To: Distribution

From: C. D. Erickson, S. G. Bender

Date: August 2, 1978

Subject: Installation of Multics Documentation Macros

Many people have expressed an interest in using the documentation compose macros. Even though they are special case macros (i.e., following Honeywell and, more specifically, Multics documentation standards), they offer many features that are useful for a variety of documentation needs. For example, they automatically translate a heading into the following format based on what level (0, 1, 2, 3, or 4) the user specifies:

0 FULL CAPS, CENTERED (section title)
also begin new page and set page counter to 1
1 FULL CAPS, UNDERLINED
2 First Caps, Underlined
3 FULL CAPS
4 First Caps

They also put the proper white space around the heading and check the room left on the page before placing the paragraph heading and related text on the page. Then, using the heading level lines, they generate a table of contents automatically. For module descriptions, they generate header lines automatically and ignore all heading lines except the module names (i.e., all the repeated headings like "Usage" and "Notes") for table of contents generation. For addenda work, they handle point pages generation and produce required blank pages automatically.

Therefore, we would like to install them as a part of the standard system in >unbundled (which is where compose lives). It is planned that a future release of Multics will offer more "general purpose" macros that let the user define specific actions, e.g., a section heading that is flush right in italics.

The only documentation on the macros will be in info segments also installed in >unbundled. Not all of the macros have info segments; if the macro is merely called by another macro, it has no info segment. The macros and info segments that should be installed are listed on the following pages.

---

compin macros

collating.compin
  coll_page.compin
  coll_cont.compin
conditional_space_1.compin
  conditional_space_2.compin
  conditional_space_3.compin
  conditional_space_4.compin
  conditional_space_5.compin
dot_page.compin
  dot_page_off.compin
fig.compin
  fig_on.compin
  fig_index.compin
  fig_header.compin
  fig_get_no.compin
init.compin
  init_mpm.compin
  init_plm.compin
init_prose_layout.compin
  init_module_layout.compin
  mpm_prose_layout.compin
  mpm_module_layout.compin
  plm_prose_layout.compin
  plm_module_layout.compin
10h.compin
  10toc.compin
  10exact.compin
  10setup.compin
11h.compin
  12h.compin
  13h.compin
  14h.compin
  11toc.compin
  12toc.compin
  13toc.compin
  14toc.compin
  11exact.compin
  12exact.compin
  13exact.compin
  14exact.compin
  11setup.compin
  12setup.compin
  13setup.compin
  14setup.compin
  11mh.compin
  12mh.compin
  13mh.compin
  14mh.compin
  setbox.compin
  prose_box.compin
  prose_box_off.compin

```
  output.compin
  exact_output.compin
preface.compin
  pf_cont.compin
  pf.compin
tab.compin
  tab_on.compin
  tab_index.compin
  tab_get_no.compin
  tab_header.compin
toc.compin
  toc_on.compin
  toc_header.compin
```

info segments

```
collating.info
  coll_page.info
  coll_cont.info
compart.info
  compose.artwork.info
dot_page.info
  dot_page_off.info
fig.info
fig_get_no.info
fig_index.info
fig_on.info
init.info
  init_plm.info
  init_mpm.info
l0exact.info
l0h.info
l0toc.info
  l1toc.info
  l2toc.info
  l3toc.info
  l4toc.info
l1exact.info
  l2exact.info
  l3exact.info
  l4exact.info
l1h.info
  l2h.info
  l3h.info
  l4h.info
l1mh.info
  l2mh.info
  l3mh.info
  l4mh.info
l0setup.info
  l1setup.info
  l2setup.info
```

```
    13setup.info
    14setup.info
  macros.info
    compose_macros.info
    macro_use.info
    macro.info
    formatting_macros.info
  preface.info
    pf.info
    pf_cont.info
  tab.info
  tab_get_no.info
  tab_index.info
  tab_on.info
  toc_on.info
```

The info segments are on the following pages. They are given in the order listed above except for the one "general" info segment, macros.info, which is presented first for obvious reasons and the artwork info segment, which is given last.

07/11/78 macro use

Function:
A set of compose macros, developed for use by the Multics documentation people, is availabe for general use. Since the macros were developed as special purpose items, they follow current Honeywell standards and specific Multics documentation format rules. However, many of the macros are general enough to be useful tools for any users who are doing online documentation of almost any variety.

This info segment briefly describes the actions of the various macros, lists the macros according to function (general purpose, figure and table, addenda, or miscellaneous), and tells users how to include the macros in their search rules. More detailed information on a specific macro can be found in the info segment for that macro (e.g., dot_page.info).

Macro actions:
1. provide proper spacing around paragraph title (i.e., 3 spaces above and 2 below for oversize page, 2 spaces above and 1 below for standard size page; see init.info for more information)
2. generate proper formatting of paragraph title, based on the following Honeywell standard:
   level 0: section name; full caps, centered, new page
   level 1: full caps, underlined
   level 2: initial caps, underlined
   level 3: full caps
   level 4: initial caps
3. ensure sufficient space on current page for new paragraph title and related text
4. generate Table of Contents automatically, based on heading level macros used
5. generate header lines for command or subroutine descriptions (identified as "module" text type in macros; see 11h.info for more information on modules)
6. generate table and figure titles with proper spacing between title and item (according to Honeywell standards, table title is above table and figure title is below figure)
7. keep a table counter and a figure counter so the number in the title is incremented automatically and user can make references without knowing the actual number of the table or figure
8. generate table and figure Table of Contents segments automatically, based on the table and figure macros used
9. generate dot pages for addendum (e.g., pages 2.1 and 2.2 between existing pages 2 and 3)

General purpose macros:
init_mpm, init_plm, init
Initialize macro environment (one of these is required in order to use any other macros).
10h, 11h, 12h, 13h, 14h
Generate section and paragraph headings (including module type headings; see 11h.info for more information); maintain Table of Contents.

l0exact, l1exact, l2exact, l3exact, l4exact
l0toc, l1toc, l2toc, l3toc, l4toc
Generate section and paragraph headings that contain special strings; maintain Table of Contents entries that contain special strings.
toc_on
Turn on Table of Contents generation.

Figure and table macros:
fig_on, fig, fig_get_no, fig_index
Turn on/generate entries in a Figures Table of Contents.
tab_on, tab, tab_get_no, tab_index
Turn on/generate entries in a Tables Table of Contents.

Addenda macros:
dot_page, dot_page_off
Begin/end a set of addendum pages with page numbers of the form <page_count>.<addendum_page_count> or <section_number>-<page_count>.<addendum_page_count> (e.g., 3.1 or 5-3.1).
l0setup, l1setup, l2setup, l3setup, l4setup
Generate header lines so module segments can be output properly without beginning on first page (only needed for module type of text; see l1h.info for more information on modules).
collating, coll_page, coll_cont
Generate proper format for collating page(s), including special footers. (Collating page consists of instructions for removing and adding pages in addendum.)

Miscellaneous macros:
preface, pf, pf_cont
Generate proper format for preface page(s), including special footers.
l1mh, l2mh, l3mh, l4mh
Generate midpage headers for module type of text; maintain Table of Contents.

Macro location:
The compose macros are in the >unbundled directory. The user must add this directory to the various search mechanisms in order to use the macros. To do this, type the following commands (or put them in your start_up.ec):

```
add_search_paths compose >unbundled -after -working_dir
add_search_rules >unbundled -after working_dir
```

Special words:
There are many special words used throughout the macros. Any person writing variations of these macros or separate macros that will be used with the documentation macros must be aware of these special words.

Builtin variable words--
CallingFileName
CallingLineNo
Date
FileName
FrontPage
HeadSpace
Indent
InputFileName
PageLine
PageNo
PageWidth
ParamPresent
Parameter
VMargHeader
Widow

Active function words--
index
length
mod
reverse
substr
underline
uppercase
verify

User interface words--
add_date
add_letter
draft
draft_date
section
style

Macro entryname words--

addendum	l0exact
addendum_off	l0h
conditional_space_1	l0setup
conditional_space_2	l0toc
conditional_space_3	l1toc
dot_page	l2toc
dot_page_off	l3toc
exact_output	l4toc
fig	mpm_module_layout
fig_get_no	mpm_prose_layout

fig_header
fig_index
fig_on
init
init_module_layout
init_mpm
init_plm
init_prose_layout
plm_module_layout
plm_prose_layout
prose_box
prose_box_off
tab_get_no
tab_header
tab_on
toc_header
toc_on

Internal words--

BREAKER
BREAKER_LOOP
DOCUMENT_TYPE
DOT_PAGE
ENTRY
FIG
FIGING
FIGURE_COUNT
FIG_REF
FILENAME
INDEX
INIT
INIT_MPM
INIT_PLM
LENGTH
LEVEL
LEVEL_FILL
LEVEL_SIZE
box_length
box_line
box_word
call_header
check_blank
end_space
fig_loop
fig_output_index
figing
figure
figure-title
figure_count
figure_section
figure_title
figure_title*
figure_title*L
figure_titleI
figure_titleL
tab_index
tab_loop
tab_output_index
tabing
table
table_count
table_section
table_title*
table_title*L
table_titleI
table_titleL
table_titleN
title_title
toc_line
toc_line_
toc_line_1
toc_line_2
tocing

LSN
LinesLeft
MACROS_OK
MACRO_INIT
MARK
MPM
N
NEXT_FIG_TITLE
NEXT_FRONT
NEXT_PAGE
NEXT_TAB_TITLE
NO_SPLIT
ORDER_NUMBER
PAGE_TAB
PART
PART_LENGTH
PART_LOOP
PART_START
PLM
figure_titleN
form_1_or_2
form_1_or_2_init
form_1_or_2_midpage_header_init
form_1_or_2_midpage_header_mpm
form_1_or_2_midpage_header_plm
form_1_or_2_mpm
form_1_or_2_plm
form_3_or_4
form_3_or_4_init
form_3_or_4_mpm
form_3_or_4_plm
form_5
form_5_init_module
form_5_init_prose
form_5_mpm_module
form_5_mpm_prose
form_5_plm_module
form_5_plm_prose

SECTION
SECTION_INDENT
SECTION_IN_TOC
SETUP
SET_TITLE
TABING
TABLE_COUNT
TAB_REF
TEXT_TYPE
TOC
TOCING
TOC_REF
TOTAL_LINES
TR_VEC
addendum
bad_arg
increment_count
init_space
l0exact_section_name
l0h_section_name
l0setup_section_name
l1h_parms
l1mh_parms
l2h_parms
l2mh_parms
l3h_parms
l3mh_parms
l4h_parms
l4mh_parms
no_section
section_head
section_no

This page intentionally left blank.

07/11/78 collating.compin
coll_page.compin
coll_cont.compin

Function:
This compin macro initializes a collating page (or pages) for an addendum to a manual. It:
1) generates the heading, standard "remove and insert" sentence, and remove and insert column headings (see "Notes" below)
2) generates the proper footers for the first and succeeding pages

Syntax:

```
.ifi collating "copyright_year"  OR  .ifi coll_page "copyright_year"
.srv file_no "Honeywell_file_number"
.srv add_letter "what_addendum"
.srv add_date "publish_date"
```

and at the top of the second page (to set the proper footers):

```
.ifi coll_cont
```

Arguments:

copyright_year is the copyright date for this addendum, not the entire manual.

Honeywell_file_number is the four-character file number as defined in Table 1-3 in Part III, "Writing and Editing," of the Honeywell Publications Standards.

what_addendum is a single capital letter, indicating which addendum this is (i.e., A is the first addendum; B, second; etc.).

publish_date is the month and year, in the form mm/yy, that this addendum is published (generally, "published" here means "brought to the printer"). By convention, leading zeros are not used so a publish date of August 1978 would be 8/78 not 08/78.

Notes:
The column headings were set up assuming the user's collating instructions segment uses ".inl 5" and puts the first column at the margin and the second at column 41 (so tabs can be used easily).

Example:
The following compin segment:

```
.ifi init_mpm "AT58"
.ifi collating "1978"
.srv file_no "1L13"
.srv add_letter "A"
.srv add_date "8/78"
.inl 5
iii through vii                    iii through vii
.spb
```

2-1 through 2-4	2-1, 2-2 2-3, 2-3.1 2-3.2, 2-4

.spb

.
.
.
.

produces the following compout:

COLLATING INSTRUCTIONS

To update this manual, remove old pages and insert new pages as follows:

Remove	Insert
iii through vii	iii through vii
2-1 through 2-4	2-1, 2-2 2-3, 2-3.1 2-3.2, 2-4

.
.
.

 File No.: 1L13

DRAFT: MAY BE CHANGED 8/78 08/26/78 AT58A

07/11/78 dot_page.compin
dot_page_off.compin

Function:
The dot_page macro initializes addendum footer lines. It:
1) turns on the date and addendum letter portion of the footers
2) turns on an automatic counter for "point" pages
3) generates blank backup pages if necessary (see "Blanks:" below)
The dot_page_off macro resets the page counter to normal (whatever type of page numbers were in use before using dot_page).
BEWARE! These macros cause page breaks.

Syntax:

```
.ifi dot_page
.srv add_date "publish_date"
.srv add_letter "what_addendum"
```

and at the top of the page where normal counter resumes:

```
.ifi dot_page_off
{.srv add_date ""}
{.srv add_letter ""}
```

Notice that the ".srv" lines are required with the dot_page macro but optional with the dot_page_off macro. This is true because point pages only exist in an addendum, but not all addendum pages need be point pages.

Arguments:
publish_date is the month and year, in the form mm/yy, that this addendum is published (generally, "published" here means "brought to the printer"). By convention, leading zeros are not used so a publish date of August 1978 would be 8/78 not 08/78.
what_addendum is a single capital letter, indicating which addendum this is (i.e., A is the first addendum; B, second; etc.).

Notes:
In order to add addendum pages, knowledge of the current and correct placement of all text on the pages involved is necessary. Also, keep in mind that use of either of these macros forces a new page so placement of the actual ".ifi" line is crucial.

Blanks:
The dot_page macro assumes that the material will be submitted for printing and therefore both sides of the printed page must be considered. If a blank page is required, it is labeled as such ("This page intentionally left blank.") and the proper addendum footers -- without a page number -- are generated.

Examples:
Assume compose encounters the following lines in the compin segment as it is formatting the twelfth page of the section (numbered either 12 or <section_number>-12).

```
.ifi dot_page
.srv add_date "8/78"
.srv add_letter "B"
```

adds the addendum publish date in the left side of the footer; adds the addendum letter to the order number in the footer; and turns on the point page page number counter so the next page number is 12.1, next is 12.2, etc. (or <section_number>-12.1, etc. if you have put in a ".srv section N" line earlier), until:

```
.ifi dot_page_off
```

turns off the point page counter and returns to normal page numbers so the next page number is 13 (or <section_number>-13). Since no new expressions where given to the add_date and add_letter reference names, the footer continues to use those values.

07/11/78 fig.compin

Function:
This compose macro:
1) increments the figure counter;
2) outputs a centered, perhaps multiline figure title below the figure;
3) outputs appropriate spacing between figure and figure title;
4) adds the figure title to the Figures Table of Contents.

The figure counter is maintained on a per-section basis for manuals done in sections (those in which the 'section' variable was given for the 10h.compin macro), and on a per-manual basis for all other manuals.

Syntax: .ifi fig "figure-title"

Arguments:
figure-title is the title of the figure, as it is to appear in the Figures Table of Contents (first caps, no underlines).

Examples:
The following lines generate a simple figure preceding some text.

```
.spb 3
.brn 6

     ¦¯¯¯¯¯¯¯¯¯¦            ¦¯¯¯¯¯¯¯¯¯¯¦
     ¦__________¦            ¦__________¦
.spb
.ifi fig "Two Boxes"
.ur The two boxes in Figure %figure% above ...
```

Notes:
After the fig macro is used, the 'figure' compose variable is set to the figure number of the last figure, and 'figure_title' is set to the title of this figure.

The centered figure label has the form:
Figure %figure%. Figure Title
or:
Figure %figure%. First Part of Figure Title
Second Part of Figure Title

'figure' can be used in a .ur line to have the text refer to the last figure. Use the fig_get_no.compin macro to refer to a figure that appears later in the text.

Long figure titles can be forced to break at certain points by using an exclamation point (!) where the break is to occur. For example:

```
.ifi fig "This Is The Longest Figure Title!Seen So Far"
```

Each exclamation point is replaced by a space in the title assigned to 'figure_title', and in the title placed in the Figures Table of Contents.

07/11/78 fig_get_no.compin

Function:
This compose macro gets the number of a figure that will appear later in the current section of a section manual (one in which the 'section' variable was given with the 10h macro), or later in the manual for a nonsection manual. The number is assigned to the 'figure' compose variable. 'figure' can be used in a .ur line to reference a figure that appears below in the text.

Syntax: .ifi fig_get_no n

Arguments:
n is an integer; the number of the nth figure after the last one output is assigned to 'figure'.

Examples:
The following lines reference the next three figures that will appear in the text.

```
.ifi fig_get_no 1
.ur Figure %figure%,
.ifi fig_get_no 2
.ur Figure %figure%, and
.ifi fig_get_no 3
.ur Figure %figure% illustrate these results.
```

07/11/78 fig_index.compin

Function:
This compose macro adds a title to the Figures Table of Contents without outputting the title in a centered figure label. It can be used when a noncentered title must be output below a figure, or when the title must be output in a specific way. The macro should be inserted just AFTER a figure label has been output below the figure.

Syntax: .ifi fig_index "figure-title"

Arguments:
figure-title is the title of the figure, as it is to appear in the Figures Table of Contents (first caps, no underlines).

Examples:
The following lines produce a figure with a left-justified figure label.

```
.ifi fig_get_no 1
.spb
.ur Figure %figure%:  Figure Data, 1975
.spb 3
.ur Figure %figure% above reflects the 1975 data.
.ifi fig_index "Figure Data, 1975"
```

07/11/78 fig_on.compin

Function:
This compose macro turns on the automatic generation of a Figures Table of Contents. This macro should be used ONCE in the 'book' compin segment. If the 'book' segment is named my_book.compin, then the Figures Table of Contents segment that is generated is named my_book.fig.compin.

Syntax: .ifi fig_on

Example:
The following is a sample 'book' compin segment named AAnn_book.compin.

```
.ifi init_mpm "AAnn"
.ifi toc_on
.ifi fig_on
.ifi tab_on
.ifi AAnn.tp
.ifi AAnn.pf
.ifi s1
.ifi s2
      .
      .
      .
.ifi AAnn_book.toc
.ifi AAnn_book.fig
.ifi AAnn_book.tab
```

07/11/78 init, init_plm, init_mpm

Function:
These compose macros initialize the segment so the other documentation macros can be used and perform the following:
1) set up proper page size and vertical margins (see "Output pages:" below)
2) indent the left margin 0
3) turn on fill mode
4) align both the left and right margins
5) make all exclamation points (!) translate to spaces in output (see "Notes:" below)

Syntax: .ifi init {"footer_info"}
or .ifi init_plm {"footer_info"}
or .ifi init_mpm {"footer_info"}

Arguments:
footer_info may be a manual order number or any other information that the user wants in the bottom right-hand corner of every page. If no footer_info argument is given, a null character string is used. (The footer_info character string is called ORDER_NUMBER in the macros) See also "Default footer information:" below.

Output pages:
The three initializing macros generate slightly different output pages.
init 'standard ' size pages (to fit 8-1/2 x 11 paper), using all the default page definitions and vertical margins
init_plm 'standard' size pages, using default page definitions but NOT default vertical margins
init_mpm oversize pages suitable for reduction to 83% of their printed size, using no default vertical margins (when reduced, these pages also fit 8-1/2 x 11 paper)

Notes:
One of the 'init' macros (init.compin, init_plm.compin, or init_mpm.compin) MUST be the first line in the segment in order to use all the other documentation macros.

The translation of an exclamation point into a space is used to force related characters or words to appear on the same line of output (e.g., Figure!3, Dr.!Johnson). To actually output an exclamation point, do the following:

```
.trf !!
An exclamation point (!) is used...
.trf !
```

Default footer information:
The 'init' macros automatically set two variables that become part of the footer line:

```
.srv draft "DRAFT: MAY BE CHANGED"
.srv draft_date "%Date%"
```

To remove this information, put the following lines after the init_mpm or init_plm line:

```
.srv draft ""
.srv draft_date ""
```

07/11/78 10exact.compin

Function:
This compose macro performs part of the functions of 10h. It:
1) initializes a new section of the manual;
2) generates a section heading on a new page without translating the section-title to uppercase;
3) does NOT output the section title in the Table of Contents.

It is used when a section-title contains a literal string that must be kept in lowercase characters. It should be inserted at the beginning of a section of the manual.

Syntax: .ifi 10exact "section-title"

Arguments:
section-title is the title of the section, exactly as it is to appear in the section heading.

Notes:
Use 10toc.compin to put a section title in the Table of Contents.

Examples:
A section on the exec_com control language could begin with...

```
.ifi init
.ifi 10exact "THE exec_com CONTROL LANGUAGE"
.ifi 10toc "The exec_com Control Language"
```

07/11/78 10h.compin

Function:
This compose macro:
1) initializes a new section of a manual;
2) generates a section heading that is full caps and centered (see "Section numbering:" below);
3) sets the page counter to 1;
4) adds the section title to the Table of Contents.

This macro should be inserted at the beginning of each section of the manual.

Syntax:

```
.ifi 10h "title-of-section"

  or, for numbered sections (see "Section numbering:" below)
.srv section "section-no"
.ifi 10h "title-of-section""
```

Arguments:
title-of-section is the title of the section as it is to appear in the Table of Contents (i.e., first caps, no underlines). It is translated to uppercase when output in the section heading.
section-no is either a section number (e.g., 2) or an appendix letter (e.g., A). This value is output (e.g., SECTION 2 or APPENDIX A) in a centered line preceding the section title line and is used in page numbers, figure numbers, and table numbers (e.g., the first page, figure, or table number when section-no is 2 is 2-1; when section-no is A, A-1).

Section numbering:
If you use numbered sections (i.e., use .srv section "section-no") the macros automatically use Arabic numbers in the section heading and as part of the page numbers, figure numbers, and table numbers (e.g., 2-1).

You can get Roman numerals in the section heading (e.g., SECTION II preceding the section title line) by using the following line immediately BEFORE the 10h line:

```
.srv style "roman"
```

The section-no portion of the page number, figure number, and table number will still be Arabic.

Examples:
The first section of the MPM Commands might begin with:

```
.ifi init_mpm "AG92"
.srv section "1"
.ifi 10h "Multics Command Environment"
.spb
```

A section of a manual that does not use section numbers might begin with:

```
.spb
.ifi init "My Manual"
.ifi 10h "Manual of Instructions"
.spb
An appendix of a manual might begin with:
.spb
.ifi init "My Manual"
.srv section "A"
.ifi 10h "Summary of Operation Properties"
```

Notes:
The section title is translated to uppercase when output in the section heading. Use the 10exact and 10toc macros for section titles containing literal strings that should not be translated to uppercase.

07/11/78 10toc.compin, 11toc.compin, 12toc.compin, 13toc.compin, 14toc.compin

Function:
These compose macros add a section or paragraph title to the Table of Contents at the appropriate level.

They are useful for handling titles that contain literal strings, and for including section and paragraph titles in the Table of Contents for sections that have not been written yet. The macros should be inserted immediately AFTER the title is output in the text, or inserted where the unwritten section or paragraph belongs.

Syntax: .ifi 1Xtoc "title"

Arguments:
title is the section-title or paragraph-title to be added to the Table of Contents. It will be added without translation or underlining.

Notes:
The 1Xexact.compin macros can be used to output a title containing literals without translation or underlining, and without adding the title to the Table of Contents.

Examples:
A paragraph describing the qedx editor might begin:

.ifi 11exact "USING THE qedx COMMAND"
.ifi 11toc "Using the qedx Command"

07/11/78 l1exact.compin, l2exact.compin, l3exact.compin, l4exact.compin

Function:
These compose macros perform part of the functions of l1h, l2h, l3h, and l4h. They:
1) generate level 1 through level 4 paragraph headings WITHOUT any translation or underlining of the paragraph title;
2) provide appropriate spacing around the paragraph headings;
3) ensure there is sufficient room on the current page for a new paragraph;
4) do NOT add the paragraph title to the Table of Contents.

Syntax: .ifi lXexact "paragraph-title"

Arguments:
paragraph-title is the title of the paragraph, exactly as it is to appear when output in the paragraph heading.

Notes:
Use l1toc.compin, l2toc.compin, l3toc.compin, and l4toc.compin to put a paragraph-title in the Table of Contents at the appropriate level.

It is not appropriate to use the lXexact macros for a module-paragraph-title since the lXh macros do not translate or underline such titles anyway. lXexact macros should only be used for a nonmodule paragraph-title. A module-paragraph-title has one of the following forms:
Name: name
Names: name1, name2, ..., nameN
Entry: name$offset
Entries: name1$offset1, ..., nameN$offsetN

Examples:
A section describing the compose text formatter might contain:

```
.ifi init_mpm "AZ98"
.srv section 4
.ifi l0exact "WORDPRO TEXT COMPOSER"
.ifi l0toc "WORDPRO Text Composer"
     The following...
.ifi l1exact "compose COMMAND"
.ifi l1toc "compose Command"
     The compose command ...
.ifi l1h "Terminology"
```

07/11/78 11h.compin, 12h.compin, 13h.compin, 14h.compin

Function:
These compose macros:
1) generate level 1 through level 4 paragraph headings;
2) provide appropriate spacing around the paragraph-title;
3) add the paragraph-title to the Table of Contents;
4) ensure there is sufficient room for a new paragraph on the current page.

Syntax: .ifi 1Xh "paragraph-title"
or
.ifi 1Xh "module-paragraph-title"

Arguments:
module-paragraph-title has one of the following forms (See "Module and prose modes:" below):
Name: name
Names: name1, name2, ..., nameN
Entry: name$offset
Entries: name1$offset1, ..., nameN$offsetN
paragraph-title is any other form of paragraph title, just as it is to appear in the Table of Contents (first caps, no underlines). The title is translated to uppercase and/or underlined when output in the paragraph heading, depending upon which macro is used (See "Macro translations:" below).

Macro translations:
The following translations are performed on the paragraph-title when it is output in a paragraph heading:

MACRO	TRANSLATION
11h	All uppercase, underlined
12h	Underlined
13h	All uppercase
14h	No translation

Module and prose modes:
When a module-paragraph-title is given, the formatting macros switch from prose- to module-description mode (.srv text_type "module"). This results is the following:
1) the module-paragraph-title is output in the paragraph heading exactly as given without any translation, no matter which macro was used.
2) Name or Names module-paragraph-title forms cause a new module description to begin on a new page, with the first (or only) name placed in a box heading at the top of each page.
3) Table of Contents entries for a module-paragraph-title exclude the underlined caption (e.g., Names: is excluded).
4) A (nonmodule) paragraph-title is NOT included in the Table of Contents when in module mode.

The macros return to prose-description mode when:
1) one of the initialization macros (init_mpm, init_plm, or init) is used, or
2) a new section begins (macro 10h), or
3) the user sets the text_type variable (.srv text_type "prose").

Examples:
A section in the MPM Commands might contain:

```
.ifi init_mpm "AG92"
.srv section 1
.ifi 10h "Multics Command Environment"
     The Multics command environment ...
.ifi 11h "Reference to Commands by Function"
     The Multics command repertoire ...
.ifi 12h "Access to the System"
.spb
.spb
```

A module description with a standard size page might contain:

```
.spb
.ifi init_plm "AN80"
.ifi 11h "Names:  check_mst, ckm"
     This command is used to read one ...
.ifi 12h "Usage"
     check_mst input_args control_args
.spb 1
where:
.spb 1
.inl 12
.unl 12
1.   input_args
.brf
are arguments of the form ...
.spb
.unl 12
2.   control_args
     .
     .
     .
.inl 0
.ifi 12h "Notes"
     The control arguments ...
.ifi 12h "Entry:  check_mst$test"
     This entry point may be used to test the ...
```

*/11/78 11mh.compin, 12mh.compin, 13mh.compin, 14mh.compin

Function:
These compose macros are for use with 'Name' modules only. They perform the same actions as the 1Xh macros do with modules except they create midpage headers so you can get more than one 'Name' item on a single page. (These macros were used throughout the active functions portion of the MPM Commands.)

Syntax:

```
.ifi 1Xmh "Name-module-paragraph-title"
```

Arguments:
Name-module-paragraph-title has one of the following forms:
Name: name
Names: name1, name2, ..., nameN

Examples:
The active functions portion of the MPM Commands might contain:

```
.ifi init_mpm "AG92"
.srv section "2"
.l0h "Active Functions"
.ifi af_intro_material
.ifi l1mh "Name: and"
     The and active function...
.ifi l1mh "Name: ceil"
     The ceil active function...
```

07/11/78 10setup.compin, 11setup.compin, 12setup.compin, 13setup.compin, 14setup.compin

Function:
The setup macros set up a proper environment and generate the header lines (boxes at the top of each page that you get when in module mode) but NOT the 'Name:' heading line in the text itself (use of '.ifi 1Xh "Name: ..."' generates both.)
Use of these macros is generally limited to special addenda work e.g., a 40-page command description in which only the last five pages change and they are written to a new segment or new material is being added to the description but kept in a separate segment.

Syntax: .ifi 1Xsetup "Name-module-paragraph-title"

Arguments:
Name-module-paragraph-title has one of the following forms:
Name: name
Names: name1, name2, ..., nameN

Examples:
Assume that my_command is about 40 pages long and a summary portion is being added in addendum A following page 3-256. The following lines could be the beginning of that summary segment:

```
.ifi init_mpm "ZZnn"
.srv section 3
.ifi 11setup "Name: my_command"
.brp 256
.ifi dot_page
.srv add_letter "A"
.srv add_date "7/78"
.ifi 12h "Summary"
     A summary of the my_command is...
```

7/11/78 preface.compin
pf.compin
pf_cont.compin

Function:
This compose macro initializes a preface for a manual. It:
1) generates the centered preface heading and the proper spacing around the heading
2) does NOT put anything in the table of contents (since the preface is not supposed to be put in it anyway)
3) generates the proper footers for the first and succeeding pages of the preface
   a) first page:
      copyright notice and Honeywell file number
   b) second and succeeding pages:
      lowercase roman numerals, beginning with iii, plus order number

Syntax:
```
.srv file_no "Honeywell-file-number"
.ifi preface "copyright-year(s)"  OR  .ifi pf "copyright-year(s)"
```
and, for the top of the second page (to reset the footers, etc.):
```
.ifi pf_cont
```

Arguments:
Honeywell-file-number is the four-character file number as defined in Table 1-3 in Part III, "Writing and Editing," of the Honeywell Publications Standards.
copyright-year(s) is the copyright date for the manual. If more than one year must be given, separate the years by a comma-space combination (e.g., .ifi preface "1973, 1977").

Example:
The following could be the preface of the MPM Commands:
```
.ifi init_mpm "AG92"
.srv file_no "1L13"
.ifi preface "1975, 1977"
     Primary reference for user and
       .
       .
       .
.ifi pf_cont
     The MPM I/O manual
       .
       .
       .
       .
```

07/11/78 tab.compin

Function:
This compose macro:
1) increments the table counter;
2) outputs a centered, perhaps multiline table title above the table;
3) outputs appropriate spacing between table title and table;
4) adds the table title to the Tables Table of Contents.

The table counter is maintained on a per-section basis for manuals done in sections (those in which the 'section' variable was given for the 10h.compin macro), and on a per-manual basis for all other manuals.

Syntax: .ifi tab "table-title"

Arguments:
table-title is the title of the table, as it is to appear in the Tables Table of Contents (first caps, no underlines).

Examples:
The following lines generate a simple table following some text:

```
These relationships are shown in the table below.
.ifi tab "Relationships Table"
     NAME            PLACE
     Paris           France
     Rome            Italy
       .
       .
       .
```

Notes:
After the tab macro is used, the 'table' compose variable is set to the table number of the last table, and 'table_title' is set to the title of this table.

The centered table heading has the form:

```
  Table %table%.  Table Title
```

or:

```
  Table %table%.  First Part of Table Title
       Second Part of Table Title
```

'table' can be used in a .ur line to have the text refer to the last table. Use the tab_get_no.compin macro to refer to a table that is to appear later in the text.

Long table titles can be forced to break at certain points by using an exclamation point (!) where the break is to occur. For example:

```
.ifi tab "This Is The Longest Table Title!Seen So Far"
```

Each exclamation point is replaced by a space in the title assigned to 'table_title', and in the title placed in the Tables Table of Contents.

07/11/78 tab_get_no.compin

Function:
This compose macro gets the number of a table that will appear later in the current section of a section manual (one in which the 'section' variable was given with the 10h macro) or later in the manual for a nonsection manual. The number is assigned to the 'figure' compose variable. 'table' can be used in a .ur line to reference a table that appears below in the text.

Syntax: .ifi tab_get_no n

Arguments:
n is an integer; the number of the nth table after the last one output is assigned to 'table'.

Examples:
The following lines reference the next three tables that will appear in the text.

```
.ifi tab_get_no 1
.ur Table %table%,
.ifi tab_get_no 2
.ur Table %table%, and
.ifi tab_get_no 3
.ur Table %table% show these results.
```

07/11/78 tab_index.compin

Function:
This compose macro adds a title to the Tables Table of Contents without outputting the title in a centered table heading. It can be used when a noncentered title must be output above a table, or when the title must be output in a specific way. The macro should be inserted just AFTER a table heading has been output above the table.

Syntax: .ifi tab_index "table-title"

Arguments:
table-title is the title of the table, as it is to appear in the Tables Table of Contents (first caps, no underlines).

Examples:
The following lines produce a table with a left-justified table heading.

```
.ifi tab_get_no 1
.spb 3
.brn 10
.ur Table %table%:  Table Data, 1975
.ifi tab_index "Table Data, 1975"
```

07/11/78 tab_on.compin

Function:
This compose macro turns on the automatic generation of an Tables Table of Contents. This macro should be ONCE in the 'book' compin segment. If the 'book' segment is named my_book_.compin, then the Tables Table of Contents segment that is generated is named my_book.tab.compin.

Syntax: .ifi tab_on

Example:
The following is a sample 'book' compin segment named AAnn_book.compin.

```
.ifi init_mpm "AAnn"
.ifi toc_on
.ifi fig_on
.ifi tab_on
.ifi AAnn.tp
.ifi AAnn.pf
.ifi s1
.ifi s2
      .
      .
      .
.ifi AAnn_book.toc
.ifi AAnn_book.fig
.ifi AAnn_book.tab
```

07/11/78 toc_on.compin

Function:
This compose macro turns on the automatic generation of a Table of Contents. This macro should be used ONCE in the 'book' compin segment. If the 'book' segment is named my_book.compin, then the Table of Contents segment that is generated is named my_book.toc.compin.

Syntax: .ifi toc_on

Example:
The following is a sample 'book' compin segment named AAnn_book.compin.

```
.ifi init_mpm "AAnn"
.ifi toc_on
.ifi fig_on
.ifi tab_on
.ifi AAnn.tp
.ifi AAnn.pf
.ifi s1
.ifi s2
      .
      .
      .
.ifi AAnn_book.toc
.ifi AAnn_book.fig
.ifi AAnn_book.tab
```

07/11/78 frequently used artwork

This info segment contains information on some of the artwork constructs available within the compose command. It permits the user to insert certain overstruck character patterns into an input file and to display them as various symbols and line art features. The default mode for artwork is ASCII. For devices with plotting capability, a device driver table must be checked. The device tables are found in >unbundled. The correct device must be made known to compose via the -dv control argument.

Two very important things to remember when using artwork:
1) adjust and fill must be off (i.e., precede the artwork with .fif)
2) enclose the artwork in an artwork block (i.e., .bba before the artwork and .bea after)

Some of the most frequently used symbols for artwork are as follows: (Note: <BS> stands for the ASCII backspace character, octal value 010, and indicates that the characters adjoining are overstruck.)

Diamonds:

/<BS>" Begin a +45 degree slant rule

\<BS>" Begin a -45 degree slant rule

/<BS>~ End the +45 degree slant rule

\<BS>~ End the -45 degree slant rule

Vertex:

<<BS>" Insert a diamond left vertex

><BS>" Insert a diamond right vertex

v<BS>" End a diamond with bottom vertex

^<BS>" Begin a diamond with top vertex

Semicircle:

(<BS>" Insert a left semicircle

)<BS>" Insert a right semicircle

One line:

¦<BS>1 Insert a one-high vertical bar (for between text)

¦<BS>~ A one line vertical rule

¦<BS>~<BS>v A one line vertical rule with down arrow

¦<BS>~<BS>^ A one line vertical rule with up arrow

¦<BS>~<BS>- Insert a one line vertical rule and begin a horizontal rule

¦<BS>~<BS>* Insert a one line vertical rule and end a horizontal line

Horizontal rules:

-<BS>" Begin a horizontal rule

-<BS>* End a horizontal rule

-<BS>< Begin a horizontal rule with left arrow

><BS>* End a horizontal rule with right arrow

¦<BS>~<BS>- End a line vertical rule and end a horizontal rule

-<BS>*<BS>¦<BS>v End a horizontal rule and a vertical rule with a down arrow

Vertical rules:

¦<BS>" Begin a vertical rule

¦<BS>~ End a vertical rule

¦<BS>~<BS>- End a vertical rule and begin a horizontal rule

¦<BS>~<BS>* End a vertical rule and end a horizontal rule

¦<BS>~<BS>v End a vertical rule with a down arrow

-<BS>*<BS>¦<BS>v End a horizontal rule and a vertical rule with a down arrow

Arrows:

|<BS>~<BS>v A one line vertical rule with down arrow

|<BS>~<BS>^ A one line vertical rule with up arrow

-<BS><<BS>> Insert a left arrow and a right arrow

-<BS>< Begin a horizontal rule with left arrow

><BS>* End a horizontal rule with right arrow

-<BS>*<BS>|<BS>v End a horizontal and a vertical rule with a down arrow

Superscript:

S<BS>^ Raise to superscript level

S<BS>v Return to line base

Subscript:

H<BS>v Drop a half line

H<BS>^ Return to line base

Box:

-<BS>| Upper left box corner

*<BS>| Upper right box corner

-<BS>~ Lower left box corner

*<BS>~ Lower right box corner

Lozenge:

-<BS>/ Upper left begin horizontal and begin diagonal

*<BS>\ Upper right end horizontal and begin diagonal

/<BS>\<BS>~ Left OR right vertex (works for both points)

-<BS>\<BS>~ Lower left begin horizontal and end diagonal

-<BS>*<BS>/<BS>~ Lower right end horizontal and end diagonal

Miscellaneous:
For the following constructs, the n after the backspace controls the height of the artwork. The value of n can be:

1 - 0	to get artwork from 1 to 10 lines high
a - z	11 to 36 lines high
A - Z	31 to 56 lines high

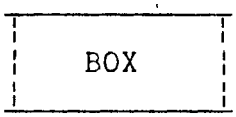
(<BS>n	Left parenthesis
)<BS>n	Right parenthesis
{<BS>n	Left brace
}<BS>n	Right brace
[<BS>n	Left bracket
]<BS>n	Right bracket
X<BS>1	A multiplication sign for one line
o<BS>1	(Letter "o" overstruck with 1) - bullet
=<BS>1	(Equals overstruck with 1) - concatenate

Examples:
Two simple artwork examples are shown below, both show the lines in the compin segment followed by the output as generated on an ASCII device.

```
--compin--
 .fif
 .bba
 .in 5
 +          *

     BOX
 =          *
 .bea
 .fin

--compout--
```



```
--compin--
 .fif
 .bba
 .inl 10
 .unl 5
 ð    Item 1
 .spf
 .unl 5
 ð    Item 2
 .spf
 .unl 5
 ð    Item 3
 .inl
 .bea
 .fin

--compout--

      ⊠   Item 1

      ⊠   Item 2

      ⊠   Item 3
```