The code of the program for modeling the FHN neuron generator on memristors. The code is written in MATLAB.

```
function J = crnt1(U)

% This function defines the current through filament as a function of voltage

% global variable of model T(temperature) t(Time) x (filament state)

global T t w

Fi_el=0.9; % eV potential barrier for electrons in Pool-Frenkel mech
Fi_ion=0.7; % eV potential barrier for ion hopes
q=1.6e-19; % electron charge
k=1.38e-23; % Boltzman factor
T=300;
t=0.01;
A=1e8*exp(1-300/T); % coeff for Ohmic current
B=2.5e21; % coeff for Pool-Frenkel current

if U>=0
   j_lin=A*U;
   j_nonlin=B*U*exp(-(q/(k*T))*(Fi_el-0.03*sqrt(U)));
else
   j_lin=A*abs(U);
   j_nonlin=B*abs(U)*exp(-(q/(k*T))*(Fi_el-0.03*sqrt(abs(U))));
```

```
end
            J=(w*j_lin+(1-w)*j_nonlin)*4*10^(-11);
end

            function dy1 = FHNmod1(t1,y1)

global etta1 epsilon1 al be
dy1 = zeros(3,1);
global T t w

Fi_el=1; % eV potential barrier for electrons in Pool-Frenkel mech
Fi_ion=0.8; % eV potential barrier for ion hopes
q=1.6e-19; % electron charge
k=1.38e-23; % Boltzman factor
T=300;
t=0.01;
A=1e8*exp(1-300/T); % coeff for Ohmic current
B=2.5e21; % coeff for Pool-Frenkel current
Vset = -3;
Vreset = 3;

if y1(1)>=0
   j_lin=A*y1(1);
   j_nonlin=B*y1(1)*exp(-(q/(k*T))*(Fi_el-0.03*sqrt(y1(1))));
else
   j_lin=A*abs(y1(1));
```

```
    j_nonlin=B*abs(y1(1))*exp(-(q/(k*T))*(Fi_el-0.03*sqrt(abs(y1(1)))));
end
            J=(w*j_lin+(1-w)*j_nonlin)*4*10^(-11);
            dy1(1) = 1.138*J-y1(2);
            dy1(2) = epsilon1*(g(y1(1))-y1(2)-etta1);
if y1(1)<Vset
            dy1(3) = 1e13*t*0.01*(exp(q*(-Fi_ion-0.08*y1(1))/(k*T))*(1-(2*y1(3)-1)^(20)));
elseif y1(1)>Vreset
            dy1(3)=0;
else
            dy1(3) = -1e13*t*0.01*(exp(q*(-Fi_ion+0.08*y1(1))/(k*T))*(1-(2*y1(3)-1)^(20)));
end

function G=g(x)
      al=0.5;
      be=1.5;
      if x<0
        G=al*x;
      else
        G=be*x;
      end
    end


clear all; clc;
global etta1 epsilon1 al be w
```

```
etta1=0.48;

epsilon1 = 0.24;

al=0.5;

be=1.5;

w=1;

t0=0.1;

tf=100;

y0=[0.5 1 0];


[T1,Y1]=ode23s('FHNmod1',[t0:0.02:tf],y0,1e-6);


k = 3;

 Uout1 = Y1(:,1)*k;

 Uout2 = Y1(:,2)*k;

 Uout3 = Y1(:,3)*k;


 len = length(t0:0.02:tf);

  for i = 1:len

        J1(i) = crnt1(Uout1(i));

  end

  for i = 1:len

        J2(i) = crnt1(Uout2(i));

  end

  for i = 1:len

        J3(i) = crnt1(Uout3(i));

  end
```

```
figure('NumberTitle', 'off', 'Name', '')
 hold on;
plot(T1,Uout1, 'g-','LineWidth',2)
plot (T1,Uout2, 'r-','LineWidth',2)
plot(T1,Uout3, 'b-','LineWidth',2)
xlabel('Time (ms)');
ylabel('Voltage (V)');
legend('u','v','x')
 hold off;

figure('NumberTitle', 'off', 'Name', '')
 hold on;
plot(Y1(:,1),Y1(:,2), 'k-','LineWidth',3)
xlabel('V');
ylabel('u');
 hold off;

figure('NumberTitle', 'off', 'Name', '')
 hold on;
semilogy(Uout1,J1, 'b-')
semilogy(Uout2,J2, 'r-')
semilogy(Uout3,J3, 'g-')
xlabel('Voltage (V)');
ylabel('Current (A)');
legend('memristor')
```

```
hold off;
```