Basel Mohamed

asu

lab2

1. set state= NORMAL;

Question 1

Full code

1. set state1=TIME;
2. set m=0,h=0, D=1,M=1, Y=2000.
3. For each input do
4. switch(state)
5. Case NORMAL: {
6. If (input ==c) state=UPDATE
7. If (input ==b) state=ALARM
8. If (input ==a)
9. If (state1==TIME)
10. state1=DATE
11. else
12. state1=TIME
13. }
14. Case UPDATE:
15. {
16. int count =0;
17. while (count!=5){
18. if(input==a){
19. count++;
20. }
21. if (input==b){
22. if (count==0){
23. if (m60!=59){
24. m60++;
25. }
26. elseif (m60==59){
27. m60=0;
28. }
29. }
30. if (count==1){
31. if (h!=23){
32. h24++;
33. }
34. elseif (h==23){
35. h24=0;
36. }
37. }
38. if (count==2){
39. if (D31!=31&&(M12!=2||M12!=4||M12!=6||M12!=9||M12!=11)){
40. D31++;
41. }
42. elseif (D31==31){
43. D31=1;
44. }
45. if (M12==2&&D31!=28){
46. D31++;
47. }
48. elseif(M12==2&&D31==28){
49. D31=1;
50. }
51. if ((M12==4||M12==6||M12==9||M12==12)&&D31!=30){
52. D31++;
53. }
54. else if (D31==30){
55. D31=1;
56. }
57. }
58. if (count==3){
59. if (M12!=12){
60. M12++;
61. }
62. else if (M12==12){
63. M12=1;
64. }
65. if (count==4){
66. Y++;
67. }
68. }
69. if (input==d ){
70. state=normal;
71. }
72. }
73. if (count==5){
74. state=normal;
75. }
76. }
77. Case ALARM: {
78. int count=0;
79. while (count!=2&&input!=D){
80. if (input==b){
81. count++;
82. }
83. else if (input==c){
84. if (count=0){
85. if (mA60==60){
86. mA60=1
87. }
88. else
89. mA60++;
90. }
91. `if (count=1){
92. if (hA24==23){
93. hA24=0
94. }
95. else
96. hA24++;
97. }
98. if (count==2){
99. state=normal
100. }
101. }
102. if (input==d){
103. state=normal;
104. }
105. }
106. if (input==a){
107. ring ()
108. }
109. }
110. DisplayDate() {print Y+"-"+M+"-"+D}
111. DisplayTIME() {print h+":"+m}A

I have added if conditions to make sure that months that have 31 day and months who have 30 day in case of update or set alarm does not pass limits

And if next step button if it repeated more than predicted times it return you to the normal state

Question two

To make test cases to make sure we got the highest percent of branch coverage I have designed 5 test cases

t.c1 {c,b,a,b,a,b,b,a,b,a,b,a}

this test case is to test every branch in update state even when we passed the limited count of next step state

t.c2 {c,b,a,b,a,b,d}

this test case is to test of case is to test if user want to force stop updating clock {the user stops in day step }

t.c3 {a,a }

test case is used to branch cover normal state

t.s4 {b,b,b,b,b,b,b,c,b,b,c }

in alarm I used variable b as input to inc hours and minutes according to cout and c to get next step ( minute -> HOUR )till it finish next step phase (count>=2)

t.c5 {b,b,b,b,b,c,b,d}

twst case is used to force stop alarm display and return to normal state

t.c6 {b,b,b,b,b,b,b,c,b,b, ,a,d}

it ring before it return to normal state