RCSL Nr.: 31-D666

Udgave: November 1981

Forfatter: Lis Clement

Titel:

ALGOL Coroutine System
Testudskrivning
Brugervejledning

RCSL 42-i1591

A/S REGNECENTRALEN
af 1979

Nøgleord:

RC8000, ALGOL8, Coroutine System, testoutput, procedurer.

Resumé:

Dette skrift beskriver en metode til flexibel udskrivning af test filer genereret af RC8000 Coroutine System.

(24 trykte sider)

Udgivet af A/S Regnecentralen af 1979, København

RCSL 42-i1591

INDHOLDSFORTEGNELSE SIDE

1. INTRODUKTION 1.

Dette skrift beskriver en metode til fleksibel udskrivning af testfiler genereret af RC8000 Coroutine System. Brugervejledningen til coroutinesystemet, ref. [1], angiver at filen kan udskrives med fp-kommandoen

```
print <fil> word words.8
```

men det resulterer i en ren numerisk udskrift, der er meget besværlig at fortolke. Formålet med den udskrivningsmetode, der beskrives heri, er at lette fortolkningen af testoutput samt at muliggøre udvælgelse af individer på en nem måde.

Metoden er første gang anvendt i forbindelse med Scroll Mode Mapping (SMM) modulet i CENTERNET, hvilket er grunden til at den individuelle del af metoden er illustreret med det, der anvendes til SMM testoutput udskrivning.

Teksterne til procedurerne test_print og print_record findes i henholdsvis bilag B og C.

Manualen igennem anvendes termen bs-fil om begrebet baggrundslager fil (engelsk: backing storage file).

2. METODEN 2.

Metoden består af en standarddel og en brugerafhængig del.

Standarddelen sørger ud fra en passende trimning for at

- positionere i filen til først ønskede individ
- læse individerne
- udvælge de individer, der skal udskrives
- afslutte læsningen.

Den brugerafhængige del kan betragtes på to forskellige niveauer, den faste del og den varierende del. Den faste del er den udskrivningsmetode, der skal anvendes på de udvalgte individer, mens den varierende del er den trimning, der er nødvendig for i hvert enkelt tilfælde at få positioneret filen, samt at få udvalgt de ønskede individer fra den korrekte fil.

Standarddelen er implementeret som en external procedure - test_print - hvor positioneringen angives i parametrene, mens udvælgelsen af hvilke individer, der skal udskrives, sker ved hjælp af et logisk udtryk, der beregnes for hvert individ (Jensens Device). De udvalgte individer sendes ét ad gangen til en udskrivningsprocedure, som er med som parameter til test_print proceduren.

Den brugerafhængige del består af

- program hvorfra test_print proceduren kaldes med passende (= varierende) parametre
- procedure til udskrivning af testindividerne.

2.1 Proceduren test_print 2.1

Proceduren test_print positionerer bs-filen med det angivne dokumentnavn til det individ, der er udpeget af rec_no, hvorpå individerne læses og overspringes indtil tidsangivelsen i individet

er større end eller lig med cur_clock. Herefter undersøges udskrivningsbetingelsen og hvis den er sand kaldes proceduren print_record; dette gentages for de følgende individer indtil den logiske filafslutning, der indtræffer når blot en af de følgende betingelser er opfyldt:

- fysisk filafslutning
- tidsangivelsen i det aktuelle individ udpeger en dato, der er tidligere end 1. januar 1981 eller senere end dags dato
- tidsangivelsen i det aktuelle individ ligger før det foregående individs tidsangivelse.

Opstår andre statusfejl end end_of_document under læsning af testfilen, afsluttes programmet med standardproceduren std_error.

2.1.1 Kald 2.1.1

Den eksterne procedure test_print kaldes således:

```
test_print(doc, rec_no, coroutine, type, cur_clock, print,
          print_record)
```

hvor

doc — (kald værdi, string)
dokumentnavnet på den bs-fil, der indeholder test individerne. (Læses kun én gang).

rec_no — (kald og retur værdi, heltal)
individ nummer
kald værdi:
= 0 start med første individ i filen
> 0 start efter rec_no individer i filen
< 0 start |rec_no| individer fra den logiske fil afslutning
retur værdi:
individ number på sidst læste individ + 1

coroutine (retur værdi, heltal)
coroutine identifikation
retur værdi:
coroutine identifikation i det sidst læste individ (6. heltal). NB: Kan være negativ jf. ref. [1].

type (retur værdi, heltal)
individ type
retur værdi:
individ typen i det sidst læste individ (1. heltal)

cur_clock (kald og retur værdi, heltal)
klokken på formen ttmmss
kald værdi:
= 0 start med rec_no individet
> 0 start med det første individ efter rec_no individet, hvis klokke er større end eller lig med cur_clock
retur værdi:
den (formatterede) klokke i det sidst læste individ (7. + 8. heltal = sidste long)

print (kald værdi, logisk udtryk)
udskrivningsbetingelse, der udregnes på basis af parametrene rec_no, coroutine, type og cur_clock, som opdateres for hvert individ, der læses (Jensens Device)
= sand medfører at proceduren print_record kaldes
= falsk medfører at det aktuelle individ springes over

print_record (procedure)
procedure, der kaldes såfremt udskrivningsbetingelsen for det aktuelle individ er sand. Se desuden afsnit 2.2.

2.1.2 Fejludskrifter

Et kald af proceduren test_print kan resultere i følgende udskrifter på current output:

> relative record number greater than max record number,
> i.e. <kald værdien af recno> >= <antal individer>

Denne udskrift kommer, hvis der er <antal individer> i den logiske fil og proceduren er blevet bedt om at starte et større antal individer fra afslutningen. Proceduren fortsætter som om rec_no = 0.

> file exhausted; record number <antal individer>

Denne udskrift kommer, hvis proceduren er blevet kaldt med en kombination af rec_no og cur_clock, som ikke kan opfyldes inden den logiske filafslutning. Proceduren afslutter.

2.2 Proceduren print_record

Proceduren print_record er brugerafhængig og funktionen af den kan derfor ikke beskrives. Det eneste der står fast er, at dens parametre skal erklæres som følger:

```
procedure print_record (record, rec_no, cur_date, cur_clock);
value                          rec_no, cur_date, cur_clock ;
integer array          record                              ;
integer                        rec_no, cur_date, cur_clock ;
```

samt at den kaldes af test_print, når udskrivningsbetingelsen (print) er sand for det aktuelle individ.

Navnet print_record er blot det formelle navn, som er anvendt i erklæringen af proceduren test_print; der kan vælges et hvilket som helst navn for udskrivningsproceduren. Parametrene til

print_record er alle kaldeparametre, som ikke må ændres af proceduren:

record	heltalsvektor (1:8) det testindivid, der skal udskrives
rec_no	heltal det aktuelle individnummer talt fra starten af den fysiske fil (startende med nummer 1)
cur_date	heltal, ååmmdd den aktuelle dato læst fra record (og fortolket)
cur_clock	heltal, ttmmss det aktuelle klokkeslet læst fra record (og fortolket).

Som et eksempel på en brugerdefineret print_record er valgt den, der benyttes til testfiler produceret af CENTERNET modulet SMM. I dette tilfælde er proceduren external.

2.2.1 Eksempel på print_record procedure

Den print_record procedure, der anvendes til SMM, producerer en udskrift, som er illustreret i eksempel 1.

```
HHMMSS COROUTINE    OPERATION    SEMAPHORE    WSELECT1 WSELECT2    WTIME           RECNO  REC(1)  REC(2)  REC(3)  REC(4)
                                               OP ADDR OPLENGTH PRIORITY                   REC(5)  REC(6)  REC(7)  REC(8)
                                              COROUTINE
                                              CURSTATE   SYMBOL   ACTION NEWSTATE
                                               H WORDS    STATE          MESSADDR
                                               H WORDS  MAX HWS     MODE
                                                 STATUS
                                                 ACTION
                                               H WORDS   OP CODE PORTADDR MESSADDR
                                               H WORDS    STATUS   RESULT MESSADDR
==============================================================================================================================

DATE: 81 10 19
161225 CENTRALOGIC ACTIV                      START INPUT                               1     128        1       8  424676
                                                                                                0        0  259586-3711351
       START INPUT WAIT        STARTINPUT            0        0        0                2       8        0       0       0
                                                                                                6        1  259586-3711239
       CENTRALOGIC ACTIV                      NET INPUT                                 3     128        2       0       0
                                                                                                6        0  259586-3711239
       NET INPUT   WAIT        INPUTANSWER           0        0        0                4       8        0       0       0
                                                                                                4        2  259586-3711239
       CENTRALOGIC ACTIV                      APPLICAT  1                               5     128        3       0       0
                                                                                                4        0  259586-3711239
       APPLICAT  1 WAIT        WAITMES               0    28532        0                6       8        0   28532       0
                                                                                               -1        5  259586-3710765
       CENTRALOGIC ACTIV                      TERMINAL  1                               7     128        4   28532       0
                                                                                               -1        0  259586-3710765
       TERMINAL  1 WAIT        TERMTASK  1       -1984        0       -1                8       8-8386624       0      -1
                                                                                               11        4  259586-3710461
       TERMINAL  1 WAIT EXIT   TERMTASK  1      420465        0        0                9      16   420465       0       0
                                                                                               11        4  259586-3710461
       TERMINAL  1 MAIN LOOP                         1        0        1        2      10    1025        1       0       1
                                                                                                2        4  259586-3710461
       TERMINAL  1 PERFORM                           1                                 11    1029        1       0       0
                                                                                                0        4  259586-3710221
161226 TERMINAL  1 SENDMESSAGE                -6842731     8192        0    45526      12    1030-6842731    8192       0
                                                                                            45526        4  259586-3706751
       CENTRALOGIC COR TO SEM  IMPLICITPAS TERMINAL  1                                 13      64        4       0       0
                                                                                               -8        0  259586-3706544
       CENTRALOGIC ACTIV                      APPLICAT  2                              14     128        5       0       0
                                                                                               -8        0  259586-3706544
       APPLICAT  2 WAIT        WAITMES               0    28570        0               15       8        0   28570       0
                                                                                               -1        5  259586-3708544
       CENTRALOGIC ACTIV                      TERMINAL  2                              16     128        6   28570       0
                                                                                               -1        0  259586-3706544
       TERMINAL  2 WAIT        TERMTASK  2       -1984        0       -1               17       8-8386624       0      -1
                                                                                               12        6  259586-3708462
```

Eksempel 1: Starten af en udskrivning fra SMM's print_record procedure.

Kommentarer til SMM's print_record procedure:

- I starten af udskriften laves en overskrift, hvoraf de 3 første linier hører til de individer, der dannes af coroutinesystemet, mens de sidste svarer til de brugerafhængige individer.

- Overskriften styres af et antal own variable: last_clock, last_date, tab_1, tab_2 samt width. De samme variable anvendes i de følgende kald til at formattere udskriften.

- Hver gang dato ændrer sig udskrives en linie med den nye dato (cur_date).

- I starten af hver individudskrift er der plads til klokkeslet (cur_clock), som kun udskrives, når det skifter.

- Coroutine identifikationen, individtypen (operation) og semaforidentifikationen udskrives i klar tekst.

- Det aktuelle individnummer udskrives sammen med det ufortolkede individindhold.

- De interessante felter i et individ udskrives i den 'fortolkede' del af udskriften.

- Der anvendes en standard variable terms til oplysning, om det antal terminaler, der blev benyttet ved opstarten af SMM. (Værdien har betydning for fortolkningen af coroutine- og semaforidentifikationerne).

- De to dele af wait_select udskrives hver især, enten som ikke-negative tal (dvs. ufortolket) eller som et minus efterfulgt af de sidste 23 bits opfattet som et positivt heltal. Denne metode skulle lette fortolkningen af wait_select i de tilfælde, hvor wait_select opfattes som en maske.

2.3 Program

Hvilket program, der skal udføres for at få kaldt proceduren test_print, er afhængigt af hvordan den aktuelle testfil skal behandles. I det følgende gives et antal eksempler på programmer til illustrering af nogle af de muligheder, der ligger i test_print proceduren.

Det simpleste program er det, hvor samtlige individer fra start og til logisk filafslutning udskrives:

```
begin
  test_print(<:test:>,0,0,0,0,true,print_rec)
end
```

Såfremt udskrivningsproceduren er en med standard variable som for eksempel den, der anvendes til SMM, skal disse variable initieres inden kaldet af test_print. Hvis der desuden kun ønskes udskrevet fra individ nr. 200 til individ nr. 300 kan det gøres med følgende program:

```
begin
integer rec_no;
  terms:= 3; <* antal terminaler *>
  rec_no:= 199;
  test_print(<:smmtest:>,rec_no,0,0,0,rec_no <= 300,
                                        print_record);
end
```

Ønskes samtlige 'signal'- og 'wait'-individer udskrevet, såfremt de ikke er genereret af centralogic (dvs. coroutine nr. 0), kan det ske med følgende program:

```
begin
integer cor,type;
  test_print(<:test:>,0,cor,type,0,
            cor <> 0 and (type = 4 or type = 5 or type = 8),
                                        print_rec);
end
```

Vides det, at tidsintervallet fra klokken 13 til 14 rummes i de sidste 2000 individer i testfilen, og ønskes blot de brugergenererede individer udskrevet fra dette tidsrum, kan det gøres på følgende måde:

```
begin
integer type,cur_clock;
  cur_clock:= 13 00 00;
  test_print(<:fil:>,-2000,0,type,cur_clock,
              type >= 1024 and cur_clock <= 14 00 00,
                                        print_rec)
end
```

Som det fremgår af ovenstående eksempler, giver Jensens Device rige muligheder for at trimme udskriften af testfiler og samtidig er selekteringsmetoden enkel, idet print parametren blot skal overholde reglerne for logiske udtryk, som kendes fra ALGOL8.

A. REFERENCER

[1] RCSL Nr. 31-D639:
ALGOL Coroutine System, Brugervejledning
Jesper Andreas Tågholt, Oktober 1981
Resumé: Denne manual beskriver et coroutine system til brug under ALGOL8.
(50 trykte sider).

[2] RCSL No 43-GL11697:
CENTERNET, RC8000 Scroll Mode Mapping Module (SMM)
Reference Manual
Lis Clement, November 1981
Abstract: This document describes the functions and the internal structure of the Scroll Mode Mapping module of CENTERNET. The SMM module maps the SC and VTP formats/ protocols into the conventions of the standard terminal interface, known as the way the RC822 terminal is accessed from an internal process in RC8000.
(106 printed pages).

B. PROGRAMTEKSTEN TIL test_print B.

```
EXTERNAL
PROCEDURE TEST_PRINT(DOC,REC_NO,COROUTINE,TYPE,CUR_CLOCK,PRINT,PRINT_RECORD);
STRING                DOC                                                  ;
INTEGER                   REC_NO,COROUTINE,TYPE,CUR_CLOCK                  ;
BOOLEAN                                                  PRINT             ;
PROCEDURE                                                      PRINT_RECORD ;
BEGIN

<*************************************************************************
 *                                                                       *
 * PARAMETERS:                                                           *
 * ----------                                                            *
 *                                                                       *
 * DOC           CALL:   DOCUMENT NAME OF THE BACKING STORAGE FILE       *
 *                       HOLDING THE TEST RECORDS                        *
 *                                                                       *
 * REC_NO *)             RECORD NUMBER                                   *
 *               CALL:   = 0  START AT FIRST RECORD                      *
 *                       > 0  START AFTER 'REC_NO' RECORDS               *
 *                       < 0  START -'REC_NO' RECORDS FROM LOGICAL END-  *
 *                            OF-DOCUMENT                                *
 *               RETURN: LAST READ RECORD NUMBER                         *
 *                                                                       *
 * COROUTINE *)          COROUTINE NUMBER                                *
 *               RETURN: COROUTINE NUMBER IN LAST READ RECORD            *
 *                                                                       *
 * TYPE *)               RECORD TYPE                                     *
 *               RETURN: RECORD TYPE IN LAST READ RECORD                 *
 *                                                                       *
 * CUR_CLOCK *)          CURRENT CLOCK, HHMMSS                           *
 *               CALL:   = 0  START AT 'REC_NO' RECORD                   *
 *                       > 0  START AT THE FIRST RECORD AFTER 'REC_NO'    *
 *                            RECORD, WHERE CLOCK IS GREATER THAN OR     *
 *                            EQUAL TO 'CUR_CLOCK'                       *
 *               RETURN: CLOCK IN LAST READ RECORD                       *
 *                                                                       *
 * PRINT                 PRINT CONDITION                                 *
 *                       = TRUE  IF CURRENT RECORD SHOULD BE PRINTED     *
 *                       = FALSE IF CURRENT RECORD IS SKIPPED            *
 *                                                                       *
 *                       THE CONDITION IS EVALUATED FOR EACH POSSIBLE    *
 *                       RECORD (JENSENS DEVICE), SEE BELOW              *
 *                                                                       *
 * PRINT_RECORD          PROCEDURE WITH FORMAL PARAMETERS AS FOLLOWS     *
 *                       (THEY MAY NOT BE CHANGED BY THE PROCEDURE):     *
 *                       RECORD    INTEGER ARRAY (1:8)                   *
 *                                 THE RECORD TO BE PRINTED              *
 *                       REC_NO    INTEGER                               *
 *                                 CURRENT RECORD NUMBER                 *
 *                       CUR_DATE  INTEGER                               *
 *                                 CURRENT DATE, YYMMDD                  *
 *                       CUR_CLOCK INTEGER                               *
 *                                 CURRENT CLOCK, HHMMSS                 *
 *                                                                       *
 *  *) THE PARAMETERS ARE CHANGED WHENEVER A RECORD IS READ; I.E. THE    *
 *     ACTUAL PARAMETERS CAN BE USED AS OPERANDS IN THE PRINT CONDITION  *
 *     IN ORDER TO DECIDE WHETHER SUCH A RECORD MAY BE PRINTED OR NOT    *
 *     (JENSENS DEVICE).                                                 *
 *                                                                       *
 *                                                                       *
 * FUNCTION:                                                             *
 * --------                                                              *
 *                                                                       *
 * AFTER POSITIONING ACCORDING TO 'REC_NO' THE RECORDS ARE SCANNED       *
 * UNTIL 'CUR_CLOCK' CONDITION IS SATISFIED.                             *
 * SUBSEQUENTLY THE READING OF RECORDS STARTS. THE PRINT CONDITION IS    *
 * EVALUATED FOR EACH RECORD AND - IF IT IS TRUE - THE PRINT_RECORD      *
 * PROCEDURE IS CALLED.                                                  *
 * THE READING STOPS AT LOGICAL END-OF-DOCUMENT.                         *
 *                                                                       *
 * LOGICAL END-OF-DOCUMENT IS MET WHEN AT LEAST ONE OF THE FOLLOWING     *
 * CONDITIONS IS FULFILLED                                               *
 *    - PHYSICAL END-OF-DOCUMENT                                         *
 *    - 'CUR_DATE' < 80 01 01 OR 'CUR_DATE' > TO DAY'S DATE              *
 *    - TIME IN CURRENT RECORD LESS THAN IN PREVIOUS RECORD (OR LESS     *
 *      THAN ZERO)                                                       *
 *                                                                       *
 * ALL EXCEPTIONS BESIDES 'END OF DOCUMENT' ARE TREATED BY STD_ERROR;    *
 * I E. THE PROGRAM STOPS WITH A RUN TIME ALARM.                         *
 *                                                                       *
```

```
*                                                                             *
* EXAMPLES:                                                                   *
* --------                                                                    *
*                                                                             *
* TEST_PRINT(<:TEST:>,0,0,0,TRUE,PRINT_REC);                                  *
* ALL RECORDS UNTIL LOGICAL END-OF-DOCUMENT IN THE T    RECORD FILE           *
* DOCUMENT 'TEST' IS PRINTED BY MEANS OF THE PROCEDUR  PRINT_REC.             *
*                                                                             *
* REC:= 200;                                                                  *
* TEST_PRINT(<:TEST:>,REC,0,0,0,REC <= 300,PRINT_REC);                        *
* IF REC < 301 THEN  WRITE(OUT,<:<10>FILE EXHAUSTED AT RECORD:>,REC);         *
* THE RECORDS FROM NUMBER 201 THROUGH 300 ARE PRINTED; BUT IF THERE IS        *
* NOT AT LEAST 300 RECORDS IN THE (LOGICAL) FILE A WARNING IS WRITTEN         *
* ON CURRENT OUTPUT.                                                          *
*                                                                             *
* REC:= -1000;                                                                *
* TEST_PRINT(<:TESTFILE:>,REC,COR,TYPE,0,                                     *
*            COR = 1 OR (COR MOD 5 = 2 AND                                    *
*                        (TYPE = 4 OR TYPE = 5 OR TYPE = 8)),PRINT);          *
* AMONG THE LATEST 1000 RECORDS IN 'TESTFILE' ALL RECORDS CONCERNING          *
* COROUTINE 1 ARE PRINTED BUT IF THE RECORDS ARE PRODUCED BY THE CO-          *
* ROUTINES 2, 5, 8, ... ONLY SIGNAL AND WAIT RECORDS ARE PRINTED; NO          *
* OTHER RECORDS ARE PRINTED.                                                  *
*                                                                             *
* TEST_PRINT(<:TEST:>,2000,COR,TYPE,13 00 00,COR <> 0 OR COR <> 7 OR          *
*                                          TYPE >= 1024,PRINT_REC);           *
* THE 'TEST' FILE IS STARTED AT THE FIRST RECORD GENERATED AT 1 P.M.          *
* (OR LATER) AFTER RECORD NUMBER 2000 (EXCLUSIVE). ALL USER RECORDS           *
* FROM ALL COROUTINES OR SYSTEM GENERATED RECORDS NOT PRODUCED BY             *
* CENTRALLOGIC OR COROUTINE 7, ARE PRINTED.                                   *
*                                                                             *
*******************************************************************************>
```

```
INTEGER
CUR_DATE,            <* YYMMDD, CORRESPONDING TO LAST_TIME                  *>
P_CUR_CLOCK,         <* VALUE OF PARAMETER CUR_CLOCK                        *>
REL_REC_NO,          <* NUMBER OF RECORDS FROM LOGICAL END-OF-DOCUMENT      *>
RES,                 <* VALUE OF LAST CALL OF NEXT_RECORD                  *>
TO_DAY;              <* TO DAY'S DATE, YYMMDD                               *>

INTEGER FIELD
COR,                 <* FIELD IN TEST RECORD                               *>
OPERATION;           <* FIELD IN TEST RECORD                                *>

LONG
LAST_TIME;           <* LAST READ TIME VALUE FROM TEST RECORD'              *>

LONG FIELD
TIME;                <* FIELD IN TEST RECORD                               *>

INTEGER ARRAY FIELD
IAF;                 <* TEST RECORD AS INTEGER ARRAY                       *>

ZONE
IN(128,1,READ_ERROR);<* ZONE FOR TEST RECORD DOCUMENT                      *>
```

```
PROCEDURE READ_ERROR(IN,STATUS,HALFWORDS);
ZONE                 IN                 ;
INTEGER                 STATUS,HALFWORDS ;
BEGIN
INTEGER ARRAY
ZONE_DESCR(1:20);    <* ZONE DESCRIPTION AREA                              *>
  IF LOGAND(STATUS,1 SHIFT 18 <* END OF DOCUMENT *>) <> 0 THEN
  BEGIN
    GET_ZONE6(IN,ZONE_DESCR);
    ZONE_DESCR(14):= ZONE_DESCR(19);  <* RECORD_BASE:= BASE_BUFFER_AREA *>
    ZONE_DESCR(16):= 16;              <* RECORD_LENGTH:= 16 HW          *>
    SET_ZONE6(IN,ZONE_DESCR);
    IN.IAF.OPERATION:= 3;
    IN.IAF.COR:= -1;
    IN.IAF.TIME:= -1;
    STATUS:= 0;
    HALFWORDS:= 128*4;
  END
  ELSE
  STD_ERROR(IN,STATUS,HALFWORDS);
END;
```

```
INTEGER
PROCEDURE NEXT_RECORD;
BEGIN
REAL R;
  NEXT_RECORD:= 0;
  INREC6(IN,16);
  REC_NO:= REC_NO + 1;
  TYPE:= IN.IAF.OPERATION;
  IF TYPE > 2 THEN
  BEGIN
    COROUTINE:= ABS IN.IAF.COR;
    IF IN.IAF.TIME < LAST_TIME THEN
    NEXT_RECORD:= 1;
    LAST_TIME:= IN.IAF.TIME;
    CUR_DATE:= SYSTIME(4,LAST_TIME/10000,R);
    CUR_CLOCK:= R;
    IF CUR_DATE < 80 01 01 OR
       CUR_DATE > TO_DAY   THEN
    NEXT_RECORD:= 2;
  END;
END;



  BEGIN
  REAL R,TIME;
    SYSTIME(1,0,TIME);
    TO_DAY:= SYSTIME(4,TIME,R);
  END;

  IAF:=            0;
  OPERATION:=      2;
  COR:=           12;
  TIME:=          16;



  OPEN(IN,4,DOC,1 SHIFT 18);
  P_CUR_CLOCK:= CUR_CLOCK;
  LAST_TIME:= 0;

  IF REC_NO < 0 THEN
  BEGIN
    REL_REC_NO:= -REC_NO;
    REC_NO:= 0;
    REPEAT UNTIL NEXT_RECORD > 0;
    IF REC_NO > REL_REC_NO THEN
    REC_NO:= REC_NO - 1 - REL_REC_NO
    ELSE
    BEGIN
      WRITE(OUT,<:<10>RELATIVE RECORD NUMBER GREATER THAN MAX RECORD NUMBER, I.E.:>,
      REL_REC_NO,<: >=:>,REC_NO,<:<10>:>);
      REC_NO:= 0;
    END;
  END;

  SETPOSITION(IN,0,REC_NO//32 << I.E. RECORDS PER SEGMENT *>);
  INREC6(IN,(REC_NO MOD 32)*16);

  COROUTINE:=
  CUR_DATE:=
  CUR_CLOCK:= 0;
  LAST_TIME:= 0;

  REPEAT
    RES:= NEXT_RECORD;
  UNTIL CUR_CLOCK >= P_CUR_CLOCK OR RES > 0;

  IF RES = 0 THEN
  BEGIN
    REPEAT
      IF PRINT THEN
      PRINT_RECORD(IN.IAF,REC_NO,CUR_DATE,CUR_CLOCK);
    UNTIL NEXT_RECORD > 0;
  END
  ELSE
  WRITE(OUT,<:<10>FILE EXHAUSTED; RECORD NUMBER:>,REC_NO,<:<10>:>);

  WRITE(OUT,FALSE,768);

  CLOSE(IN,TRUE);

END;  <*TEST PRINT*>
END
```

C. PROGRAMTEKSTEN TIL print_record C.

```
EXTERNAL
PROCEDURE PRINT_RECORD(RECORD,REC_NO,CUR_DATE,CUR_CLOCK);
VALUE                          REC_NO,CUR_DATE,CUR_CLOCK ;
INTEGER ARRAY          RECORD                            ;
INTEGER                        REC_NO,CUR_DATE,CUR_CLOCK ;
BEGIN

OWN
INTEGER
TERMS;                 <* STANDARD VARIABLE HOLDING ACTUAL NUMBER OF       *>
                       <* TERMINALS                                        *>
                       <* NB THIS DECLARATION MUST BE THE FIRST OWN DE-     *>
                       <* CLARATION IN THIS PROCEDURE                       *>
                       <* IN THE CATALOG AN ENTRY MUST BE SET AS            *>
                       <*   TERMS=SET BS PRINTRECORD 1 576.U 0 4.U          *>
                       <**************************************************>


OWN
INTEGER
LAST_CLOCK,            <* LAST PRINTED CLOCK, HHMMSS                       *>
LAST_DATE,             <* LAST PRINTED DATE,  YYMMDD                       *>
TAB_1,                 <* FIRST TABULATOR, CHARACTER POSITION              *>
TAB_2,                 <* SECOND TABULATOR, CHARACTER POSITION             *>
WIDTH;                 <* PAGE WIDTH (CHARACTERS)                          *>

INTEGER FIELD
ACTION,                <* USER FIELD IN TEST RECORD                        *>
ADDR,                  <* FIELD IN TEST RECORD                             *>
COR,                   <* FIELD IN TEST RECORD                             *>
COR_NO,                <* FIELD IN TEST RECORD                             *>
CUR_STATE,             <* USER FIELD IN TEST RECORD                        *>
HALFWORDS,             <* USER FIELD IN TEST RECORD                        *>
LENGTH,                <* FIELD IN TEST RECORD                             *>
MAX_HALFWORDS,         <* USER FIELD IN TEST RECORD                        *>
MESS_ADDR,             <* USER FIELD IN TEST RECORD                        *>
MODE,                  <* USER FIELD IN TEST RECORD                        *>
NEW_ACTION,            <* USER FIELD IN TEST RECORD                        *>
NEW_STATE,             <* USER FIELD IN TEST RECORD                        *>
OP_CODE,               <* USER FIELD IN TEST RECORD                        *>
OPERATION,             <* FIELD IN TEST RECORD                             *>
PORT_ADDR,             <* USER FIELD IN TEST RECORD                        *>
PRIORITY,              <* FIELD IN TEST RECORD                             *>
RESULT,                <* USER FIELD IN TEST RECORD                        *>
SELECT_1,              <* FIELD IN TEST RECORD                             *>
SELECT_2,              <* FIELD IN TEST RECORD                             *>
SEM,                   <* FIELD IN TEST RECORD                             *>
STATE_AFTER,           <* USER FIELD IN TEST RECORD                        *>
STATE_BEFORE,          <* USER FIELD IN TEST RECORD                        *>
STAT,                  <* USER FIELD IN TEST RECORD                        *>
STATUS,                <* USER FIELD IN TEST RECORD                        *>
SYMBOL,                <* USER FIELD IN TEST RECORD                        *>
W_TIME;                <* FIELD IN TEST RECORD                             *>

LONG FIELD
TIME;                  <* FIELD IN TEST RECORD                             *>



INTEGER
PROCEDURE WRITE_COR(COR_NO);
VALUE                  COR_NO ;
INTEGER                COR_NO ;
BEGIN

INTEGER COR_INDEX;

  COR_INDEX:= (IF COR_NO < 0             THEN               - 1  ELSE
               IF COR_NO < 3             THEN             COR_NO  ELSE
               IF COR_NO < 3 + TERMS*2   THEN (COR_NO MOD 2 + 3) ELSE
               IF COR_NO = 3 + TERMS*2   THEN                 5  ELSE
               IF COR_NO = 4 + TERMS*2   THEN                 6  ELSE
                                                             -2       ) + 3;
  COR_INDEX:=
  WRITE(OUT,<: :>,CASE COR_INDEX OF (<:????????????:>,
                                     <:           :>,
                                     <:CENTRALOGIC:>,
                                     <:START INPUT:>,
                                     <:NET INPUT  :>,
                                     <:TERMINAL :>,
                                     <:APPLICAT :>,
                                     <:NPM MESSAGE:>,
                                     <:NPM ANSWER :>));
  IF COR_NO >= 3 AND COR_NO < 3 + TERMS*2 THEN
  COR_INDEX:= COR_INDEX + WRITE(OUT,<<0D>,((COR_NO - 1)//2);

  WRITE_COR:= COR_INDEX;
END;
```

```
OPERATION:=      2;

ADDP:=
SELECT_1:=
COR_NO:=
CUR_STATE:=
HALFWORDS:=
STATUS:=
ACTION:=         4;

LENGTH:=
SELECT_2:=
SYMBOL:=
STATE_BEFORE:=
MAX_HALFWORDS:=
OP_CODE:=
STAT:=           6;

W_TIME:=
PRIORITY:=
NEW_ACTION:=
STATE_AFTER:=
MODE:=
PORT_ADDR:=
RESULT:=         8;

SEM:=
NEW_STATE:=
MESS_ADDR:=     10;

COR:=           12;

TIME:=          16;
```

```
IF LAST_DATE = 0 AND LAST_CLOCK = 0 THEN
BEGIN
  LAST_DATE:= -1;
  TAB_1:= - 1 + WRITE(OUT,<:<10>HHMMSS COROUTINE   OPERATION   SEMAPHORE   :>);
  TAB_2:= TAB_1 + WRITE(OUT,<:WSELECT1 WSELECT2    WTIME          :>);
  WIDTH:= TAB_2 + WRITE(OUT,<:RECNO  REC(1)  REC(2)  REC(3)  REC(4):>);
  WRITE(OUT,"SP",TAB_2 + 6 - WRITE(OUT,"NL",1,"SP",TAB_1,<: OP ADDR OPLENGTH PRIORITY:>),
          <:  REC(5)  REC(6)  REC(7)  REC(8):>);
  WRITE(OUT,"NL",1,"SP",TAB_1,<:COROUTINE:>);
  WRITE(OUT,"NL",1,"SP",TAB_1,<:CURSTATE   SYMBOL   ACTION NEWSTATE:>);
  WRITE(OUT,"NL",1,"SP",TAB_1,<: H WORDS    STATE          MESSADDR:>);
  WRITE(OUT,"NL",1,"SP",TAB_1,<: H WORDS  MAX HWS     MODE:>);
  WRITE(OUT,"NL",1,"SP",TAB_1,<:  STATUS:>);
  WRITE(OUT,"NL",1,"SP",TAB_1,<:  ACTION:>);
  WRITE(OUT,"NL",1,"SP",TAB_1,<: H WORDS  OP CODE PORTADDR MESSADDR:>);
  WRITE(OUT,"NL",1,"SP",TAB_1,<: H WORDS   STATUS   RESULT MESSADDR:>);
  WRITE(OUT,"NL",1,"=",WIDTH,"NL",1);
END;
IF RECORD_OPERATION > 2 THEN
BEGIN
  IF CUR_DATE <> LAST_DATE THEN
  WRITE(OUT,<:<10>DATE::>,<< DD DD DD >,CUR_DATE);
  IF CUR_CLOCK <> LAST_CLOCK THEN
  WRITE(OUT,"NL",1,<<DDDDDD>,CUR_CLOCK)
  ELSE
  WRITE(OUT,"NL",1,<:       :>);
  LAST_DATE:= CUR_DATE;
  LAST_CLOCK:= CUR_CLOCK;
END
ELSE
WRITE(OUT,"NL",1,<:       :>);
```

```
WRITE_COR(IF RECORD.OPERATION <= 2 THEN -1 ELSE ABS RECORD.COR);

WRITE(OUT,<: :>,IF RECORD.OPERATION =    1 THEN <:SIGNAL DATA:> ELSE
                IF RECORD.OPERATION =    2 THEN <:WAIT DATA  :> ELSE
                IF RECORD.OPERATION =    4 OR
                   RECORD.OPERATION =    5 THEN <:SIGNAL     :> ELSE
                IF RECORD.OPERATION =    8 THEN <:WAIT       :> ELSE
                IF RECORD.OPERATION =   16 OR
                   RECORD.OPERATION =   18 THEN <:WAIT EXIT  :> ELSE
                IF RECORD.OPERATION =   64 THEN <:COR TO SEM :> ELSE
                IF RECORD.OPERATION =  128 THEN <:ACTIV      :> ELSE
                IF RECORD.OPERATION = 1025 THEN <:MAIN LOOP  :> ELSE
                IF RECORD.OPERATION = 1026 THEN <:ANSWERINPUT:> ELSE
                IF RECORD.OPERATION = 1027 THEN <:SENDOUTPUT :> ELSE
                IF RECORD.OPERATION = 1028 THEN <:OUTPUTSTAT :> ELSE
                IF RECORD.OPERATION = 1029 THEN <:PERFORM    :> ELSE
                IF RECORD.OPERATION = 1030 THEN <:SENDMESSAGE:> ELSE
                IF RECORD.OPERATION = 1031 THEN <:SENDANSWER :> ELSE
                                                <:???????????:>     );

WRITE(OUT,<: :>);
IF RECORD.OPERATION < 128 AND RECORD.OPERATION > 2 THEN
BEGIN
  WRITE(OUT,IF RECORD.SEM < -9 THEN       <:???????????:> ELSE
            IF RECORD.SEM < 11 THEN
               (CASE RECORD.SEM + 10 OF (<:READY       :>,
                                         <:IMPLICITPAS:>,
                                         <:SEM -7      :>,
                                         <:FREE        :>,
                                         <:VIRTUEL FKR:>,
                                         <:SEM -4      :>,
                                         <:DELAY       :>,
                                         <:WAITANSPOOL:>,
                                         <:WAITMES     :>,
                                         <:WAITMESPOOL:>,
                                         <:LARGEPOOL   :>,
                                         <:MESSAGEPOOL:>,
                                         <:SMALLPOOL   :>,
                                         <:INPUTANSWER:>,
                                         <:ATTANSWER   :>,
                                         <:STARTINPUT  :>,
                                         <:OPENPOOL    :>,
                                         <:EVENTPOOL   :>,
                                         <:TELEGRAPOOL:>,
                                         <:LETTERPOOL  :>))ELSE
           IF RECORD.SEM < 11 + TERMS THEN <:TERMTASK :> ELSE
                                         <:???????????:>     );
  IF RECORD.SEM >= 11 AND
     RECORD.SEM < 11 + TERMS THEN
  WRITE(OUT,<<DD>,RECORD.SEM - 10);
END
ELSE
WRITE(OUT,<:            :>);

  WRITE(OUT,"SP",TAB_2 - TAB_1 - (
  IF RECORD.OPERATION = 8 THEN
  WRITE(OUT,<< -DDDDDDD>,IF RECORD.SELECT_1 < 0 THEN -(RECORD.SELECT_1 EXTRACT 23)
                                                ELSE   RECORD.SELECT_1,
                         IF RECORD.SELECT_2 < 0 THEN -(RECORD.SELECT_2 EXTRACT 23)
                                                ELSE   RECORD.SELECT_2,
            RECORD.W_TIME)
  ELSE
  IF RECORD.OPERATION =  4 OR RECORD.OPERATION =  5 OR
     RECORD.OPERATION = 16 OR RECORD.OPERATION = 18 THEN
  WRITE(OUT,<< -DDDDDDD>,RECORD.ADDR,RECORD.LENGTH,RECORD.PRIORITY)
  ELSE
  IF RECORD.OPERATION = 64 OR RECORD.OPERATION = 128 THEN
  WRITE_COR(ABS RECORD.COR_NO)
  ELSE
  IF RECORD.OPERATION = 1025 THEN
  WRITE(OUT,<< -DDDDDDD>,RECORD.CUR_STATE,RECORD.SYMBOL,RECORD.NEW_ACTION,RECORD.NEW_STATE)
  ELSE
  IF RECORD.OPERATION = 1026 THEN
  WRITE(OUT,<< -DDDDDDD>,RECORD.HALFWORDS,RECORD.STATE_BEFORE,"SP",9,RECORD.MESS_ADDR)
  ELSE
  IF RECORD.OPERATION = 1027 THEN
  WRITE(OUT,<< -DDDDDDD>,RECORD.HALFWORDS,RECORD.MAX_HALFWORDS,RECORD.MODE)
  ELSE
  IF RECORD.OPERATION = 1028 THEN
  WRITE(OUT,<< -DDDDDDD>,RECORD.STATUS)
  ELSE
  IF RECORD.OPERATION = 1029 THEN
  WRITE(OUT,<< -DDDDDDD>,RECORD.ACTION)
  ELSE
  IF RECORD.OPERATION = 1030 THEN
  WRITE(OUT,<< -DDDDDDD>,RECORD.HALFWORDS,RECORD.OP_CODE,RECORD.PORT_ADDR,RECORD.MESS_ADDR)
  ELSE
  IF RECORD.OPERATION = 1031 THEN
  WRITE(OUT,<< -DDDDDDD>,RECORD.HALFWORDS,RECORD.STAT,RECORD.RESULT,RECORD.MESS_ADDR)
  ELSE
  0),

  << DDDDD>,REC_NO,<<-DDDDDDD>,RECORD(1),RECORD(2),RECORD(3),RECORD(4),
  "NL",1,"SP",TAB_2 + 5,RECORD(5),RECORD(6),RECORD(7),RECORD(8));

END; <*PRINT RECORD*>
END
```

LÆSERBEMÆRKNINGER

Titel: ALGOL Coroutine System
Testudskrivning
Brugervejledning

RCSL Nr.: 31-D666

A/S Regnecentralen af 1979 bestræber sig på at forbedre kvalitet og brugbarhed af sine publikationer. For at opnå dette ønskes læserens kritiske vurdering af denne publikation.

Kommenter venligst manualens fuldstændighed, nøjagtighed, disposition, anvendelighed og læsbarhed:

_____

_____

_____

_____

Angiv fundne fejl (reference til sidenummer):

_____

_____

_____

_____

Hvordan kan manualen forbedres:

_____

_____

_____

_____

Andre kommentarer:

_____

_____

_____

_____

_____

Navn: _____ Stilling: _____

Firma: _____

Adresse: _____

Dato: _____

På forhånd tak!

42-i 1289

..... Fold her .....

..... Riv ikke - Fold her og hæft .....

Frankeres
som
brev

REGNECENTRALEN
af 1979
Informationsafdelingen
Lautrupbjerg 1
2750 Ballerup