```
0001  function [sim, outlist, active_state, x_global_kp1, userdata]=state_mainfn(sim, ...
0002                                          inlist, x_global, state, statename, userdata)
0003    printf("defining state %s (#%d) ... userdata(1)=%s\ n", statename, state, userdata(1) );
0004
0005    // define names for the first event in the simulation
0006    events = 0;
0007
0008    // demultiplex x_global
0009    [sim, x_global] = ld_demux(sim, events, vecsize=4, invec=x_global);
0010
0011    // sample data fot output
0012    [sim, outdata1] = ld_constvec(sim, events, vec=[1200]);
0013
0014    select state
0015      case 1 // state 1
0016        // wait 10 simulation steps and then switch to state 2
0017        [sim, active_state] = ld_steps(sim, events, activation_simsteps=[10], values=[-1,2]);
0018        [sim, x_global(1)] = ld_add_ofs(sim, events, x_global(1), 1); // increase counter 1 by 1
0019      case 2 // state 2
0020        // wait 10 simulation steps and then switch to state 3
0021        [sim, active_state] = ld_steps(sim, events, activation_simsteps=[10], values=[-1,3]);
0022        [sim, x_global(2)] = ld_add_ofs(sim, events, x_global(2), 1); // increase counter 2 by 1
0023      case 3 // state 3
0024        // wait 10 simulation steps and then switch to state 1
0025        [sim, active_state] = ld_steps(sim, events, activation_simsteps=[10], values=[-1,1]);
0026        [sim, x_global(3)] = ld_add_ofs(sim, events, x_global(3), 1); // increase counter 3 by 1
0027    end
0028
0029    // multiplex the new global states
0030    [sim, x_global_kp1] = ld_mux(sim, events, vecsize=4, inlist=x_global);
0031
0032    // the user defined output signals of this nested simulation
0033    outlist = list(outdata1);
0034  endfunction
```