```
PROGRAM MAIN
 1
           iCycleCount : INT;
           cycleTrigger : Tc2 Standard . R TRIG;
 4
           shiftRegister : WORD := 0;
           shiftInput : BOOL;
 8
 9
           coverFrontEdge : Tc2_Standard . R_TRIG;
10
           coverBackEdge : Tc2 Standard . F TRIG;
11
12
           camFrontEdge : Tc2_Standard . R_TRIG;
           camBackEdge : Tc2_Standard . F_TRIG ;
13
14
15
           state : StateType := StateType . S0_SHEET_NOT_EXPECTED;
           nextState : StateType := StateType . S0_SHEET_NOT_EXPECTED ;
16
17
           first : BOOL := TRUE;
18
19
          resetTimer : TON;
20
           fault : BOOL := FALSE ;
21
22
           flasher : TON;
23
24
           i : INT := 0;
25
       END VAR
26
       // Every cycle we need to see if there is a cover sheet being loaded
       // It takes 3 cycles for a cover sheet to get to the scorer
       // If there is a cover sheet in the scorer we need to check
 3
 4
                       - that there is a sheet there during the first pulse
 5
                       - that there isn't a sheet present during the second pulse
       // At all other times we shouldn't see a sheet in the scorer for either
 6
       pulse
 7
8
       Global . bCoverSheetDetected := NOT Global . bCoverFeedNoSheet;
9
10
       cycleTrigger (CLK := Global . bCycleTrigger);
       coverFrontEdge (CLK := Global . bScorerSheetDetected );
11
       coverBackEdge (CLK := Global . bScorerSheetDetected );
       camFrontEdge (CLK := Global . bScorerCam);
13
       camBackEdge (CLK := Global . bScorerCam);
14
15
       resetTimer (IN := fault, PT := T#1S);
17
       flasher (IN := NOT flasher . Q , PT := T#250MS);
18
19
       IF cycleTrigger . Q THEN
20
           iCycleCount := iCycleCount + 1;
           shiftRegister := SHL (shiftRegister, 1);
21
22
           shiftRegister . 0 := Global . bCoverSheetDetected;
23
       END_IF ;
24
25
       CASE state OF
26
           StateType . S0_SHEET_NOT_EXPECTED :
27
               {f IF} first {f THEN}
28
                   fault := FALSE;
```

```
END IF
30
31
               IF shiftRegister . 1 THEN
32
                   nextState := StateType . S1 SHEET SEARCH;
33
                   ADSLOGSTR (msgCtrlMask := ADSLOG MSGTYPE STRING, msgFmtStr :=
        'S0 -> S1', strArg := '');
               ELSIF coverFrontEdge . Q AND NOT shiftRegister . 0 THEN
34
35
                   nextState := StateType . S8 FAULT;
36
                   ADSLOGSTR (msgCtrlMask := ADSLOG MSGTYPE STRING , msgFmtStr :=
        'S0 -> S8', strArg := '');
37
               END_IF
38
39
           StateType . S1 SHEET SEARCH :
40
               IF camFrontEdge . Q THEN
41
                   {\tt IF} Global . bScorerSheetDetected {\tt THEN}
42
                       nextState := StateType . S2 BREAK SEARCH ;
43
                       ADSLOGSTR (msgCtrlMask := ADSLOG MSGTYPE STRING , msgFmtStr
         := 'S1 -> S2', strArg := '');
44
                   ELSE
45
                       nextState := StateType . S8 FAULT;
46
                      ADSLOGSTR (msgCtrlMask := ADSLOG MSGTYPE STRING , msgFmtStr
        := 'S1 -> S8', strArg := '');
                   END IF
48
               END IF
49
50
           StateType . S2 BREAK SEARCH :
51
               IF camFrontEdge . Q THEN
52
                   IF NOT Global . bScorerSheetDetected THEN
53
                       nextState := StateType . S3 BREAK COMPLETE;
54
                       ADSLOGSTR (msgCtrlMask := ADSLOG MSGTYPE STRING , msgFmtStr
        := 'S2 -> S3', strArg := '');
55
                   ELSE
56
                       nextState := StateType . S8_FAULT ;
57
                       ADSLOGSTR (msgCtrlMask := ADSLOG MSGTYPE STRING , msgFmtStr
         := 'S2 -> S8', strArg := '');
58
                   END IF
59
               END_IF
60
61
           StateType . S3 BREAK COMPLETE:
62
               IF camBackEdge . Q THEN
63
                   IF NOT Global . bScorerSheetDetected THEN
64
                       nextState := StateType . S4 CYCLE COMPLETE;
65
                       ADSLOGSTR (msgCtrlMask := ADSLOG MSGTYPE STRING , msgFmtStr
         := 'S3 -> S4', strArg := '');
66
                   ELSE
67
                       nextState := StateType . S8 FAULT;
68
                       ADSLOGSTR (msgCtrlMask := ADSLOG MSGTYPE STRING , msgFmtStr
         := 'S3 -> S8', strArg := '');
69
                   END IF
               END IF
70
71
72
           StateType . S4 CYCLE COMPLETE:
                IF cycleTrigger . Q THEN
73
74
                   nextState := StateType . SO SHEET NOT EXPECTED;
75
                   ADSLOGSTR (msgCtrlMask := ADSLOG MSGTYPE STRING , msgFmtStr :=
       'S4 -> S0', strArg := '');
```

```
END_IF
 77
 78
            StateType . S8 FAULT :
79
               Global . bMisfeedAlarm := TRUE;
80
                fault := TRUE;
81
                IF flasher . Q THEN
82
                   Global . bMisfeedAlarm2 := TRUE;
83
84
                   Global . bMisfeedAlarm2 := FALSE;
85
                END_IF
86
                IF resetTimer . Q THEN
87
                   Global . bMisfeedAlarm := FALSE;
88
                    Global . bMisfeedAlarm2 := FALSE;
89
                   nextState := StateType . S4_CYCLE_COMPLETE;
                   ADSLOGSTR (msgCtrlMask := ADSLOG_MSGTYPE_STRING , msgFmtStr :=
90
        'S8 -> S4', strArg := '');
91
                END_IF
92
        END_CASE
93
 94
        IF state <> nextState THEN
95
            first := TRUE;
96
            state := nextState;
 97
            FOR i := 32 TO 2 BY -1 DO
98
               Global . stateHistory [ i ] := Global . stateHistory [ i - 1 ];
99
            END FOR
100
            Global . stateHistory [ 1 ] := state;
101
        ELSE
102
            first := FALSE;
103
        END_IF
104
```