

```

1  /*-----*\
2  Use :
3  \*-----*/
4  void FCM_Main()
5  {
6
7  // Name: Loop, Type: Loop: While 1
8  while (1)
9  {
10
11 // Name: Call Component Macro, Type: Call Component Macro: led_7seg_quad1::ShowDigit(3, unita, 0)
12 FCD_ofca1_led_7seg_quad1_ShowDigit(3, FCV_UNITA, 0);
13
14 // Name: Call Component Macro, Type: Call Component Macro: led_7seg_quad1::ShowDigit(2, decine, 0)
15 FCD_ofca1_led_7seg_quad1_ShowDigit(2, FCV_DECINE, 0);
16
17 // Name: Call Component Macro, Type: Call Component Macro: led_7seg_quad1::ShowDigit(1, centinaia, 0)
18 FCD_ofca1_led_7seg_quad1_ShowDigit(1, FCV_CENTINAIA, 0);
19
20 // Name: Call Component Macro, Type: Call Component Macro: led_7seg_quad1::ShowDigit(0, migliaia, 1)
21 FCD_ofca1_led_7seg_quad1_ShowDigit(0, FCV_MIGLIAIA, 1);
22
23 // Name: Call Component Macro, Type: Call Component Macro: led_7seg_quad2::ShowDigit(3, diecimila, 0)
24 FCD_ofca2_led_7seg_quad2_ShowDigit(3, FCV_DIECIMILA, 0);
25
26 // Name: Call Component Macro, Type: Call Component Macro: led_7seg_quad2::ShowDigit(2, centomila, 0)
27 FCD_ofca2_led_7seg_quad2_ShowDigit(2, FCV_CENTOMILA, 0);
28
29 // Name: Call Component Macro, Type: Call Component Macro: led_7seg_quad2::ShowDigit(1, mega, 1)
30 FCD_ofca2_led_7seg_quad2_ShowDigit(1, FCV_MEGA, 1);
31
32 // Name: Call Component Macro, Type: Call Component Macro: led_7seg_quad2::ShowDigit(0, diecimega, 0)
33 FCD_ofca2_led_7seg_quad2_ShowDigit(0, FCV_DIECIMEGA, 0);
34
35 // Name: Calculation, Type: Calculation:
36 // unita = unita + 1
37 // unita = unita % 10
38 FCV_UNITA = FCV_UNITA + 1;
39 FCV_UNITA = FCV_UNITA % 10;
40
41 // Name: Decision, Type: Decision: unita = 0?
42 if (FCV_UNITA == 0)
43 {
44
45 // Name: Calculation, Type: Calculation:
46 // decine = decine + 1
47 // decine = decine % 10
48 FCV_DECINE = FCV_DECINE + 1;
49 FCV_DECINE = FCV_DECINE % 10;
50

```

```

51 // Name: Decision, Type: Decision: decine = 0?
52 if (FCV_DECINE == 0)
53 {
54
55 // Name: Calculation, Type: Calculation:
56 // centinaia = centinaia + 1
57 // centinaia = centinaia % 10
58 FCV_CENTINAIA = FCV_CENTINAIA + 1;
59 FCV_CENTINAIA = FCV_CENTINAIA % 10;
60
61 // Name: Decision, Type: Decision: centinaia = 0?
62 if (FCV_CENTINAIA == 0)
63 {
64
65 // Name: Calculation, Type: Calculation:
66 // migliaia = migliaia + 1
67 // migliaia = migliaia % 10
68 FCV_MIGLIAIA = FCV_MIGLIAIA + 1;
69 FCV_MIGLIAIA = FCV_MIGLIAIA % 10;
70
71 // Name: Decision, Type: Decision: migliaia = 0?
72 if (FCV_MIGLIAIA == 0)
73 {
74
75 // Name: Calculation, Type: Calculation:
76 // diecimila = diecimila + 1
77 // diecimila = diecimila % 10
78 FCV_DIECIMILA = FCV_DIECIMILA + 1;
79 FCV_DIECIMILA = FCV_DIECIMILA % 10;
80
81 // Name: Decision, Type: Decision: diecimila = 0?
82 if (FCV_DIECIMILA == 0)
83 {
84
85 // Name: Calculation, Type: Calculation:
86 // centomila = centomila + 1
87 // centomila = centomila % 10
88 FCV_CENTOMILA = FCV_CENTOMILA + 1;
89 FCV_CENTOMILA = FCV_CENTOMILA % 10;
90
91 // Name: Decision, Type: Decision: centomila = 0?
92 if (FCV_CENTOMILA == 0)
93 {
94
95 // Name: Calculation, Type: Calculation:
96 // mega = mega + 1
97 // mega = mega % 10
98 FCV_MEGA = FCV_MEGA + 1;
99 FCV_MEGA = FCV_MEGA % 10;
100

```

```

101 // Name: Decision, Type: Decision: mega = 0?
102 if (FCV_MEGA == 0)
103 {
104
105 // Name: Calculation, Type: Calculation:
106 // diecimega = diecimega + 1
107 // diecimega = diecimega % 10
108 FCV_DIECIMEGA = FCV_DIECIMEGA + 1;
109 FCV_DIECIMEGA = FCV_DIECIMEGA % 10;
110
111 // } else {
112 // }
113 // } else {
114 // }
115 // } else {
116 // }
117 // } else {
118 // }
119 // } else {
120 // }
121 // } else {
122 // }
123 // } else {
124 // }
125 // } else {
126 // }
127 // } else {
128 // }
129 // } else {
130 // }
131 // } else {
132 // }
133 // } else {
134 // }
135 // } else {
136 // }
137 // } else {
138 // }
139 // } else {
140 // }
141 // } else {
142 // }
143 // } else {

```