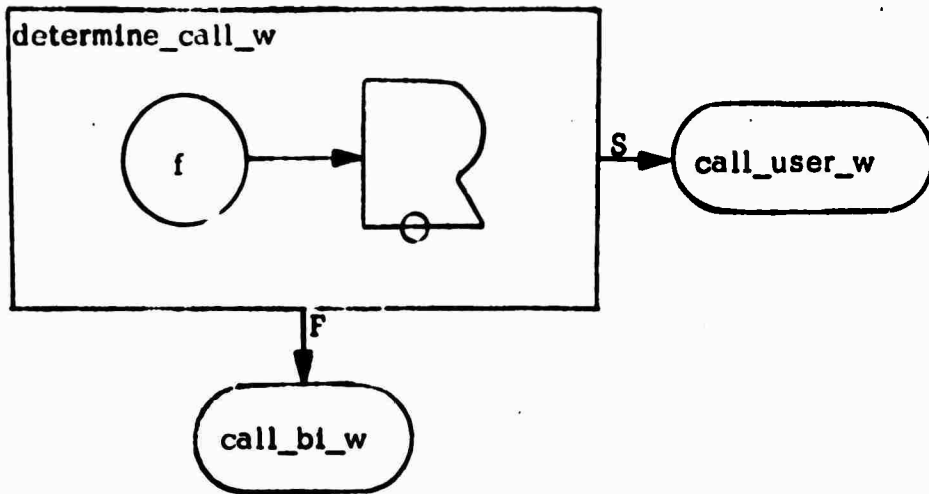
F  
error

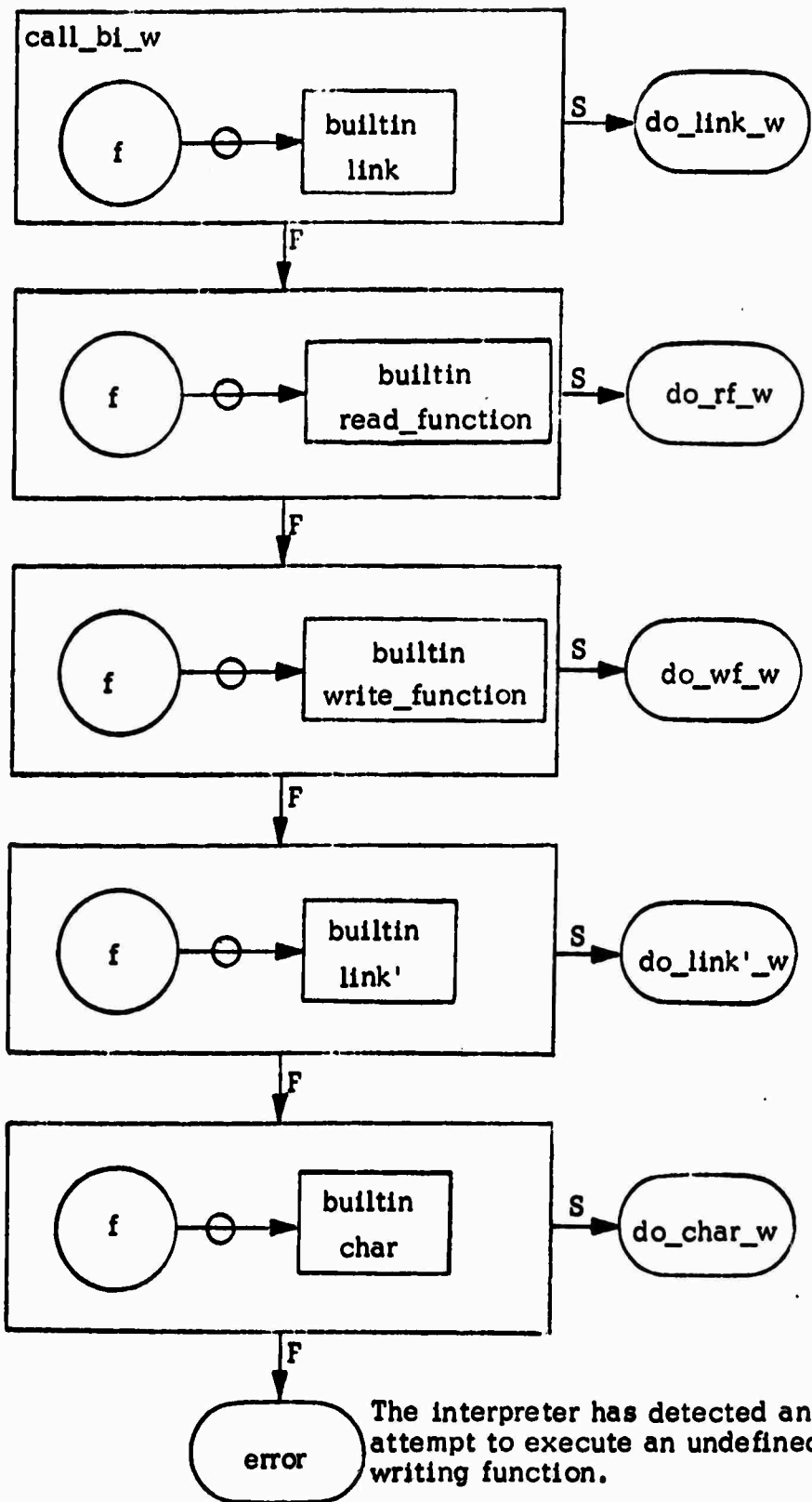


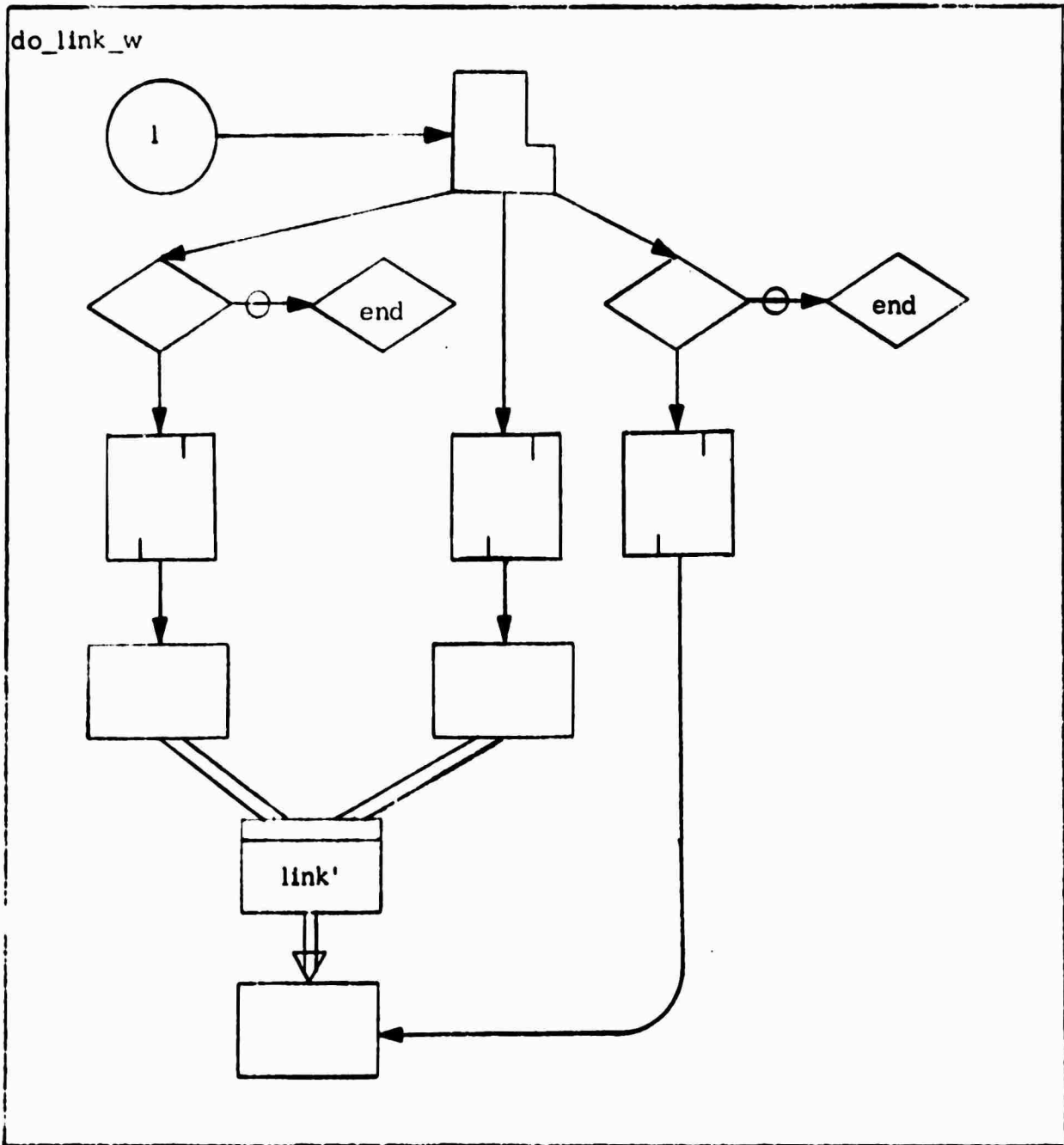






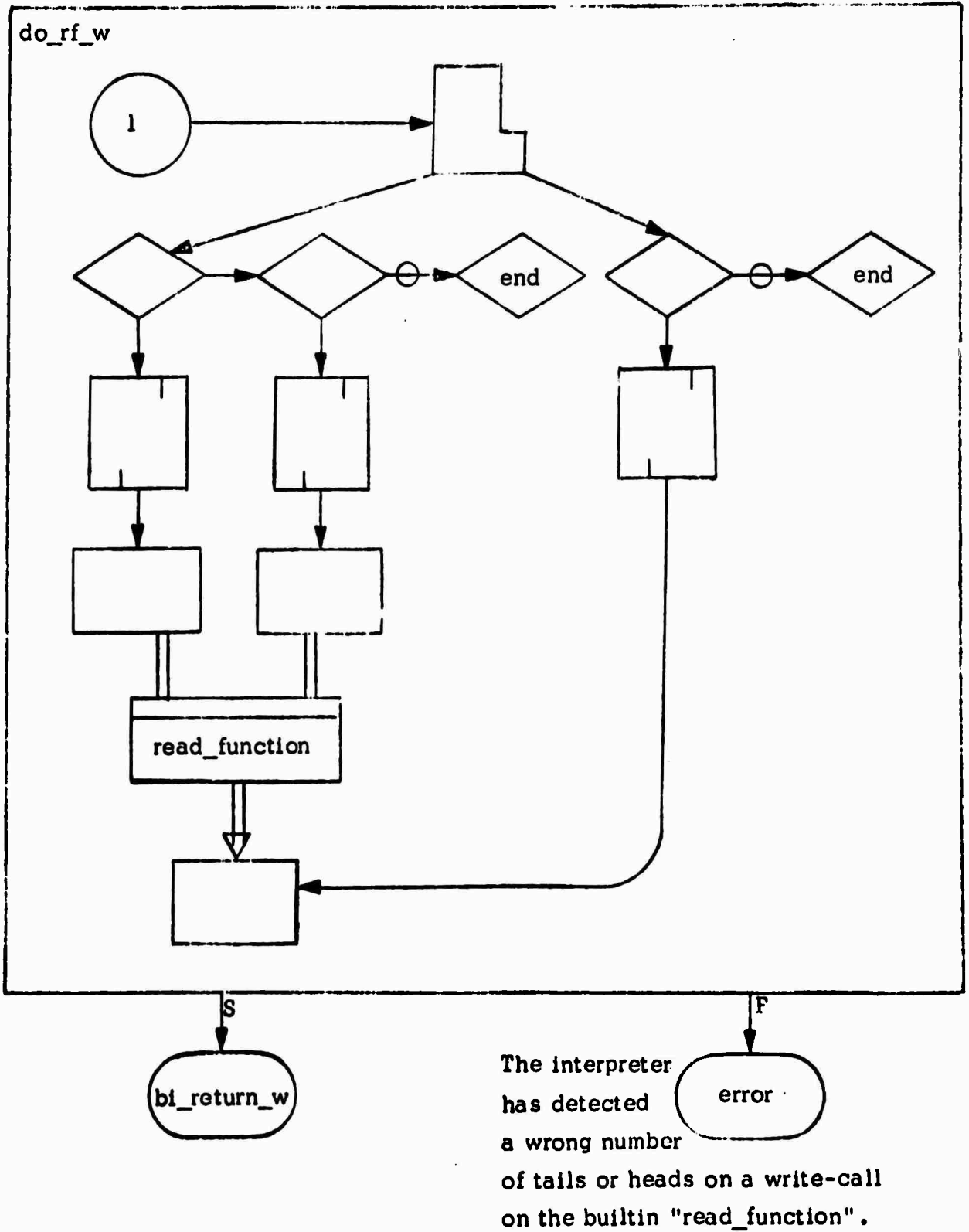


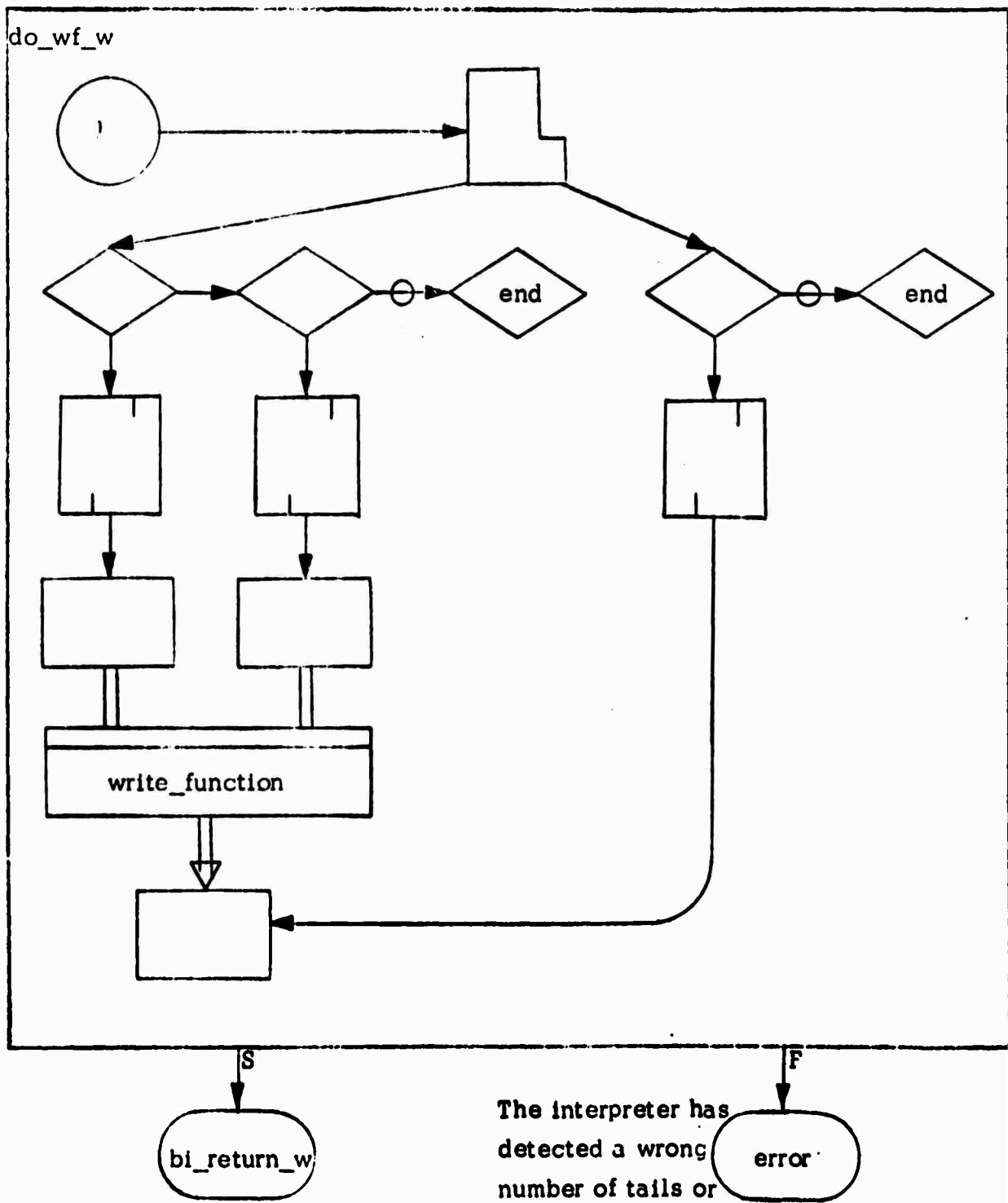




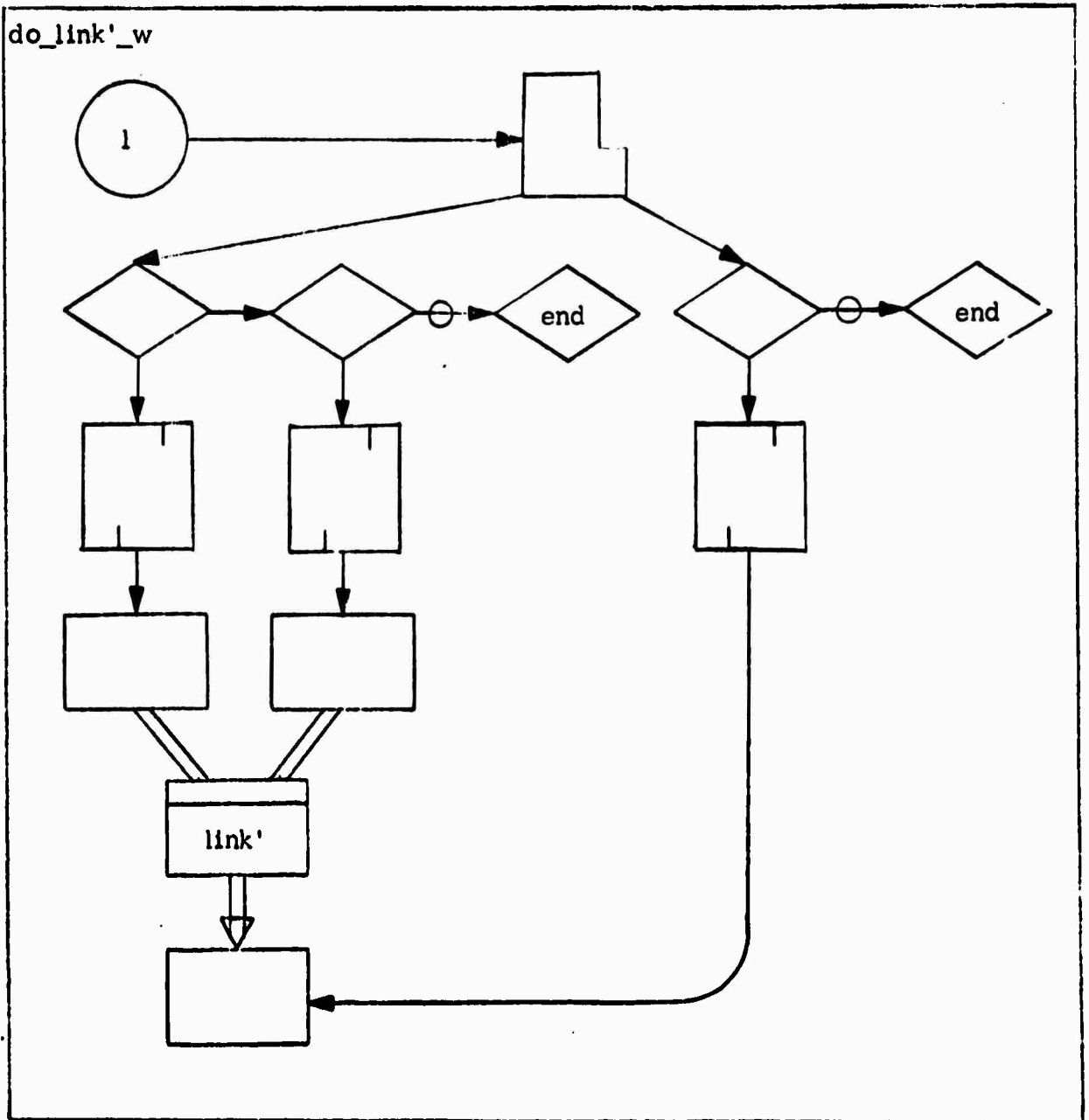
S  
 bi\_return\_w

F  
 The interpreter has detected a wrong number of tails or heads on a write-call on the builtin "link".  
 error





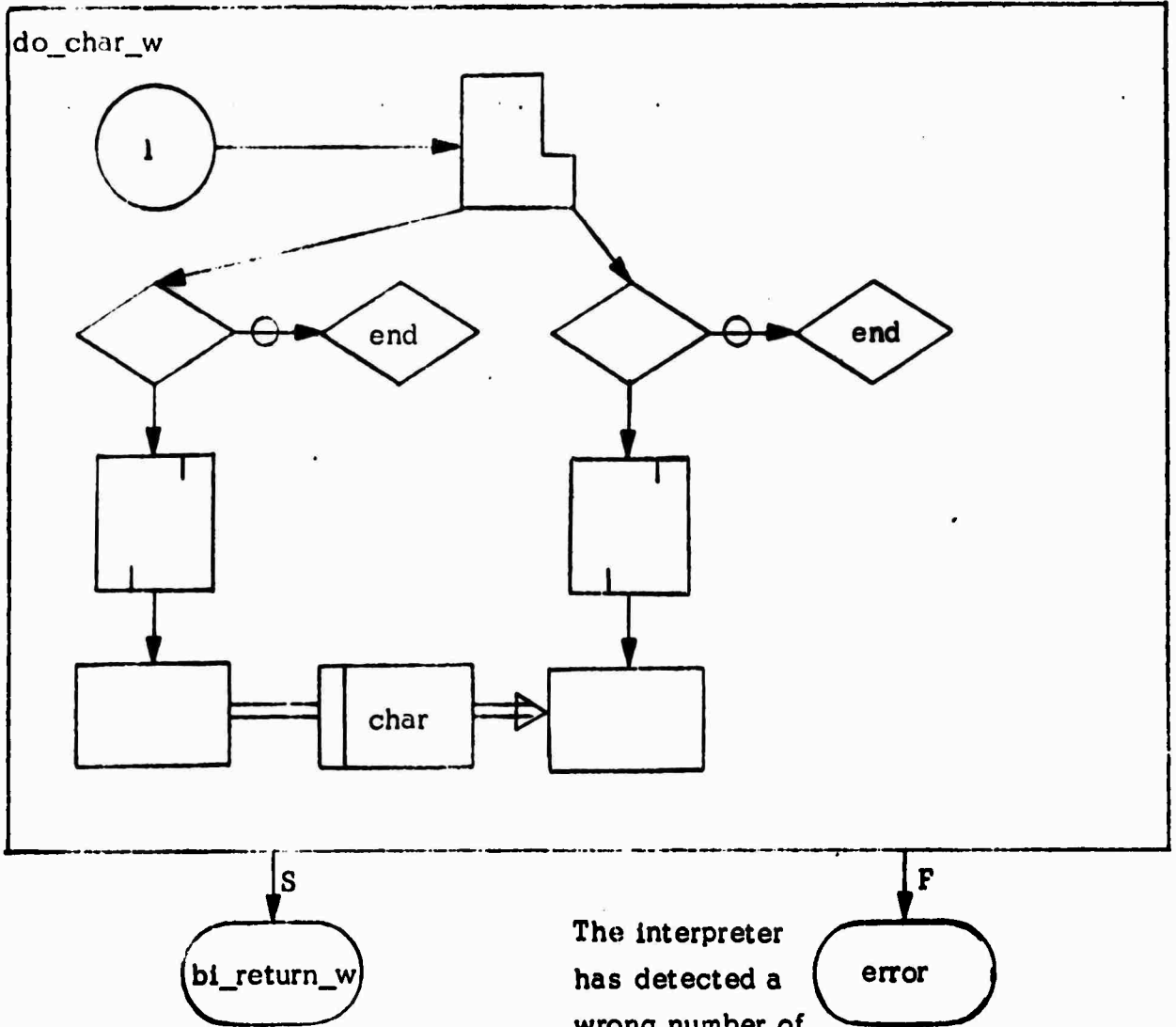
The interpreter has detected a wrong number of tails or heads on a write-call on the builtin "write\_function".



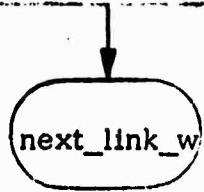
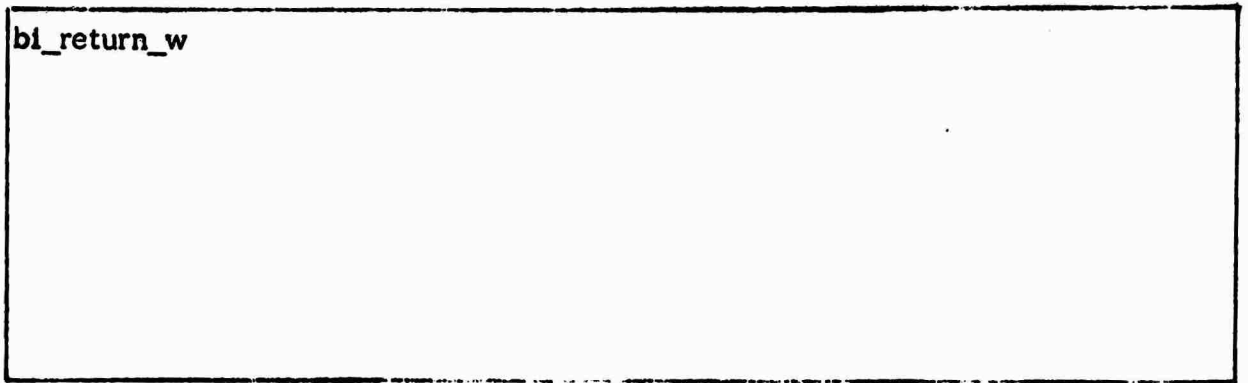
S  
bi\_return\_w

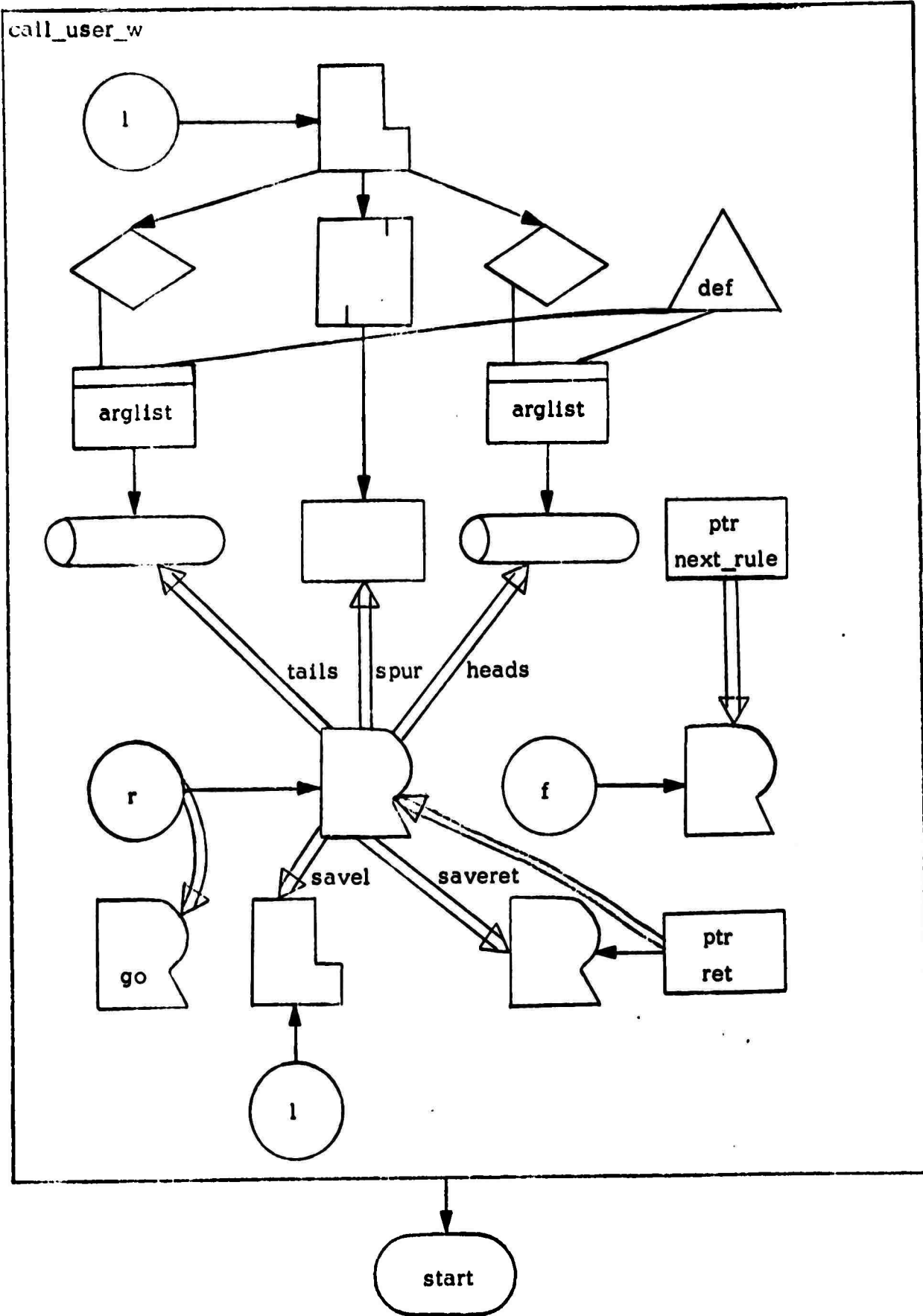
F  
error.

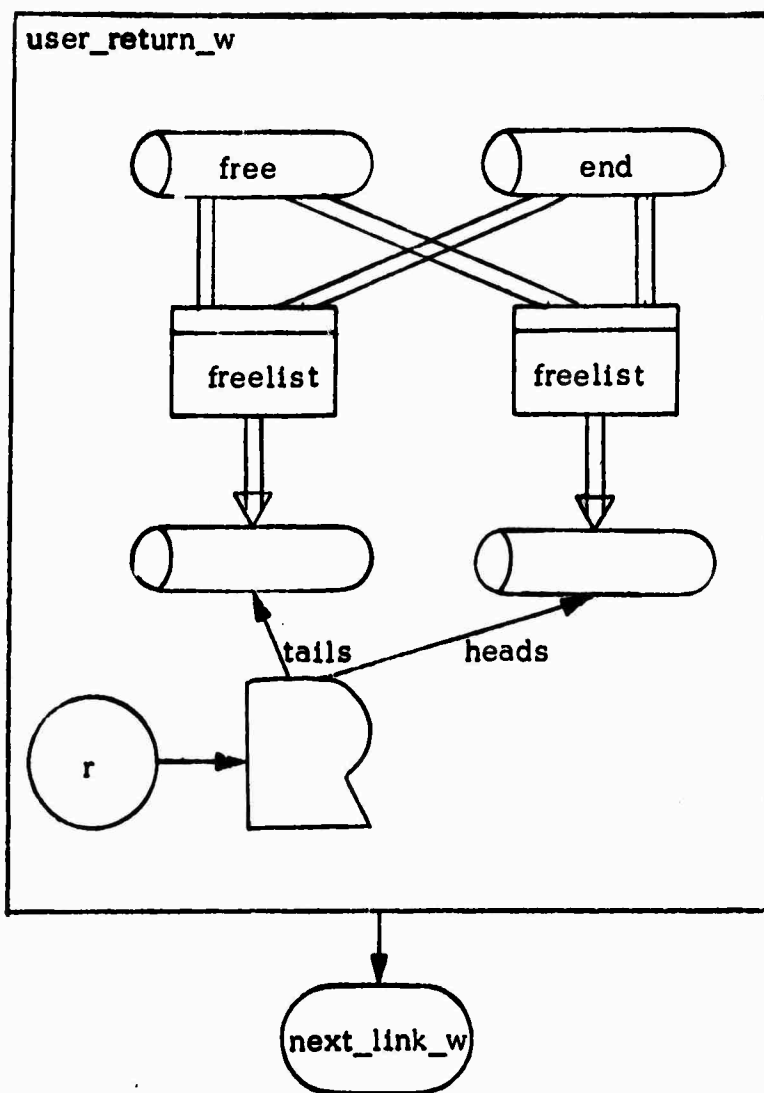
The interpreter has detected a wrong number of tails or heads on a write-call on the builtin "link'".

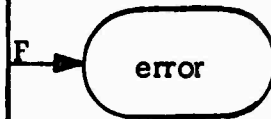
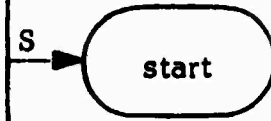
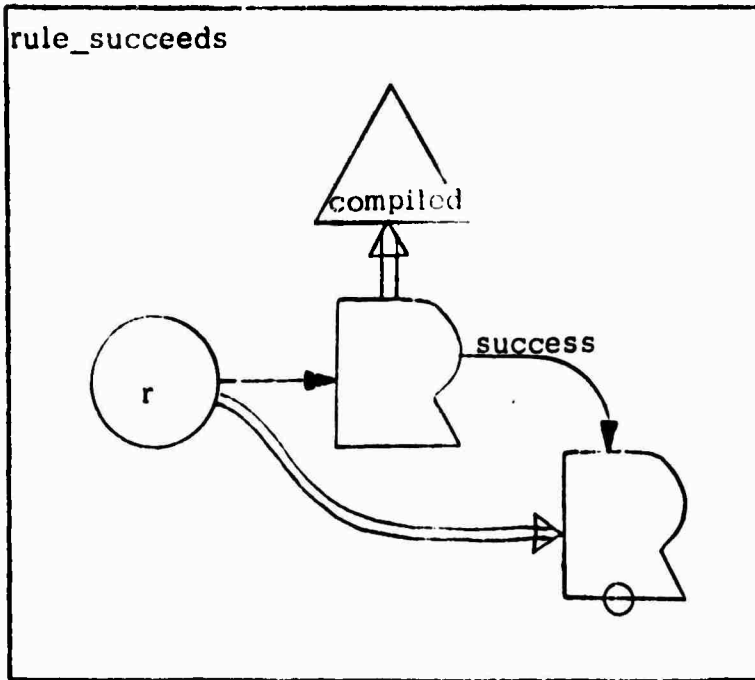


The interpreter has detected a wrong number of tails or heads on a write-call on the builtin "char".

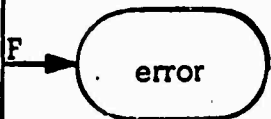
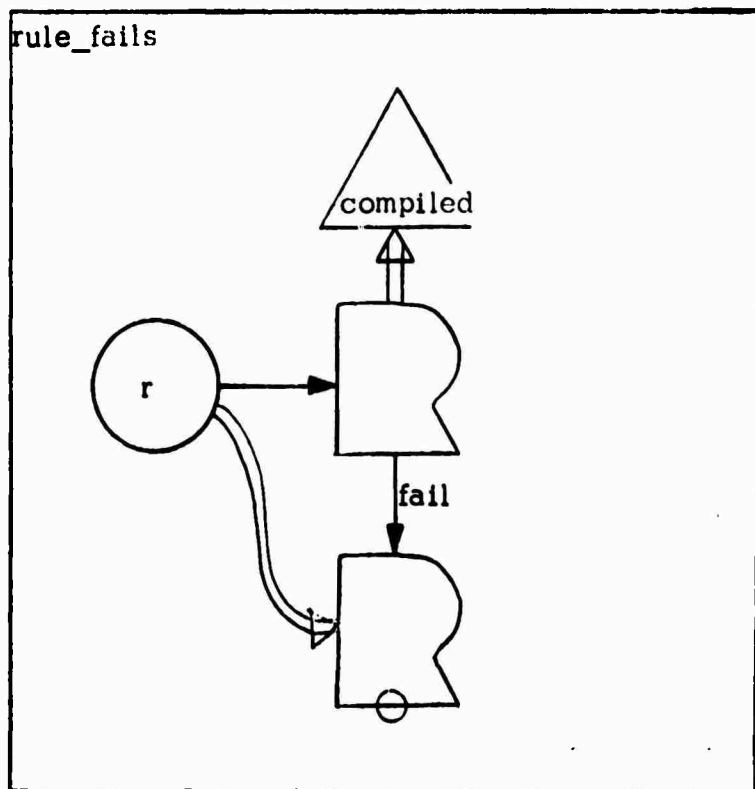




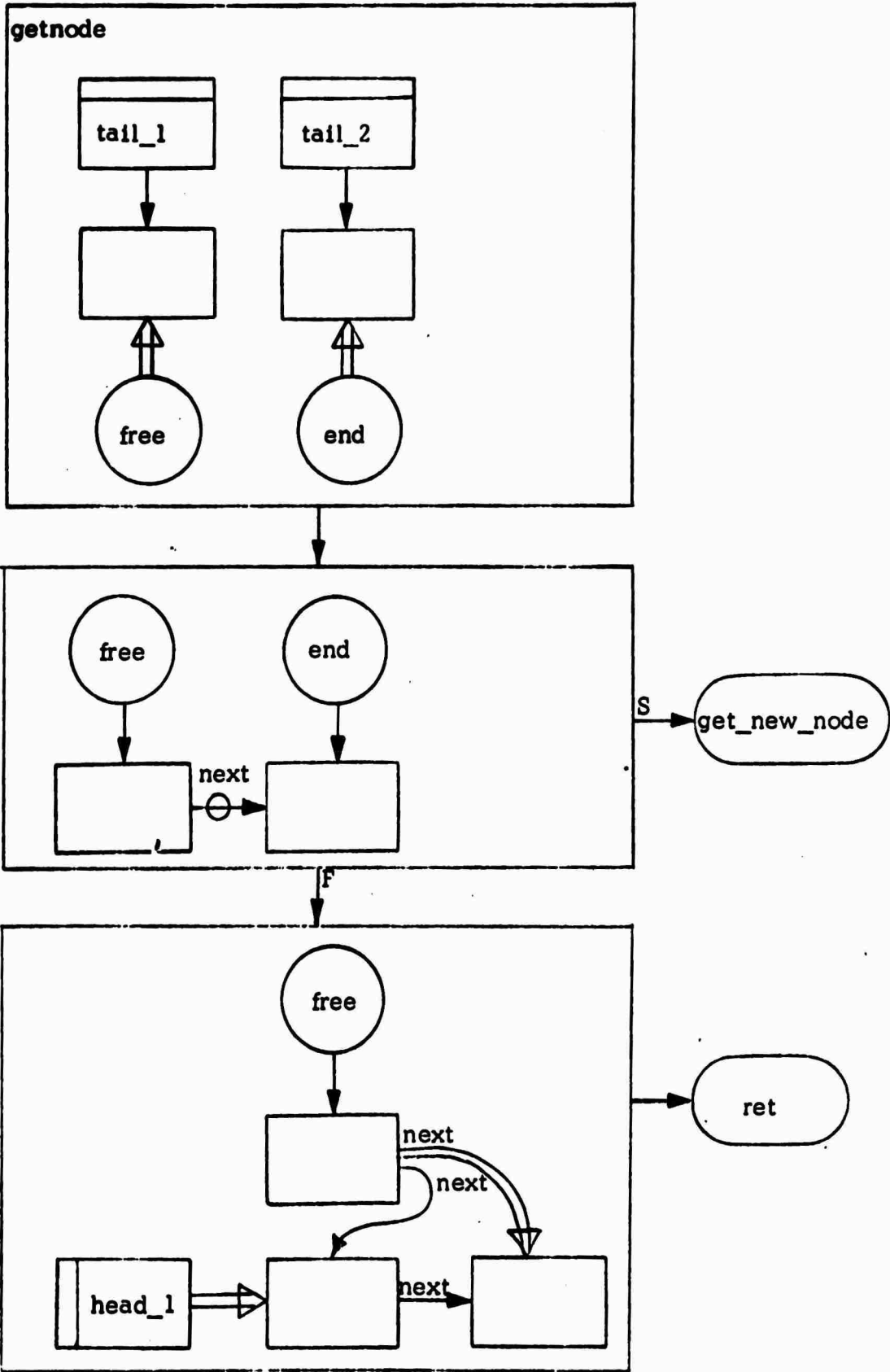


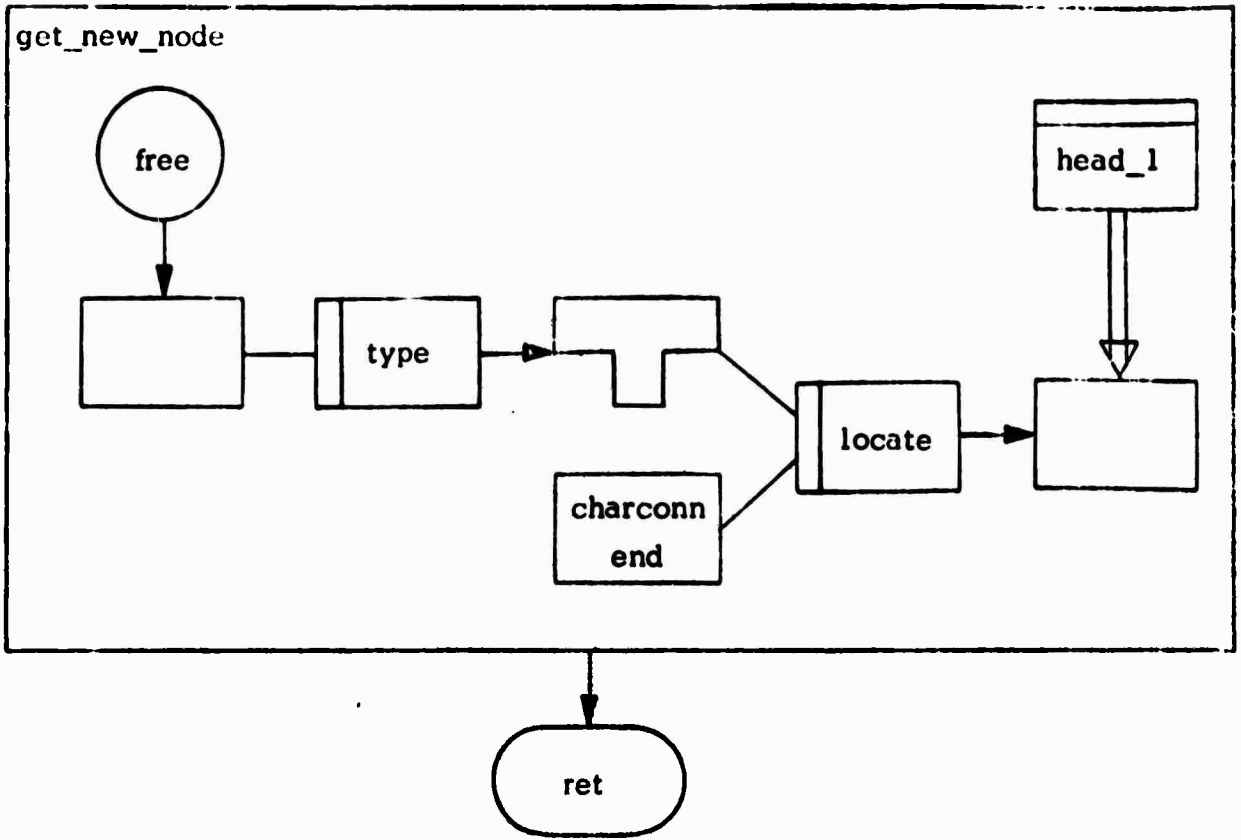


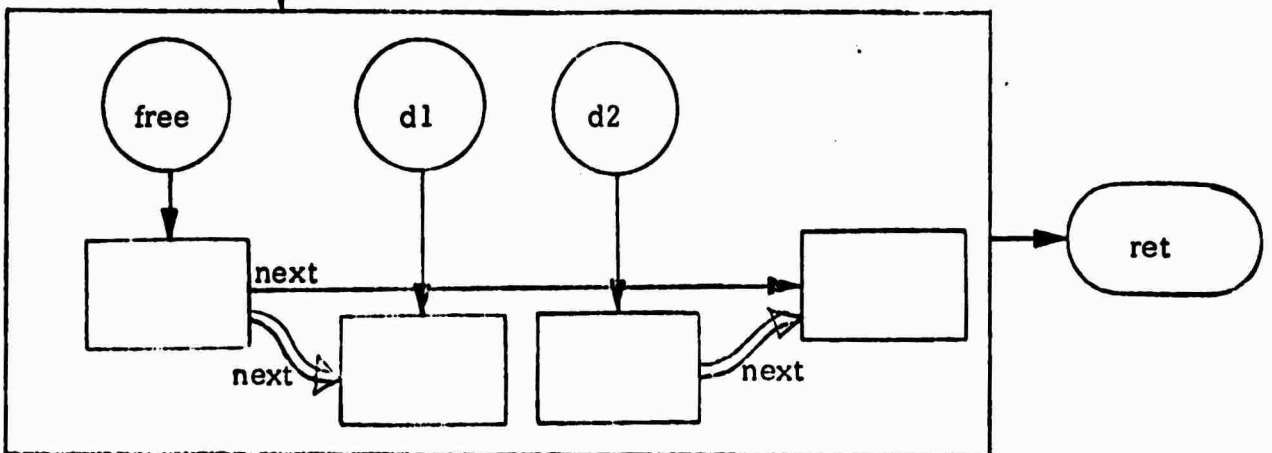
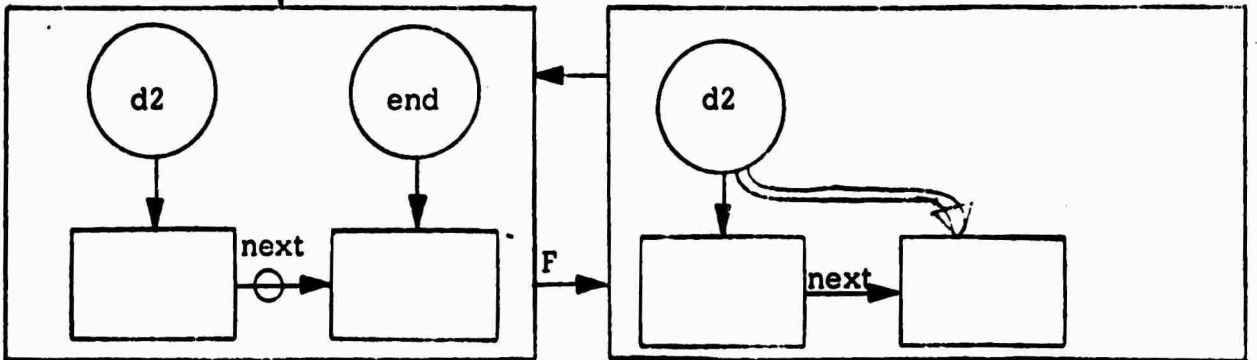
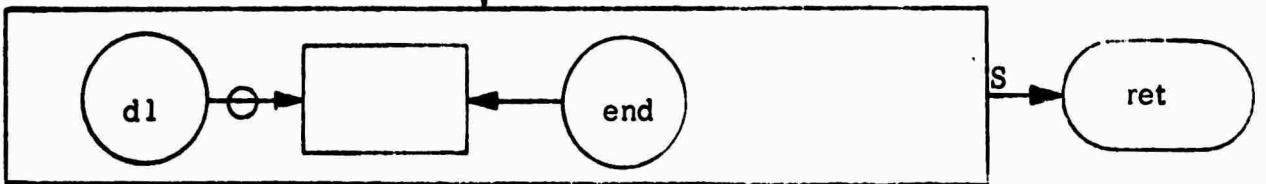
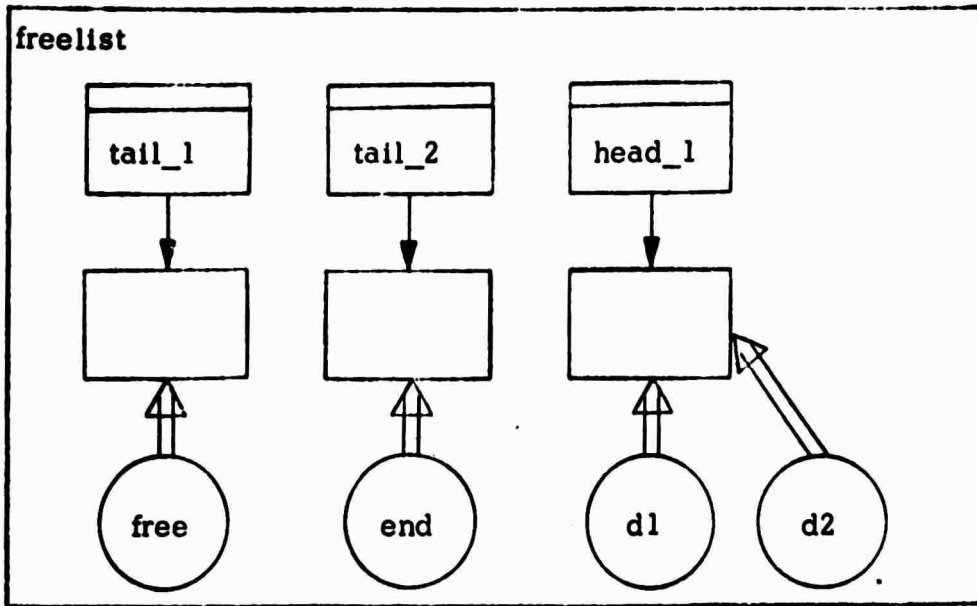
The interpreter has detected the "success" exit from a rule leads to "...", which is not a rule.

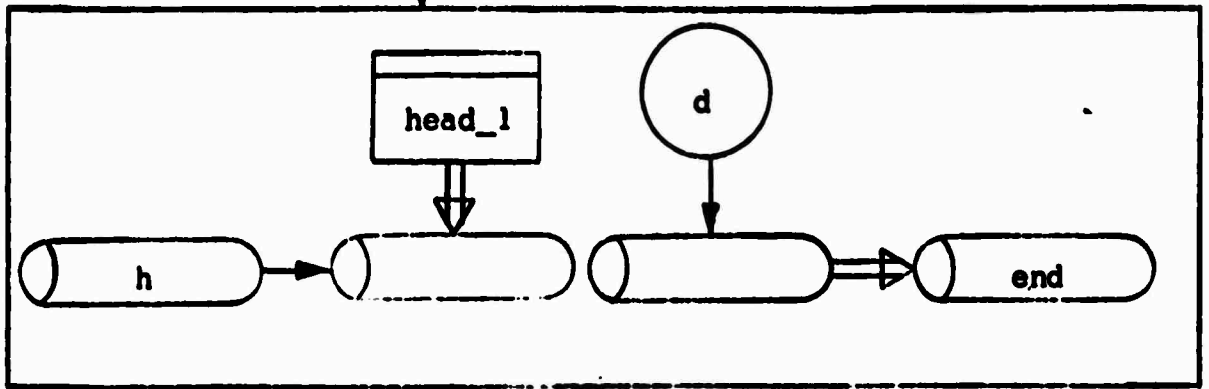
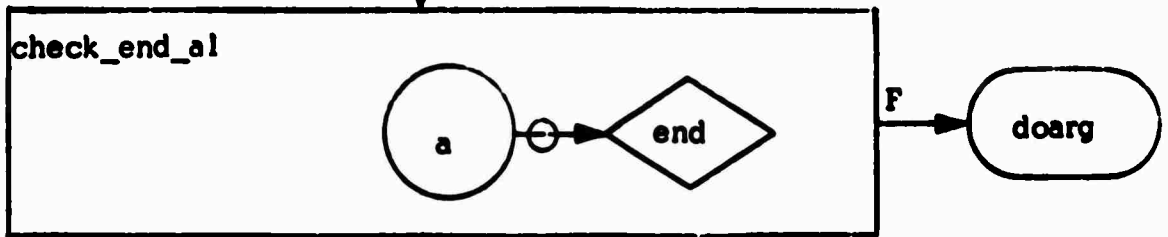
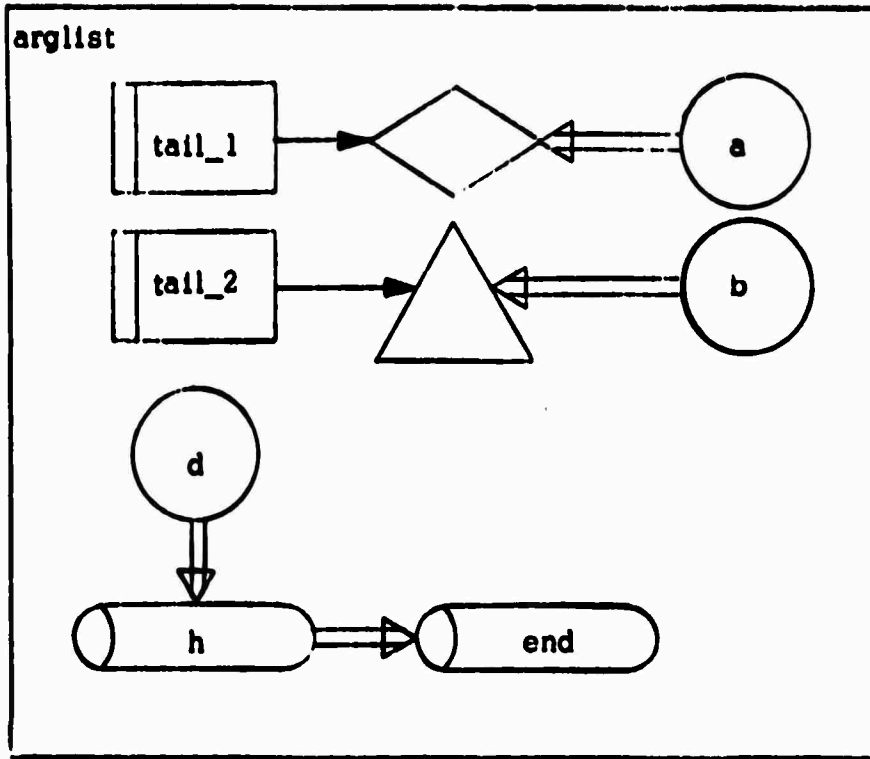


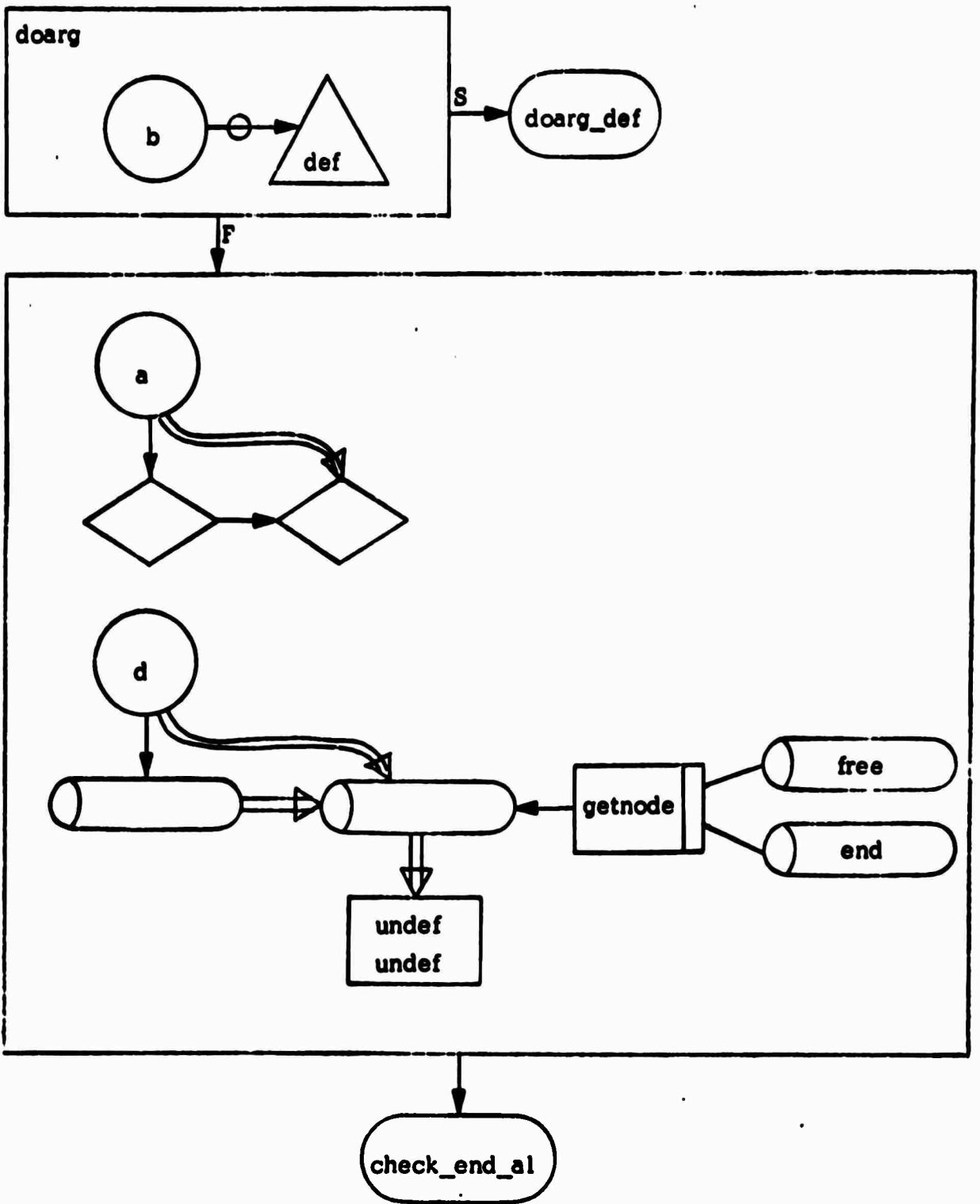
The interpreter has detected the "fail" exit from a rule leads to "...", which is not a rule.



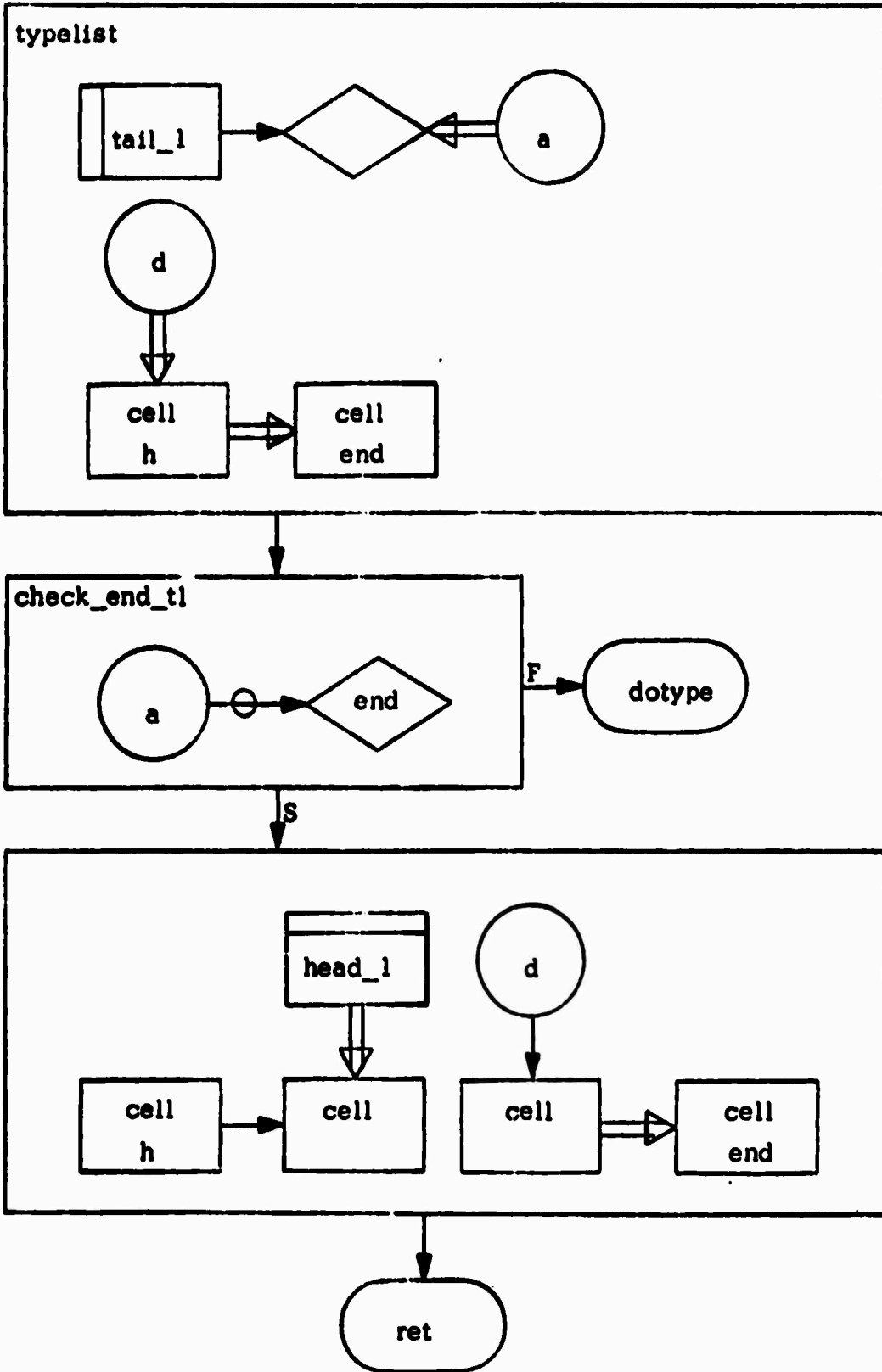




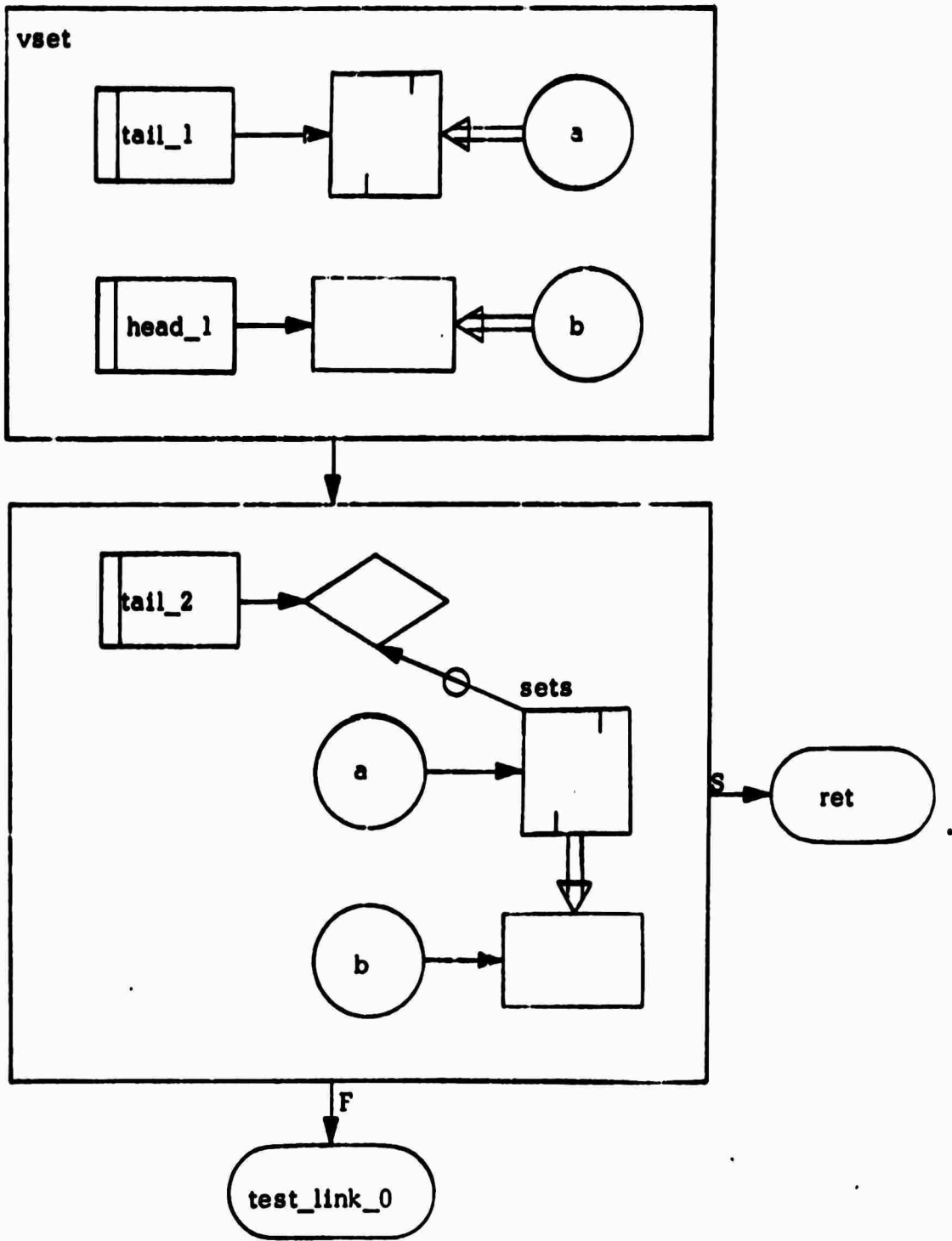


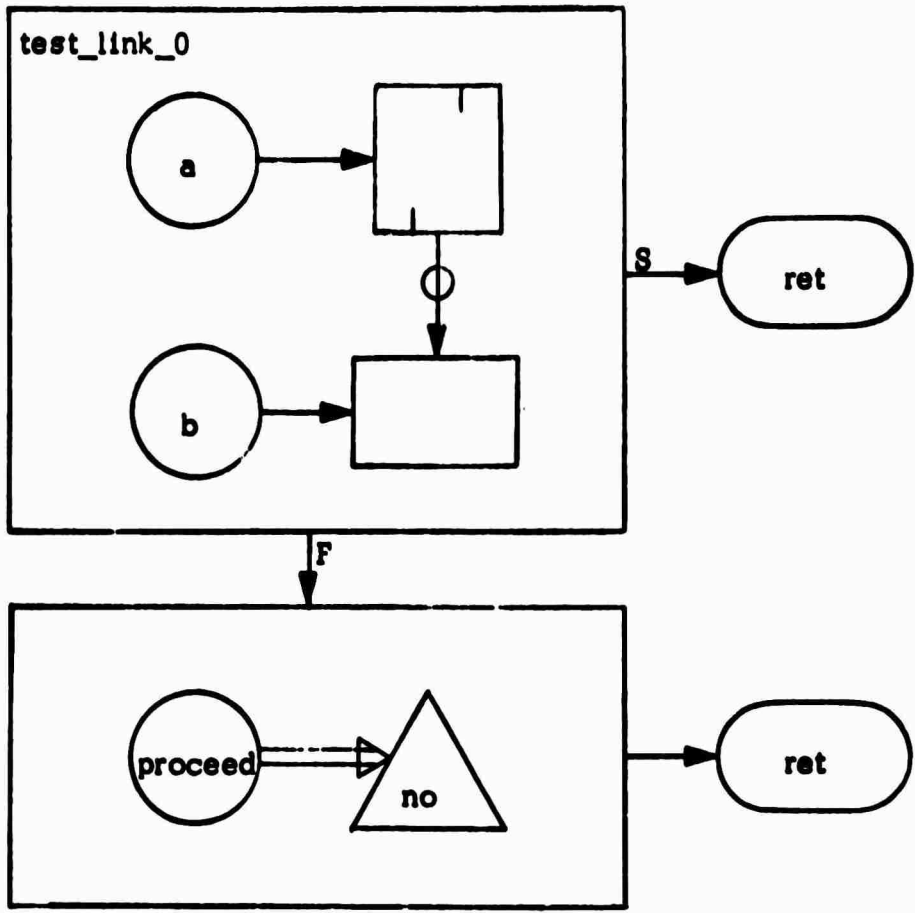


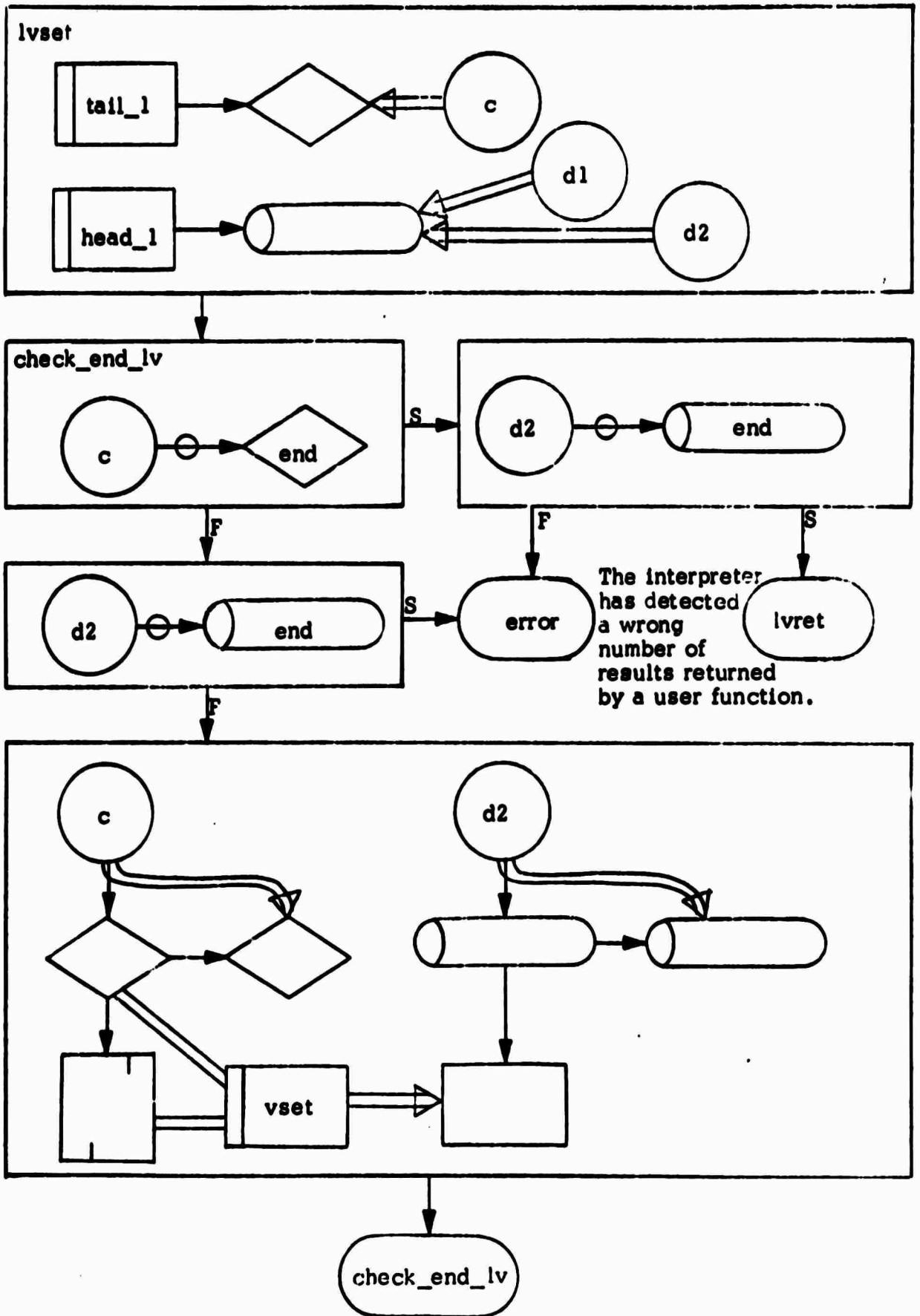


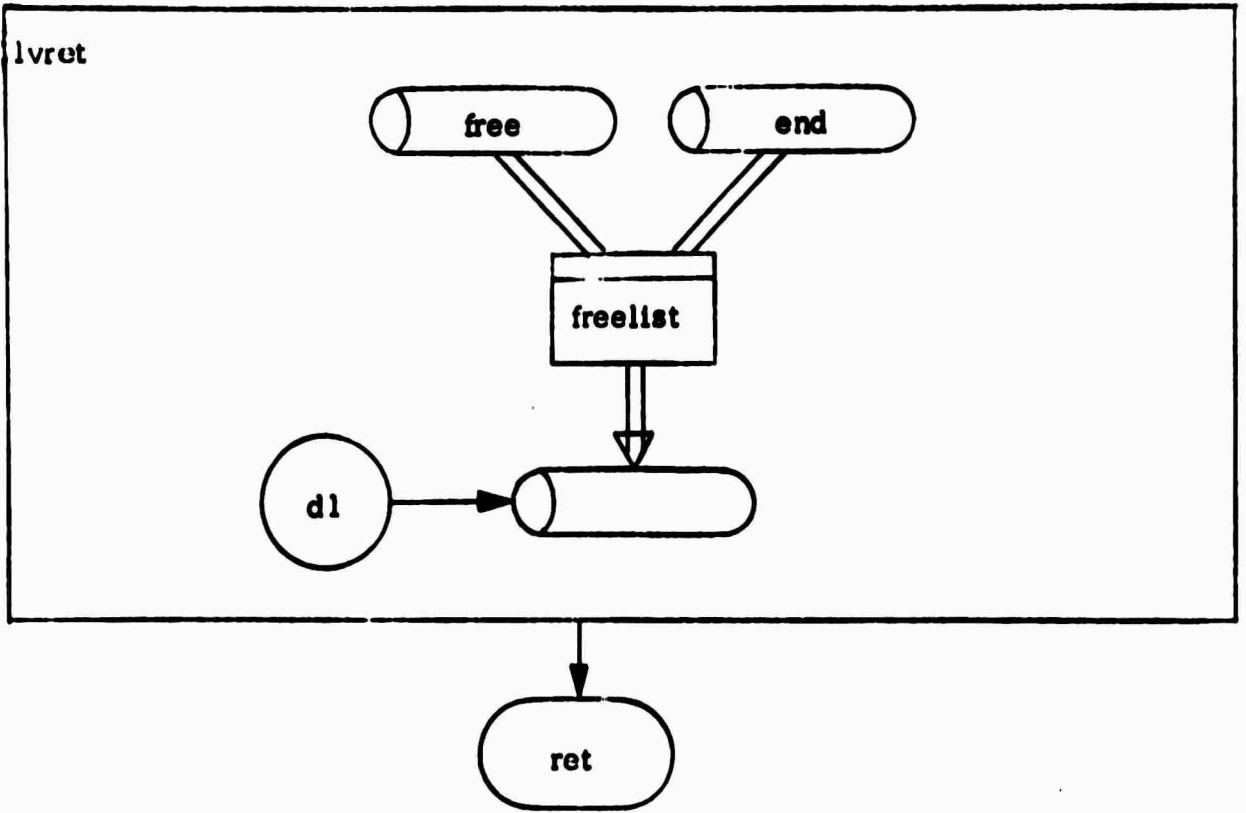


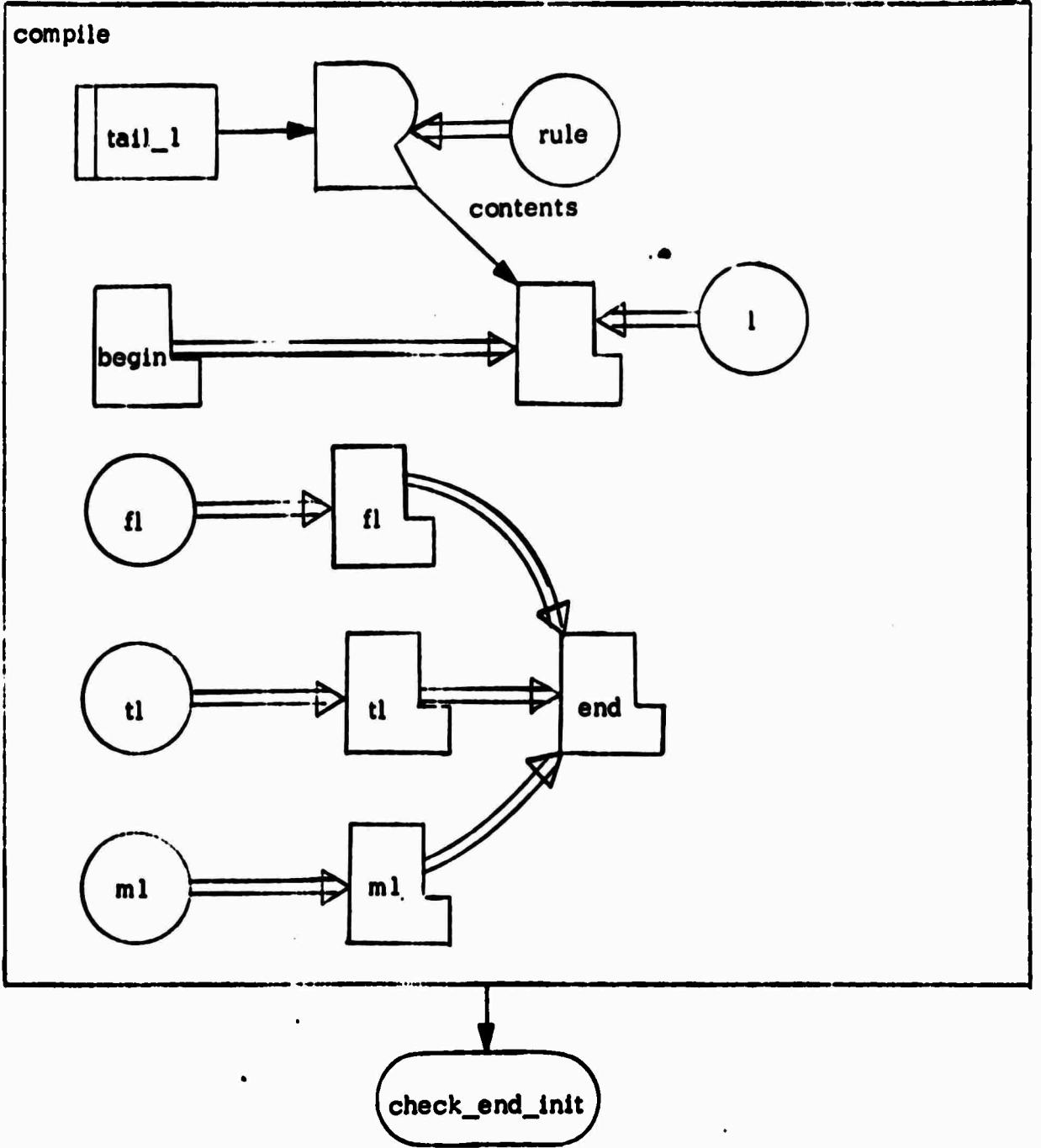


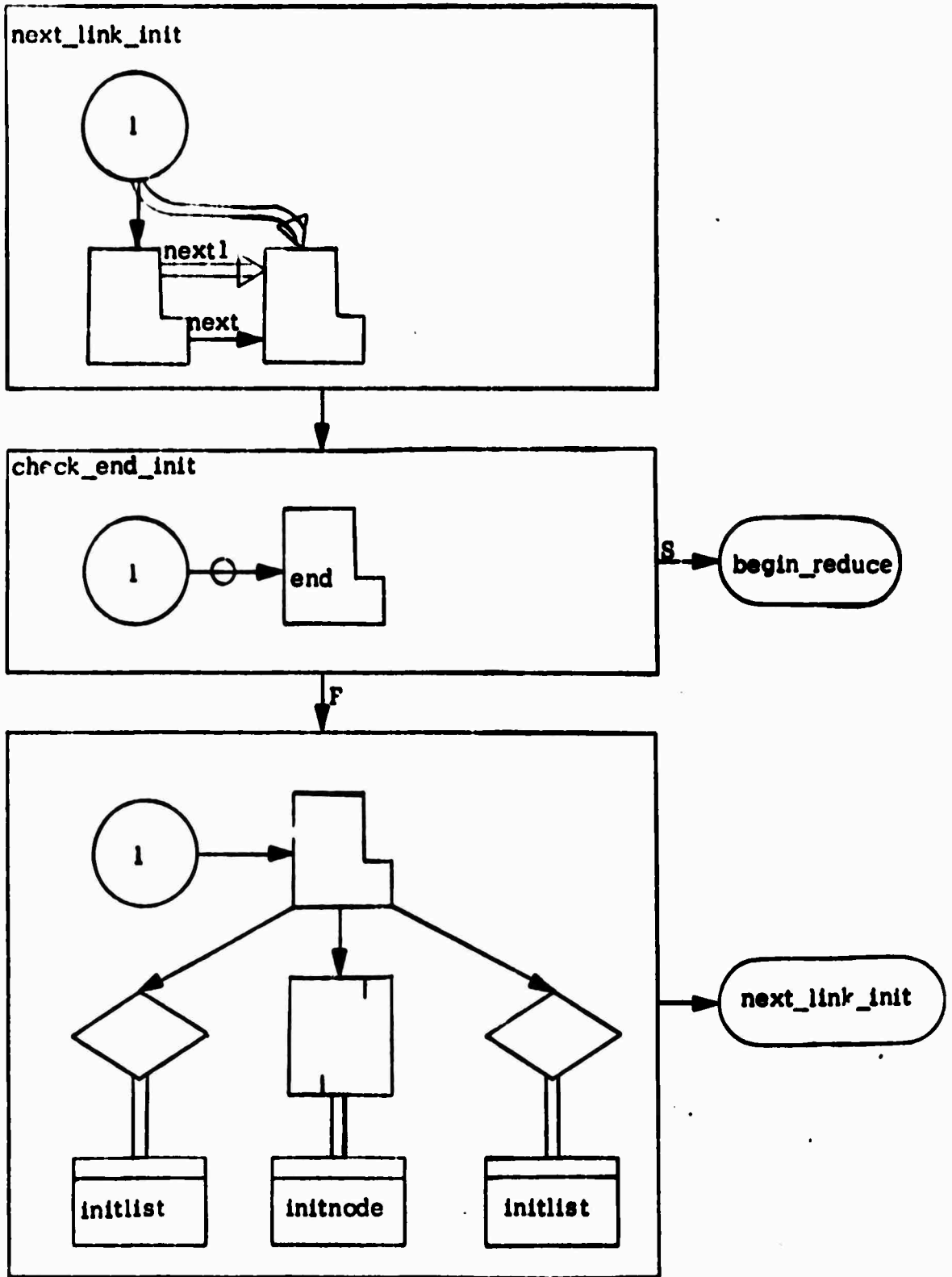


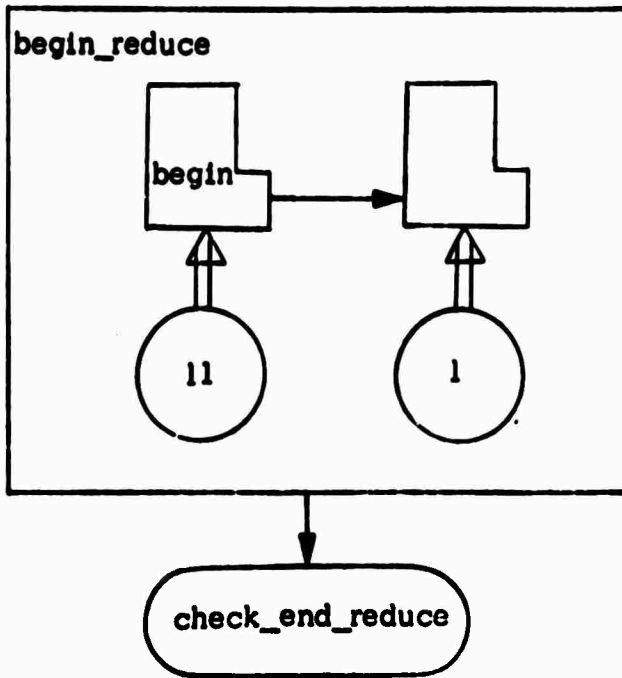


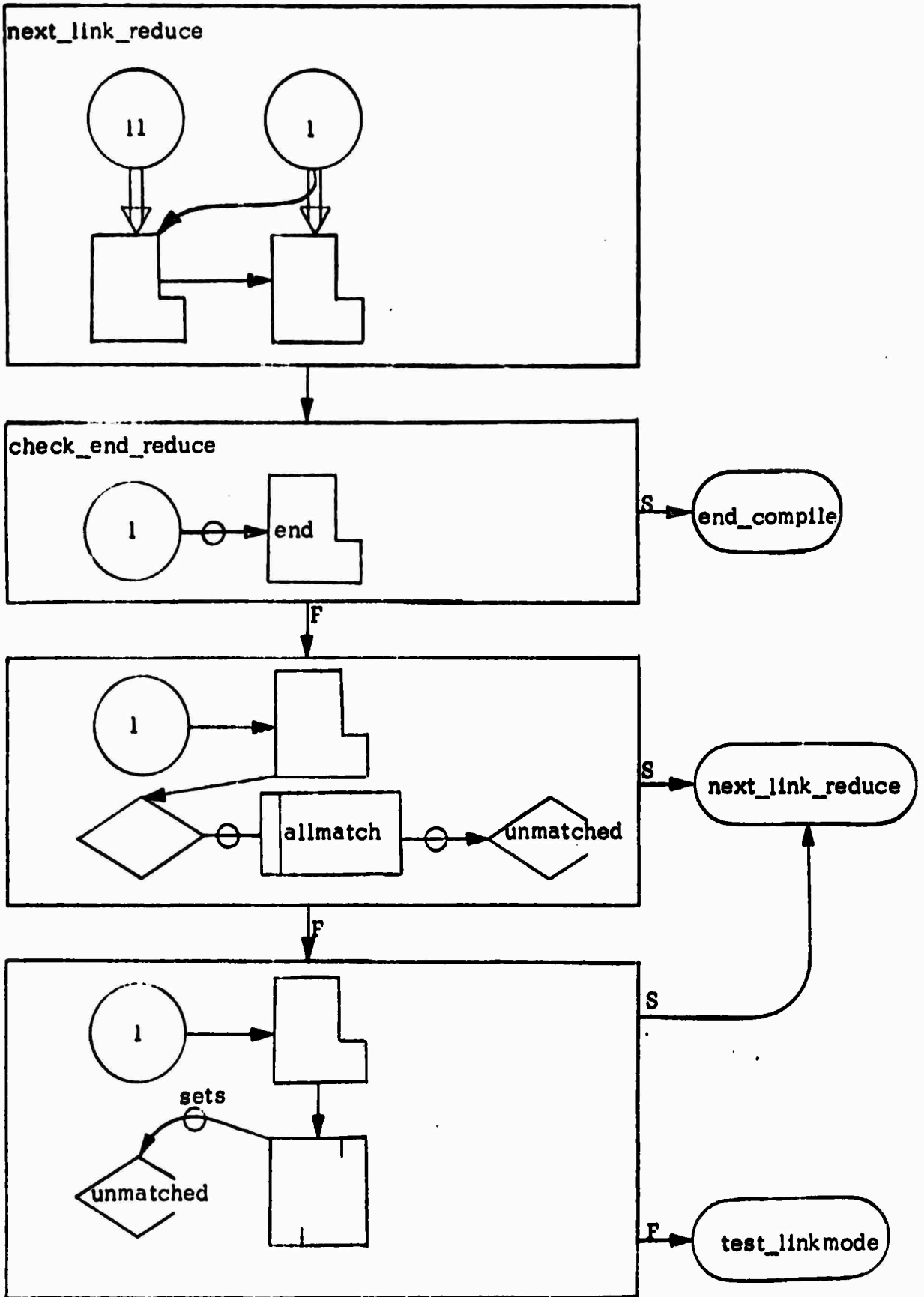


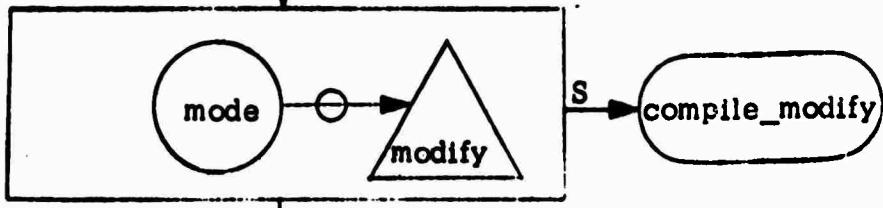
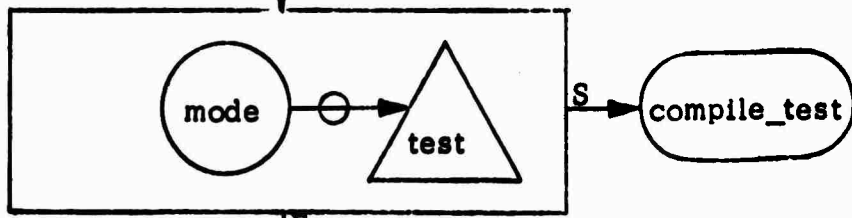
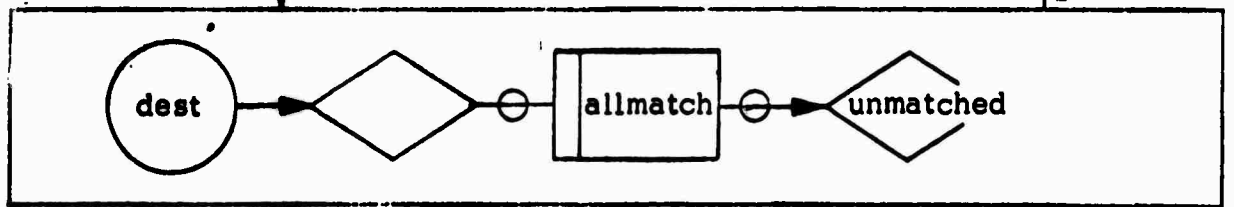
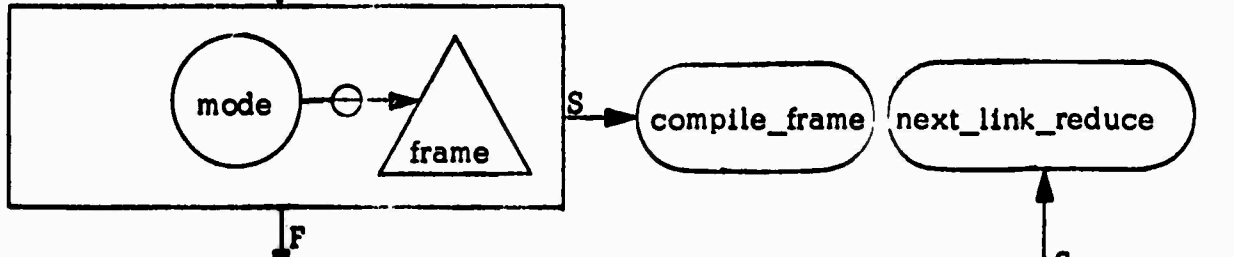
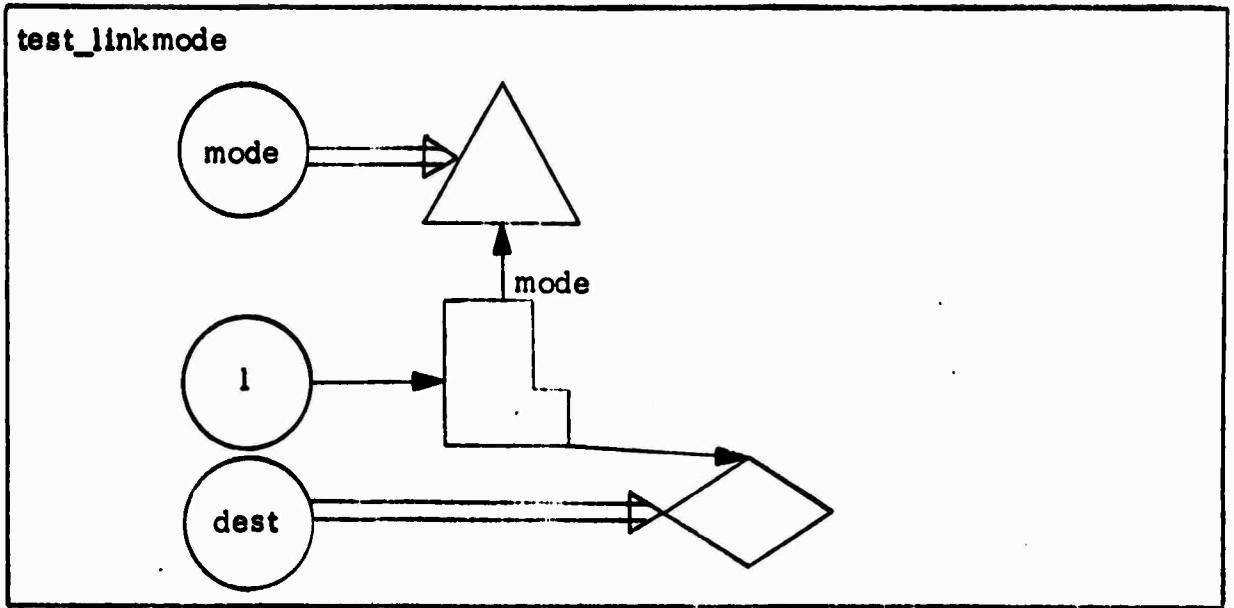




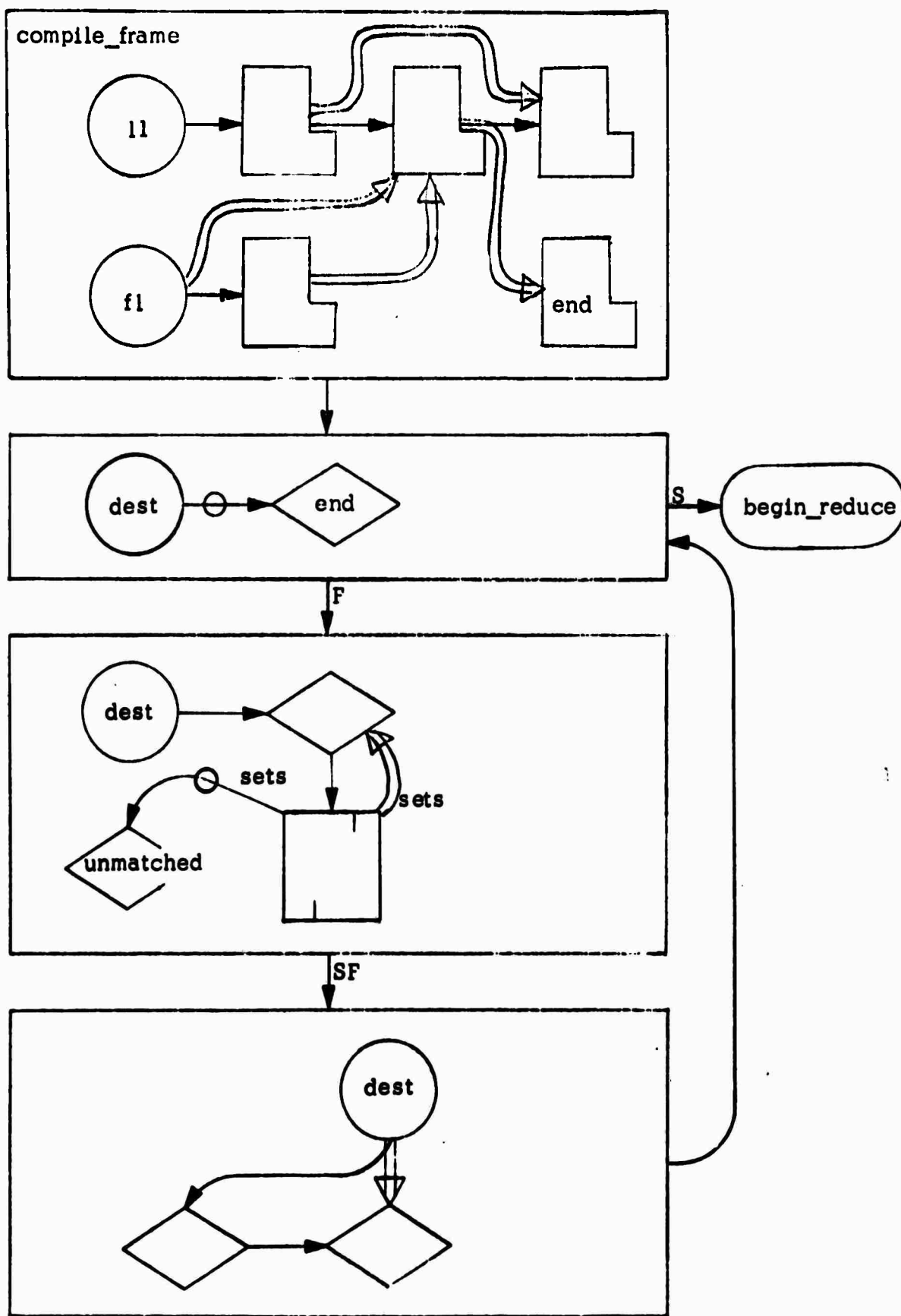


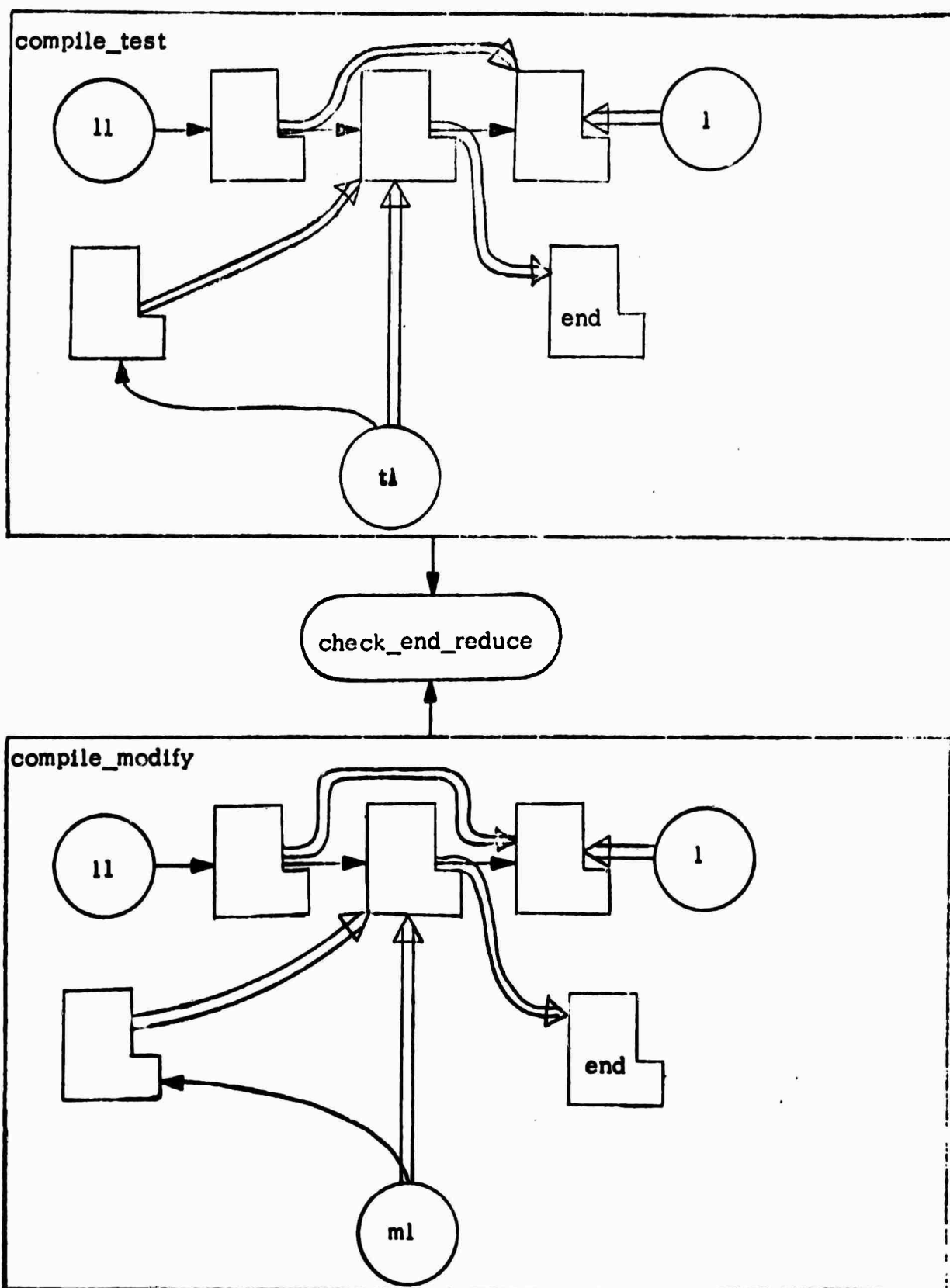


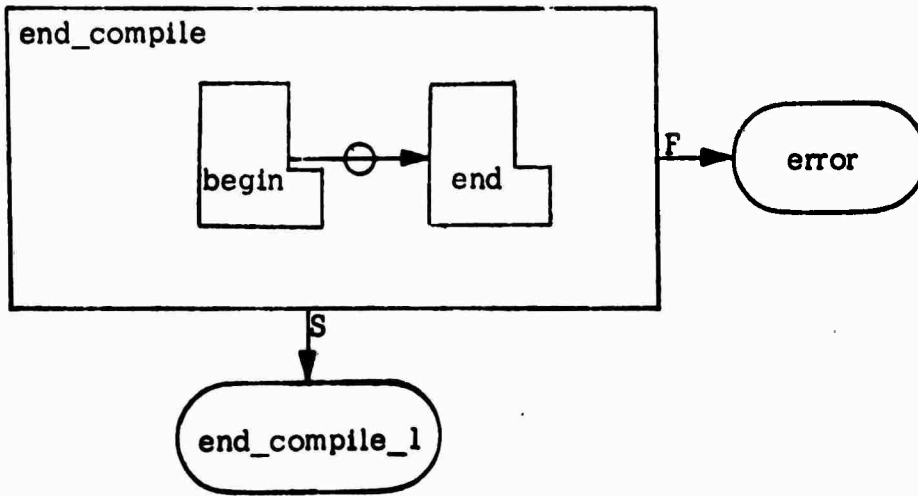




**error** The compilation phase of the interpreter has detected the mode of a link is "...", which is neither "flag frame", "flag test", nor "flag modify".







The compilation phase of the interpreter has detected that a rule contains an unreachable node.

