EFFECTS OF H.V.D.C. TRANSMISSION LINE ON THE TRANSIENT PERFORMANCE OF AN A.C. POWER SYSTEM

VOLUME 2

By

SYED MUHAMMAD AHMED

This volume forms part of the thesis with above title submitted for the degree of

DOCTOR OF PHILOSOPHY IN

THE FACULTY OF APPLIED SCIENCE

UNIVERSITY OF NEWCASTLE UPON TYNE

NOVEMBER 1969

CONTENTS

APPENDIX 1

DIGITAL PROGRAMME TO COMPUTE CONVERTER CHARACTERISTICS

Identifiers Used:

For all the identifiers representing complex quantities the d - x axis components are indicated, if not defined elsewhere, by the last suffices d and q respectively.

ec - Voltage behind the commutating reactance, p.u.

ec1, ec2, ec3 - Different values of "ec".

es - Effective voltage behind the supply system, p.u.

es1, es2, es3 - Different values of "es".

vy - Fundamental component of the voltage at star point of the equivalent circuit, p.u.

vl - Fundamental component of the voltage at the line side of the converter transformer, p.u.

vr - Voltage at which "vl" is controlled, p.u.

iv - Alternating current in the valve side of the transformer, p.u.

ip, iq - Direct and quadrature axis components of "iv".

il - Alternating current in the line side of the transformer, p.u.

is - Current supplied by the generator, p.u.

xv, xl, xt - Reactances in the valve, line and tertiary branches of the star equivalent circuit for the transformer, p.u.

xs - Reactance of the supply system, p.u.

xl - Leakage reactance of the converter transformer, p.u.

xc - Commutating reactance, p.u.

xa - Reactance associated with the mean sub-transient reactance of the synchronous compensator, p.u.

qt - Total reactive power of the synchronous compensator, p.u.

qs - Reactive power delivered by the supply system, p.u.

qy - Total reactive power delivered by the synchronous compensator at star point, p.u.

qr - Total reactive power consumed by the d.c. system, p.u.

vd, id, pd - Voltage, current and power of the d.c. system, p.u.

Z - Cosine of the delay angle of the converter.

u - Angle between the bus-bar voltage and the fundamental phase current (displacement angle), radians.

VdB, IdB, PdB - Base voltage, current and power of the d.c. system respectively, actual units.

VaB - Base voltage of the a.c. system, actual units.

mvarf - Reactive power output of the fitter bank at system frequency and at the nominal voltage of the valve side of the transformer, actual units.

S - Capacitive reactance of the filter bank, p.u.

nn - Turn ratio of the converter transformer.

n - Turn ratio of the ideal transformer.

Id, VD, Es, PD, Qt, Qs, Qr, U, Ec - The values of "id", "vd", "es", "pd", "qt", "qs", "qr", "u" and "ec" respectively, actual units.

Procedures Used:

1. Procedure SYSTMVOLT (id, ec, vd, vr):

The input parameters of this procedures are id, ec, vd and vr and the generator voltage "es" is calculated when the procedure is called. For example, in the main programme the initial value of "es" is calculated as:

vd, vr - are read from the data

id - is calculated from the values of pd and v́d given in the data

ec - is calculated from the given values of "Z" and "xc".

The procedure then calculates the d-q axis components of the current on the a.c. side of the rectifier, and proceeds to compute the fundamental components of voltage and current at the star point of the equivalent circuit of the converter transformer by taking into consideration the reactive power "qy" supplied by the synchronous compensator. Finally, by solving the a.c. circuit on the L.H.S. of the star point the required value of "es" is calculated, If synchronous compensators are not used in the H.V.D.C. system, the calculations of "qy" can be skipped over by giving the marker "rs" a value 1,000.

2. Procedure c:

This procedure gives the instruction of "carriage return" to the computer.

3. Procedure underline (m, n):

This procedure is to underline any text during the print out of the results. The input parameters "m" and "n" of the procedure indicate the starting and finishing point of the line.

Actions of the Programme:

The programme reads and stores the data, as given in Table 4-1, Vol. 1, and then calculates the commutating reactance using either of the equations as instructed by the marker "w". Subsequently, the generator voltage "es", for the desired power and normal voltage of the converter, is calculated by calling the procedure "SYSTMVOLT" and is stored for comparing other computed voltages.

To draw the characteristics of the converter, "id" is given a series of different values. Then for each value of id, a value of "ec" (ec1 = 1.0) is used for the first calculation, and the corresponding value of "es" (= es1) is computed by calling the procedure SYSTMVOLT. A second value of "ec" (ec2 = ec1 - 0.2) is considered to calculate its corresponding value of "es" (= es2) as above. The third value of "ec" (= ec3) is found from the above results, and similarly "es" (= es3) is computed. This process is continued, using the latest two sets of values of "ec" and "es" to predict the next iterate for "ec", until the calculated value of "es" is equal to its initial value within the prefixed tolerance of 10^{-6}. At this stage the latest p.u. values of the system variables are printed in the prescribed format.

In case the actual values of the computed variables are also required, the marker "EF" is given a value 1 and consequently the calculations for conversion comes in the main loop and the results in actual units are printed on a separate sheet under the proper headings. After considering all the values of "id", the programme finishes.

VERBATIM

```
DHEG18 CASE 1(CHARACTERISTICS**OF**CONVERTERS)→
begin library A0,A6;
      open(20); open(30);
      writetext(30,[[20s]AHMAD*DHEG18*HVDC[cc]]);

      begin real ec,es,vd,id,ip,iq,xv,x1,xt,xs,xa,xl,xc,z,
                 pi,qt,qs,qy,S,n,VdB,IdB,pd,mvarf,w,R,qr,
                 nn,PdB,es1,es2,vr,es3,VaB,ec1,ec2,ec3,
                 u,i1d,i1q,v1d,v1q,x;

           array Id,VD,Es,PD,Qt,Qy,Qs,Qr,U,EC[1:100];
           integer f,i,EF,FE,re,rs,rl;
           L2:
           rl:=read(20);
           re:= read(20);
           rs:= read(20);
           EF:=read(20);
           FE:=read(20);
           vd:=read(20);
           pd:=read(20);
           z:=read(20);
           x1:=read(20);
           xa:=read(20);
           xv:=read(20);
           xt:=read(20);
           xl:=read(20);
           n:=read(20);
           PdB:=read(20);
           mvarf:=read(20);
           VaB:=read(20);
           VdB:=read(20);
           xs:=read(20);
           w :=read(20);
           vr:=read(20);
           R:=read(20);

           pi:=3.142;
           IdB:=PdB/VdB;
           nn:=VdBxpi/(VaBxzx3xsqrt(2));
           S:=mvarf/PdB;
           i:=1;

           if w=3 then
           xc:=xv+(xa+xt)xx1/(xa+xt+x1)
           else
           f:=format([-nddd.dddd]);
```

```
begin
      real procedure SYSTMVOLT (id,ec,vd,vr);
      real id,ec,vd,vr;
      begin
            real ecd,ecq,is,es,esd,esq,vy,vyd,
                 vyq,iv,ivd,ivq,isd,isq,x2,x3,
                 Ad,Aq,Bd,Bq,Cd,Cq,Dd,Dq,a,b,
                 c,d,e,m,g,h,k;
            pd:=vdxid;
            iv:=id;
            u:=vd/ec;
            if u≥1 then
            begin
                  u:=1;
                  ip:=id;
                  iq:=0;
                  goto L3;
            end
            else
            if u≤-1 then
            begin
                  u:=-1;
                  ip:=-id;
                  iq:=0;
                  goto L3;
            end;
            ip:=uxid;
            iq:=sqrt(ivt2-ipt2);
            ip:= if re=-1 then -ip else  ip;
            L3:
            ecd:=ec;
            ecq:=0;
            qr:=ecdxiq;
            ivd:=ip;
            ivq:=-iq;
            if rs=1000then
            begin
                  v1d:=ecdxn;
                  v1q:=ecqxn;
                  i1d:=ivd/n;
                  i1q:=ivq/n;
                  qt:=0;
                  qy:=0;
                  goto L1;
            end;
            comment calculations for qt and
                    qy start;
            vyd:=ecd+(xc-xv)xivq;
            vyq:=-(xc-xv)xivd;
            vy :=sqrt(vydt2+vyqt2);
            Ad:=ivd+vyq/x1;
            Aq:=-(ivq-vyd/x1);
            Bd:=vyd+xt/x1xvyd-xtxivq;
            Bq:=vyq+xtxivd+xt/x1xvyq;
```

```
Cq:=-1/(n×x1);
Cd:=0;
Dd:=-xt/(n×x1);
Dq:=0;
a:=vr×(Bd×Cq+Bq×Cd-Ad×Dq-Aq×Dd);
b:=vr×(Bd×Cd-Bq×Cq+Ad×Dd-Aq×Dq);
c:=vr↑2×(Cd×Dd-Cq×Dq+Ad×Bd-Aq×Bq);
x2:=(-a×c+b×sqrt(a↑2-c↑2+b↑2))/
    (b↑2+a↑2);
x3:=(-a×c-b×sqrt(a↑2-c↑2+b↑2))/
    (b↑2+a↑2);
x:=if abs(x2)>abs(x3)then x3 else x2;
d:=Ad×Bq+Aq×Bd;
e:=Cq×Dd+Dq×Cd;
m:=Bq×Cd+Cq×Bd;
g:=Aq×Dd+Dq×Ad;
h:=Ad×Dd-Aq×Dq;
k:=Bd×Cd-Bq×Cq;
qt:=d+vr↑2×e+vr×sqrt(1-x↑2)×(m+g)
    +vr××x(h-k);
qy:=vy↑2/(2××t)×(sqrt(1+4××t×qt/
    (vy↑2))-1);
comment end of calculations for qt
          and qy;
i1d:=(ivd-qy×vyq/(vy↑2))/n;
i1q:=(ivq+qy×vyd/(vy↑2))/n;
v1d:=vr×sqrt(1-x↑2);
v1q:=vr××;
L1:
isd:=i1d-S×v1q;
isq:=i1q+S×v1d;
is:=sqrt(isd↑2+isq↑2);
esd:=v1d-xs×isq;
esq:=v1q+xs×isd;
es:=sqrt(esd↑2+esq↑2);
qs:=isq×esd-isd×esq;

    SYSTMVOLT :=es;
end SYSTMVOLT;

procedure underlin(m,n);value m,n;integerm,
          n;
begin integer i;for i:=1step1until m+n-1 do
     charout(30,if i < m then 64 else
    30)end;

procedure c;writetext(30,[[c]]);


id:=pd/vd;
comment calculations of initial value of es;
ec:=(vd+pi/6×id××c)/z;
es:=SYSTMVOLT(id,ec,vd,vr);
```

```
copytext(20,30,[;;]);c;underlin(0,121);c;
writetext(30,[[6s]id[8s]es[8s]vd[8s]pd
          [8s]ip[8s]iq[8s]qt[8s]qy[8s]
          s[8s]qr[8s]u[8s]ec[cc]]);
underlin(0,121);c;
for id:=.1 step .1 until 1.60001 do
begin
      ec1:=1.0;
      comment iteration procedure to
      find ec;
      vd:=ec1xz-pi/6xidxxc;
      es1:=SYSTMVOLT(id,ec1,vd,vr);
      if abs(es-es1)>10↑(-6) then
      begin
            ec2:=ec1-.2;
            vd:=ec2xz-pi/6xidxxc;
            es2:=SYSTMVOLT(id,ec2,vd,vr);
            if abs(es-es2)>10↑(-6) then
            begin
                  L:ec3:=(es-es1)/(es2-es1)
                  x(ec2-ec1)+ec1;
                  vd:=ec3xz-pi/6xidxxc;
                  es3:=SYSTMVOLT(id,ec3,vd,
                       vr);
                  if abs(es-es3)>10↑(-6)
                  then
                  begin
                        es1:=es2;
                        es2:=es3;
                        ec1:=ec2;
                        ec2:=ec3;
                        goto L;
                  end
                  else ec:=ec3;
            end
            else  ec:=ec2;
      end
      else ec:=ec1;
      comment iteration procedure ends;
      if rI=1 then vd:=vd-RXid;
      if EF=1 then
      begin
            comment conversion of p.u.
            values into actual units;
            Id[i]:=idxIdB;
            VD[i]:=vdxVdB;
            PD[i]:=pdxPdB;
            Es[i]:=esxVaB;
            Qt[i]:=qtxPdB;
            Qy[i]:=qyxPdB;
            Qs[i]:=qsxPdB;
            Qr[i]:=qrxPdB;
            U[i]:=u;
            EC[i]:=nnxecxVaB;
            i:=i+1;
      end;
```

```
                write(30,f,id);
                write(30,f,es);
                write(30,f,vd);
                write(30,f,pd);
                write(30,f,ip);
                write(30,f,iq);
                write(30,f,qt);
                write(30,f,qy);
                write(30,f,qs);
                write(30,f,qr);
                write(30,f,u);
                write(30,f,ec);
                writetext(30,[[cc]]);
        end;
        underlin(0,121);c;
        if EF=1 then
        begin
                writetext(30,[[p]]);
                copytext(20,30,[;;]);c;
                underlin(0,118);c;
                writetext(30,[[6s]Id[8s]
                Es[8s]VD[8s]PD[8s]Qt[8s]
                Qy[8s]Qs[8s]Qr[8s]U[8s]
                EC[cc]]);
                underlin(0,118);c;
                for i:=1 step 1 until 16 do
                begin
                        write(30,f,Id[i]);
                        write(30,f,Es[i]);
                        write(30,f,VD[i]);
                        write(30,f,PD[i]);
                        write(30,f,Qt[i]);
                        write(30,f,Qy[i]);
                        write(30,f,Qs[i]);
                        write(30,f,Qr[i]);
                        write(30,f,U[i]);
                        write(30,f,EC[i]);
                        writetext(30,[[cc]]);
                end;
                underlin(0,118);c;
        end;
end;
If FE=0 then
begin
        writetext(30,[[p]]);
        goto L2;
end;
close(20);
close(30);
end→
```

APPENDIX 2

DIGITAL PROGRAMME TO SIMULATE D.C. LINE IN LOAD FLOW STUDY OF A.C. SYSTEM

Identifiers Used:

The identifiers of this programme, which have not been defined earlier, are given below:

xn - Commutating reactance, when synchronous compensators are connected with the tertiary windings of the transformers, p.u.

vdl, vd2 - Sending and receiving ends average voltages of the d.c. system, p.u.

vdn - Average d.c. voltage when the converter is on N.V. control, p.u.

idcl, idc2 - Direct currents of the rectifier and invertor, p.u.

Zl, Z2 - Cosine of the rectifier and invertor delay angles, respectively.

KR, KI - Gains of the rectifier and invertor C.C. regulators, respectively.

Ir, Irl - Reference currents of rectifier and invertor C.C. regulators, p.u.

k - Current margin, p.u.

ul, u2 - Displacement factors of sending and receiving ends, respectively.

vl, v2, v3 - Values of d.c. voltage during iteration, p.u.

il, i2, i3 - Values of direct current during iteration, p.u.

qs, qsl, qs2 - Reactive power consumed by the converter, rectifier and invertor respectively p.u.

ibd, ibq - Alternating current components on the a.c. side of the rectifier, p.u.

idd, idq - Alternating current components on the a.c. side of the invertor, p.u.

v10, v20 - Sending and receiving ends a.c. bus-voltages, p.u.

n1, n2 - Turn ratios of the sending and receiving end on-load tap-changing transformer.

ninc - Increment of the turn ratio.

nmax, nmin - Maximum and minimum limits of ni (i = 1,2).

zmax, zmin - Maximum and minimum limits of the power factor.

Procedures Used:

The programme uses all the procedures which have previously been discussed. The remaining procedure SWAP (a, b); assigns to "a" the value of "b" and to "b" the initial value of "a".

Action of the Programme

The programme reads and stores the data, as given in Table 4-2, Vol. 1, and then prints the headings of the required results. For a positive value of "k", which is the difference between the reference currents of converters A and B, the converter "A" works as a rectifier and B as an invertor, whilst for a negative value of "k" the operation is reversed. In the latter case, the programme calls the procedure SWAP replacing in turn Z1, Ir, Kr and n1 - the values associated with the rectifier - to the corresponding values Z2, Ir1, KI and n2 of the invertor.

The value of "idc" is found by solving iteratively the characteristic equations of the invertor on C.E.A. control and of the rectifier on C.C. control, while the values of the required variables are stored. The programme then proceeds to compute the characteristics of the rectifier and invertor when they are on N.V. and C.C. controls respectively, and stores them. At this stage the comparison is made between the characteristics of the rectifier and invertor on N.V. and C.E.A. controls respectively. If the former is greater than the latter, the rectifier and invertor operate on C.C. and C.E.A. controls respectively, otherwise the rectifier will be on N.V. control with the invertor on C.C. control. Finally, the corresponding stored values of the variables will be printed.

If marker F = 1, the transformer on-load tap changer is considered by the programme, which then compares the computed power factor with its maximum and minimum values and changes the turn ratio within its prefixed limits, if required.

The a.c. bus-bar voltage on either side is varied and the above process is repeated to find the relationship between the a.c. bus voltage and the a.c. components of the converter current.

VERBATIM

```
DHEG18 CASE 2(TO**SIMULATE**H.V.D.C.** LINK** IN**
                LOAD**FLOW**STUDY)→
begin library A0,A6;
      open(20); open(30);
      writetext(30,[[20s]AHMAD*DHEG18*HVDC[cc]]);
      begin real idc,xv,x1,xt,xs,xa,xl,xn,r,J1,J2,u,
            u1,u2,qs,qs1,qs2,isd,isq,ibq,ibd,idq,idd,xc,pi,
            S,n,pd,mvarf,PdB,es1,es2,es3,KI,KR,vr,Ir,k,ec1,
            ec2,ec3,vd,z1,z2,v1,v2,v3,V10,V20,R1,R2,i1,i2,
            i3,vdn,aa,ab,ac,ad,ba,bb,bc,bd,ca,cb,cc,cd,ce,
            cf,vd2,idc1,idc2,t1,t2,t3,tt,vx,Ir1,n1,n2,ninc,
            nmax,nmin,zmax,zmin;
            integer F,F1,f,rs,re;
            x1:=read(20);
            xa:=read(20);
            xv:=read(20);
            xt:=read(20);
            xl:=read(20);
            n1:=read(20);
            n2:=read(20);
            F :=read(20);
            ninc:=read(20);
            nmax:=read(20);
            nmin:=read(20);
            PdB:=read(20);
            mvarf:=read(20);
            xs:=read(20);
            tt:=read(20);
            z1:=read(20);
            z2:=read(20);
            k:=read(20);
            r:=read(20);
            KR:=read(20);
            KI:=read(20);
            Ir:=read(20);
            R1:=read(20);
            R2:=read(20);
            j1:=read(20);
            j2:=read(20);
            zmax:=read(20);
            zmin:=read(20);
            t1:=read(20);
            t2:=read(20);
            t3:=read(20);
            F1:=read(20);
            Ir1:=Ir-k;
            V10:=t1;
            V20:=tt;
            if F1=1 then t1:=tt;
            pI:=3.142;
            S:=mvarf/PdB;
            xn:=xv+(xa+xt)xx1/(xa+xt+x1);
            f:=format([-nddd.dddd]);
```

```
begin
      procedure SWAP(a,b);
      real a,b;
      begin
            real X;
            X:=a;
            a:=b;
            b:=X;
end SWAP;
real procedure SYSTMVOLT(id,ec,vd,vr);
real id,ec,vd,vr;
begin
      real ecd,ecq,is,es,esd,esq,vy,vyd,
      vyq,iv,ivd,ivq,x2,x3,Ad,Aq,Bd,Bq,
      Cd,Cq,Dd,Dq,a,b,c,d,e,m,g,h,k,i1d,
      i1q,v1q,v1d,x,qt,qy,ip,iq,qr;
      pd:=vdXid;
      iv:=id;
      u:=vd/ec;
      if u>1 then
      begin
            u:=1;
            ip:=id;
            iq:=0;
            goto L3;
      end
      else
      If u<-1 then
      begin
            u:=-1;
            ip:=-id;
            iq:=0;
            goto L3;
      end;
      Ip:=uXid;
      iq:=sqrt(ivT2-ipT2);
      ip:= if re=-1 then -ip else  ip;
      L3:
      ecd:=ec;
      ecq:=0;
      qr:=ecdXiq;
      ivd:=ip;
      ivq:=-iq;
      if rs=1000then
      begin
            v1d:=ecdXn;
            v1q:=ecqXn;
            i1d:=ivd/n;
            i1q:=ivq/n;
            qt:=0;
            qy:=0;
            goto L1;
      end;
      comment calculations for qt
      and qy start;
```

```
vyd:=ecd+(xc-xv)×1vq;
vyq:=-(xc-xv)×1vd;
vy :=sqrt(vyd↑2+vyq↑2);
Ad:=1vd+vyq/x1;
Aq:=-(1vq-vyd/x1);
Bd:=vyd+xt/x1×vyd-xt×1vq;
Bq:=vyq+xt×1vd+xt/x1×vyq;
Cq:=-1/(n×x1);
Cd:=0;
Dd:=-xt/(n×x1);
Dq:=0;
a:=vr×(Bd×Cq+Bq×Cd-Ad×Dq-Aq×Dd);
b:=vr×(Bd×Cd-Bq×Cq+Ad×Dd-Aq×Dq);
c:=vr↑2×(Cd×Dd-Cq×Dq+Ad×Bd-Aq×Bq);
x2:=(-a×c+b×sqrt(a↑2-c↑2+b↑2))/
(b↑2+a↑2);
x3:=(-a×c-b×sqrt(a↑2-c↑2+b↑2))/
(b↑2+a↑2);
x:=if abs(x2)>abs(x3) then
x3 else x2;
d:=Ad×Bq+Aq×Bd;
e:=Cq×Dd+Dq×Cd;
m:=Bq×Cd+Cq×Bd;
g:=Aq×Dd+Dq×Ad;
h:=Ad×Dd-Aq×Dq;
k:=Bd×Cd-Bq×Cq;
qt:=d+vr↑2×e+vr×sqrt(1-x↑2)×
(m+g)+vr××x×(h-k);
qy:=vy↑2/(2×xt)×(sqrt(1+4××xt×qt/
(vy↑2))-1);
comment end of calculations for
qt and qy;
i1d:=(ivd-qy×vyq/(vy↑2))/n;
i1q:=(ivq+qy×vyd/(vy↑2))/n;
v1d:=vr×sqrt(1-x↑2);
v1q:=vr×x;
L1:
isd;=i1d-S×v1q;
isq:=i1q+S×v1d;
is:=sqrt(isd↑2+isq↑2);
esd:=v1d-xs×isq;
esq:=v1q+xs×isd;
es:=sqrt(esd↑2+esq↑2);
qs:=isq×esd-isd×esq;

      SYTMVOLT :=es;
end SYSTMVOLT;

procedure underlin(m,n);value m,n;
integer m,n;
begin integeri;for i:=1 step 1 until
      m+n-1 do charout(30,if i<m then
      64 else 30)
end;
```

```
procedure c;writetext(30,[[c]]);
coptext(20,30,[;;]);c;underlIn(0,121);c;
writetext(30,[[5s]V10[6s]V20[7s]idc[8s]vd
[8s]pd[8s]ibq[6s]Ibd[8s]u1[8s]idq[6s]idd
[8s]u2[8s]n1[cc]]);
underlIn(0,121);c;
comment test for inversion;
If k<0 then
begin
        SWAP(z1,z2);
        SWAP(Ir,Ir1);
        SWAP(KR,KI);
        SWAP(n1,n2);
end;
comment the values of V10 or V20 are
varied;
for vx:=t1 step t2 until t3 do
begin
        if F1=1 then V20:=vx
        else
        V10:=vx;
        comment computation of invertor
        characteristics with C.E.A.control;
        v1:=-1.00;
        idc:=Ir;
        L13:idc1:=idc;
        xc:=x1;
        ec1:=(-v1+p1/6xidcxxc)/z2;
        re:=R1;
        rs:=j1;
        vr:=1 ;
        n:=n2;
        es1:=SYSTMVOLT(idc,ec1,v1,vr);
        if abs(V20-es1)>10↑(-3) then
        begin
                v2:=v1- .2;
                ec2:=(-v2+p1/6xidcxxc)/z2;
                es2:=SYSTMVOLT(idc,ec2,v2,vr);
                if abs(V20-es2)>10↑(-3) then
                begin
                        L3: v3:=(V20-es1)/
                        (es2-es1)x(v2-v1)+v1;
                        ec3:=(-v3+p1/6xidcxxc)/z2;
                        es3:=SYSTMVOLT
                        (idc,ec3,v3,vr);
                        if abs(V20-es3)>10↑
                        (-3) then
                        begin
                                es1:=es2;
                                es2:=es3;
                                v1:=v2;
                                v2:=v3;
                                goto L3;
                        end
```

```
                                    else
                                    vd:=v3;
                            end
                            else
                            vd:=v2;
                    end
                    else
                    vd:=v1;
                    vd2:=vd;
                    comment the above computed values
                    are stored;
                    ba:=isd;
                    bb:=isq;
                    bc:=qs;
                    bd:=u ;
                    comment computation of rectifier
                    characteristics with C.C.control;
                    vd:=-vd2;
                    i1:=idc-0.1;
                    xc:=xl;
                    vr:=1;
                    ec1:=(-vd+i1x(p1/6xxc-r)+KRx
                    (Ir-i1))/z1;
                    if KR=0 then
                    ec1:=(vd+I1x(p1/6xxc+r))/z1;
                    re:=R2;
                    rs:=j2;
                    L12: n:=n1;
                    es1:=SYSTMVOLT(i1,ec1,vd,vr);
                    if abs(V10-es1)>10↑(-3)then
                    begin
                            i2:=i1-.01;
                            ec2:=(-vd+i2x(p1/6xxc-r)+
                            KRx(Ir-i2))/z1;
                            if KR=0 then ec2:=(vd+i2x
                            (p1/6xxc+r))/z1;
                            es2:=SYSTMVOLT(i2,ec2,vd,vr);
                            if abs(V10-es2)>10↑(-3)then
                            begin
                                    L4:i3:=(V10-es1)
                              /(es2-es1)x(i2-i1)+i1;
                                    ec3:=(-vd+i3x(p1/6xxc
                                    -r)+KRx(Ir-i3))/z1;
                            if KR=0 then ec3:=
                            (vd+i3x(p1/6xxc+r))/z1;
                            es3:=SYSTMVOLT
                            (i3,ec3,vd,vr);
                            ifabs(V10-es3)>10↑(-3)
                            then
```

```
                begin
                        es1:=es2;
                        es2:=es3;
                        i1:=i2;
                        i2:=i3;
                        goto L4;
                end
                else
                idc:=i3;
                        end
                        else
                        idc:=i2;
                end
                else
                idc:=i1;
                idc2:=idc;
                comment iterations to find the
                correct value of the direct
                current of the line;
                if abs(idc1-idc2)>=-3 then
                begin
                        idc:=idc2;
                        goto L13;
                end;
                If F=1 then
                comment tap-changing calculations
                start;
                begin
                        if u<zmin then
                        begin
                                if n1> nmax then goto L16
                                else
                                begin
                                        n1:=n1+ninc;
                                        goto L12;
                                end
                        end
                        else
                        if u>zmax then
                        begin
                                if n1<nmin then goto L16
                                else
                                begin
                                        n1:=n1-ninc;
                                        goto L12;
                                end
                        end
                end;
                comment end of tap-changing
                calculations;
                if KR=0 and  abs(vd)>abs(V20xz2-p1/
                6xxcxidc) then goto  FAIL;
```

```
comment the above computed values
are stored;
L16:ca:=isd;
cb:=isq;
cc:=qs;
cd:=u;
ce:=idc;
cf:=vd;
if KR=0 then goto L15;
comment computation of rectifier
characteristics with N.V.control;
v1:=1;
ec1:=v1+idcx(pi/6xxc+r);
es1:=SYSTMVOLT(idc,ec1,v1,vr);
if abs(V10-es1)>10-3 then
begin
        v2:=v1-.2;
        ec2:=v2+idcx(pi/6xxc+r);
        es2:=SYSTMVOLT(idc,ec2,v2,vr);
        if abs(V10-es2)>10-3then
        begin
                L7: v3:=(V10-es1)/(es2-
                es1)x(v2-v1)+v1;
                ec3:=v3+idcx(pi/6xxc+r);
                es3:=SYSTMVOLT(idc,ec3,
                v3,vr);
                if abs(V10-es3)>10-3 then
                begin
                        es1:=es2;
                        es2:=es3;
                        v1:=v2;
                        v2:=v3;
                        goto L7;
                end
                else
                vdn:=v3;
        end
        else
        vdn:=v2;
end
elsevdn:=v1;
if vdn<0 then goto FAIL;
comment the above computed values
are stored;
aa:=isd;
ab:=isq;
ac:=qs ;
ad:=u ;
if abs(vdn)< abs(vd) then
begin
        comment computation of
        Invertor characteristics
        with C.C control;
```

```
vd:=-vdn;
i1:=idc-0.1;
xc:=x';
ec1:=(-vd+i1xp1/6xxc+KIx
(Ir1-i1))/z2;
re:=R1;
rs:=i1;
vr:=1;
n:=n2;
1:=SYSTMVOLT(i1,ec1,vd,vr);
if abs(V20-es1)>10-3then
begin
        i2:=i1-.01;
        ec2:=(-vd+i2xp1/6xxc+KIx
        (Ir1-i2))/z2;
        es2:=SYSTMVOLT
        (i2,ec2,vd,vr);
        if abs (V20-es2)>10-3then
        begin
                L6:i3:=(V20-es1)/
                (es2-es1)x(i2-i1)
                +i1;
                ec3:=(-vd+i3xp1/
                6xxc+KIx(Ir1-i3))
                /z2;
                es3:=SYSTMVOLT
                (i3,ec3,vd,vr);
                if abs(V20-es3)
                >10-3then
                begin
                        es1:=es2;
                        es2:=es3;
                        i1:=i2;
                        i2:=i3;
                        goto L6;
                end
                else
                idc:=i3;
        end
        else
        idc:=i2;
end
else
idc:=i1;
vd:=vdn;
pd:=vdxidc;
idq:=isd;
idd:=isq;
qs2:=qs;
u2:=u;
ibq:=aa;
ibd:=ab;
qs1:=ac ;
u1 :=ad;
```

```
end
else
L15:
begin
        ibq:=ca;
        ibd:=cb;
        qs1:=cc;
        u1:=cd;
        idc:=ce;
        vd:=cf;
        idq:=ba;
        idd:=bb;
        qs2:=bc;
        u2:=bd;
        pd:=idc×vd;
end;
If k<0 then
begin
        SWAP(ibq,idq);
        SWAP(ibd,idd);
        SWAP(u1,u2);
        SWAP(V10,V20);
end;
write(30,f,V10);
write(30,f,V20);
write(30,f,idc);
write(30,f,vd);
write(30,f,pd);
write(30,f,ibq);
write(30,f,ibd);
write(30,f,u1);
write(30,f,idq);
write(30,f,idd);
write(30,f,u2);
write(30,f,n1);
writetext(30,[[cc]]);
if k<0 then
SWAP(V10,V20);
    end;
end;
FAIL:
    end;
    close(20);
    close(30);
end→
```

APPENDIX 3

DIGITAL PROGRAMME TO STUDY THE TRANSIENT BEHAVIOUR OF H.V.D.C. SYSTEM

Identifiers Used:

The identifiers which have not been defined in Appendices 1 and 2 are given below:

Tl - Time constant of the d.c. line, sec.

T - Time constant of the converters, sec.

degtorad - Multiplying factor to convert degrees into radians.

radtodeg - Multiplying factor to convert radians into degrees.

n - Total number of differential equations.

t - Independent variable (time).

y - An array containing the values of the dependent variables (solution) at "t".

range - The procedure replaces the values in the array "y" by the values at t + range replacing "t" by t + range.

acc - Tolerated truncation error.

h - Interval of integration.

comptime - Computing time.

ecr, eci - Control signals of rectifier and invertor respectively, p.u.

Zr, Zi - Cosine of the delay angles of the rectifier and invertor respectively.

Zo - Cosine of the extinction angle of the invertor.

xdc - Inductive reactance of the d.c. line, p.u.

R - Resistance of the d.c. line, p.u.

idcl - New value of the d.c. line current, p.u.

K - Multiplying factor to give a step-rise to vql or Ir.

t1, t2 - Time intervals, sec.

vql, vq2 - Quadrature axis components of a.c. voltages across rectifier and invertor, respectively, p.u.

vdlO, vqlO - Direct and quadrature axis components of sending end a.c. bus voltage with respect to common reference frame, p.u.

vd20, vd20 - Direct and quadrature axis components of receiving end a.c. bus voltage with respect to common reference frame, p.u.

AnglO - Phase difference between sending end bus-bar and reference bus, degrees.

M,Fl Markers for different instructions.

Procedures Used:

The procedures in this programme which have not been defined earlier are given below:

1. Procedure r(x):

This procedure, when called for, reads from the data the values of the identifiers written in the parenthesis.

2. Procedure SYSTMVOLT (id, vd, ec, ivq, ivd):

This is the modified form of the procedure SYSTMVOLT discussed in Appendix I. id, vd and ec are the input parameters of the procedure which calculates ivd and ivq. For example, the transient values of the rectifier current components, on the a.c. side, are computed as:

id - is replaced by the last value of idc

vd - is replaced by the last value of vdl

and, ec - is replaced by the last value of ecl.

The procedure SYSTMVOLT then computes the displacement factor and the magnitude of the alternating current, and thus finds the required current components of the rectifier, ivd and ivq.

3. Procedure KM3 (n,t,y,range,acc,h):

This procedure uses the method of Kutta-Merson to solve n simultaneous first order differential equations, as discussed in Sec. 2-2-4, Vol. I.

4. Procedure fn (t,y,f):

This procedure is written by the user for a particular problem and is called by the procedure KM3 during computation. For this programme all the first order differential equations involved in the solution of the problem are written in transfer functional form where the arrays "f" and "y" represent the input and output of the integrating amplifiers respectively.

5. Procedure PRINT:

This procedure prints the computed values of the identifiers in the "write" statements given in the body of the procedure.

Action of the Programme

The programme reads and stores the data which is given in Table 4-3, Vol I, and then proceeds to print the headings of the required variables. To find the initial values the programme computes the d.c. voltage, current and control signals of both the rectifier and invertor, and then calls the procedure SYSTMVOLT to calculate the respective alternating current components. Subsequently,

the procedure PRINT is called and the computed values of all the identifiers, specified in the procedure, are printed under the appropriate headings.

During the transient state, a series of time intervals are taken and at each interval the values of the required variables are computed. A switch statement is written within the body of the programme which gives a step rise to the a.c. bus-bar voltage or reference current when t > t1, and at t > t2 the raised value is normalised. The differential equations are solved for idc, M1 and M2, by calling the procedure KM3, and the values of the rectifier and invertor control signals together with the other required variables are calculated to match with the latest value of idc. The programme then takes the next value of t, and this process continues until the prefixed computing time is reached. At the end of each step the values of the required variables are printed.

VERBATIM

EG 18 CASE 3 (TRANSIENT**BEHAVIOUR**OF**H.V.D.C.**SYSTEM)→

```
begin library A0,A6,A1,A13,A14;
 open(20);open(30);open(10);
begin real VdB,pi,vd1,idc,vd2,R,xc,T1,degtorad,
           radtodeg,T,KR,Ir,M1,M2,h,acc,range,
           comptime,t,n,Ang10,vq1,ecr,eci,zr,
           VaB,nn,V10,vq10,vd10,u,ibq,z0,ibd,
           xdc,V20,K,t1,t2,A,ec1,ec2,vq20,vd20,
           idq,idd,vq2,KI,k,V,idc1;
      integer M,F1;
      array y[1:6];
      procedure r(x); real x; x:=read(20);
      procedure  SYSTMVOLT(id,vd,ec,ivq,ivd);
      real id,vd,ec,ivq,ivd;
      begin   real iv,u1;
              u1:=vd/ec;
              if u1≥1 then
              begin
                      u:=1;
                      ivq:=id;
                      ivd:=0;
              end
              else
              if u1≤-1 then
              begin
                      u:=-1;
                      ibq:=-id;
                      ivd:=0;
              end
              else
              u:=u1;
              iv:=id;
              ivq:=ivxu;
              ivd:=sqrt(iv↑2-ivq↑2);
      end SYSTMVOLT;
      procedure underlin(m,n);value m,n;
                integer m,n;
      begininteger i;fori:=1step1until1m+n-1do
                     charout(30,if i<m then
                     64else30)
      end;

      procedure c;writetext(30,[[c]]);
      procedure KM3 (n,t,y,range,acc,h);
      value n,range,acc;
      integer n; real t,range,acc,h; array y;
      begin integer j; real t1,hn,inc,error;
                       boolean finish;
            array z,f0,f1,f2[1:n];
```

```
t1:=t+range; finish:=false;
 L1:if hx(t+h-t1)≥0.0 then begin hn:=h;
          h:=t1-t;
 finish:=true end;
 fn(t,y,f0);
 L2:for j:=1 step 1 until n do z[j]
          :=y[j]+f0[j]xh/3;
fn(t+h/3,z,f1);
for j:=1 step 1 until n do z[j]:=y[j]+
          (f0[j]+f1[j])xh/6;
fn(t+h/3,z,f1);
for j:=1 step 1 until n do z[j]:=y[j]+
          (0.125xf0[j]+0.375xf1[j])xh;
fn(t+h/2,z,f2);
for j:=1 step 1 until n do z[j]:=y[j]+
          (0.5xf0[j]-1.5xf1[j]+2.0xf2[j])xh;
fn(t+h,z,f1);
inc:=h;
for j:=1 step 1 until n do
begin f1[j]:=y[j]+(f0[j]+4.0xf2[j]+f1[j])
                  xh/6;
error:=0.2xabs(f1[j]-z[j]);
if error>acc then begin h:=h/2; finish:=
                                  false;goto L2 end;
if error>0.025xacc then inc:=0.0
end;
t:=t+h;
          h:=h+inc;
          for j:=1 step 1 until n do y[j]:=f1[j];
          if finish then h:=hn else goto L1;fn(t,y,f0);
end KM3;
procedure fn(t,y,f);real t;array y,f;
begin
          f[1]:=1/T1x(idc1-y[1]);
          f[2]:=1/Tx(y[1]-y[2]);
          f[3]:=1/Tx(y[2]-y[3]);
          f[4]:=1/Tx(ecr-y[4]);
          f[5]:=1/Tx(eci-y[5]);
end fn;

procedure PRINT;
begin
          write(30,F1,t);
          write(30,F1,ibq);
          write(30,F1,ibd);
          write(30,F1,ecr);
          write(30,F1,vd1);
          write(30,F1,idc);
          write(30,F1,idq);
          write(30,F1,idd);
          write(30,F1,eci);
          write(30,F1,vd2);
          write(30,F1,Ir );
```

```
        write(30,F1,vq2);
         write(30,F1,vq1);
         c;
end;
M:=read(20);
r(K);r(VdB);r(Ang10);r(vd1);r(R);r(VaB);r(xc);
r(KR);r(z0);r(range);r(acc);r(T);r(h);(comptime);
r(Ir);r(zr);r(k);r(KI);r(V10);r(V20);r(xdc);r(t1);
r(t2);
copytext(20,30,[;;]);c;underlin(0,118);
writetext(30,[[8s]t[6s]ibq[5s]ibd[5s]ecr[5s]vd1
[4s]idc[5s]idq[5s]idd[5s]eci[5s]vd2[6s]Ir[5s]
vq2[5s]vq1[c]]);
underlin(0,118); c;
F1:=format([-nnd.ddd]);
pi:=3.142;
T1:=xdc/(100XRXpi);
degtorad:=pi/180;
radtodeg:=180/pi;
nn:=VdBXpi/(VaBXzrX3Xsqrt(2));
comment steady state calculations of the system;
vq10:=V10Xcos(Ang10Xdegtorad);
vd10:=V10Xsin(Ang10Xdegtorad);
vq1:=sqrt(vq10↑2+vd10↑2);
if M=1 then V:=vq1;
If M=2 then V:=Ir ;
vq20:=V20;
vd20:=0;
vq2:=vq20;
M1:=vq1Xzr;
ecr:=M1;
idc:=(Ir-ecr/KR);
vd1:=M1-pi/6XxcXidc;
vd2:=-(vd1-idcXR);
eci:=vd2+pi/6XxcXidc;
z0:=(-eci+pi/6XxcXidc)/vq2;
M2:=eci;
y[1]:=y[2]:=y[3]:=idc1:=idc;
t:=0;
y[4]:=M1;
y[5]:=M2;
SYSTMVOLT(idc,vd1,vq1,ibq,ibd);
SYSTMVOLT(idc,vd2,vq2,idq,idd);
PRINT;
comment end of steady state calculations;
begin
      comment start of transient state
                calculations;
      for n:=5 while t<comptime+.0001 do
      begin
            comment instructions to give a
                    step rise to vq1 or
                    Ir at required interval
                    of time;
```

```
if t≤t1then A:=V
 else
 if t≤t2then A:=V×K
 else
 A:=V;
 if M=1 then vq1:=A;
 if M=2 then Ir :=A;
 idc:=(vd1+vd2)/R;
 if idc<0 then idc:=0;
 KM3 (n,t,y,range,acc,h);
 comment limits imposed on the control
          signals;
 idc:=y[3];
 if idc<k×Ir then ecr:=vq1
 else
 if idc≥Ir then ecr:=vq1×.087
 else
 ecr:=KR×(Ir-idc);
 if ecr>vq1 then ecr:=vq1;
 M1:=y[4];
 if M1>vq1 then M1:=vq1;
 vd1:=M1-pi/6××c×idc;
 ec1:=KI×(k×Ir-idc);
 ec2:=-vq2×z0+pi/6××c×idc;
 if idc≥Ir×k then ec1:=ec2
 else
 if (ec1+ec2)>0 then ec1:=-vq2×0.087
 else
 ec1:= ec1+ec2;
 M2:=y[5];
 vd2:=M2-pi/6××c×idc;
 idc1:=(vd1+vd2)/R;
 if idc1<0 then idc1:=0;
 SYSTMVOLT(idc,vd1,vq1,ibq,ibd);
 SYSTMVOLT(idc,vd2,vq2,idq,idd);
 PRINT;
                end;
          end;
     end;
     close(20);close(30);close(10);
end →
```

APPENDIX 4

DIGITAL PROGRAMME TO DRAW THE SWING CURVES OF AN A.C. SYSTEM

Identifiers Used:

The identifiers of this programme which have not been defined earlier are given as:

gl,g2,gf - Conductances of the generator, transmission line and shunt fault respectively, p.u.

bl,b2,bf - Susceptance of the generator, transmission line and shunt fault respectively, p.u.

Angl - Phase displacement of the sending end a.c. bus-bar with respect to the common reference frame, degrees.

Ang2 - Rotor angle, degrees.

Ang3 - Phase displacement of the receiving end a.c. bus-bar with respect to the common reference frame, degrees.

Cosang 2, Sinang 2 - Cosine and sine of Ang2, respectively.

ed10, eq10 - d - q axis components of the generator terminal voltage with respect to the common reference frame, p.u.

id10, iq10 - d - q axis components of the generator current with respect to the common reference frame, p.u.

ed11, eq11 - d - q axis components of the generator terminal voltage with respect to Park's reference frame, p.u.

id11, iq11 - d - q axis components of the generator current with respect to Park's reference frame, p.u.

Gef1 - Terminal voltage of the generator on open circuit and at normal speed, p.u.

Gefd1O, Gefq1O - d - q axis components of Gef1 with respect to the common reference frame, p.u.

xd1, xq1 - d - q axis components of the generator reactance with respect to the Park's reference frame, p.u.

phid11, phiq11 - d - q axis components of the total armature flux linkages, p.u.

Pel - Positive phase sequence armature power, p.u.

Pi - Input mechanical power, p.u.

xdI1 - Direct-axis component of the transient reactance of the generator, p.u.

Pow Kvel - Accelerating power of the rotor, p.u.

Hl - Inertia constant of the generator.

ml - $H1_{/\pi f}$

t - Independent variable (time in sec)

Kdl - Damping coefficient.

tr - Reclosing time of the circuit breaker, sec.

tfc - Fault clearing time of the circuit breaker, sec.

V1O, V2O - Sending end and receiving end a.c. bus voltages respectively, p.u.

TdoI1 - Direct axis open-circuit field time constant, sec.

V1Od, V1Oq and V2Od, V2Oq - d - q axis component of V1O and V2O respectively with respect to the common reference frame, p.u.

IDl, IQ1 - d - q axis components of the injected current, p.u.

Xℓ1, Xℓ2 - Inductive reactance of the generator and transmission line, respectively, p.u.

vt - Voltage proportional to the generator terminal voltage, p.u.

vr - Reference terminal voltage for the generator, p.u.

avrin - Input voltage of the comparator, p.u.

vs - Equivalent voltage of the stabiliser, p.u.

kvt - Gain of the machine.

k7 - Gain of the stabiliser.

kmb - Gain of the comparator.

kav - Gain of the exciter.

T7, T6, T5, T4 - Time constants of the stabiliser, exciter and amplifiers 1 and 2, respectively, sec.

ef - Exciter voltage, p.u.

ein - Input voltage to the amplifier 1, p.u.

avrlim - Upper limit of the open circuit terminal voltage, p.u.

EF - Marker to exclude or include A.V.R.

F, F1 - Formats.

F4 - Marker to select the parameters for the network calculations.

n - Total number of differential equations.

Y - Array for the differential equations.

Yn (n = 1, 2,3,4) - Arrays for admittance matrices.

Zn (n = 1, 2,3,4) - Arrays for the inverse of the admittance matrices.

An (n = 1, 2,3,4) - Arrays for the admittances of the system.

Procedures Used

The procedures in this programme that have not been discussed earlier, are given below:

1. Procedure inversion (n,a,b,f singular):

This procedure finds the inverse matrix "b" of any "n" dimensional matrix "a" while keeping the pivot at its minimum allowable value "f". In case of singularity in the matrix, the procedure goes to the label "singular" and the programme terminates. This inversion procedure has been discussed in some detail in Sec. 6-3-2, Vol. I.

2. Procedure w (g,x):

The procedure consists of a string "g" and a real parameter "x", and whenever this is called, the string "g" is printed together with the latest value of the identifier "x", in the prescribed format. If the absolute value of x > 999 then the word "INFINITY" is printed against the string.

3. Procedure fn (t,y,f):

This procedure is written for each individual problem, and is called by the procedure KM3. For this particular programme all the first order differential equations of the synchronous generator and those of its excitation control system are written in transfer functional form. Some of the algebraic equations are also written in the procedure body in such a way to get the right values of the variables used in the differential equations. Array f and real identifier "t" are the input parameters while array y gives the output of the procedure - the arrays f and y representing the input and output of the integrating amplifier, and t the independent variable (time).

4. Procedure PRINT:

This procedure,when called for, prints the latest values of the identifiers mentioned in the write statements of its body.

5. Procedure Z CHANGE (X):

This procedure has been written to change the existing matrix into the new matrix "X".

6. Procedure SETARRAY (i):

All the elements of the required matrix "i" are written in the body of this procedure and when this is called, the required matrix is set up.

7. Procedure STORAPARA (a)

This procedure stores the parameters, which are given in its block, in an array "a".

8. Procedure READ Y (G):

The procedure reads the values of the identifiers written within the "write" statements of its block, and then calls the procedure w(g,x), a number of times, to write the strings and the values of the identifiers specified therein.

Action of the Programme

The computer after reading and storing the data, which is given in Table 5-1, Vol. I, prints the values of computing time, range, accuracy, inertia constant of the generator, infinite bus voltage, fault clearing time and reclosing time, etc.

Then admittance matrices for the different configurations of the system are set up by calling the procedure SETARRAY (Yn) - (n = 1,2,3,4). Similarly by calling the procedure STORARRAY (An) - (n = 1,2,3,4) the admittances of the system are stored - the suffices 1,2,3,4 indicating PRE-FAULT, FAULT, PRE-RECLOSURE and RECLOSURE configurations of the system.

The programme then proceeds to the inversion of the above mentioned admittance matrices. In case of singularity in any matrix, "INVERSION FAILURE - N" (N = 1,2,3,4) is printed out, where N denotes the number of the matrix, and the programme terminates.

The steady state values of the generator variables are computed by considering the prefault admittances of the generator and transmission line respectively, and they are printed. The excitation control system of the generator can be ignored by putting EF = 1; otherwise its calculations will be in the main loop of the programme.

To compute the machine variables during the transient state, time "t" is varied from zero to a prefixed computing time and a switch statement is included in the programme which allots new values to the inverse matrix "Zn" and to the network admittances "An" corresponding to positions when the line is on fault, when the fault is cleared, and when the faulted line is reconnected. Then by using nodal analysis, the programme solves the network for the current components of the generator with respect to the common reference frame. These components are transformed to Park's reference frame, and at this stage the procedure KM3 is called to solve the differential equations. The results are printed under the appropriate headings. The programme then takes the next value of t and the above cycle is repeated until the prefixed value of the computing time is reached.

VERBATIM

```
EG18 CASE 4(TO**DRAW**SWING**CURVES**OF**AN**A.C.**SYSTEM)→
begin library A0,A6,A12,A13,A14;
      open(10); open(20); open(30);
      begin real t,g1,g2,gf,b1,b2,bf,Ang1,Ang2,Ang3,
            Cosang2,Sinang2,comptime,ed10,eq10,eq11,
            Gefd10,Gefq10,Gef1,acc,h,range,xd1,
            phid11,Pi1,id11,iq11,id10,iq10,xdI1,Pe1,
            PowKve11,m1,H1,Kd1,TdoI1,degtorad,radtodeg,
            tr,tfc,V10q,V10d,V10,V20q,V20d,V20,xq1,
            IQ1,ID1,X11,X12,pi,phiq11,vt,vr,avrin,kvt,
            vs,T7,T6,T5,T4,k7,ef,ein,kmb,kav,G,avrlim,
            X1,X2;

      integer EF,F,F1,RECYCLE,F4,n,N;

      array y[1:7],Z,Z1,Z2,Z3,Z4,Y1,Y2,Y3,Y4
      [1:2,1:2],A1,A2,A3,A4[1:2,1:2];

      procedure inversion(n,a,b,f,singular);
      value n; integer n; array a,b; real f;
      label singular;
      begin integer i,j,k,l; real piv, w,z;
            array c,e,q[1:n]; integer array
            p[1:n];
      for i:=1 step 1 until n do
      begin z:=0;
            for j:=1 step 1 until n do
            if abs(a[i,j])>z then
            z:=abs(a[i,j]);
            if z=0 then go to singular ;
            q[i]:=z:=1/z;
            for j:=1 step 1 until n do
            b[i,j]:=a[i,j]×z
      end;
      for k:1 step 1 until n do
      begin piv:=0;
            for i:=k step 1 until n do
            begin w:=b[i,k];
                  if abs(w)>abs(piv)then
                  begin piv:=w;l:=i
            end;
      p[k]:=l;
      if abs(piv)≤f then go to
      singular ;
      if p[k]≠k
      then for j:=1 step 1 until
      n do
      begin z:=b[l,j]; b[l,j]:=b[k,j];
            b[k,j]:=z
      end;
```

```
for j:=1 step 1 until n do
begin if j=k then
      begin e[j]:=1.0/piv;
            c[j]:=1.0
      end else
      begin e[j]:= -b[k,j]/piv;
            c[j]:=b[j,k]
      end;
      b[k,j]:= b[j,k]:=0.0
            end;
            for i:=1 step 1 until n do
            for j:=1step 1 until n do
            b[i,j]:=b[i,j]+c[i]xe[j]
      end k;
      for k:=n step -1 until 1 do
      begin l:=p[k];
            for i:=1 step 1 until n do
            begin z:=b[i,l]; b[i,l]:=b[i,k];
            b[i,k]:=z
            end
      end;
      for k:=1 step 1 until n do
      begin z:=q[k];
            for i:=1 step 1 until n do
            b[i,k]:=b[i,k]xz;
      end
end inversion;

procedure r(x);realx; x:=read(20);

procedure KM3 (n,t,y,range,acc,h);
value n,range,acc;
integer n; real t,rango,acc,h; array y;
begin integer j; real t1,hn,inc,error;
      boolean finish;
      array z,f0,f1,f2[1:n];
      t1:=t+range; finish:=false;
      L1:if hxx((t+h-t1)≥0.0 then
      begin hn:=h; h:=t1-t; finish:=true
      end;
      fn(t,y,f0);
      L2:for j:=1 step 1 until n do
      z[j]:=y[j]+f0[j]xh/3;
      fn(t+h/3,z,f1);
      for j:=1 step 1 until n do
      z[j]:=y[j]+(f0[j]+f1[j])xh/6;
      fn(t+h/3,z,f1);
      for j:=1 step 1 until n do
      z[j]:=y[j]+(0.125xf0[j]+0.375xf1[j])xh;
      fn(t+h/2,z,f2);
      for j:=1 step 1 until n do
      z[j]:=y[j]+(0.5xf0[j]-1.5xf1[j]+2.0xf2[j[])
      xh;
```

```
fn(t+h,z,f1);
        inc:=h;
        for j:=1 step 1 until n do
        begin f1[j]:=y[j]+(f0[j]+4.0xf2[j]+f1[j]
              xh/6;
              error:=0.2xabs(f1[j]-z[j]);
              if error>acc then
              begin h:=h/2; finish:=false;
                    goto L2
              end;
              if error>0.025xacc then inc:=0.0
        end;
        t:=t+h;
        h:=h+inc;
        for j:=1 step 1 until n do y[j]:=f1[j];
        if finish then h:=hn else goto
        L1;fn(t,y,f0);
end KM3;


procedure underlin(m,n);value m,n;integer m,n;
begin integer i;for i:=1 step 1 until m+n-1 do
      charout(30,if i<m then 64 else 30)
end;

procedure c;writetext(30,[[c]]);

procedure  w(g,x); value x; real x; string g;
begin  writetext(30,g);  if abs(x)>999 then
       writetext(30,[*INFINITY*;])  else
       write(30,format([-nnd.ddddds;]),x);
end;

procedure fn(t,y,f); real t; array y,f;

begin
      f[1]:=(Gef1-xd1xid11-phid11)/TdoI1;
      phid11:=y[1]-xdI1xid11;
      eq11:=phid11+xq1xid11;
      Pe1:=iq11xeq11;
      PowKvel1:=Pi1-Pe1-Kd1xy[2];
      f[2]:=PowKvel1xm1;
      f[3]:=-y[2];
      if EF =1 then goto L6;
      comment calculations of excitation
      control;
      phiq11:=iq11xxq1;
      vt:=sqrt(phiq11↑2+phid11↑2);
      avrin:=vr-vtxkvt;
      f[7]:=vs/T7;
      vs:=(k7xef)/T7-y[7];
      ein:=avrinxkmb-vs;
      f[4]:=(ein-y[4])/T4;
      [5]:=(y[4]-y[5])/T5;
```

```
                f[6]:=(y[5]-y[6])/T6;
                ef:=kavxy[6];
                Gef1:=Gxef;
                if Gef1>avrlim then Gef1:=avrlim;
                If Gef1<0.00 then Gef1:=0.00;
                L6:
        end;

        procedure PRINT;
        begin
                write(30,F,t);
                write(30,F,PowKvel1);
                write(30,F,y[2]);
                write(30,F,Pe1);
                write(30,F,y[3]xradtodeg);
                c;
        end;

        procedure Z CHANGE(x); value x; array x;
        begin integer yy,zz;
                for yy:=1,2 do
                begin for zz:=1 step 1 until 2 do
                        Z[zz,yy]:=x[zz,yy];
                end;
        end;

        procedure SET ARRAY(i); array i;
        begin
                i[1,1]:=g1+g2+gf;
                i[1,2]:=b1+b2+bf;
                i[2,1]:=-b1-b2-bf;
                i[2,2]:=g1+g2+gf;
        end;

        procedure STORPARA(a); array a;
        begin
                a[1,1]:=g1;
                a[1,2]:=b1;
                A12,1]:=g2;
                a[2,2]:=b2;
        end;

        procedure  READ Y(G); string G;
        begin
                g1:=read(20);
                g2:=read(20);
                gf:=read(20);
                b1:=read(20);
                b2:=read(20);
                bf:=read(20);
                writetext(30,G);
                writetext(30,[[cc]]);
```

```
            w([NEAR*CONDUCTANCE],g1); space(30,10);
             w([LINE*CONDUCTANCE],g2); c;
             w([NEAR*SUSCEPTANCE],b1); space(30,10);
             w([LINE*SUSCEPTANCE],b2); c;
             w([FAULT*CONDUCTANCE],gf); c;
             w([FAULT*SUSCEPTANCE],bf); c; c;
             underlin(0,118); c;
end;

RECYCLE:=read(20);
for N:=1step 1 until RECYCLE do
begin
            writetext(30,[[p]EG18****BASIC
            ******A.C******SYSTEM[c]]);
            c; underlin(0,118); c; c;
            r(comptime);r(xd1);r(TdoI1);r(Kd1);r(H1);
            r(V20);r(acc);r(h);r(Ang1);r(tr);r(tfc);
            r(range);r(xq1);r(xdI1);r(X11); r(X12);
            r(V10);r(Ang2);r(Ang3);r(pi);r(G);r(kvt);
            r(T7);r(k7);r(kmb);r(T4);r(T5);r(T6);
            r(kav);r(avrlim);
            ef:=read(20);
            radtodeg:=180.0000/pi;
            degtorad:=pi/180.0000;
            m1:=pi×50.000/H1;
            F:=format([-nnd.dddd]);
            F1:=format[-nn.dddddd];
            F4:=0;
            F4:=1;

            w([COMPTIME******],comptime);
            w([RANGE********],range);
            w([ACCURACY******],acc);
            w([EST*H*********],h);
            w([INF*BUS*VOLTS*],V20);
            w([FAULT*CLEAR***],tfc);
            w([LINE*RECLOSE**],tr);

            c; underlin(0,118); c;
            READY([PRE*FAULT*ADMITTANCES]);
            SET ARRAY(Y1);
            STORPARA(A1);
            READY([FAULT*ADMITTANCES]);
            SET ARRAY(Y2);
            STORPARA(A2);
            READY([PRE*RECLOSE*ADMITTANCES]);
            SET ARRAY(Y3);
            STORPARA(A3);
            READY([RECLOSURE*ADMITTANCES]);
            SET ARRAY(Y4);
            STORPARA(A4);
            copytext(20,30,[;;]); c; underlin(0,118);
```

```
writetext(30,[[c]***TIME****POWKVEL***REL
*******PE******ROTOR[c]***SECS******P.U
******VEL* *****P.U*******ANGLE[cc]]);

underlin(0,118);  c;
comment start of steady state calculations;
begin
      inversion(2,Y1,Z1,0,INVFAIL 1);
      inversion(2,Y2,Z2,0,INVFAIL 2);
      inversion(2,Y3,Z3,0,INVFAIL 3);
      inversion(2,Y4,Z4,0,INVFAIL 4);
      g1:=A1[1,1];
      b1:=A1[1,2];
      g2:=A1[2,1];
      b2:=A1[2,2];
      X1:=abs((Ang1-Ang3)xdegtorad);
      X2:=abs((Ang2-Ang1)xdegtorad);
      V20q:=V20xcos(Ang3);
      V20d:=V20xsin(Ang3);

      V10q:=V10xcos(Ang1xdegtorad);
      V10d:=V10xsin(Ang1xdegtorad);
      Pe1:=(V10xV20xsin(X1))/X12;
      Gef1:=Pe1xX11/(V10xsin(X2));
      Gefq10:=Gef1xcos(Ang2xdegtorad);
      Gefd10:=Gef1xsin(Ang2xdegtorad);
      y[3]:=Ang2xdegtorad;
      iq10:=g1x(Gefq10-V10q)-b1x
      (V10d-Gefd10);
      id10:=b1x(V10q-Gefq10)+g1x
      (Gefd10-V10d);
      iq11:=iq10xcos(y[3])+id10xsin
      (y[3]);
      id11:=-iq10xsin(y[3])+id10xcos
      (y[3]);
      phid11:=Gef1-xd1xid11;
      y[1]:=phid11+xdI1xid11;
      eq11:=phid11+xq1xid11;
      Pi1:=Pe1;
      PowKvel1:=y[2]:=0.00000;
      t:=0.00000;
      if EF =1 then goto L7;
      ef:=Gef1/G;
      y[6]:=y[5]:=y[4]:=ein:=ef/kav;
      vs:=0;
      y[7]:=k7xef/T7;
      avrin:=ein/kmb;
      vr:=avrin+vtxkvt;
      L7:PRINT;
      comment end of steady state
      calculations;

      Z CHANGE(Z1);
```

```
comment start of transient state
calculations;

begin
      for n:=7 while t<comptime-
      range do
      begin
            if t>-0.0001 and F4=1
            then
begin
      ZCHANGE(Z2);
      F4:=2;
end
else if t>tfc-0.0001
and F4=2 then
begin
      ZCHANGE(Z3);
      F4:=3;
end
else if t>tr-0.0001
and F4=3 then
begin
      ZCHANGE(Z4);
      F4:=4;
end;
Cosang2:=cos(y[3]);
Sinang2:=sin(y[3]);
eq10:=eq11XCosang2;
ed10:=eq11XSinang2;

if F4=1 then
begin
      g1:=A1[1,1];
      b1:=A1[1,2];
      g2:=A1[2,1];
      b2:=A1[2,2];
end
else if F4=2 then
begin
      g1:=A2[1,1];
      b1:=A2[1,2];
      g2:=A2[2,1];
      b2:=A2[2,2];
end

else if F4=3 then
begin
      g1:=A3[1,1];
      b1:=A3[1,2];
      g2:=A3[2,1];
      b2:=A3[2,2];
end
```

```
 else if F4=4 then
begin
        g1:=A4[1,1];
        b1:=A4[1,2];
        g2:=A4[2,1];
        b2:=A4[2,2];
 end;

 IQ1:=eq10Xg1+ed10Xb1+
 V20qXg2+V20dXb2;
 ID1:=-eq10Xb1+ed10Xg1-
 V20qXb2+V20dXg2;
 V10q:=IQ1XZ[1,1]+ID1X
 Z[1,2];
 V10d:=IQ1XZ[2,1]+ID1X
 Z[2,2];
 iq10:=g1X(eq10-V10q)-
 b1X(V10d-ed10);
 id10:=b1X(V10q-eq10)+
 g1X(ed10-V10d);
 iq11:=iq10XCosang2+
 id10XSinang2;
 id11:=-iq10XSinang2+
 id10XCosang2;
 KM3(n,t,y,range,acc,h);
 comment end of transient
 state calculations;

 PRINT;
      end;
end;
goto QQ;
INVFAIL 1: writetext(30,[INVERSION*
FAILURE*1]); go to QQ;
INVFAIL 2: writetext(30,[INVERSION*
FAILURE*2]); go to QQ;
INVFAIL 3: writetext(30,[INVERSION*
FAILURE*3]); go to QQ;
INVFAIL 4: writetext(30,[INVERSION*
FAILURE*4]); go to QQ;
    end;
end;
QQ:
    end;
    close(10);close(20);close(30);
end→
```

APPENDIX 5

DIGITAL PROGRAMME TO DRAW THE STABILITY BOUNDARIES OF AN A.C. SYSTEM

Identifiers Used:

The identifiers of this programme which have not been defined earlier are given as below:

delta - Rotor angle of the synchronous generator, degrees.

delacc - Prefixed tolerance to compare the increment in the rotor angle, degrees.

DELINC - Original increment given to the rotor angle, degrees.

dell - Identifier used to store the original values of delta.

delmax and delmin - Maximum and minimum values of the delta respectively, degrees.

delinc - Increment given to delta during search programme, degrees.

last deg - Identifier used for the last value of the delta, degrees.

tfc max and tfc min - Maximum and minimum values of the fault clearing times, respectively, sec.

tfcdec - Decrement in the fault clearing time, sec.

Procedures Used:

All the procedures discussed in Appendix 4 have been used with the exception of procedure "PRINT".

Action of the Programme:

This programme includes the swing-curve calculations, as discussed in Appendix 4, together with the tests to find the stability state of the synchronous machine under study. A number of fault clearing times are selected and the programme finds the corresponding critical points of stability by varying the rotor angle of the machine.

Initially a guess of the rotor angle is made, and this is included in the data of the problem. The result of the test of the system at this initial value tells whether the system is stable or unstable and the programme alters the rotor angle accordingly to approach the stability limit. If the conditions for the stable or unstable positions as laid down in the programme are not fulfilled, the rotor angle is given a prefixed increment, and the whole process is repeated until the stability boundary is crossed. However, the increment of the rotor angle is reduced for every crossing, and its sign is reversed to continue the search for the required critical point for a given fault-clearing time. Similarly, the critical values for the other fault-clearing times are calculated and the pre-fault conditions are printed. When all the values of fault clearing times are exhausted the programme finishes.

VERBATIM

```
EG18**CASE5(TO**DRAW**STABILITY**BOUNDARIES**OF**AN**A.C.
                    **SYSTEM)→

begin library A0,A6,A12,A13,A14;
      open(10); open(20); open(30);
      begin real t,g1,g2,gf,b1,b2,bf,Ang1,Ang3,Cosang2,
      Sinang2,comptime,ed10,eq10,eq11,Gefd10,Gefq10,
      Gef1,acc,h,range,xd1,Phid11,Pi1,id11,iq11,id10,
      iq10,dI1,Pe1,PowKvel1,m1,H1,Kd1,TdoI1,degtorad,
      radtodeg,tr,tfc,V10q,V10d,V10,V20q,
      V20d,V20,xq1,IQ1,ID1,pi,delta,delacc,DELINC,
      del1,delmin,delinc,delmax,lastdeg,tfcmax,
      tfcdec,tfcmin;
      integer F,F1,RECYCLE,F4,n,N;
      array y[1:4],Z,Z1,Z2,Z3,Z4,Y1,Y2,Y3,
            Y4[1:2,1:2],A1,A2,A3,A4[1:2,1:2];
      procedure inversion(n,a,b,f,singular);
      value n; integer n; array a,b; real f;
      label singular;
      begin integer i,j,k,l; real piv, w,z;
            array c,e,q[1:n]; integer array p[1:n];
            for i:=1 step 1 until n do
      begin z:=0;
            forj:=1 step 1 until n do
            if abs(a[i,j])>z then
            z:=abs(a[i,j]);
            if z=0 then go to singular ;
            q[i]:=z:=1/z;
            for j:=1 step 1 until n do
            b[i,j]:=a[i,j]xz
      end;
      for k:=1 step 1 until n do
      begin piv:=0;
            for i:=k step 1 until n do
            begin w:=b[i,k];
                  if abs(w)>abs(piv)then
                  begin piv:=w;l:=i end
      end;
      p[k]:=l;
      if abs(piv)≤f then go to singular ;
      if p[k]≠k
      then for j:=1 step 1 until n do
      begin z:=b[l,J]; b[l,J]:=
            b[k,j]; b[k,j]:=z
      end;
      for j:=1 step 1 until n do
      begin if j=k then
            begin e[j]:=1.0/piv;
            c[j]:=1.0 end
      else  begin e[j]:= -b[k,j]/piv;
                  c[j]:=b[j,k]
            end ;
```

```
                b[k,j]:= b[j,k]:=0.0
        end;
        for i:=1 step 1 until n do
        for j:=1step 1 until n do
        b[i,j]:=b[i,j]+c[i]xe[j]
end k;
for k:=n step -1 until 1 do
begin l:=p[k];
        for i:=1 step 1 until n do
        begin z:=b[i,l];
                b[i,l]:=b[i,k];
                b[i,k]:=z end
        end;
        for k:=1 step 1 until n do
        begin z:=q[k];
                for i:=1 step 1 until n do
                b[i,k]:=b[i,k]xz;
        end
end inversion;
procedure r(x);realx; x:=read(20);
procedure KM3 (n,t,y,range,acc,h);
value n,range,acc;
integer n; real t,range,acc,h; array y;
begin integer j; real t1,hn,inc,error;
        boolean finish;
        array z,f0,f1,f2[1:n];
        t1:=t+range; finish:=false;
        L1:if hx(t+h-t1)>0.0 then
        begin hn:=h; h:=t1-t; finish:=true
        end;
        fn(t,y,f0);
        L2:for j:=1 step 1 until n do
        z[j]:=y[j]+f0[j]xh/3;
        fn(t+h/3,z,f1);
        for j:=1 step 1 until n do
        z[j]:=y[j]+(f0[j]+f1[j])xh/6;
        fn(t+h/3,z,f1);
        for j:=1 step 1 until n do
        z[j]:=y[j]+(0.125xf0[j]+0.375xf1[j])xh;
        fn(t+h/2,z,f2);
        for j:=1 step 1 until n do
        z[j]:=y[j]+(0.5xf0[j]-1.5xf1[j]+2.0xf2[j])xh;
        fn(t+h,z,f1);
        inc:=h;
        for j:=1 step 1 until n do
        begin f1[j]:=y[j]+(f0[j]+4.0xf2[j]+f1[j])xh/6;
                error:=0.2xabs(f1[j]-z[j]);
                if error>acc then begin h:=h/2;
                                        finish:=false;
                                        goto L2
                                end;
                if error>0.025xacc then inc:=0.0
end;
t:=t+h;
h:=h+inc;
```

```
        for j:=1 step 1 until n do
        y[j]:=f1[j];
        if finish then h:=hn else
        goto L1;fn(t,y,f0);
end KM3;
procedure underlin(m,n);value m,n;
integerm,n;
begin integer i;fori:=1step1until
        m+n-1do
        charout(30,if1<mthen64else30)
end;
procedure c;writetext(30,[[c]]);
procedure  w(g,x); value x;
realx; string g;
begin  writetext(30,g);  if abs(x)>999 then
       writetext(30,[*INFINITY*;]) else
       write(30,format([-nnd.ddddds;]),x);
end;
procedure fn(t,y,f); real t; array y,f;
begin
        f[1]:=(Gef1-xd1xid11-phid11)/TdoI1;
        phid11:=y[1]-xdI1xid11;
        eq11:=phid11+xq1xid11;
        Pe1:=iq11xeq11;
        PowKvel1:=Pi1-Pe1-Kd1xy[2];
        f[2]:=PowKvel1xm1;
        f[3]:=-y[2];
end;
procedure Z CHANGE(x); value x; array x;
begin integer yy,zz;
        for yy:=1,2 do
        begin for zz:=1 step 1 until 2 do
              Z[zz,yy]:=x[zz,yy];
        end;
end;
procedure SET ARRAY(i); array i;
begin
        i[1,1]:=g1+g2+gf;
        i[1,2]:=b1+b2+bf;
        i[2,1]:=-b1-b2-bf;
        i[2,2]:=g1+g2+gf;
end;
procedure STORPARA(a); array a;
begin
        a[1,1]:=g1;
        a[1,2]:=b1;
        a[2,1]:=g2;
        a[2,2]:=b2;
end;
procedure  READ Y(G); string G;
begin
        g1:=read(20);
        g2:=read(20);
```

```
gf:=read(20);
b1:=read(20);
b2:=read(20);
bf:=read(20);
writetext(30,0);
c;c;
w([NEAR*CONDUCTANCE],g1);space(30,10);
w([LINE*CONDUCTANCE],g2); c;
w([NEAR*SUSCEPTANCE],b1); space(30,10);
w([LINE*SUSCEPTANCE],b2); c;
w([FAULT*CONDUCTANCE],gf); c;
w([FAULT*SUSCEPTANCE],bf); c; c;
underlin(0,118); c;
end;
RECYCLE:=read(20);
for N:=1step 1 until RECYCLE do
begin
      writetext(30,[[p]EG18**STABILITY**
      BOUNDARY*****A.C***SYSTEM[c]);
      c; underlin(0,118); c; c;
      r(comptime);r(xd1);r(TdoI1);r(Kd1);
      r(H1);r(V20);r(acc);r(h);r(Ang1);r(tr);
      r(Gef1);r(range);r(xq1);r(xdI1);r(V10);
      r(Ang3);r(pi);r(delacc);
      r(DELINC);r(tfcmax);r(tfcdec);r(tfcmin);
      r(delmin);
      radtodeg:=180.0000/pi;
      degtorad:=pi/180.0000;
      m1:=pix50.000/H1;
      F:=format([-nnd.dddd]);
      F1:=format[-nn.dddddd]);
      F4:=0;
      F4:=1;
      w([COMPTIME******],comptime);
      w([RANGE*********],range)  ;
      w([ACCURACY******],acc);
      w([EST*H*********],h);
      w([INF*BUS*VOLTS*],V20);
      w([SEND*BUS*VOLTS*],V10);
      w([LINE*RECLOSE**],tr);
      c; underlin(0,118); c;
      READY([PRE*FAULT*ADMITTANCES]);
      SET ARRAY(Y1);
      STORPARA(A1);
      READY([FAULT*ADMITTANCES]);
      SET ARRAY(Y2);
      STORPARA(A2);
      READY([PRE*RECLOSE*ADMITTANCES]);
      SET ARRAY(Y3);
      STORPARA(A3);
      READY([RECLOSURE*ADMITTANCES]);
      SET ARRAY(Y4);
      STORPARA(A4);
```

```
copytext(20,30,[;;]);c;underlin(0,118);
writetext(30,[[c]*****CRIT***CLEAR*****
ROTOR****POWER[c]*****TIME***SECS****
ANGLE*****P.U[cc]]);
underlin(0,118);  c;
inversion(2,Y1,Z1,0,INVFAIL 1);
inversion(2,Y2,Z2,0,INVFAIL 2);
inversion(2,Y3,Z3,0,INVFAIL 3);
inversion(2,Y4,Z4,0,INVFAIL 4);
g1:=A1[1,1];b1:=A1[1,2];
g2:=A1[2,1];b2:=A1[2,2];
delinc:=DELINC;
comment calculations for stability
boundaries start;
for tfc:=tfcmax step-tfcdecuntiltfcmin do
begin integer state,UP,DOWN,stable,unstable,
                working,aim;
      stable:=1;
      unstable:=2;
      UP:=1;
      DOWN:=-1;
      delmax:=-180;
      aim:=UP;
      for delta:=delmin step delinc until
                delmax do
comment swing*-curve calculations
      start;
begin del1:=delta;
      V20q:=V20xcos(Ang3xdegtorad);
      V20d:=V20xsin(Ang3xdegtorad);
      V10q:=V10xcos(Ang1xdegtorad);
      V10d:=V10xsin(Ang1xdegtorad);
      Gefq10:=Gef1xcos(deltax
            degtorad);
      Gefd10:=Gef1xsin(deltax
            degtorad);
      y[3]:=deltaxdegtorad;
      lastdeg:=y[3];
      iq10:=g1x(Gefq10-V10q)-b1x
      (V10d-Gefd10);
      id10:=b1x(V10q-Gefq10)+g1x
      (Gefd10-V10d);
      iq11:=iq10xcos(y[3])+id10x
      sin(y[3]);
      id11:=-iq10xsin(y[3])+id10x
      cos(y[3]);
      phid11:=Gef1-xd1xid11;
      y[1]:=phid11+xdI1xid11;
      eq11:=phid11+xq1xid11;
      Pi1:=Pe1:=iq11xeq11;
      PowKvel1:=y[2]:=0.00000;
      t:=0.00000; Z CHANGE(Z1);
      state:=0;
```

```
F4:=1;
working:=0;
for n:=3 while t<comptime-
range do
begin if t>-0.0001 and F4=1
      then begin ZCHANGE(Z2);
                 F4:=2; end
      else if t>tfc-0.0001 and
      F4=2 thenbegin ZCHANGE
                     (Z3);
                 F4:=3;
           end
else if t>tr-0.0001 and
F4=3 then
begin ZCHANGE(Z4);F4:=4;
end;
Cosang2:=cos(y[3]);
Sinang2:=sin(y[3]);
eq10:=eq11xCosang2;
ed10:=eq11xSinang2;
if F4=1 then
begin g1:=A1[1,1];
      b1:=A1[1,2];
      g2:=A1[2,1];
      b2:=A1[2,2];
end
else if F4=2 then
begin
      g1:=A2[1,1];
      b1:=A2[1,2];
      g2:=A2[2,1];
      b2:=A2[2,2];
end
else if F4=3 then
begin g1:=A3[1,1];
      b1:=A3[1,2];
      g2:=A3[2,1];
      b2:=A3[2,2];
end
else if F4=4 then
begin
      g1:=A4[1,1];
      b1:=A4[1,2];
      g2:=A4[2,1];
      b2:=A4[2,2];
end;

IQ1:=eq10xg1+ed10xb1+V20
qxg2+V20dxb2;
ID1:=-eq10xb1+ed10xg1
-V20qxb2+V20dxg2;
```

```
V10q:=IQ1XZ[1,1]+ID1
XZ[1,2];
V10d:=IQ1XZ[2,1]+ID1
XZ[2,2];
iq10:=g1x(eq10-V10q)
-b1x(V10d-ed10);
id10:=b1x(V10q-eq10)
+g1x(ed10-V10d);
iq11:=iq10xCosang2+id
10xSinang2;
id11:=-iq10xSinang2+id
10xCosang2;
comment end of swing
curve calculations;
KM3(n,t,y,range,acc,h);
comment testing for the
stable or unstable state
of the system;
if abs(y[3]) >pi  then
state:=unstable
else if abs(lastdeg)
-abs(y[3])>0 and
t> 1 then state:=stable;
lastdeg:=if abs(y[3])>
abs(lastdeg)
then y[3] else lastdeg;
If state=stable and
aim= UP then
goto NEXTDEL;
If state≠ working then
begin if (state=unstable
      and aim= UP)
      or (state=stable
      and aim= DOWN)
      then
      begin aim:=-aim;
            delinc:=
            -delinc/2.000;
            delmin:=
            delta+delinc;
            delmax:=
            -delmax;
            if abs
            (delinc)<
                  delacc then
            goto PRINT;
            gotoNEXTDEL;
                  end;
            end;
      end;
      NEXTDEL:n:=n;
end;
```

```
                PRINT :write(30,F,t);
                       write(30,F,tfc);
                       write(30,F,del1);
                       write(30,F,Pi1);
                       c;
                       delmin:=del1+DELINC;
                       delinc:=DELINC/2;
          end;
          comment end of stability boundaries
                  calculations;
                  goto QQ;
                  INVFAIL 1: writetext(30,
                  [INVERSION*FAILURE*1]); go to QQ;
                  INVFAIL 2: writetext(30,
                  [INVERSION*FAILURE*2]); go to QQ;
                  INVFAIL 3: writetext(30,
                  [INVERSION*FAILURE*3]); go to QQ;
                  INVFAIL 4: writetext(30,
                  [INVERSION*FAILURE*4]); go to QQ;
        end;
QQ:end;close(10);close(20);close(30);
end→
```

APPENDIX 6

DIGITAL PROGRAMME TO DRAW THE SWING-CURVES OF AN A.C.-D.C. SYSTEM AND AN A.C. SYSTEM

Identifiers Used:

The identifiers of this programme, which have not been defined earlier, are given below:

ec1, ec2 - Invertor C.C. and C.E.A. control signals respectively, p.u.

Q1 - Total reactive power supplied by the generator, p.u.

Qac - Reactive power consumed by the a.c. network, p.u.

Qdc - Reactive power consumed by the d.c. system, p.u.

inc - Step decrement of the reference current, p.u.

M - Number of steps to normalise the reference current.

A - Multiplying factor to raise the reference current.

Irc - Raised value of the reference current, p.u.

Z1, Z2 - Highest and lowest limits of the cosine of the delay angle, respectively.

rect - Counter for the rotor angle.

F, F1 - Different formats used in writing the results.

FE - Marker to exclude the d.c. calculations.

EF - Marker to include the instruction to punch the computed values of the required variables.

EA - Marker to exclude the calculations of the excitation control system.

X - Array to store the computed values of the rotor angle.

Procedures Used:

The procedures in this programme, which have not been discussed earlier, are given below:

1. Procedure SYSTMVOLT (id,vd,q,d,ec,Iq,Id):

This is the further modification of the procedure SYSTMVOLT discussed in Appendix 4. In its present form id, vd, q,d, and ec are the input parameters of the procedure, and it computes the current components Id and Iq in the common reference frame. For example, in the main programme, ibd and ibq, the current components on the a.c. side of the rectifier are calculated by this procedure as:

id - is replaced by the last value of idc

vd - is replaced by the last value of vdl

d & q - are replaced by the last values of vl0d and vl0q respectively

ec - is replaced by the last value of V_{lO}.

The procedure first calculates the d-q axis components of the rectifier current with respect to the converter reference frame, then after finding the phase displacement between the common and converter reference frames, computes ibd and ibq by axis transformation.

2. Procedure fn (t,y,f):

In this procedure the differential equations of the generator, d.c. system and excitation control are written in such an order that the last two sets of equations, together or independently, can be skipped over if desired.

3. Procedure PRINT:

In this procedure "write" statements are written to print the latest values of the required identifiers.

Action of the Programme

In this case the h.v.d.c. programme of Appendix 3, after some modifications, has been included in the a.c. swing-curve programme of Appendix 4. At each time interval the d.c. section of the programme calculates the a.c. components of the converter current with respect to the common reference frame and thus the node, at which the d.c. line is connected, is subjected to nodal analysis to find the generator currents and consequently the other required variables.

In this programme the following modifications have been made to the above mentioned h.v.d.c. and a.c. swing curve programmes:

1. The a.c. swing-curve programme has been modified to compute directly the steady-state values of the rotor angle, the generator open-circuit voltage and the receiving end bus-voltage, from the known values of the "P" and "Q" components of the sending end bus-voltage and the power delivered through the d.c. line.
2. In this programme provision has been made to include or exclude, together or independently, the d.c. system and the excitation control.
3. In the d.c. portion a special statement has been included which initially gives a step rise to the reference current whenever a short circuit is applied, and then if the rotor angle starts decreasing, brings the raised value back to normal in steps.

VERBATIM

```
EG18 CASE 6(TO**DRAW**SWING**CURVES**OF**AN**A.C.-D.C.**
                  SYSTEM**AND**AN**A.C.**SYSTEM)→
begin library A0,A6,A12,A13,A14;
      open(10); open(20); open(30);
      begin real ibd,ibq,idq,idd,t,g1,g2,gf,b1,b2,bf,Ang1,
            Ang2,Ang3,Cosang2,Sinang2,comptime,ed10,eq10,
            eq11,Gefd10,Gefq10,Gef1,acc,h,range,xd1,phid11,
            Pi1,id11,iq11,id10,iq10,xdI1,Pe1,PowKvel1,m1,H1,
            Kd1,TdoI1,degtorad,radtodeg,tr,tfc,V10q,V10d,
            V10,V20q,V20d,V20,xq1,IQ1,ID1,VaB,Ir1,VdB,pi,
            vd1, idc,vd2,R,xc,KR,Ir,idc1,Ang,ecr,eci,ec1,
            ec2,T1,KI,xdc,z0,k,nn,Irc,Pac,Pdc,Q1,x11,x12,
            inc,M,u,vt,vr,avrin,kvt,vs,T7,T6,phiq11,Gefd1,
            Gefq1,T5,T4,k7,ef,ein,kmb,kav,G,avrlim,Qac,
            Qdc,A,zr,zi,z1,z2;
      integer F,F1,RECYCLE,F4,n,N,FE,EF,EA,rect;
      array y[1:8],Z,Z1,Z2,Z3,Z4,Y1,Y2,Y3,Y4[1:2,1:2],
      A1,A2,A3,A4[1:2,1:2],X[0:505];
      procedure inversion(n,a,b,f,singular);
      value n; integer n; array a,b; real f;
      label singular;
      begin integer i,j,k,l; real piv, w,z;
            array c,e,q[1:n]; integer array p[1:n];
            for i:=1 step 1 until n do
      begin z:=0;
            forj:=1 step 1 until n do
            if abs(a[i,j])>z then
            z:=abs(a[i,j]);
            if z=0 then
            go to singular; q[i]:=z:=1/z;
            for j:=1 step 1 until n do
            b[i,j]:=a[i,j]×z
      end;
      for k:=1 step 1 until n do
      begin piv:=0;
            for i:=k step 1 until n do
            begin w:=b[i,k];
                  if abs(w)>abs(piv)then
                  begin piv:=w;l:=i end
            end;
      p[k]:=l;
      if abs(piv)≤f then go to singular ;
      if p[k]≠k
      then for j:=1 step 1 until n do
      begin z:=b[l,j]; b[l,j]:=b[k,j];
            b[k,j]:=z
      end;
      for j:=1 step 1 until n do
      begin if j=k then
            begin e[j]:=1.0/piv;
                  c[j]:=1.0
            end
```

```
                else begin e[j]:= -b[k,j]/piv;
                           c[j]:=b[j,k]
                     end ;
                b[k,j]:= b[j,k]:=0.0
end;
for i:=1 step 1 until n do
for j:=1step 1 until n do
b[I,j]:=b[I,j]+c[i]×e[j]
end k;
for k:=n step -1 until 1 do
begin l:=p[k];
      for i:=1 step 1 until n do
      begin z:=b[i,l]; b[i,l]:=b[i,k];
            b[i,k]:=z
      end
end;
for k:=1 step 1 until n do
begin z:=q[k];
for i:=1 step 1 until n do b[i,k]:=b
      [i,k]×z;
end
end inversion;
procedure  SYSTMVOLT(id,vd,q,d,ec,Iq,Id);
real id,vd,q,d,ec,Iq,Id;
begin   real zc,zs,iv,ivd,ivq,u1;
      ec:=sqrt(q↑2+d↑2);
      zc:=q/ec;
      zs:=d/ec;
      u1:=vd/ec;
      if u1≥1 then
      begin u:=1;
            ivq:=id;
            ivd:=0;
            goto L3;
      end
      else if u1≤-1 then
      begin u:=-1;
            ivq:=-id;
            ivd:=0;
            goto L3;
      end
      else
      u:=u1;
            iv:=id;
            ivq:=iv×u;
            ivd:=sqrt(iv↑2-ivq↑2);
   L3:Iq:=ivq×zc-ivd×zs;
            Id:=ivq×zs+ivd×zc;
end SYSTMVOLT;
procedure r(x);realx; x:=read(20);
procedure KM3 (n,t,y,range,acc,h);
value n,range,acc;
integer n; real t,range,acc,h; array y;
```

```
begin integer j; real t1,hn,inc,error;
      boolean finish;
      array z,f0,f1,f2[1:n];
      t1:=t+range; finish:=false;
  L1: if hx(t+h-t1)≥0.0 then
      begin hn:=h; h:=t1-t; finish:=true
      end;
      fn(t,y,f0);
 L2: for j:=1 step 1 until n do
      z[j]:=y[j]+f0[j]xh/3;
      fn(t+h/3,z,f1);
      for j:=1 step 1 until n do
      z[j]:=y[j]+(f0[j]+f1[j])xh/6;
      fn(t+h/3,z,f1);
      for j:=1 step 1 until n do
      z[j]:=y[j]+(0.125xf0[j]+0.375x
      f1[j])xh; fn(t+h/2,z,f2);
      for j:=1 step 1 until n do
      z[j]:=y[j]+(0.5xf0[j]-1.5xf1[j]+2.0
      xf2[j])xh;
      fn(t+h,z,f1);
      inc:=h;
      for j:=1 step 1 until n do
      begin f1[j]:=y[j]+(f0[j]+4.0xf2[j]+f1[j])
            xh/6;
            error:=0.2xabs(f1[j]-z[j]);
            if error>acc then
            begin h:=h/2; finish:=false; goto L2
            end;
            If error>0.025xacc then inc:=0.0
      end;
      t:=t+h;
      h:=h+inc;
      for j:=1 step 1 until n do y[j]:=f1[j];
      If finish then h:=hn
      else goto L1;fn(t,y,f0);
end KM3;
procedure underlin(m,n);valuem,n;integerm,n;
begin integeri;fori:=1steplluntil1m+n-1do
      charout(30,if1<mthen64else30)
end;
procedure c;writetext(30,[[c]]);
procedure  w(g,x); value x; realx; string g;
begin  writetext(30,g);  if abs(x)>999 then
       writetext(30,[*INFINITY*;])  else
       write(30,format([-nnd.ddddds;]),x);
end;
procedure fn(t,y,f); real t; array y,f;
begin f[1]:=(Gef1-xd1xdd11-phidI1)/TdoI1;
      phid11:=y[1]-xdI1xid11;
      eq11:=phid11+xq1xid11; Pe1:=1q11xeq11;
      PowKve11:=Pi1-Pe1-Kd1xy[2];
      f[2]:=PowKve11xm1;
```

```
f[3]:=-y[2];
if FE=1 then goto LA;
f[4]:=1/T1x(idc1-y[4]);
LA:if EA =1 then goto LB;
phiq11:=iq11xxq1;
vt:=sqrt(phiq11↑2+phid11↑2);
avrin:=vr-vtxkvt;
f[5]:=vs/T7;
vs:=(k7xef)/T7-y[5];
ein:=avrinxkmb-vs;
f[6]:=(ein-y[6])/T4;
f[7]:=(y[6]-y[7])/T5;
f[8]:=(y[7]-y[8])/T6;
ef:=kavxy[6];
Gef1:=Gxef;
if Gef1>avrlim then Gef1:=avrlim;
if Gef1<0.00 then Gef1:=0.00;
LB:end;
   procedure PRINT;
   begin write(30,F,t);
         write(30,F,PowKvel1);
         write(30,F,y[2]);
         write(30,F,Pe1);
         write(30,F,Q1);
         write(30,F,Pac);
         write(30,F,y[3]xradtodeg);
         write(30,F,idc);
         write(30,F,vd1);
         write(30,F,vd2);
         write(30,F,Pdc);
         write(30,F,Ir);
         write(30,F,V10);
         c;
         if EF=1 then
         begin
               write(10,F,ibq);
               write(10,F,ibd);
               write(10,F,idq);
               write(10,F,idd);
               write(10,F,ecr);
               write(10,F,ec1);
               writetext(10,[[c]]);
               end;
   end;
   procedure Z CHANGE(x); value x; array x;
   begin integer yy,zz;
         for yy:=1,2 do begin for zz:=
                                1 step 1 until 2 do
                                Z[zz,yy]:=x[zz,yy];
                          end;
   end;
   procedure SET ARRAY(i); array i;
   begin
         i[1,1]:=g1+g2+gf;
```

```
            i[1,2]:=b1+b2+bf;
            i[2,1]:=-b1-b2-bf;
            i[2,2]:=g1+g2+gf;
end;
procedure STORPARA(a); array a;
begin
            a[1,1]:=g1;
            a[1,2]:=b1;
            a[2,1]:=g2;
            a[2,2]:=b2;
end;
procedure  READ Y(G); string G;
begin
            g1:=read(20);
            g2:=read(20);
            gf:=read(20);
            b1:=read(20);
            b2:=read(20);
            bf:=read(20);
            writetext(30,G);
            writetext(30,[[cc]]);
            w([NEAR*CONDUCTANCE],g1); space(30,10);
            w([LINE*CONDUCTANCE],g2); c;
            w([NEAR*SUSCEPTANCE],b1); space(30,10);
            w([LINE*SUSCEPTANCE],b2); c;
            w([FAULT*CONDUCTANCE],gf); c;
            w([FAULT*SUSCEPTANCE],bf); c; c;
            underlin(0,118); c;
end;
RECYCLE:=read(20);for N:=1step 1 until
RECYCLE do
begin writetext(30,[[p]EG18****BASIC***NUMBER
            **TWO**AC/DC***SYSTEM[c]];
            c; underlin(0,118); c; c;
            r(comptime);r(xd1);r(TdoI1);r(Kd1);
            r(H1);r(acc);r(h);r(tr);r(tfc);r(range);
            r(xq1);r(xdI1);r(Pe1);r(Ang3);r(pi);r(R);
            r(xc);r(KR);r(Pdc);r(vd1);r(xdc);r(KI);
            r(k);r(VdB);r(VaB);r(Q1);r(x11);r(x12);r(M);
            r(V10);r(G);r(kvt);r(T7);r(kmb);r(T4);r(T5);
            r(T6);r(kav);r(avrlim);r(k7);r(A);r(z0);
            r(z1);r(z2);
            FE:=read(20);
            EF:=read(20);
            EA:=read(20);
            n:=read(20);
            radtodeg:=180.0000/pi;
            degtorad:=pi/180.0000;
            m1:=pix50.000/H1;
            F:=format([-nnd.dddd]);
            F1:=format[-nn.dddddd];
            F4:=0;
            nn:=VdB/VaBxpi/(3xsqrt(2));
            F4:=1;
```

```
T1:=xdc/(100XRxpi);
rect:=0;
w([COMPTIME******],comptime);
space(30,10);  w([REF. CURRENT********],Ir);
c; w([RANGE*********],range)   ;
space(30,10);w([D.C.LINE*VOLTAGE***],vdi);c;
w([ACCURACY******],acc);
space(30,10); w([CURRENT*MARGIN*****],k);c;
w([EST*H*********],h);
space(30,10); w([D.C.LINE*RESISTANCE],R);c;
w([INF*BUS*VOLTS*],V20);
space(30,10);w([D.C.LINE*INDUCTANCE],xdc);c;
w([FAULT*CLEAR***],tfc);
space(30,10);w([RECT.GAIN**********],KR);c;
w([LINE*RECLOSE**],tr);
space(30,10);w([INVT.GAIN**********],KI);c;
c; underlin(0,118); c;
READY([PRE*FAULT*ADMITTANCES]);
SET ARRAY(Y1);
STORPARA(A1);
READY([FAULT*ADMITTANCES]);
SET ARRAY(Y2);
STORPARA(A2);
READY([PRE*RECLOSE*ADMITTANCES]);
SET ARRAY(Y3);
STORPARA(A3);
READY([RECLOSURE*ADMITTANCES]);
SET ARRAY(Y4);
STORPARA(A4);
copytext(20,30,[;;]); c; underlin(0,118);
writetext(30,[[c]***TIME****POWKVEL****
REL*******PE****REACT*******AC****ROTOR
****DIRECT****RECT*****INVT*******DC****
**IR******V10[c]***SECS******P.U ******
VEL*******P.U*****POWER******POWER**ANGLE
****CURRENT***VOLT*****VOLT******POWER
*****P.U****,p.u[cc]]);
underlin(0,118);  c;
comment steady state calculations start;
begin inversion(2,Y1,Z1,0,INVFAIL 1);
      inversion(2,Y2,Z2,0,INVFAIL 2);
      inversion(2,Y3,Z3,0,INVFAIL 3);
      inversion(2,Y4,Z4,0,INVFAIL 4);
g1:=A1[1,1];
b1:=A1[1,2];
g2:=A1[2,1];
b2:=A1[2,2];
comment calculation of receiving end
A.C.bus-voltage and its phase-
displacement;
if FE=1 then Pdc:=0;
```

```
Pac:=Pe1-Pdc;
idc:=Pdc/vd1;
Qdc:=sqrt(idc↑2-(idc×vd1/V10)↑2)×V10;
Qac:=Q1-Qdc;
V20d:=(-Pac××12)/V10;
V20q:=V10-(Qac××12)/V10;
V20:=sqrt(V20q↑2+V20d↑2);
if V20q≥0.00 then Ang1:=arctan(
V20d/V20q)
else
Ang1:=pi+arctan(V20d/V20q);
V20q:=V20×cos(Ang3);
V20d:=V20×sin(Ang3);
V10q:=V10×cos(Ang1);
V10d:=V10×sin(Ang1);
if FE=1 then
begin vd1:=0;
      vd2:=0;
      Ir:=0;
      idc1:=0;
      ibq:=0;
      ibd:=0;
      idq:=0;
      idd:=0;
      goto L6;
end;
comment calculations of D.C. system
        variables;
ecr:=vd1+pi/6××c×idc;
Ir:=idc+ecr/KR;
Ir1:=Ir;
Irc:=Ir×A;
inc:=(Irc-Ir)/M;
eci:=-V20×z0+pi/6××c×idc;
y[4]:=idc1:=idc;
zr:=ecr/V10;
zi:=eci/V20;
vd2:=V20×zi-pi/6××c×idc;
SYSTMVOLT(idc,vd1,V10q,V10d,V10,ibq,
ibd);
SYSTMVOLT(idc,vd2,V20q,V20d,V20,idq,
idd);
comment calculations of GEF1 and
rotors angle;
L6:
Pi1:=Pe1;
Gefd1:=(Pe1××11)/V10;
Gefq1:=V10+(Q1××11)/V10;
Gef1:=sqrt(Gefd1↑2+Gefq1↑2);
if Gefq1≥0.00 then Ang :=arctan
(Gefd1/Gefq1)
else
Ang:=pi+arctan(Gefd1/Gefq1);
```

```
Ang2:=-(Ang-Ang1);
Gefq10:=Gef1xcos(Ang2);
Gefd10:=Gef1xsin(Ang2);

comment setting up the initial
conditions;
y[3]:=Ang2;
X[rect]:=y[3];
iq10:=g1x(Gefq10-V10q)-b1x
(V10d-Gefd10);
id10:=b1x(V10q-Gefq10)+g1x
(Gefd10-V10d);
iq11:=iq10xcos(y[3])+id10x
sin(y[3]);
id11:=-iq10xsin(y[3])+id10x
cos(y[3]);
phid11:=Gef1-xd1xid11;
y[1]:=phid11+xdI1xid11;
eq11:=phid11+xq1xid11;
Z CHANGE (Z1);
PowKvel1:=y[2]:=0.00000;
t:=0.00000;
if EA =1 then goto LC;
phiq11:=iq11xxq1;
vt:=sqrt(phid11↑2+phiq11↑2);
ef:=Gef1/G;
y[8]:=y[7]:=y[6]:=ein:=ef/kav;
vs:=0;
y[5]:=k7xef/T7;
avrin:=ein/kmb;
vr:=avrin+vtxkvt;
LC:PRINT;
ed10:=eq11xsin(y[3]);
eq10:=eq11xcos(y[3]);
Q1:=abs(eq10xid10-ed10xiq10);
comment end of steady state
calculations;
comment start of transient state
calculations;
begin for n:=n while t<comptime
      -range do
      begin if t>-0.0001 and F4=1
            then begin ZCHANGE(Z2);
                       Ir:=Irc;F4:=2;
                 end
      else if t>tfc-0.0001
      and F4=2 then
      begin ZCHANGE(Z3);
            F4:=3;
      end
      else if t>tr-0.0001
      and F4=3 then
      begin ZCHANGE(Z4);
            F4:=4;
```

```
end;
Cosang2:=cos(y[3]);
Sinang2:=sin(y[3]);
eq10:=eq11XCosang2;
ed10:=eq11XSinang2;
if F4=1 then
begin g1:=A1[1,1];
      b1:=A1[1,2];
      g2:=A1[2,1];
      b2:=A1[2,2];
end
else if F4=2 then
begin g1:=A2[1,1];
      b1:=A2[1,2];
      g2:=A2[2,1];
      b2:=A2[2,2];
end
else if F4=3 then
begin g1:=A3[1,1];
      b1:=A3[1,2];
      g2:=A3[2,1];
      b2:=A3[2,2];
end
else if F4=4 then
begin g1:=A4[1,1];
      b1:=A4[1,2];
      g2:=A4[2,1];
      b2:=A4[2,2];
end;
comment nodal-analysis;

IQ1:=eq10Xg1+ed10Xb1+
V20qXg2+V20dXb2-ibq;
ID1:=-eq10Xb1+ed10Xg1
-V20qXb2+V20dXg2-ibd;
V10q:=IQ1XZ[1,1]+ID1X
Z[1,2];
V10d:=IQ1XZ[2,1]+ID1X
Z[2,2];
V10:=sqrt(V10d↑2+V10q↑
2);
iq10:=g1X(eq10-V10q)-
b1X(V10d-ed10);
id10:=b1X(V10q-eq10)+
g1X(ed10-V10d);
iq11:=iq10XCosang2+id
10XSinang2;
id11:=-iq10XSinang2+
id10XCosang2;
if FE=1 then
```

```
begin
KM3(n,t,y,range,acc,h);
goto L;
end;
KM3(n,t,y,range,acc,h);
comment instruction to
normalise the raised
reference current;
X[rect+1]:=y[3];
if abs(X[rect])>abs
(X[rect+1]) then
begin Ir:=Ir-inc;
      if Ir<Ir1 then
      Ir:=Ir1;
end;
rect:=rect+1;
idc:=y[4];
comment calculations of
control signals;
if idc<(Ir-k) then
ecr:=V10xz1
else if idc>Ir then
ecr:=V10xz2
else ecr:=KRx(Ir-idc);
if ecr>V10 then
ecr:=V10;
zr:=ecr/V10;
ec1:=KIx((Ir-k)-idc);
ec2:=-V20xz0+p1/6xxcxidc;
if idc>(Ir-k) then
ec1:=ec2
else if (ec1+ec2)>0 then
ec1:=-V20xz2
else ec1:= ec1+ec2;
z1:=ec1/V20;
vd1:=V10xzr-p1/6xxcxidc;
if vd1<0 then
vd1:=0;
vd2:=V20xz1-p1/6xxcxidc;
Pdc:=idcxvd1;
idc1:=(vd1+vd2)/R;
if idc1<0 then
idc1:=0;
SYSTMVOLT(idc,vd1,V10q,V1
0d,V10,ibq,ibd);
SYSTMVOLT(idc,vd2,V20q,
V20d,V20,idq,idd);
L: Pac:=Pe1-Pdc;
Q1:=abs(eq10Xid10 -ed10x
iq10);
PRINT;
comment end of transient
state calculations;
end;
end;
```

```
                                    go to QQ;
                                    INVFAIL 1: writetext(30,[INVERSION*
                                    FAILURE*1]); go to QQ;
                                    INVFAIL 2: writetext(30,[INVERSION*
                                    FAILURE*2]); go to QQ;
                                    INVFAIL 3: writetext(30,[INVERSION*
                                    FAILURE*3]); go to QQ;
                                    INVFAIL 4: writetext(30,[INVERSION*
                                    FAILURE*4]); go to QQ;
                            end;
                        end;
                    QQ:end;
                    close(10);close(20);close(30);
end→→
```

APPENDIX 7

DIGITAL PROGRAMME TO DRAW THE STABILITY BOUNDARIES OF AN A.C.-D.C. SYSTEM AND OF AN A.C. SYSTEM

Identifiers Used:

All the identifiers used in this programme have been defined earlier.

Procedures Used:

All the procedures used in this programme have been discussed earlier.

Action of the Programme:

This programme is basically the same as that for the stability boundaries of the a.c. system discussed in Appendix 5, except for the following modifications:

1. In this programme the a.c. swing-curve calculations have been replaced by the a.c.-d.c. calculations which have been discussed in Appendix 6.
2. To test for the stable position of the a.c.-d.c. system, the stability criteria of Appendix 5 have been modified by adding the conditions of normality of the reference current.

VERBATIM

```
EG18*CASE 7(TO*DRAW**STABILITY**BOUNDARIES*OF*AN*A.C.-D.C.
                *SYSTEM*AND*AN*A.C.*SYSTEM)→
begin library A0,A6,A12,A13,A14;
      open(10); open(20); open(30);
      begin real ibd,ibq,idq,idd,t,g1,g2,gf,b1,b2,bf,Ang1,
            Ang3,Cosang2,Sinang2,comptime,ed10,eq10,eq11,
            Gefd10,Gefq10,Gef1,acc,h,range,xd1,phid11,Pi1,
            id11,iq11,id10,iq10,xdI1,PA1,PowKve11,m1,H1,Kd1,
            TdoI1,degtorad,radtodeg,tr,V10q,V10d,V10,V20q,
            V20d,V20,xq1,IQ1,ID1,VaB,Ir1, VdB,p1,vd1, idc,
            vd2,R,xc,KR,Ir,idc1,ecr,ec1,ec1,ec2,Ang,T1,KI,
            xdc,z0,k,nn,Irc,Pac,Pdc,xl1,xl2,inc,M,u,vt,vr,
            avrin,kvt,vs,T7,T6,Gefd,Gefq,phiq11,T5,T4,k7,
            ef,ein,kmb,kav,G,avrlim,Qac,A,delta,delacc,
            DELINC,del1,delmin,delinc,delmax,tfcdec,
            tfcmin,tfcmax,lastdeg,C,B,tfc,zr,zi,z1,z2,Q1,
            Qdc,D;
      integer F,F1,RECYCLE,F4,n,N,FE,EF,EA,rect;
      array y[1:8],Z,Z1,Z2,Z3,Z4,Y1,Y2,Y3,Y4[1:2,1:2],
      A1,A2,A3,A4[1:2,1:2],X[0:505];
      procedure inversion(n,a,b,f,singular);
      value n; integer n; array a,b; real f;
      label singular;
      begin integer i,j,k,l; real piv, w,z;
            array c,e,q[1:n]; integer array p[1:n];
            for i:=1 step 1 until n do
      begin z:=0;
            forj:=1 step 1 until n do
            if abs(a[i,j])>z then z:=abs(a[i,j]);
            if z=0 then go to singular ;
            q[i]:=z:=1/z;
            for j:=1 step 1 until n do
            b[i,j]:=a[i,j]xz
      end;
      for k:=1 step 1 until n do
      begin piv:=0;
            for i:=k step 1 until n do
            begin w:=b[i,k];
                  if abs(w)>abs(piv)then
                  begin piv:=w;l:=i end
            end;
            p[k]:=l;
            if abs(piv)≤f then go to singular ;
            if p[k]≠k
            then for j:=1 step 1 until n do
            begin z:=b[l,j]; b[l,j]:=b[k,j];
                  b[k,j]:=z
            end;
            for j:=1 step 1 until n do
            begin if j=k then
```

```
                begin e[j]:=1.0/piv;
                      c[j]:=1.0
                end
                else begin e[j]:= -b[k,j]/piv;
                           c[j]:=b[j,k]
                     end ;
                b[k,j]:= b[j,k]:=0.0
end;
for i:=1 step 1 until n do
for j:=1step 1 until n do
b[i,j]:=b[i,j]+c[i]xe[j]
                end k;
                for k:=n step -1 until 1 do
                begin l:=p[k];
                      for i:=1 step 1 until n do
                      begin z:=b[i,l]; b[i,l]:=b[i,k];
                            b[i,k]:=z
                      end
                end;
                for k:=1 step 1 until n do
                begin z:=q[k];
                      for i:=1 step 1 until n do
                      b[i,k]:=b[i,k]xz;
                end
end inversion;
procedure  SYSTMVOLT(id,vd,q,d,ec,Iq,Id);
real id,vd,q,d,ec,Iq,Id;
begin   real zc,zs,iv,ivd,ivq,u1;
        ec:=sqrt(q↑2+d↑2);
        zc:=q/ec;
        zs:=d/ec;
        u1:=vd/ec;
        if u1≥1 then
        begin u:=1;
              ivq:=id;
              ivd:=0;
              goto L3;
        end
        else if u1≤-1 then
        begin u:=-1;
              ibq:=-id;
              ivd:=0;
              goto L3;
        end
        else u:=u1;
        iv:=id;
        ivq:=ivxu;
        ivd:=sqrt(iv↑2-ivq↑2);
L3:     Iq:=ivqxzc-ivdxzs;
        Id:=ivqxzs+ivdxzc;
end SYSTMVOLT;
procedure r(x);realx; x:=read(20);
procedure KM3 (n,t,y,range,acc,h);
```

```
value n,range,acc;
integer n; real t,range,acc,h; array y;
begin integer j; real t1,hn,inc,error;
      boolean finish;
      array z,f0,f1,f2[1:n];
      t1:=t+range; finish:=false;
      L1:if hx(t+h-t1)≥0.0 then
      begin hn:=h; h:=t1-t;
            finish:=true
      end;
      fn(t,y,f0);
      L2:for j:=1 step 1 until n do
      z[j]:=y[j]+f0[j]xh/3;
      fn(t+h/3,z,f1);
      for j:=1 step 1 until n do
      z[j]:=y[j]+(f0[j]+f1[j])xh/6;
      fn(t+h/3,z,f1);
      for j:=1 step 1 until n do
      z[j]:=y[j]+(0.125xf0[j]+0.375xf1[j])xh;
      fn(t+h/2,z,f2);
      for j:=1 step 1 until n do
      z[j]:=y[j]+(0.5xf0[j]-1.5xf1[j]+2.0xf2[j])
      xh;
      fn(t+h,z,f1);
      inc:=h;
      for j:=1 step 1 until n do
      begin f1[j]:=y[j]+(f0[j]+4.0xf2[j]+f1
            [j])xh/6;
            error:=0.2xabs(f1[j]-z[j]);
            if error>acc then
            begin h:=h/2; finish:=false;
                  goto L2
            end;
            if error>0.025xacc then
            inc:=0.0
      end;
      t:=t+h;
      h:=h+inc;
      for j:=1 step 1 until n do
      y[j]:=f1[j];
      if finish then h:=hn else goto L1;
      fn(t,y,f0);
end KM3;
procedure underlin(m,n);valuem,n;integerm,n;
begin integeri;fori:=1step1until1m+n-1do
      charout(30,if1<mthen64else30)
end;
procedure c;writetext(30,[[c]]);
procedure  w(g,x); value x; realx; string g;
begin  writetext(30,g);  if abs(x)>999 then
      writetext(30,[*INFINITY*;])  else
      write(30,format([-nnd.ddddds;]),x);
end;
```

```
procedure fn(t,y,f); real t; array y,f;
begin f[1]:=(Gef1-xd1xid11-phid11)/TdoI1;
      phid11:=y[1]-xdI1xid11;
      eq11:=phid11+xq1xid11;
      Pe1:=iq11xeq11;
      PowKve11:=Pi1-Pe1-Kd1xy[2];
      f[2]:=PowKve11xm1;
      f[3]:=-y[2];
      if FE=1 then goto LA;
      f[4]:=1/T1x(idc1-y[4]);
LA:if EA =1 then goto LB;
      comment calculations of the
      excitation control;
      phiq11:=iq11xxq1;
      vt:=sqrt(phiq11↑2+phid11↑2);
      avrin:=vr-vtxkvt;
      f[5]:=vs/T7;
      vs:=(k7xef)/T7-y[5];
      ein:=avrinxkmb-vs;
      f[6]:=(ein-y[6])/T4;
      f[7]:=(y[6]-y[7])/T5;
      f[8]:=(y[7]-y[8])/T6;
      ef:=kavxy[8];
      Gef1:=Gxef;
      if Gef1>avrlim then Gef1:=avrlim;
      if Gef1<0.00 then Gef1:=0.00;
LB:   end;
      procedure Z CHANGE(x); value x; array x;
      begin integer yy,zz;
            for yy:=1,2 do
            begin for zz:=1 step 1 until 2 do
                  Z[zz,yy]:=x[zz,yy];
            end;
      end;
      procedure SET ARRAY(i); array i;
      begin i[1,1]:=g1+g2+gf;
            i[1,2]:=b1+b2+bf;
            i[2,1]:=-b1-b2-bf;
            i[2,2]:=g1+g2+gf;
      end;
      procedure STORPARA(a); array a;
      begin a[1,1]:=g1;
            a[1,2]:=b1;
            a[2,1]:=g2;
            a[2,2]:=b2;
      end;
      procedure  READ Y(G); string G;
      begin g1:=read(20);
            g2:=read(20);
            gf:=read(20);
            b1:=read(20);
            b2:=read(20);
            bf:=read(20);
```

```
            writetext(30,G);
            writetext(30,[[cc]]);
            w([NEAR*CONDUCTANCE],g1); space(30,10);
            w([LINE*CONDUCTANCE],g2); c;
            w([NEAR*SUSCEPTANCE],b1); space(30,10);
            w([LINE*SUSCEPTANCE],b2); c;
            w([FAULT*CONDUCTANCE],gf); c;
            w([FAULT*SUSCEPTANCE],bf); c; c;
            underlin(0,118); c;
end;
RECYCLE:=read(20);
for N:=1step 1 until RECYCLE do
begin writetext(30,[[p]EG18****BASIC***NUMBER
      **TWO**AC/DC***SYSTEM[c]]);
      c; underlin(0,118); c; c;
      r(comptime);r(xd1);r(TdoI1);r(Kd1);
      r(H1);r(acc);r(h);r(tr);
      r(range);r(xq1);r(xdI1);r(Ang3);r(pi);
      r(R);r(xc);r(KR);r(vd1);r(xdc);r(KI);
      r(k);r(VdB);r(VaB);r(x11);r(x12);r(M);
      r(V10);r(G);r(kvt);r(T7);r(kmb);r(T4);
      r(T5);r(T6);r(kav);r(avrlim);r(delacc);
      r(A);r(DELINC);r(tfcmax);r(tfcdec);
      r(tfcmin);
      r(delmin);r(Gef1);r(z0);r(z1);r(z2);
      r(Ang1);
      r(k7);
      FE:=read(20);
      EF:=read(20);
      EA:=read(20);
      n :=read(20);
      radtodeg:=180.0000/pi;
      degtorad:=pi/180.0000;
      m1:=pi*50.000/H1;
      F:=format([-nnd.dddd]);
      F1:=format[[-nn.dddddd]];
      F4:=0;
      nn:=VdB/VaB*pi/(3*sqrt(2));
      F4:=1;
      T1:=xdc/(100*R*pi);
      rect:=0;
      C:=V10;
      B:=vd1;
      D:=Gef1;
      w([COMPTIME******],comptime);
      space(30,10);
      w([REF. CURRENT********],Ir);c;
      w([RANGE*********],range)   ;
      space(30,10);  w([D.C.LINE*VOLTAGE***],
      vd1);c;
      w([ACCURACY******],acc);
      space(30,10);
      w([CURRENT* MARGIN*****],k);c;
      w([EST*H*********],h);
```

```
space(30,10);
w([D.C.LINE *RESISTANCE],R);c;
w([INF*BUS*VOLTS*],V20);
space(30,10);
w([D.C.LINE* INDUCTANCE],xdc);c;
w([FAULT*CLEAR***],tfc);
space(30,10);
w([RECT. GAIN**********],KR);c;
w([LINE*RECLOSE**],tr);
space(30,10);
w([INVT.GAIN **********],KI);c;
c; underlin(0,118); c;
READY([PRE*FAULT*ADMITTANCES]);
SET ARRAY(Y1);
STORPARA(A1);
READY([FAULT*ADMITTANCES]);
SET ARRAY(Y2);
STORPARA(A2);
READY([PRE*RECLOSE*ADMITTANCES]);
SET ARRAY(Y3);
STORPARA(A3);
READY([RECLOSURE*ADMITTANCES]);
SET ARRAY(Y4);
STORPARA(A4);
copytext(20,30,[;;]); c; underlin(0,118);
writetext(30,[[c]*****CRIT***CLEAR*****
ROTOR****POWER[c]*****TIME***SECS*****
ANGLE*****P.U[cc]]);
underlin(0,118);  c;
inversion(2,Y1,Z1,0,INVFAIL 1);
inversion(2,Y2,Z2,0,INVFAIL 2);
inversion(2,Y3,Z3,0,INVFAIL 3);
inversion(2,Y4,Z4,0,INVFAIL 4);
g1:=A1[1,1];
b1:=A1[1,2];
g2:=A1[2,1];
b2:=A1[2,2];
comment calculations for stability
boundaries start;
delinc:=DELINC;
for tfc:=tfcmax step -tfcdec until
tfcmin do
begin integer state,UP,DOWN,stable,
      unstable,working,aim;
      stable:=1;
      unstable:=2;
      UP:=1;
      DOWN:=-1;
      delmax:=-180;
      aim:=UP;
      comment swing curve calculations
      start;
      for delta:=delmin step delinc until
      delmax do
```

```
begin
      del1:=delta;
      vd1:=B;
      V10:=C;
      Gef1:=D;
      rect:=0;
      Ang:=-(delta-Ang1)xdegtorad;
      Gefd:=Gef1xsin(Ang);
      Gefq:=Gef1xcos(Ang);
      P11:=Pe1:=GefdxV10/x11;
      if FE =1 then Pdc:=0
      else Pdc:= 1/2xPe1;
      idc:=Pdc/vd1;
      Q1:=(Gefq-V10)xV10/x11;
      Qdc:=sqrt(idc↑2-(idcxvd1/V10)
      ↑2)xV10;
      Qac:=Q1-Qdc;
      Pac:=Pe1-Pdc;
      V20d:=(-Pacxx12)/V10;
      V20q:=V10-(Qacxx12)/V10;
      V20:=sqrt(V20q↑2+V20d↑2);
      V20q:=V20xcos(Ang3xdegtorad);
      V20d:=V20xsin(Ang3xdegtorad);
      V10q:=V10xcos(Ang1xdegtorad);
      V10d:=V10xsin(Ang1xdegtorad);

      if FE=1 then
      begin vd1:=0;
            vd2:=0;
            Ir:=0;
            Ir1:=0;
            idc:=0;
            ibq:=0;
            ibd:=0;
            idq:=0;
            idd:=0;
            goto L6;
      end;
      comment calculations of D.C.
      variables;
      ecr:=vd1+pi/6xxcxidc;
      Ir:=idc+ecr/KR;
      Ir1:=Ir;
      Irc:=IrxA;
      inc:=(Irc-Ir)/M;
      eci:=-V20xz0+pi/6xxcxidc;
      y[4]:=idc1:=idc;
      zr:=ecr/V10;
      zi:=eci/V20;
      vd2:=V20xz1-pi/6xxcxidc;
      SYSTMVOLT(idc,vd1,V10q,V10d,
      V10,ibq,ibd);
      SYSTMVOLT(idc,vd2,V20q,V20d,
```

```
                V20,idq,idd);
L6:             Gefq10:=Gef1xcos(deltax
                degtorad);
                Gefd10:=Gef1xsin(deltax
                degtorad);
                y[3]:=deltaxdegtorad;
                lastdeg:=y[3];
                X[rect]:=y[3];
                iq10:=g1x(Gefq10-V10q)-b1x
                (V10d-Gefd10);
                id10:=b1x(V10q-Gefq10)+g1x
                (Gefd10-V10d);
                iq11:=iq10xcos(y[3])
                +id10xsin(y[3]);
                id11:=-iq10xsin(y[3])
                +id10xcos(y[3]);
                phid11:=Gef1-xd1xid11;
                y[1]:=phid11+xdI1xid11;
                eq11:=phid11+xq1xid11;
                PowKve11:=y[2]:=0.00000;
                t:=0.00000;
                phiq11:=iq11xxq1;
                vt:=sqrt(phiq11↑2+phid11↑2);
                if EA =1 then goto LC;
                ef:=Gef1/G;
                y[8]:=y[7]:=y[6]:=ein:=ef/kav;
                vs:=0;
                y[5]:=k7xef/T7;
                avrin:=ein/kmb;
                vr:=avrin+vtxkvt;
LC:             Z CHANGE(Z1);
                state:=0;
                F4:=1;
                working:=0;
                for n:=n while t<comptime-
                range do
                begin If t>-0.0001
                      and F4=1 then
                      begin ZCHANGE(Z2);Ir:=Irc;
                            F4:=2;
                      end
                      else if t>tfc-0.0001
                      and F4=2  then
                      begin ZCHANGE(Z3);
                            F4:=3;
                      end
                      else if t>tr-0.0001
                      and F4=3 then
                      begin ZCHANGE(Z4);
                            F4:=4;
                end;
                Cosang2:=cos(y[3]);
                Sinang2:=sin(y[3]);
```

```
eq10:=eq11xCosang2;
ed1C:=eq11xSinang2;
IF F4=1 then
begin g1:=A1[1,1];
      b1:=A1[1,2];
      g2:=A1[2,1];
      b2:=A1[2,2];
end

else if F4=2 then
begin g1:=A2[1,1];
      b1:=A2[1,2];
      g2:=A2[2,1];
      b2:=A2[2,2];
end

else if F4=3 then
begin g1:=A3[1,1];
      b1:=A3[1,2];
      g2:=A3[2,1];
      b2:=A3[2,2];
end
else if F4=4 then
begin g1:=A4[1,1];
      b1:=A4[1,2];
      g2:=A4[2,1];
      b2:=A4[2,2];
end;
comment nodal analysis
of the network;

IQ1:=eq10xg1+ed10xb1+
V20qxw2+V20dxb2-ibq;
ID1:=-eq10xb1+ed10x
g1-V20qxb2+V20dxg2-ibd;
V10q:=IQ1xZ[1,1]+ID1xZ
[1,2];
V10d:=IQ1xZ[2,1]+ID1xZ
[2,2];
iq10:=g1x(eq10-V10q)-b1x
(V10d-ed10);
id10:=b1x(V10q-eq10)
+g1x(ed10-V10d);
iq11:=iq10xCosang2+id10
xSinang2;
id11:=-iq10xSinang2+id10
xCosang2;
if FE=1 then
begin KM3(n,t,y,range,
      acc,h);
      goto L;
end;
V10:=sqrt(V10d↑2+V10q↑2);
V20:=sqrt(V20d↑2+V20q↑2);
KM3(n,t,y,range,acc,h);
```

```
comment instructions to
normalise the raised
reference current;
X[rect+1]:=y[3];
if abs(X[rect])>abs(X
[rect+1]) then
begin Ir:=Ir-inc;
if Ir<Ir1 then
Ir:=Ir1;
end;
rect:=rect+1;
comment calculation of
the control signals and
other variables of
converters;
idc:=y[4];
if idc<(Ir-k) then
ecr:=V10xz1
else
if idc>Ir then
ecr:=V10xz2
else ecr:=KRx(Ir-idc);
if ecr>V10 then
ecr:=V10;
zr:=ecr/V10;
ec1:=KIx((Ir-k)-idc);
ec2:=-V20xz0+pi/6xxcxidc;
if idc>(Ir-k) then
eci:=ec2
else
if (ec1+ec2)>0 then
eci:=-V20xz2
else eci:= ec1+ec2;
zi:=eci/V20;
vd1:=V10xzr-pi/6xxcxidc;
if vd1<0 then
vd1:=0;
vd2:=V20xzi-pi/6xxcxidc;
idc1:=(vd1+vd2)/R;
if idc1<0 then
idc1:=0;
SYSTMVOLT(idc,vd1,V10q,
V10d,V10,ibq,ibd);
SYSTMVOLT(idc,vd2,V20q,
V20d,V20,idq,idd);
comment end of swing
curve calculations;
comment test for
stable and unstable
state of the system;
L:
if abs(y[3]) >pi  then
state:=unstable
```

```
else if abs(lastdeg)-abs
(y[3])>0 and t≥ 1 and
Ir=Ir1  then
state:=stable;
lastdeg:=if abs(y[3])>
abs(lastdeg)
then y[3] else lastdeg;
if state=stable and
aim= UP
then goto NEXTDEL;
if state≠ working then
begin if (state=unstable
      and aim= UP) or
      (state=stable and
      aim=DOWN) then
      begin aim:=-aim;
            delinc:=
            -delinc/2.000;
            delmin:=
            delta+
            delinc;
            delmax:=
            -delmax;
            if abs(delinc)
            < delacc then
            goto PRINT;
            goto NEXTDEL;
            end;
      end;
      NEXTDEL:n:=n;
end;
PRINT :write(30,F,t);
write(30,F,tfc);
write(30,F,del1);
write(30,F,P11);
c;
delmin:=del1+DELINC;
delinc:=DELINC/2;
end;
goto QQ;
comment end of stability boundaries
calculations;
INVFAIL 1: writetext(30,[INVERSION*FAILURE
*1]);go to QQ;
INVFAIL 2: writetext(30,[INVERSION*FAILURE
*2]);go to QQ;
INVFAIL 3: writetext(30,[INVERSION*FAILURE
*3]);go to QQ;
INVFAIL 4: writetext(30,[INVERSION*FAILURE
*4]);go to QQ;
      end;
QQ:end;close(10);close(20);close(30);
end→
```