```
 1  LIBRARY IEEE;
 2  USE  IEEE.STD_LOGIC_1164.all;
 3  USE  IEEE.STD_LOGIC_UNSIGNED.all;
 4
 5  entity MSS_Call_Center is
 6     Port (clock,resetn,start,boton,BORRAR,Fu,HISTORIAL_LLAMADAS,cont6,Re_oc,Re_des,
    igual_oc,dir_hist_igual,igual_hist1,
 7           REALIZAR_LLAMADA,FINALIZAR_LLAMADA,icuenta0,icuenta1,icuenta2,icuenta3,icuenta4,
    icuenta5,u1,u2,u3,u4,u5,u6,u7,u8,
 8           Fu_ad1,Fu_ad2,Fu_ad3,Fu_ad4,Fu_ad5,Fu_ad6,Fu_ad7,Fu_ad8,iusuario1,iusuario2,
    iusuario3,iusuario4,iusuario5,iusuario6,
 9           iusuario7,iusuario8,iusuario_ant1,iusuario_ant2,iusuario_ant3,iusuario_ant4,
    iusuario_ant5,iusuario_ant6,iusuario_ant7,
10           iusuario_ant8: IN std_logic;
11
12           incrementa,reset_reg,cuenta_a_0,incremento_u1,incremento_u2,incremento_u3,
    incremento_u4,incremento_u5,incremento_u6,
13           incremento_u7,incremento_u8,we_u1,we_u2,we_u3,we_u4,we_u5,we_u6,we_u7,we_u8,
    reset_oc,OCUPADO,reset_hist,disp_select,rhist_u1,
14           rhist_u2,rhist_u3,rhist_u4,rhist_u5,rhist_u6,rhist_u7,rhist_u8,ihist_u1,ihist_u2
    ,ihist_u3,ihist_u4,ihist_u5,ihist_u6,ihist_u7,
15           ihist_u8,led1,led2,led3,led4,led5,led6,led7,led8,clock_ff_d: OUT std_logic;
16
17           selec_u1,selec_u2,selec_u3,selec_u4,selec_u5,selec_u6,selec_u7,selec_u8: OUT
    std_logic_vector (1 downto 0);
18
19           enable_reg: OUT std_logic_vector (5 downto 0));
20
21  end MSS_Call_Center;
22
23  ARCHITECTURE sol OF MSS_Call_Center IS
24
25  Type estado is (Ta,Tb,Tc,Td,Te,Tf,Tg,Th,Ti,Tj,Tk,Tl,Tm,Tr,Ts,Tt,Tu,Tv,Tw,Tx,Ty,T_oc,
    T_oc_apagado,T_hist,T_hist1,T_hi,T1);--falta revisar qué estados cambié
26  Signal y: estado;
27  Begin
28     Process (clock,resetn)
29     Begin
30        if resetn='0' then y<=Ta;
31        elsif (clock'event and clock='1') then
32           case y is
33              when Ta => if start='1' then y<=Tb; else y<=Ta; end if;
34              when Tb => if start='1' then y<=Tb; else y<=Tc; end if;
35              when Tc => if start='1' then y<=Td;
36                         elsif boton='1' then y<=Tf;
37                         elsif (BORRAR='1') then y<=Tr;
38                         elsif Fu='1' then y<=Tk;
39                         elsif HISTORIAL_LLAMADAS='1' then y<=T_hist;
40                         else y<=Tc; end if;
41              when Td => if start='1' then y<=Td; else y<=Ta; end if;
42              when Tf => y<=Te;
43              when Te => if boton='1' then y<=Te; else y<=Tg; end if;
44              when Tg => if cont6='0' then y<=Th; else y<=Ti; end if;
45              when Th => y<=Tc;--Por el incremento, es una salida
46              when Ti => if Re_oc='1' then y<=T_oc;
47                         elsif Re_des='1' then y<=Tj;
48                         elsif BORRAR='1' then y<=Tr;
49                         else y<=Ti; end if;
50
51              when T_oc => if (igual_oc='1') then y<=T_oc_apagado; else y<=T_oc; end if;
    --estado de transicion para poder apagar la señal de ocupado
52              when T_oc_apagado => y<=Tc;
53              when T_hist => if HISTORIAL_LLAMADAS='1' then y<=T_hist; else y<=T_hist1; end
     if;
54              when T_hist1 => if dir_hist_igual='1' then y<=Tc;--se sobre entiende que
    sólo un usuario está seleccionado antes de que presione el botón del historial
55                              else y<=T_hi;end if;
56              when T_hi => if igual_hist1='1' then y<=T1;
57                           else y<=T_hi; end if;
58              when T1 => y<=T_hist1;
59
60              when Tj => if REALIZAR_LLAMADA='1' then y<=Tj; else y<=Tl; end if;
61              when Tl => y<=Ts;--Porque hay una salida (que el led se encienda)
62
63              when Tk => if FINALIZAR_LLAMADA='1' then y<=Tk; else y<=Tm; end if;
64
65              when Tm => y<=Tt; --Porque hay una salida (que el led se apague)
66
67              when Tr => if BORRAR='1' then y<=Tr; else y<=Ty; end if;
68              when Ts => y<=Tu;
69              when Tu => y<=Tw;
```

```
70                when Tw => y<=Tc;
71                when Tt => y<=Tv;
72                when Tv => y<=Tx;
73                when Tx => y<=Tc;
74                when Ty => y<=Tc;
75
76             end case;
77          end if;
78       end Process;
79       Process(y)
80       Begin
81          case y is
82                when Ta => incrementa<='0';reset_reg<='1';cuenta_a_0<='0';--Condicion
    inicial, si en un estado cambia esa condicion
83                           incremento_u1<='0';incremento_u2<='0';incremento_u3<='0';
    incremento_u4<='0';--solo lo hace en ese estado, al momento
84                           incremento_u5<='0';incremento_u6<='0';incremento_u7<='0';
    incremento_u8<='0';we_u1<='0';--de pasar a otro estado todo
85                           we_u2<='0';we_u3<='0';we_u4<='0';we_u5<='0';we_u6<='0';we_u7<='0';
    we_u8<='0';--vuelve a la condicion inicial
86                           selec_u1<="00";selec_u2<="00";selec_u3<="00";selec_u4<="00";
    selec_u5<="00";selec_u6<="00";selec_u7<="00";selec_u8<="00";
87                           reset_oc<='1'; OCUPADO<='0';reset_hist<='1';disp_select<='0';
88                when Tb => incrementa<='0';reset_reg<='1';cuenta_a_0<='0';--Condicion
    inicial, si en un estado cambia esa condicion
89                           incremento_u1<='0';incremento_u2<='0';incremento_u3<='0';
    incremento_u4<='0';--solo lo hace en ese estado, al momento
90                           incremento_u5<='0';incremento_u6<='0';incremento_u7<='0';
    incremento_u8<='0';we_u1<='0';--de pasar a otro estado todo
91                           we_u2<='0';we_u3<='0';we_u4<='0';we_u5<='0';we_u6<='0';we_u7<='0';
    we_u8<='0';--vuelve a la condicion inicial
92                           selec_u1<="00";selec_u2<="00";selec_u3<="00";selec_u4<="00";
    selec_u5<="00";selec_u6<="00";selec_u7<="00";selec_u8<="00";
93                           reset_oc<='1'; OCUPADO<='0';reset_hist<='1';disp_select<='0';
94                           rhist_u1<='1';rhist_u2<='1';rhist_u3<='1';rhist_u4<='1';rhist_u5<=
    '1';rhist_u6<='1';rhist_u7<='1';rhist_u8<='1';
95                           ihist_u1<='0';ihist_u2<='0';ihist_u3<='0';ihist_u4<='0';ihist_u5<=
    '0';ihist_u6<='0';ihist_u7<='0';ihist_u8<='0';
96                when Tc => incrementa<='0';reset_reg<='1';cuenta_a_0<='0';--Condicion
    inicial, si en un estado cambia esa condicion
97                           incremento_u1<='0';incremento_u2<='0';incremento_u3<='0';
    incremento_u4<='0';--solo lo hace en ese estado, al momento
98                           incremento_u5<='0';incremento_u6<='0';incremento_u7<='0';
    incremento_u8<='0';we_u1<='0';--de pasar a otro estado todo
99                           we_u2<='0';we_u3<='0';we_u4<='0';we_u5<='0';we_u6<='0';we_u7<='0';
    we_u8<='0';--vuelve a la condicion inicial
100                          selec_u1<="00";selec_u2<="00";selec_u3<="00";selec_u4<="00";
    selec_u5<="00";selec_u6<="00";selec_u7<="00";selec_u8<="00";
101                          reset_oc<='1'; OCUPADO<='0';reset_hist<='1';disp_select<='0';
102                          rhist_u1<='1';rhist_u2<='1';rhist_u3<='1';rhist_u4<='1';rhist_u5<=
    '1';rhist_u6<='1';rhist_u7<='1';rhist_u8<='1';
103                          ihist_u1<='0';ihist_u2<='0';ihist_u3<='0';ihist_u4<='0';ihist_u5<=
    '0';ihist_u6<='0';ihist_u7<='0';ihist_u8<='0';
104               when Td =>
105               when Tf => if icuenta0='1' then enable_reg<="000001";
106                          elsif icuenta1='1' then enable_reg<="000010";
107                          elsif icuenta2='1' then enable_reg<="000100";
108                          elsif icuenta3='1' then enable_reg<="001000";
109                          elsif icuenta4='1' then enable_reg<="010000";
110                          elsif icuenta5='1' then enable_reg<="100000";end if;
111               when Te => enable_reg<="000000";
112               when Tg =>
113               when Th => incrementa<='1';
114               when Ti =>
115
116               when T_oc => reset_oc<='0'; OCUPADO<='1';
117               when T_oc_apagado => OCUPADO<='0';
118               when T_hist => selec_u1<="10";selec_u2<="10";selec_u3<="10";selec_u4<="10";
    selec_u5<="10";selec_u6<="10";selec_u7<="10";selec_u8<="10";
119                              ihist_u1<='0';ihist_u2<='0';ihist_u3<='0';ihist_u4<='0';
    ihist_u5<='0';ihist_u6<='0';ihist_u7<='0';ihist_u8<='0';
120                              rhist_u1<='0';rhist_u2<='0';rhist_u3<='0';rhist_u1<='0';
    rhist_u5<='0';rhist_u6<='0';rhist_u7<='0';rhist_u8<='0';--solamente en estos estados el
    contador para mostrar el historial
121               when T_hist1 => selec_u1<="10";selec_u2<="10";selec_u3<="10";selec_u4<="10";
    selec_u5<="10";selec_u6<="10";selec_u7<="10";selec_u8<="10";--debe funcionar por eso es
    '0', ya que con '1' el reset se activa
122                          rhist_u1<='0';rhist_u2<='0';rhist_u3<='0';rhist_u1<='0';rhist_u5<=
    '0';rhist_u6<='0';rhist_u7<='0';rhist_u8<='0';
123                          ihist_u1<='0';ihist_u2<='0';ihist_u3<='0';ihist_u4<='0';ihist_u5<=
    '0';ihist_u6<='0';ihist_u7<='0';ihist_u8<='0';disp_select<='1';
```

```
124                 when T_hi => selec_u1<="10";selec_u2<="10";selec_u3<="10";selec_u4<="10";
    selec_u5<="10";selec_u6<="10";selec_u7<="10";selec_u8<="10";
125                              rhist_u1<='0';rhist_u2<='0';rhist_u3<='0';rhist_u1<='0';rhist_u5<=
    '0';rhist_u6<='0';rhist_u7<='0';rhist_u8<='0';
126                              selec_u1<="10";selec_u2<="10";selec_u3<="10";selec_u4<="10";
    selec_u5<="10";selec_u6<="10";selec_u7<="10";selec_u8<="10";
127                              ihist_u1<='0';ihist_u2<='0';ihist_u3<='0';ihist_u4<='0';ihist_u5<=
    '0';ihist_u6<='0';ihist_u7<='0';ihist_u8<='0';disp_select<='1';
128                              reset_hist<='0';--empieza la cuenta hasta cuatro, el numero ya
    está seleccionado, se sobre entiende que el usuario ya esta seleccionado, y el adress
    empieza en 0
129                 when T1 => if u1='1' then ihist_u1<='1';--incremento para cada adress de la
    ram
130                              elsif u2='1' then ihist_u2<='1';
131                              elsif u3='1' then ihist_u3<='1';
132                              elsif u4='1' then ihist_u4<='1';
133                              elsif u5='1' then ihist_u5<='1';
134                              elsif u6='1' then ihist_u6<='1';
135                              elsif u7='1' then ihist_u7<='1';
136                              elsif u8='1' then ihist_u8<='1';end if;disp_select<='1';
137
138                 when Tj => clock_ff_d<='0';
139                              if (u1='1') then we_u1<='1';
140                              elsif (u2='1') then we_u2<='1';
141                              elsif (u3='1') then we_u3<='1';
142                              elsif (u4='1') then we_u4<='1';
143                              elsif (u5='1') then we_u5<='1';
144                              elsif (u6='1') then we_u6<='1';
145                              elsif (u7='1') then we_u7<='1';
146                              elsif (u8='1') then we_u8<='1';end if;
147                 when Tk => if Fu_ad1='1' then incremento_u1<='1';--puedo omitir
    "FINALIZAR_LLAMADA='1'" porque se sobre entiende
148                              elsif Fu_ad2='1' then incremento_u2<='1';--Solo cuando finalizo
    la llamada debo avanzar al siguiente adress de la ram
149                              elsif Fu_ad3='1' then incremento_u3<='1';--Podria yo cambiar y
    establecer 2 clocks, uno para encendido y otro para apagado
150                              elsif Fu_ad4='1' then incremento_u4<='1';--con eso evitaría que
    salgan las 'x's . La solucion es cuando un led enciende
151                              elsif Fu_ad5='1' then incremento_u5<='1';--escribir que los demas
    sean cero, pero con dos clocks evito que alguno que
152                              elsif Fu_ad6='1' then incremento_u6<='1';--esté encendido se apague
153                              elsif Fu_ad7='1' then incremento_u7<='1';
154                              elsif Fu_ad8='1' then incremento_u8<='1';end if;clock_ff_d<='0';
155                 when Tl => if (u1='1') then led1<='1';--condicion para cada usuario encender
156                              elsif (u2='1') then led2<='1';
157                              elsif (u3='1') then led3<='1';
158                              elsif (u4='1') then led4<='1';
159                              elsif (u5='1') then led5<='1';
160                              elsif (u6='1') then led6<='1';
161                              elsif (u7='1') then led7<='1';
162                              elsif (u8='1') then led8<='1'; end if;clock_ff_d<='0';
163                 when Tm => if (u1='1') then led1<='0';--condicion para cada usuario apagar
164                              elsif (u2='1') then led2<='0';
165                              elsif (u3='1') then led3<='0';
166                              elsif (u4='1') then led4<='0';
167                              elsif (u5='1') then led5<='0';
168                              elsif (u6='1') then led6<='0';
169                              elsif (u7='1') then led7<='0';
170                              elsif (u8='1') then led8<='0'; end if;clock_ff_d<='0';
171                 when Tr =>
172                 when Ts => if (u1='1') then led1<='1';--Para que mantenga el valor anterior
    y el clock del flipflop funcione
173                              elsif (u2='1') then led2<='1';
174                              elsif (u3='1') then led3<='1';
175                              elsif (u4='1') then led4<='1';
176                              elsif (u5='1') then led5<='1';
177                              elsif (u6='1') then led6<='1';
178                              elsif (u7='1') then led7<='1';
179                              elsif (u8='1') then led8<='1'; end if;clock_ff_d<='1';
180                 when Tt => if (u1='1') then led1<='0';--Para que mantenga el valor anterior
    y el clock del flipflop funcione
181                              elsif (u2='1') then led2<='0';
182                              elsif (u3='1') then led3<='0';
183                              elsif (u4='1') then led4<='0';
184                              elsif (u5='1') then led5<='0';
185                              elsif (u6='1') then led6<='0';
186                              elsif (u7='1') then led7<='0';
187                              elsif (u8='1') then led8<='0'; end if;clock_ff_d<='1';
188                              selec_u1<="01";selec_u2<="01";selec_u3<="01";selec_u4<="01";
    selec_u5<="01";selec_u6<="01";selec_u7<="01";selec_u8<="01";
189                 when Tu => if (iusuario1='1') then led1<='1';--condicion para cada usuario
```

```
      para encender el otro led
190                           elsif (iusuario2='1') then led2<='1';
191                           elsif (iusuario3='1') then led3<='1';
192                           elsif (iusuario4='1') then led4<='1';
193                           elsif (iusuario5='1') then led5<='1';
194                           elsif (iusuario6='1') then led6<='1';
195                           elsif (iusuario7='1') then led7<='1';
196                           elsif (iusuario8='1') then led8<='1'; end if;clock_ff_d<='0';
197                when Tv => clock_ff_d<='0';--condicion para cada usuario apagar el otro led
198                           selec_u1<="01";selec_u2<="01";selec_u3<="01";selec_u4<="01";
      selec_u5<="01";selec_u6<="01";selec_u7<="01";selec_u8<="01";
199                           if iusuario_ant1='1' then led1<='0';
200                           elsif iusuario_ant2='1' then led2<='0';
201                           elsif iusuario_ant3='1' then led3<='0';
202                           elsif iusuario_ant4='1' then led4<='0';
203                           elsif iusuario_ant5='1' then led5<='0';
204                           elsif iusuario_ant6='1' then led6<='0';
205                           elsif iusuario_ant7='1' then led7<='0';
206                           elsif iusuario_ant8='1' then led8<='0';end if;
207                when Tw => if (iusuario1='1') then led1<='1';--Para que mantenga el valor
      del otro led
208                           elsif (iusuario2='1') then led2<='1';
209                           elsif (iusuario3='1') then led3<='1';
210                           elsif (iusuario4='1') then led4<='1';
211                           elsif (iusuario5='1') then led5<='1';
212                           elsif (iusuario6='1') then led6<='1';
213                           elsif (iusuario7='1') then led7<='1';
214                           elsif (iusuario8='1') then led8<='1'; end if;clock_ff_d<='1';
      reset_reg<='0';cuenta_a_0<='1';
215                when Tx => clock_ff_d<='1';--Para que se mantenga el valor del otro led
216                           selec_u1<="01";selec_u2<="01";selec_u3<="01";selec_u4<="01";
      selec_u5<="01";selec_u6<="01";selec_u7<="01";selec_u8<="01";
217                           if iusuario_ant1='1' then led1<='0';
218                           elsif iusuario_ant2='1' then led2<='0';
219                           elsif iusuario_ant3='1' then led3<='0';
220                           elsif iusuario_ant4='1' then led4<='0';
221                           elsif iusuario_ant5='1' then led5<='0';
222                           elsif iusuario_ant6='1' then led6<='0';
223                           elsif iusuario_ant7='1' then led7<='0';
224                           elsif iusuario_ant8='1' then led8<='0';end if;
225                when Ty => reset_reg<='0';cuenta_a_0<='1';
226           end case;
227       end Process;
228   end sol;
```