IAEA/ANL
Interregional Training Course

Technical and Administrative Preparations Required for Shipment of Research Reactor Spent Fuel to Its Country of Origin

Argonne National Laboratory
Argonne, IL
13 - 24 January 1997

Lecture L.3.3a

Spent Fuel Classification Data PHDOSE Program Package

R. B. Pond

RERTR Program
Argonne National Laboratory

International Atomic Energy Agency
Vienna, Austria

Argonne National Laboratory
Illinois, USA

PHDOSE Program Package:
Example Input
Example Output
Input/Output Description
PHDOSE Program (Fortran)

PHDOSE Program

Calculates the photon dose rate of a line-source model (see attached figure) of spent fuel as a function of:

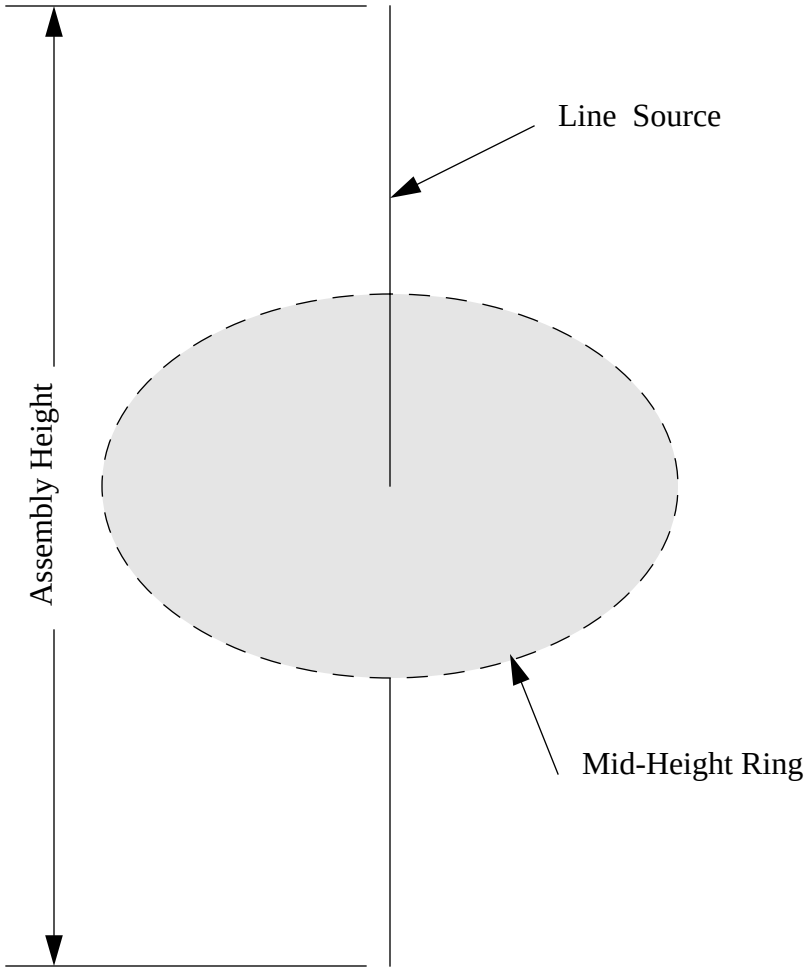
1. Fuel assembly power density, MW/kg^{235}U
    a. fuel assembly power, MW
    b. fuel assembly mass, kg^{235}U
2. U-235 Burnup, %
3. Fission product decay time, year

A dose rate estimate for an MTR-, TRIGA- or DIDO-type fuel assembly can be made based upon the line-source model dose rate. An MTR-type fuel assembly dose rate is a factor of ≈ 0.92 times the line-source model dose rate; a TRIGA-type fuel assembly is a factor of ≈ 1.04 times an MTR-type fuel assembly dose rate; and a DIDO-type fuel assembly is a factor of ≈ 1.05 times an MTR-type fuel assembly dose rate.

Calculates the photon source (ph/s), photon dose (rem/h), radioactivity (Curies) and gamma decay heat (Watts) of 12 fission products. The fission products are:

Sr-90	Y-90	Cs-133	Cs-134
Cs-137	Ba-137m	Ce-144	Pr-144
Ru-106	Rh-106	Zr-95	Nb-95

Similar data are also calculated for Co-60, since it is a possible photon source in spent fuel arising from the activation of Co-59.


Line Source
Assembly Height
Mid-Height Ring

PHDOSE PROGRAM

Purpose To calculate the photon dose rate from spent fuel.

Input (format 7D10.5)

ti	Irradiation time (d) ■ if ti = 0.0, ti is calculated from the powden and burnup parameters as: ti = burnup/powden*8, assuming 1.25 g^{235}U burned per MWd. ■ if ti ≥ 0.0, normal output is printed. ■ if ti < 0.0, ti = -ti and optional output is also printed. ■ note: if ti ≠ 0.0, ti must be within 1% of the calculated irradiation time. (≥ 1%, stop code 101)
power	Assembly power (MW)
powden	Assembly power density (MW/kg^{235}U)
td	Fission product decay time (y)
kg	Structural material mass (kg)
ppm	^{59}Co in structural material (ppm)
burnup	^{235}U burnup in fuel (%)

Normal Output

Record #1:
- Echo input (ti, power, powden, td, kg, ppm, burnup)
- Average neutron flux (n/cm^2-s)

Record #2:
- ^{235}U burned (g) [= power/powden*burnup*10]
- Fission product photon source rate (ph/s) per g^{235}U burned
- ^{60}Co photon source rate (ph/s)
- Self-protecting ^{235}U mass (g^{235}U burned)
- Fission product photon dose rate (rem/h) per g^{235}U burned
- ^{60}Co photon dose rate (rem/h)

Fission products (FP):
^{90}Sr, ^{90}Y, ^{134}Cs, ^{137}Cs, ^{137m}Ba, ^{144}Ce, ^{144}Pr, ^{106}Rh, ^{95}Zr, ^{95}Nb

Optional Output

- Material formation rates (atoms/s)
- Material loss rates (atoms/s)
- Material n0 present at time ti (atoms)
- Material n present at time td (atoms)
- Material disintegration rates (curies)
- Material/group photon source rates (ph/s)
- Material/group photon dose rates (rem/h)
- Group photon heat (watts)
- Group photon flux (ph/cm^2-s)
- Group photon source rate (ph/s)
- Group photon dose rate (rem/h)

Material (Group #) order:

^{90}Sr(1,2), ^{90}Y(1,2,3), ^{133}Cs, ^{134}Cs(2,3), ^{137}Cs(1,2), ^{137m}Ba(2), ^{144}Ce(1), ^{144}Pr(1,2,3), ^{106}Ru, ^{106}Rh(1,2,3), ^{95}Zr(2), ^{95}Nb(2), ^{60}Co(5)

Group:

#1=0.30 MeV, #2=0.63 MeV, #3=1.10 MeV,
#4=sum #1 – #3, #5=1.10 MeV (^{60}Co)

PHDOSE Example Input:

```
  -3584.0     0.025 0.0892857       3.0       4.0      10.0      40.0
```

PHDOSE Example Output:

```
0input: irrad time (d), power (MW), pow den (MW/kgU-235), decay time (y),
     asby mass (kg), Co-59 (ppm), U-235 burnup (%), avg flux (n/cm**2-s)
 -3.58400D+03 2.50000D-02 8.92857D-02 3.00000D+00 4.00000D+00 1.00000D+01
  4.00000D-01 3.29464D+12
 output: U-235 burned (g), FP source (ph/s per g), Co-60 source (ph/s),
     self-pro U-235 mass (g), FP dose (rem/h per g), Co-60 dose (rem/h)
  1.12000D+02 1.34115D+11 6.39796D+10 8.94664D+01 1.11774D+00 8.73080D-01
 material formation rates, atoms/s
  4.52545D+13 0.00000D+00 5.21987D+13 0.00000D+00 4.82975D+13 0.00000D+00
  4.29138D+13 0.00000D+00 3.12880D+12 0.00000D+00 5.07162D+13 0.00000D+00
  5.34375D+10
 material loss rates, atoms/s
  7.84987D-10 3.00501D-06 3.40007D-10 1.10129D-08 7.30132D-10 4.53000D-03
  2.82139D-08 6.68400D-04 2.17415D-08 2.31800D-02 1.22500D-07 2.28512D-07
  4.19008D-09
 material n0, atoms (time=ti)
  1.24395D+22 3.23551D+18 1.53419D+22 4.02355D+20 1.33854D+22 2.15615D+15
  1.52072D+21 6.41825D+16 1.43736D+20 1.34806D+14 4.14010D+20 2.21942D+20
  9.97537D+18
 material n, atoms (time=td)
  1.15529D+22 3.00490D+18 1.53419D+22 1.46762D+20 1.24925D+22 2.01232D+15
  1.05430D+20 4.44971D+15 1.83788D+19 1.72371D+13 3.83499D+15 4.43184D+15
  6.71965D+18
 material disintegration rates, curies
  2.44047D+02 2.44047D+02 0.00000D+00 4.22834D+01 2.46373D+02 2.46373D+02
  8.03834D+01 8.03834D+01 1.07988D+01 1.07988D+01 1.26969D-02 2.73696D-02
  7.58412D-01
 material/group photon source rates, ph/s
  4.23494D+10 3.96405D+07 4.46069D+11 1.46282D+11 2.23937D+10 3.75476D+12
  4.84990D+10 9.20697D+10 3.86510D+08 9.57160D+12 2.97419D+11 2.23956D+11
  1.28485D+11 2.57862D+10 6.11319D+10 1.37047D+11 2.07769D+10 5.40254D+08
  1.23546D+09 6.39796D+10
 material/group photon dose rates, rem/h
  1.82942D-01 3.42041D-04 1.92694D+00 1.26220D+00 3.05590D-01 3.23983D+01
  6.61829D-01 3.97725D-01 3.33503D-03 8.25893D+01 1.28480D+00 9.67452D-01
  1.10864D+00 3.51884D-01 2.64079D-01 1.18252D+00 2.83526D-01 4.66162D-03
  1.06603D-02 8.73080D-01
 group photon heat, wt
  5.59001D-02 1.38693D+00 2.07005D-02 1.46353D+00 1.12758D-02
 group photon flux, ph/cm**2-s
  8.94575D+06 1.05691D+08 9.03470D+05 1.15540D+08 4.92131D+05
 group photon source rate, ph/s
  1.16299D+12 1.37404D+13 1.17456D+11 1.50208D+13 6.39796D+10
 group photon dose rate, rem/h
  5.02394D+00 1.18560D+02 1.60283D+00 1.25187D+02 8.73080D-01
```

```
 program phdose
c PC Version
c The code below has been compiled using a Pentium processor and a Lahey
c compiler.  It should work in most PC environments.
c hand calculation of photon dose rate
      real (kind=8) dk(13),yield(13),cap(13),r(13),k(13),n0(13),n(13),
     x mrate(13)
      real (kind=8) f2d(3),mev(3),gsour(5),gflux(5),gdose(5),gheat(5)
      real (kind=8) ph(20),srate(20),drate(20)
      real (kind=8) ti,power,powden,td,kg,ppm,burnup,jmev,bpc
      real (kind=8) bu,time,dm,pflux,nflux,sums,sumd,self
      common r,n0,powden,kg,ppm,bu,time
      common /comdat/dk,cap
c parameter order in data arrays:
c      material   t(1/2)     ph/dis   groups*
c     #01 Sr-90    28.1y    #01-#02    #1-#2
c     #02 Y-90     2.67d    #03-#05    #1-#3
c     #03 Cs-133
c     #04 Cs-134   2.06y    #06-#07    #2-#3
c     #05 Cs-137   30.1y    #08-#09    #1-#2
c     #06 Ba-137m  2.55m    #10        #2
c     #07 Ce-144   284.d    #11        #1
c     #08 Pr-144   17.3m    #12-#14    #1-#3
c     #09 Ru-106   1.01y
c     #10 Rh-106   29.9s    #15-#17    #1-#3
c     #11 Zr-95    65.5d    #18        #2
c     #12 Nb-95    35.1d    #19        #2
c     #13 Co-60    5.26y    #20        #5
c     *gps, MeV: #1=0.30, #2=0.63, #3=1.10, #4=sum #1-#3, #5=Co-60 (1.10)
c fission product yield
      data yield/5.8d-2,0.d0,6.69d-2,0.d0,6.19d-2,0.d0,5.5d-2,0.d0,
     x 4.01d-3,0.d0,6.5d-2,2*0.d0/
c photons/disintegration
      data ph/4.69d-3,4.39d-6,4.94d-2,1.62d-2,2.48d-3,2.4d0,3.1d-2,
     x 1.01d-2,4.24d-5,1.05d0,1.0d-1,7.53d-2,4.32d-2,8.67d-3,1.53d-1,
     x 3.43d-1,5.2d-2,1.15d0,1.22d0,2.28d0/
c photon flux
      data pflux/7.692d-6/
c conversion: joule=wt-s per mev
      data jmev/1.60219d-13/
c conversion: becquerel per curie
      data bpc/3.7d10/
c mean photon energy, mev/ph
      data mev/0.3d0,0.63d0,1.1d0/
c fluence-to-dose factor, sv-cm**2
      data f2d/1.56d-12,3.116d-12,4.928d-12/
c flux-to-dose conversion, rem/h
      do 105 i=1,3
  105 f2d(i)=f2d(i)*3.6d5
   10 format(7d10.5)
   11 format(1x,1p6d12.5)
      open(5,file='phdose.ft5')
      open(6,file='phdose.ft6')
c read input: irrad time (d); power (MW); pow den (MW/kgU-235);
c decay time (y); asby mass (kg); Co-59 (ppm); U-235 burnup (%)
c note: if ti=0, ti is calculated in the 'time' parameter below
  100 read(5,10,ee(i)=0.0d0
```

```
program phdose
c UNIX Version
c The code below has been compiled and run on a SUN workstation
c and should work in most UNIX environments.
c hand calculation of photon dose rate
      real*8 dk(13),yield(13),cap(13),r(13),k(13),n0(13),n(13),mrate(13)
      real*8 f2d(3),mev(3),gsour(5),gflux(5),gdose(5),gheat(5)
      real*8 ph(20),srate(20),drate(20)
      real*8 ti,power,powden,td,kg,ppm,burnup,jmev,bpc
      real*8 bu,time,dm,pflux,nflux,sums,sumd,self
      common dk,cap,r,n0,powden,kg,ppm,bu,time
c parameter order in data arrays:
c      material   t(1/2)     ph/dis   groups*
c     #01 Sr-90    28.1y    #01-#02    #1-#2
c     #02 Y-90     2.67d    #03-#05    #1-#3
c     #03 Cs-133
c     #04 Cs-134   2.06y    #06-#07    #2-#3
c     #05 Cs-137   30.1y    #08-#09    #1-#2
c     #06 Ba-137m  2.55m    #10        #2
c     #07 Ce-144   284.d    #11        #1
c     #08 Pr-144   17.3m    #12-#14    #1-#3
c     #09 Ru-106   1.01y
c     #10 Rh-106   29.9s    #15-#17    #1-#3
c     #11 Zr-95    65.5d    #18        #2
c     #12 Nb-95    35.1d    #19        #2
c     #13 Co-60    5.26y    #20        #5
c     *gps, MeV: #1=0.30, #2=0.63, #3=1.10, #4=sum #1-#3, #5=Co-60 (1.10)
c decay constant, s-1
      data dk/7.816d-10,3.005d-6,0.d0,1.066d-8,7.297d-10,4.530d-3,
     x 2.821d-8,6.684d-4,2.174d-8,2.318d-2,1.225d-7,2.285d-7,4.176d-9/
c fission product yield
      data yield/5.8d-2,0.d0,6.69d-2,0.d0,6.19d-2,0.d0,5.5d-2,0.d0,
     x 4.01d-3,0.d0,6.5d-2,2*0.d0/
c capture cross section, cm**2
      data cap/1.028d-24,2.740d-24,103.2d-24,107.1d-24,0.1310d-24,
     x 0.d0,1.172d-24,0.d0,0.4550d-24,2*0.d0,3.563d-24,4.275d-24/
c photons/disintegration
      data ph/4.69d-3,4.39d-6,4.94d-2,1.62d-2,2.48d-3,2.4d0,3.1d-2,
     x 1.01d-2,4.24d-5,1.05d0,1.0d-1,7.53d-2,4.32d-2,8.67d-3,1.53d-1,
     x 3.43d-1,5.2d-2,1.15d0,1.22d0,2.28d0/
c photon flux
      data pflux/7.692d-6/
c conversion: joule=wt-s per mev
      data jmev/1.60219d-13/
c conversion: becquerel per curie
      data bpc/3.7d10/
c mean photon energy, mev/ph
      data mev/0.3d0,0.63d0,1.1d0/
c fluence-to-dose factor, sv-cm**2
      data f2d/1.56d-12,3.116d-12,4.928d-12/
c flux-to-dose conversion, rem/h
      do 105 i=1,3
  105 f2d(i)=f2d(i)*3.6d5
   10 format(7d10.5)
   11 format(1x,1p6d12.5)
c read input: irrad time (d); power (MW); pow den (MW/kgU-235);
c decay time (y); asby mass (kg); Co-59 (ppm); U-235 burnup (%)
```

```
c note: if ti=0, ti is calculated in the 'time' parameter below
  100 read(5,10,end=999) ti,power,powden,td,kg,ppm,burnup
c burnup in percent
      burnup=1.0d-2*burnup
c calculate irradiation time and average neutron flux
      time=burnup/(1.25d-3*powden)
      if(ti.eq.0.0d0) ti=time
      dm=1.0d0 - 0.5d0*burnup
      nflux=2.952d13*powden/dm
c write input data plus nflux
      write(6,20)
   20 format('0input: irrad time (d), power (MW), pow den (MW/kgU-235),
     2decay time (y),'/5x,'asby mass (kg), Co-59 (ppm), U-235 burnup (%)
     3, avg flux (n/cm**2-s)')
      write(6,11) ti,power,powden,td,kg,ppm,burnup*1.0d2,nflux
c use ti as a flag for additional printout
      nskip=0
      if(ti.lt.0.0d0) then
       ti=-ti
       nskip=1
      endif
c check if ti is consistent (<1%) with burnup and power density
      if(dabs(ti/time - 1.0d0).ge.1.0d-2) stop 101
c irradiation time (ti) in days and decay time (td) in years
      ti=ti*8.64d4
      td=td*3.1536d7
c initialize arrays
      do 101 i=1,13
  101 r(i)=0.0d0
      do 103 i=1,5
      gsour(i)=0.0d0
      gflux(i)=0.0d0
      gdose(i)=0.0d0
  103 gheat(i)=0.0d0
c in subroutines, calculate nflux in 10 steps
      bu=burnup/10.0d0
      time=ti/10.0d0
c #1: sr-90
      r(1)=3.121d16*yield(1)*power
      k(1)=dk(1) + cap(1)*nflux
c     n0(1)=r(1)/k(1)*(1.0d0 - dexp(-k(1)*ti))
      call sr90
      n(1)=n0(1)*dexp(-dk(1)*td)
      mrate(1)=dk(1)*n(1)/bpc
      srate(1)=dk(1)*n(1)*ph(1)
      srate(2)=dk(1)*n(1)*ph(2)
      drate(1)=f2d(1)*srate(1)*pflux
      drate(2)=f2d(2)*srate(2)*pflux
      gsour(1)=gsour(1) + srate(1)
      gsour(2)=gsour(2) + srate(2)
      gflux(1)=gflux(1) + srate(1)*pflux
      gflux(2)=gflux(2) + srate(2)*pflux
      gdose(1)=gdose(1) + drate(1)
      gdose(2)=gdose(2) + drate(2)
c #2: y-90
      k(2)=dk(2) + cap(2)*nflux
      n0(2)=dk(1)/k(2)*n0(1)
```

```
       if(k(2)*ti.le.4.605170186d1) n0(2)=
      x dk(1)*r(1)/k(1)*((1.0d0 - dexp(-k(2)*ti))/k(2) -
      x (dexp(-k(1)*ti) - dexp(-k(2)*ti))/(k(2)-k(1)))
       n(2)=dk(1)/dk(2)*n(1)
       if(dk(2)*td.le.4.605170186d1) n(2)=
      x dk(1)*n0(1)*(dexp(-dk(1)*td) - dexp(-dk(2)*td))/
      x (dk(2) - dk(1)) + n0(2)*dexp(-dk(2)*td)
       mrate(2)=dk(2)*n(2)/bpc
       srate(3)=dk(2)*n(2)*ph(3)
       srate(4)=dk(2)*n(2)*ph(4)
       srate(5)=dk(2)*n(2)*ph(5)
       drate(3)=f2d(1)*srate(3)*pflux
       drate(4)=f2d(2)*srate(4)*pflux
       drate(5)=f2d(3)*srate(5)*pflux
       gsour(1)=gsour(1) + srate(3)
       gsour(2)=gsour(2) + srate(4)
       gsour(3)=gsour(3) + srate(5)
       gflux(1)=gflux(1) + srate(3)*pflux
       gflux(2)=gflux(2) + srate(4)*pflux
       gflux(3)=gflux(3) + srate(5)*pflux
       gdose(1)=gdose(1) + drate(3)
       gdose(2)=gdose(2) + drate(4)
       gdose(3)=gdose(3) + drate(5)
c #3: cs-133
       r(3)=3.121d16*yield(3)*power
       k(3)=dk(3) + cap(3)*nflux
       n0(3)=r(3)/k(3)*(1.0d0 - dexp(-k(3)*ti))
       n(3)=n0(3)*dexp(-dk(3)*td)
       mrate(3)=dk(3)*n(3)/bpc
c #4: cs-134
       k(4)=dk(4) + cap(4)*nflux
c      n0(4)=r(3)*((1.0d0 - dexp(-k(4)*ti))/k(4) -
c     x   (dexp(-k(3)*ti) - dexp(-k(4)*ti))/(k(4)-k(3)))
       call cs134
       n(4)=n0(4)*dexp(-dk(4)*td)
       mrate(4)=dk(4)*n(4)/bpc
       srate(6)=dk(4)*n(4)*ph(6)
       srate(7)=dk(4)*n(4)*ph(7)
       drate(6)=f2d(2)*srate(6)*pflux
       drate(7)=f2d(3)*srate(7)*pflux
       gsour(2)=gsour(2) + srate(6)
       gsour(3)=gsour(3) + srate(7)
       gflux(2)=gflux(2) + srate(6)*pflux
       gflux(3)=gflux(3) + srate(7)*pflux
       gdose(2)=gdose(2) + drate(6)
       gdose(3)=gdose(3) + drate(7)
c #5: cs-137
       r(5)=3.121d16*yield(5)*power
       k(5)=dk(5) + cap(5)*nflux
c      n0(5)=r(5)/k(5)*(1.0d0 - dexp(-k(5)*ti))
       call cs137
       n(5)=n0(5)*dexp(-dk(5)*td)
       mrate(5)=dk(5)*n(5)/bpc
       srate(8)=dk(5)*n(5)*ph(8)
       srate(9)=dk(5)*n(5)*ph(9)
       drate(8)=f2d(1)*srate(8)*pflux
       drate(9)=f2d(2)*srate(9)*pflux
```

```
      gsour(1)=gsour(1) + srate(8)
      gsour(2)=gsour(2) + srate(9)
      gflux(1)=gflux(1) + srate(8)*pflux
      gflux(2)=gflux(2) + srate(9)*pflux
      gdose(1)=gdose(1) + drate(8)
      gdose(2)=gdose(2) + drate(9)
c #6: ba-137m
      k(6)=dk(6) + cap(6)*nflux
      n0(6)=dk(5)/k(6)*n0(5)
      if(k(6)*ti.le.4.605170186d1) n0(6)=
     x dk(5)*r(5)/k(5)*((1.0d0 - dexp(-k(6)*ti))/k(6) -
     x (dexp(-k(5)*ti) - dexp(-k(6)*ti))/(k(6)-k(5)))
      n(6)=dk(5)/dk(6)*n(5)
      if(dk(6)*td.le.4.605170186d1) n(6)=
     x dk(5)*n0(5)*(dexp(-dk(5)*td) - dexp(-dk(6)*td))/
     x (dk(6) - dk(5)) + n0(6)*dexp(-dk(6)*td)
      mrate(6)=dk(6)*n(6)/bpc
      srate(10)=dk(6)*n(6)*ph(10)
      drate(10)=f2d(2)*srate(10)*pflux
      gsour(2)=gsour(2) + srate(10)
      gflux(2)=gflux(2) + srate(10)*pflux
      gdose(2)=gdose(2) + drate(10)
c #7: ce-144
      r(7)=3.121d16*yield(7)*power
      k(7)=dk(7) + cap(7)*nflux
c     n0(7)=r(7)/k(7)*(1.0d0 - dexp(-k(7)*ti))
      call ce144
      n(7)=n0(7)*dexp(-dk(7)*td)
      mrate(7)=dk(7)*n(7)/bpc
      srate(11)=dk(7)*n(7)*ph(11)
      drate(11)=f2d(1)*srate(11)*pflux
      gsour(1)=gsour(1) + srate(11)
      gflux(1)=gflux(1) + srate(11)*pflux
      gdose(1)=gdose(1) + drate(11)
c #8: pr-144
      k(8)=dk(8) + cap(8)*nflux
      n0(8)=dk(7)/k(8)*n0(7)
      if(k(8)*ti.le.4.605170186d1) n0(8)=
     x dk(7)*r(7)/k(7)*((1.0d0 - dexp(-k(8)*ti))/k(8) -
     x (dexp(-k(7)*ti) - dexp(-k(8)*ti))/(k(8)-k(7)))
      n(8)=dk(7)/dk(8)*n(7)
      if(dk(8)*td.le.4.605170186d1) n(8)=
     x dk(7)*n0(7)*(dexp(-dk(7)*td) - dexp(-dk(8)*td))/
     x (dk(8) - dk(7)) + n0(8)*dexp(-dk(8)*td)
      mrate(8)=dk(8)*n(8)/bpc
      srate(12)=dk(8)*n(8)*ph(12)
      srate(13)=dk(8)*n(8)*ph(13)
      srate(14)=dk(8)*n(8)*ph(14)
      drate(12)=f2d(1)*srate(12)*pflux
      drate(13)=f2d(2)*srate(13)*pflux
      drate(14)=f2d(3)*srate(14)*pflux
      gsour(1)=gsour(1) + srate(12)
      gsour(2)=gsour(2) + srate(13)
      gsour(3)=gsour(3) + srate(14)
      gflux(1)=gflux(1) + srate(12)*pflux
      gflux(2)=gflux(2) + srate(13)*pflux
      gflux(3)=gflux(3) + srate(14)*pflux
```

```
      gdose(1)=gdose(1) + drate(12)
      gdose(2)=gdose(2) + drate(13)
      gdose(3)=gdose(3) + drate(14)
c #9: ru-106
      r(9)=3.121d16*yield(9)*power
      k(9)=dk(9) + cap(9)*nflux
c     n0(9)=r(9)/k(9)*(1.0d0 - dexp(-k(9)*ti))
      call ru106
      n(9)=n0(9)*dexp(-dk(9)*td)
      mrate(9)=dk(9)*n(9)/bpc
c #10: rh-106
      k(10)=dk(10) + cap(10)*nflux
      n0(10)=dk(9)/k(10)*n0(9)
      if(k(10)*ti.le.4.605170186d1) n0(10)=
     x dk(9)*r(9)/k(9)*((1.0d0 - dexp(-k(10)*ti))/k(10) -
     x (dexp(-k(9)*ti) - dexp(-k(10)*ti))/(k(10)-k(9)))
      n(10)=dk(9)/dk(10)*n(9)
      if(dk(10)*td.le.4.605170186d1) n(10)=
     x dk(9)*n0(9)*(dexp(-dk(9)*td) - dexp(-dk(10)*td))/
     x (dk(10) - dk(9)) + n0(10)*dexp(-dk(10)*td)
      mrate(10)=dk(10)*n(10)/bpc
      srate(15)=dk(10)*n(10)*ph(15)
      srate(16)=dk(10)*n(10)*ph(16)
      srate(17)=dk(10)*n(10)*ph(17)
      drate(15)=f2d(1)*srate(15)*pflux
      drate(16)=f2d(2)*srate(16)*pflux
      drate(17)=f2d(3)*srate(17)*pflux
      gsour(1)=gsour(1) + srate(15)
      gsour(2)=gsour(2) + srate(16)
      gsour(3)=gsour(3) + srate(17)
      gflux(1)=gflux(1) + srate(15)*pflux
      gflux(2)=gflux(2) + srate(16)*pflux
      gflux(3)=gflux(3) + srate(17)*pflux
      gdose(1)=gdose(1) + drate(15)
      gdose(2)=gdose(2) + drate(16)
      gdose(3)=gdose(3) + drate(17)
c #11: zr-95
      r(11)=3.121d16*yield(11)*power
      k(11)=dk(11) + cap(11)*nflux
c     n0(11)=r(11)/k(11)*(1.0d0 - dexp(-k(11)*ti))
      call zr95
      n(11)=n0(11)*dexp(-dk(11)*td)
      mrate(11)=dk(11)*n(11)/bpc
      srate(18)=dk(11)*n(11)*ph(18)
      drate(18)=f2d(2)*srate(18)*pflux
      gsour(2)=gsour(2) + srate(18)
      gflux(2)=gflux(2) + srate(18)*pflux
      gdose(2)=gdose(2) + drate(18)
c #12: nb-95
      k(12)=dk(12) + cap(12)*nflux
      n0(12)=dk(11)*r(11)/k(11)*((1.0d0 - dexp(-k(12)*ti))/k(12) -
     x (dexp(-k(11)*ti) - dexp(-k(12)*ti))/(k(12)-k(11)))
      n(12)=dk(11)*n0(11)*(dexp(-dk(11)*td) - dexp(-dk(12)*td))/
     x (dk(12) - dk(11)) + n0(12)*dexp(-dk(12)*td)
      mrate(12)=dk(12)*n(12)/bpc
      srate(19)=dk(12)*n(12)*ph(19)
      drate(19)=f2d(2)*srate(19)*pflux
```

```
      gsour(2)=gsour(2) + srate(19)
      gflux(2)=gflux(2) + srate(19)*pflux
      gdose(2)=gdose(2) + drate(19)
c #13: co-60
      r(13)=1.197d10*kg*ppm*powden/dm
      k(13)=dk(13) + cap(13)*nflux
c     n0(13)=r(13)/k(13)*(1.0d0 - dexp(-k(13)*ti))
      call co60
      n(13)=n0(13)*dexp(-dk(13)*td)
      mrate(13)=dk(13)*n(13)/bpc
      srate(20)=dk(13)*n(13)*ph(20)
      drate(20)=f2d(3)*srate(20)*pflux
      gsour(5)=srate(20)
      gflux(5)=srate(20)*pflux
      gdose(5)=drate(20)
      gheat(5)=gsour(5)*mev(3)*jmev
c sum fission product photon source and dose rates
      sums=0.0d0
      sumd=0.0d0
      do 102 i=1,19
      sums=sums + srate(i)
  102 sumd=sumd + drate(i)
      burnup=power/powden*1.0d3*burnup
      sums=sums/burnup
      sumd=sumd/burnup
c mass of burned U-235 necessary for 100 rem/h dose rate
      self=1.0d2/sumd
c write output: U-235 burned, g;
c FP source, ph/s per gU-235 burned; Co-60 source, ph/s;
c self-protected U-235 mass burned, g
c FP dose,  rem/h per gU-235 burned; Co-60 dose,  rem/h
      write(6,21)
   21 format(' output: U-235 burned (g), FP source (ph/s per g), Co-60 s
     2ource (ph/s),'/5x,'self-pro U-235 mass (g), FP dose (rem/h per g),
     3 Co-60 dose (rem/h)')
      write(6,11) burnup,sums,srate(20),self,sumd,drate(20)
      if(nskip.eq.0) go to 100
c formation rates, atoms/s
      write(6,31)
   31 format(1x,'material formation rates, atoms/s')
      write(6,11) r
c loss rates, atoms/s
      write(6,32)
   32 format(1x,'material loss rates, atoms/s')
      write(6,11) k
c atoms initially present, atoms (time=ti)
      write(6,33)
   33 format(1x,'material n0, atoms (time=ti)')
      write(6,11) n0
c atoms currently present, atoms (time=td)
      write(6,34)
   34 format(1x,'material n, atoms (time=td)')
      write(6,11) n
c material disintegration rates, curies
      write(6,35)
   35 format(1x,'material disintegration rates, curies')
      write(6,11) mrate
```

```
c photon source rates, ph/s
      write(6,36)
   36 format(1x,'material/group photon source rates, ph/s')
      write(6,11) srate
c photon dose rates, rem/h
      write(6,37)
   37 format(1x,'material/group photon dose rates, rem/h')
      write(6,11) drate
c photon heat, wt; =ph/s*mev/ph*j/mev where j=wt-s
      do 104 i=1,3
      gheat(i)=gsour(i)*mev(i)*jmev
  104 gheat(4)=gheat(4) + gheat(i)
      write(6,41)
   41 format(1x,'group photon heat, wt')
      write(6,11) gheat
c photon flux, ph/cm**2-s; sum fission product photon fluxes
      do 107 i=1,3
  107 gflux(4)=gflux(4) + gflux(i)
      write(6,42)
   42 format(1x,'group photon flux, ph/cm**2-s')
      write(6,11) gflux
c photon source rate, ph/s; sum fission product photon source
      do 106 i=1,3
  106 gsour(4)=gsour(4) + gsour(i)
      write(6,43)
   43 format(1x,'group photon source rate, ph/s')
      write(6,11) gsour
c photon dose rate, rem/h; sum fission product photon dose
      do 108 i=1,3
  108 gdose(4)=gdose(4) + gdose(i)
      write(6,44)
   44 format(1x,'group photon dose rate, rem/h')
      write(6,11) gdose
      go to 100
  999 stop
      end
c
      subroutine sr90
      real*8 dk(13),cap(13),r(13),n0(13)
      real*8 powden,kg,ppm
      real*8 bu,time,dm,nflux,t(10),c(10),n(10),k
      common dk,cap,r,n0,powden,kg,ppm,bu,time
      do 101 i=1,10
      t(i)=dfloat(i)*time
      dm=1.0d0-bu*(dfloat(i)-0.5d0)
      nflux=2.952d13*powden/dm
      k=dk(1) + cap(1)*nflux
      if(i.eq.1) c(1)=-r(1)/k
      if(i.ne.1) c(i)=(n(i-1) - r(1)/k)*dexp(k*t(i-1))
  101 n(i)=r(1)/k + c(i)*dexp(-k*t(i))
c     write(6,11) n
   11 format(1x,1p6d12.5)
      n0(1)=n(10)
      return
      end
c
      subroutine cs134
```

```
      real*8 dk(13),cap(13),r(13),n0(13)
      real*8 powden,kg,ppm
      real*8 bu,time,dm,nflux,t(10),c(10),n(10),k1,k2,del
      common dk,cap,r,n0,powden,kg,ppm,bu,time
      do 101 i=1,10
      t(i)=dfloat(i)*time
      dm=1.0d0-bu*(dfloat(i)-0.5d0)
      nflux=2.952d13*powden/dm
      k1=dk(3) + cap(3)*nflux
      k2=dk(4) + cap(4)*nflux
      del=k2-k1
      if(i.eq.1) c(1)=r(3)*(-1.0d0/k2 + 1.0d0/del)
      if(i.ne.1) c(i)=(n(i-1) - r(3)/k2)*dexp(k2*t(i-1)) +
     x r(3)/del*dexp(del*t(i-1))
  101 n(i)=r(3)*(1.0d0/k2 - dexp(-k1*t(i))/del) + c(i)*dexp(-k2*t(i))
c     write(6,11) n
   11 format(1x,1p6d12.5)
      n0(4)=n(10)
      return
      end
c
      subroutine cs137
      real*8 dk(13),cap(13),r(13),n0(13)
      real*8 powden,kg,ppm
      real*8 bu,time,dm,nflux,t(10),c(10),n(10),k
      common dk,cap,r,n0,powden,kg,ppm,bu,time
      do 101 i=1,10
      t(i)=dfloat(i)*time
      dm=1.0d0-bu*(dfloat(i)-0.5d0)
      nflux=2.952d13*powden/dm
      k=dk(5) + cap(5)*nflux
      if(i.eq.1) c(1)=-r(5)/k
      if(i.ne.1) c(i)=(n(i-1) - r(5)/k)*dexp(k*t(i-1))
  101 n(i)=r(5)/k + c(i)*dexp(-k*t(i))
c     write(6,11) n
   11 format(1x,1p6d12.5)
      n0(5)=n(10)
      return
      end
c
      subroutine ce144
      real*8 dk(13),cap(13),r(13),n0(13)
      real*8 powden,kg,ppm
      real*8 bu,time,dm,nflux,t(10),c(10),n(10),k
      common dk,cap,r,n0,powden,kg,ppm,bu,time
      do 101 i=1,10
      t(i)=dfloat(i)*time
      dm=1.0d0-bu*(dfloat(i)-0.5d0)
      nflux=2.952d13*powden/dm
      k=dk(7) + cap(7)*nflux
      if(i.eq.1) c(1)=-r(7)/k
      if(i.ne.1) c(i)=(n(i-1) - r(7)/k)*dexp(k*t(i-1))
  101 n(i)=r(7)/k + c(i)*dexp(-k*t(i))
c     write(6,11) n
   11 format(1x,1p6d12.5)
      n0(7)=n(10)
      return
```

```
      end
c
      subroutine ru106
      real*8 dk(13),cap(13),r(13),n0(13)
      real*8 powden,kg,ppm
      real*8 bu,time,dm,nflux,t(10),c(10),n(10),k
      common dk,cap,r,n0,powden,kg,ppm,bu,time
      do 101 i=1,10
      t(i)=dfloat(i)*time
      dm=1.0d0-bu*(dfloat(i)-0.5d0)
      nflux=2.952d13*powden/dm
      k=dk(9) + cap(9)*nflux
      if(i.eq.1) c(1)=-r(9)/k
      if(i.ne.1) c(i)=(n(i-1) - r(9)/k)*dexp(k*t(i-1))
  101 n(i)=r(9)/k + c(i)*dexp(-k*t(i))
c     write(6,11) n
   11 format(1x,1p6d12.5)
      n0(9)=n(10)
      return
      end
c
      subroutine zr95
      real*8 dk(13),cap(13),r(13),n0(13)
      real*8 powden,kg,ppm
      real*8 bu,time,dm,nflux,t(10),c(10),n(10),k
      common dk,cap,r,n0,powden,kg,ppm,bu,time
      do 101 i=1,10
      t(i)=dfloat(i)*time
      dm=1.0d0-bu*(dfloat(i)-0.5d0)
      nflux=2.952d13*powden/dm
      k=dk(11) + cap(11)*nflux
      if(i.eq.1) c(1)=-r(11)/k
      if(i.ne.1) c(i)=(n(i-1) - r(11)/k)*dexp(k*t(i-1))
  101 n(i)=r(11)/k + c(i)*dexp(-k*t(i))
c     write(6,11) n
   11 format(1x,1p6d12.5)
      n0(11)=n(10)
      return
      end
c
      subroutine co60
      real*8 dk(13),cap(13),r(13),n0(13)
      real*8 powden,kg,ppm
      real*8 bu,time,dm,nflux,t(10),c(10),n(10),k1,k2,r59
      common dk,cap,r,n0,powden,kg,ppm,bu,time
      do 101 i=1,10
      t(i)=dfloat(i)*time
      dm=1.0d0-bu*(dfloat(i)-0.5d0)
      nflux=2.952d13*powden/dm
      k1=       39.67d-24*nflux
      k2=dk(13) + cap(13)*nflux
      r59=1.022d19*kg*ppm*k1
      if(i.eq.1) c(1)=-r59/k2
      if(i.ne.1) c(i)=(n(i-1) - r59/k2)*dexp(k2*t(i-1))
  101 n(i)=r59/k2 + c(i)*dexp(-k2*t(i))
c     write(6,11) n
   11 format(1x,1p6d12.5)
```

```
n0(13)=n(10)
return
end
```

```
  103 gheat(i)=0.0d0
c in subroutines, calculate nflux in 10 steps
      bu=burnup/10.0d0
      time=ti/10.0d0
c #1: sr-90
      r(1)=3.121d16*yield(1)*power
      k(1)=dk(1) + cap(1)*nflux
c     n0(1)=r(1)/k(1)*(1.0d0 - dexp(-k(1)*ti))
      call sr90
      n(1)=n0(1)*dexp(-dk(1)*td)
      mrate(1)=dk(1)*n(1)/bpc
      srate(1)=dk(1)*n(1)*ph(1)
      srate(2)=dk(1)*n(1)*ph(2)
      drate(1)=f2d(1)*srate(1)*pflux
      drate(2)=f2d(2)*srate(2)*pflux
      gsour(1)=gsour(1) + srate(1)
      gsour(2)=gsour(2) + srate(2)
      gflux(1)=gflux(1) + srate(1)*pflux
      gflux(2)=gflux(2) + srate(2)*pflux
      gdose(1)=gdose(1) + drate(1)
      gdose(2)=gdose(2) + drate(2)
c #2: y-90
      k(2)=dk(2) + cap(2)*nflux
      n0(2)=dk(1)/k(2)*n0(1)
      if(k(2)*ti.le.4.605170186d1) n0(2)=
     x dk(1)*r(1)/k(1)*((1.0d0 - dexp(-k(2)*ti))/k(2) -
     x (dexp(-k(1)*ti) - dexp(-k(2)*ti))/(k(2)-k(1)))
      n(2)=dk(1)/dk(2)*n(1)
      if(dk(2)*td.le.4.605170186d1) n(2)=
     x dk(1)*n0(1)*(dexp(-dk(1)*td) - dexp(-dk(2)*td))/
     x (dk(2) - dk(1)) + n0(2)*dexp(-dk(2)*td)
      mrate(2)=dk(2)*n(2)/bpc
      srate(3)=dk(2)*n(2)*ph(3)
      srate(4)=dk(2)*n(2)*ph(4)
      srate(5)=dk(2)*n(2)*ph(5)
      drate(3)=f2d(1)*srate(3)*pflux
      drate(4)=f2d(2)*srate(4)*pflux
      drate(5)=f2d(3)*srate(5)*pflux
      gsour(1)=gsour(1) + srate(3)
      gsour(2)=gsour(2) + srate(4)
      gsour(3)=gsour(3) + srate(5)
      gflux(1)=gflux(1) + srate(3)*pflux
      gflux(2)=gflux(2) + srate(4)*pflux
      gflux(3)=gflux(3) + srate(5)*pflux
      gdose(1)=gdose(1) + drate(3)
      gdose(2)=gdose(2) + drate(4)
      gdose(3)=gdose(3) + drate(5)
c #3: cs-133
      r(3)=3.121d16*yield(3)*power
      k(3)=dk(3) + cap(3)*nflux
      n0(3)=r(3)/k(3)*(1.0d0 - dexp(-k(3)*ti))
      n(3)=n0(3)*dexp(-dk(3)*td)
      mrate(3)=dk(3)*n(3)/bpc
c #4: cs-134
      k(4)=dk(4) + cap(4)*nflux
c     n0(4)=r(3)*((1.0d0 - dexp(-k(4)*ti))/k(4) -
c    x   (dexp(-k(3)*ti) - dexp(-k(4)*ti))/(k(4)-k(3)))
```

```
      call cs134
      n(4)=n0(4)*dexp(-dk(4)*td)
      mrate(4)=dk(4)*n(4)/bpc
      srate(6)=dk(4)*n(4)*ph(6)
      srate(7)=dk(4)*n(4)*ph(7)
      drate(6)=f2d(2)*srate(6)*pflux
      drate(7)=f2d(3)*srate(7)*pflux
      gsour(2)=gsour(2) + srate(6)
      gsour(3)=gsour(3) + srate(7)
      gflux(2)=gflux(2) + srate(6)*pflux
      gflux(3)=gflux(3) + srate(7)*pflux
      gdose(2)=gdose(2) + drate(6)
      gdose(3)=gdose(3) + drate(7)
c #5: cs-137
      r(5)=3.121d16*yield(5)*power
      k(5)=dk(5) + cap(5)*nflux
c     n0(5)=r(5)/k(5)*(1.0d0 - dexp(-k(5)*ti))
      call cs137
      n(5)=n0(5)*dexp(-dk(5)*td)
      mrate(5)=dk(5)*n(5)/bpc
      srate(8)=dk(5)*n(5)*ph(8)
      srate(9)=dk(5)*n(5)*ph(9)
      drate(8)=f2d(1)*srate(8)*pflux
      drate(9)=f2d(2)*srate(9)*pflux
      gsour(1)=gsour(1) + srate(8)
      gsour(2)=gsour(2) + srate(9)
      gflux(1)=gflux(1) + srate(8)*pflux
      gflux(2)=gflux(2) + srate(9)*pflux
      gdose(1)=gdose(1) + drate(8)
      gdose(2)=gdose(2) + drate(9)
c #6: ba-137m
      k(6)=dk(6) + cap(6)*nflux
      n0(6)=dk(5)/k(6)*n0(5)
      if(k(6)*ti.le.4.605170186d1) n0(6)=
     x dk(5)*r(5)/k(5)*((1.0d0 - dexp(-k(6)*ti))/k(6) -
     x (dexp(-k(5)*ti) - dexp(-k(6)*ti))/(k(6)-k(5)))
      n(6)=dk(5)/dk(6)*n(5)
      if(dk(6)*td.le.4.605170186d1) n(6)=
     x dk(5)*n0(5)*(dexp(-dk(5)*td) - dexp(-dk(6)*td))/
     x (dk(6) - dk(5)) + n0(6)*dexp(-dk(6)*td)
      mrate(6)=dk(6)*n(6)/bpc
      srate(10)=dk(6)*n(6)*ph(10)
      drate(10)=f2d(2)*srate(10)*pflux
      gsour(2)=gsour(2) + srate(10)
      gflux(2)=gflux(2) + srate(10)*pflux
      gdose(2)=gdose(2) + drate(10)
c #7: ce-144
      r(7)=3.121d16*yield(7)*power
      k(7)=dk(7) + cap(7)*nflux
c     n0(7)=r(7)/k(7)*(1.0d0 - dexp(-k(7)*ti))
      call ce144
      n(7)=n0(7)*dexp(-dk(7)*td)
      mrate(7)=dk(7)*n(7)/bpc
      srate(11)=dk(7)*n(7)*ph(11)
      drate(11)=f2d(1)*srate(11)*pflux
      gsour(1)=gsour(1) + srate(11)
      gflux(1)=gflux(1) + srate(11)*pflux
```

```
      gdose(1)=gdose(1) + drate(11)
c #8: pr-144
      k(8)=dk(8) + cap(8)*nflux
      n0(8)=dk(7)/k(8)*n0(7)
      if(k(8)*ti.le.4.605170186d1) n0(8)=
     x dk(7)*r(7)/k(7)*((1.0d0 - dexp(-k(8)*ti))/k(8) -
     x (dexp(-k(7)*ti) - dexp(-k(8)*ti))/(k(8)-k(7)))
      n(8)=dk(7)/dk(8)*n(7)
      if(dk(8)*td.le.4.605170186d1) n(8)=
     x dk(7)*n0(7)*(dexp(-dk(7)*td) - dexp(-dk(8)*td))/
     x (dk(8) - dk(7)) + n0(8)*dexp(-dk(8)*td)
      mrate(8)=dk(8)*n(8)/bpc
      srate(12)=dk(8)*n(8)*ph(12)
      srate(13)=dk(8)*n(8)*ph(13)
      srate(14)=dk(8)*n(8)*ph(14)
      drate(12)=f2d(1)*srate(12)*pflux
      drate(13)=f2d(2)*srate(13)*pflux
      drate(14)=f2d(3)*srate(14)*pflux
      gsour(1)=gsour(1) + srate(12)
      gsour(2)=gsour(2) + srate(13)
      gsour(3)=gsour(3) + srate(14)
      gflux(1)=gflux(1) + srate(12)*pflux
      gflux(2)=gflux(2) + srate(13)*pflux
      gflux(3)=gflux(3) + srate(14)*pflux
      gdose(1)=gdose(1) + drate(12)
      gdose(2)=gdose(2) + drate(13)
      gdose(3)=gdose(3) + drate(14)
c #9: ru-106
      r(9)=3.121d16*yield(9)*power
      k(9)=dk(9) + cap(9)*nflux
c     n0(9)=r(9)/k(9)*(1.0d0 - dexp(-k(9)*ti))
      call ru106
      n(9)=n0(9)*dexp(-dk(9)*td)
      mrate(9)=dk(9)*n(9)/bpc
c #10: rh-106
      k(10)=dk(10) + cap(10)*nflux
      n0(10)=dk(9)/k(10)*n0(9)
      if(k(10)*ti.le.4.605170186d1) n0(10)=
     x dk(9)*r(9)/k(9)*((1.0d0 - dexp(-k(10)*ti))/k(10) -
     x (dexp(-k(9)*ti) - dexp(-k(10)*ti))/(k(10)-k(9)))
      n(10)=dk(9)/dk(10)*n(9)
      if(dk(10)*td.le.4.605170186d1) n(10)=
     x dk(9)*n0(9)*(dexp(-dk(9)*td) - dexp(-dk(10)*td))/
     x (dk(10) - dk(9)) + n0(10)*dexp(-dk(10)*td)
      mrate(10)=dk(10)*n(10)/bpc
      srate(15)=dk(10)*n(10)*ph(15)
      srate(16)=dk(10)*n(10)*ph(16)
      srate(17)=dk(10)*n(10)*ph(17)
      drate(15)=f2d(1)*srate(15)*pflux
      drate(16)=f2d(2)*srate(16)*pflux
      drate(17)=f2d(3)*srate(17)*pflux
      gsour(1)=gsour(1) + srate(15)
      gsour(2)=gsour(2) + srate(16)
      gsour(3)=gsour(3) + srate(17)
      gflux(1)=gflux(1) + srate(15)*pflux
      gflux(2)=gflux(2) + srate(16)*pflux
      gflux(3)=gflux(3) + srate(17)*pflux
```

```
      gdose(1)=gdose(1) + drate(15)
      gdose(2)=gdose(2) + drate(16)
      gdose(3)=gdose(3) + drate(17)
c #11: zr-95
      r(11)=3.121d16*yield(11)*power
      k(11)=dk(11) + cap(11)*nflux
c     n0(11)=r(11)/k(11)*(1.0d0 - dexp(-k(11)*ti))
      call zr95
      n(11)=n0(11)*dexp(-dk(11)*td)
      mrate(11)=dk(11)*n(11)/bpc
      srate(18)=dk(11)*n(11)*ph(18)
      drate(18)=f2d(2)*srate(18)*pflux
      gsour(2)=gsour(2) + srate(18)
      gflux(2)=gflux(2) + srate(18)*pflux
      gdose(2)=gdose(2) + drate(18)
c #12: nb-95
      k(12)=dk(12) + cap(12)*nflux
      n0(12)=dk(11)*r(11)/k(11)*((1.0d0 - dexp(-k(12)*ti))/k(12) -
     x (dexp(-k(11)*ti) - dexp(-k(12)*ti))/(k(12)-k(11)))
      n(12)=dk(11)*n0(11)*(dexp(-dk(11)*td) - dexp(-dk(12)*td))/
     x (dk(12) - dk(11)) + n0(12)*dexp(-dk(12)*td)
      mrate(12)=dk(12)*n(12)/bpc
      srate(19)=dk(12)*n(12)*ph(19)
      drate(19)=f2d(2)*srate(19)*pflux
      gsour(2)=gsour(2) + srate(19)
      gflux(2)=gflux(2) + srate(19)*pflux
      gdose(2)=gdose(2) + drate(19)
c #13: co-60
      r(13)=1.197d10*kg*ppm*powden/dm
      k(13)=dk(13) + cap(13)*nflux
c     n0(13)=r(13)/k(13)*(1.0d0 - dexp(-k(13)*ti))
      call co60
      n(13)=n0(13)*dexp(-dk(13)*td)
      mrate(13)=dk(13)*n(13)/bpc
      srate(20)=dk(13)*n(13)*ph(20)
      drate(20)=f2d(3)*srate(20)*pflux
      gsour(5)=srate(20)
      gflux(5)=srate(20)*pflux
      gdose(5)=drate(20)
      gheat(5)=gsour(5)*mev(3)*jmev
c sum fission product photon source and dose rates
      sums=0.0d0
      sumd=0.0d0
      do 102 i=1,19
      sums=sums + srate(i)
  102 sumd=sumd + drate(i)
      burnup=power/powden*1.0d3*burnup
      sums=sums/burnup
      sumd=sumd/burnup
c mass of burned U-235 necessary for 100 rem/h dose rate
      self=1.0d2/sumd
c write output: U-235 burned, g;
c FP source, ph/s per gU-235 burned; Co-60 source, ph/s;
c self-protected U-235 mass burned, g
c FP dose,  rem/h per gU-235 burned; Co-60 dose,  rem/h
      write(6,21)
   21 format(' output: U-235 burned (g), FP source (ph/s per g), Co-60 s
```

```
     2ource (ph/s),'/5x,'self-pro U-235 mass (g), FP dose (rem/h per g),
     3 Co-60 dose (rem/h)')
      write(6,11) burnup,sums,srate(20),self,sumd,drate(20)
      if(nskip.eq.0) go to 100
c formation rates, atoms/s
      write(6,31)
   31 format(1x,'material formation rates, atoms/s')
      write(6,11) r
c loss rates, atoms/s
      write(6,32)
   32 format(1x,'material loss rates, atoms/s')
      write(6,11) k
c atoms initially present, atoms (time=ti)
      write(6,33)
   33 format(1x,'material n0, atoms (time=ti)')
      write(6,11) n0
c atoms currently present, atoms (time=td)
      write(6,34)
   34 format(1x,'material n, atoms (time=td)')
      write(6,11) n
c material disintegration rates, curies
      write(6,35)
   35 format(1x,'material disintegration rates, curies')
      write(6,11) mrate
c photon source rates, ph/s
      write(6,36)
   36 format(1x,'material/group photon source rates, ph/s')
      write(6,11) srate
c photon dose rates, rem/h
      write(6,37)
   37 format(1x,'material/group photon dose rates, rem/h')
      write(6,11) drate
c photon heat, wt; =ph/s*mev/ph*j/mev where j=wt-s
      do 104 i=1,3
      gheat(i)=gsour(i)*mev(i)*jmev
  104 gheat(4)=gheat(4) + gheat(i)
      write(6,41)
   41 format(1x,'group photon heat, wt')
      write(6,11) gheat
c photon flux, ph/cm**2-s; sum fission product photon fluxes
      do 107 i=1,3
  107 gflux(4)=gflux(4) + gflux(i)
      write(6,42)
   42 format(1x,'group photon flux, ph/cm**2-s')
      write(6,11) gflux
c photon source rate, ph/s; sum fission product photon source
      do 106 i=1,3
  106 gsour(4)=gsour(4) + gsour(i)
      write(6,43)
   43 format(1x,'group photon source rate, ph/s')
      write(6,11) gsour
c photon dose rate, rem/h; sum fission product photon dose
      do 108 i=1,3
  108 gdose(4)=gdose(4) + gdose(i)
      write(6,44)
   44 format(1x,'group photon dose rate, rem/h')
      write(6,11) gdose
```

```
      go to 100
  999 stop
      end
c
      block data
      real (kind=8) dk(13),cap(13)
      common /comdat/dk,cap
c decay constant, s-1
      data dk/7.816d-10,3.005d-6,0.d0,1.066d-8,7.297d-10,4.530d-3,
     x 2.821d-8,6.684d-4,2.174d-8,2.318d-2,1.225d-7,2.285d-7,4.176d-9/
c capture cross section, cm**2
      data cap/1.028d-24,2.740d-24,103.2d-24,107.1d-24,0.1310d-24,
     x 0.d0,1.172d-24,0.d0,0.4550d-24,2*0.d0,3.563d-24,4.275d-24/
      end
c
      subroutine sr90
      real (kind=8) dk(13),cap(13),r(13),n0(13)
      real (kind=8) powden,kg,ppm
      real (kind=8) bu,time,dm,nflux,t(10),c(10),n(10),k
      common r,n0,powden,kg,ppm,bu,time
      common /comdat/dk,cap
      do 101 i=1,10
      t(i)=dfloat(i)*time
      dm=1.0d0-bu*(dfloat(i)-0.5d0)
      nflux=2.952d13*powden/dm
      k=dk(1) + cap(1)*nflux
      if(i.eq.1) c(1)=-r(1)/k
      if(i.ne.1) c(i)=(n(i-1) - r(1)/k)*dexp(k*t(i-1))
  101 n(i)=r(1)/k + c(i)*dexp(-k*t(i))
c     write(6,11) n
   11 format(1x,1p6d12.5)
      n0(1)=n(10)
      return
      end
c
      subroutine cs134
      real (kind=8) dk(13),cap(13),r(13),n0(13)
      real (kind=8) powden,kg,ppm
      real (kind=8) bu,time,dm,nflux,t(10),c(10),n(10),k1,k2,del
      common r,n0,powden,kg,ppm,bu,time
      common /comdat/dk,cap
      do 101 i=1,10
      t(i)=dfloat(i)*time
      dm=1.0d0-bu*(dfloat(i)-0.5d0)
      nflux=2.952d13*powden/dm
      k1=dk(3) + cap(3)*nflux
      k2=dk(4) + cap(4)*nflux
      del=k2-k1
      if(i.eq.1) c(1)=r(3)*(-1.0d0/k2 + 1.0d0/del)
      if(i.ne.1) c(i)=(n(i-1) - r(3)/k2)*dexp(k2*t(i-1)) +
     x r(3)/del*dexp(del*t(i-1))
  101 n(i)=r(3)*(1.0d0/k2 - dexp(-k1*t(i))/del) + c(i)*dexp(-k2*t(i))
c     write(6,11) n
   11 format(1x,1p6d12.5)
      n0(4)=n(10)
      return
      end
```

```
c
      subroutine cs137
      real (kind=8) dk(13),cap(13),r(13),n0(13)
      real (kind=8) powden,kg,ppm
      real (kind=8) bu,time,dm,nflux,t(10),c(10),n(10),k
      common r,n0,powden,kg,ppm,bu,time
      common /comdat/dk,cap
      do 101 i=1,10
      t(i)=dfloat(i)*time
      dm=1.0d0-bu*(dfloat(i)-0.5d0)
      nflux=2.952d13*powden/dm
      k=dk(5) + cap(5)*nflux
      if(i.eq.1) c(1)=-r(5)/k
      if(i.ne.1) c(i)=(n(i-1) - r(5)/k)*dnd=999) ti,power,powden,td,kg,ppm,burnup
c burnup in percent
      burnup=1.0d-2*burnup
c calculate irradiation time and average neutron flux
      time=burnup/(1.25d-3*powden)
      if(ti.eq.0.0d0) ti=time
      dm=1.0d0 - 0.5d0*burnup
      nflux=2.952d13*powden/dm
c write input data plus nflux
      write(6,20)
   20 format('0input: irrad time (d), power (MW), pow den (MW/kgU-235),
     2decay time (y),'/5x,'asby mass (kg), Co-59 (ppm), U-235 burnup (%)
     3, avg flux (n/cm**2-s)')
      write(6,11) ti,power,powden,td,kg,ppm,burnup*1.0d2,nflux
c use ti as a flag for additional printout
      nskip=0
      if(ti.lt.0.0d0) then
       ti=-ti
       nskip=1
      endif
c check if ti is consistent (<1%) with burnup and power density
      if(dabs(ti/time - 1.0d0).ge.1.0d-2) stop 101
c irradiation time (ti) in days and decay time (td) in years
      ti=ti*8.64d4
      td=td*3.1536d7
c initialize arrays
      do 101 i=1,13
  101 r(i)=0.0d0
      do 103 i=1,5
      gsour(i)=0.0d0
      gflux(i)=0.0d0
      gdosexp(k*t(i-1))
  101 n(i)=r(5)/k + c(i)*dexp(-k*t(i))
c     write(6,11) n
   11 format(1x,1p6d12.5)
      n0(5)=n(10)
      return
      end
c
      subroutine ce144
      real (kind=8) dk(13),cap(13),r(13),n0(13)
      real (kind=8) powden,kg,ppm
      real (kind=8) bu,time,dm,nflux,t(10),c(10),n(10),k
      common r,n0,powden,kg,ppm,bu,time
```

```
      common /comdat/dk,cap
      do 101 i=1,10
      t(i)=dfloat(i)*time
      dm=1.0d0-bu*(dfloat(i)-0.5d0)
      nflux=2.952d13*powden/dm
      k=dk(7) + cap(7)*nflux
      if(i.eq.1) c(1)=-r(7)/k
      if(i.ne.1) c(i)=(n(i-1) - r(7)/k)*dexp(k*t(i-1))
  101 n(i)=r(7)/k + c(i)*dexp(-k*t(i))
c     write(6,11) n
   11 format(1x,1p6d12.5)
      n0(7)=n(10)
      return
      end
c
      subroutine ru106
      real (kind=8) dk(13),cap(13),r(13),n0(13)
      real (kind=8) powden,kg,ppm
      real (kind=8) bu,time,dm,nflux,t(10),c(10),n(10),k
      common r,n0,powden,kg,ppm,bu,time
      common /comdat/dk,cap
      do 101 i=1,10
      t(i)=dfloat(i)*time
      dm=1.0d0-bu*(dfloat(i)-0.5d0)
      nflux=2.952d13*powden/dm
      k=dk(9) + cap(9)*nflux
      if(i.eq.1) c(1)=-r(9)/k
      if(i.ne.1) c(i)=(n(i-1) - r(9)/k)*dexp(k*t(i-1))
  101 n(i)=r(9)/k + c(i)*dexp(-k*t(i))
c     write(6,11) n
   11 format(1x,1p6d12.5)
      n0(9)=n(10)
      return
      end
c
      subroutine zr95
      real (kind=8) dk(13),cap(13),r(13),n0(13)
      real (kind=8) powden,kg,ppm
      real (kind=8) bu,time,dm,nflux,t(10),c(10),n(10),k
      common r,n0,powden,kg,ppm,bu,time
      common /comdat/dk,cap
      do 101 i=1,10
      t(i)=dfloat(i)*time
      dm=1.0d0-bu*(dfloat(i)-0.5d0)
      nflux=2.952d13*powden/dm
      k=dk(11) + cap(11)*nflux
      if(i.eq.1) c(1)=-r(11)/k
      if(i.ne.1) c(i)=(n(i-1) - r(11)/k)*dexp(k*t(i-1))
  101 n(i)=r(11)/k + c(i)*dexp(-k*t(i))
c     write(6,11) n
   11 format(1x,1p6d12.5)
      n0(11)=n(10)
      return
      end
c
      subroutine co60
      real (kind=8) dk(13),cap(13),r(13),n0(13)
```

```
      real (kind=8) powden,kg,ppm
      real (kind=8) bu,time,dm,nflux,t(10),c(10),n(10),k1,k2,r59
      common r,n0,powden,kg,ppm,bu,time
      common /comdat/dk,cap
      do 101 i=1,10
      t(i)=dfloat(i)*time
      dm=1.0d0-bu*(dfloat(i)-0.5d0)
      nflux=2.952d13*powden/dm
      k1=       39.67d-24*nflux
      k2=dk(13) + cap(13)*nflux
      r59=1.022d19*kg*ppm*k1
      if(i.eq.1) c(1)=-r59/k2
      if(i.ne.1) c(i)=(n(i-1) - r59/k2)*dexp(k2*t(i-1))
  101 n(i)=r59/k2 + c(i)*dexp(-k2*t(i))
c     write(6,11) n
   11 format(1x,1p6d12.5)
      n0(13)=n(10)
      return
      end
```