```
PROGRAM  MAIN
VAR
    iCycleCount  :  INT ;
    cycleTrigger  :  Tc2_Standard . R_TRIG ;

    shiftRegister  :  WORD  :=  0 ;
    shiftInput  :  BOOL ;

    coverFrontEdge  :  Tc2_Standard . R_TRIG ;
    coverBackEdge  :  Tc2_Standard . F_TRIG ;

    camFrontEdge  :  Tc2_Standard . R_TRIG ;
    camBackEdge  :  Tc2_Standard . F_TRIG ;

    state  :  StateType  :=  StateType . S0_SHEET_NOT_EXPECTED ;
    nextState  :  StateType  :=  StateType . S0_SHEET_NOT_EXPECTED ;
    first  :  BOOL  :=  TRUE ;

    resetTimer  :  TON ;
    fault :  BOOL  :=  FALSE ;

    flasher  :  TON ;

    i  :  INT  :=  0 ;
END_VAR

```

```
// 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
//              - that there is a sheet there during the first pulse
//              - 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 pulse

Global . bCoverSheetDetected  :=  NOT  Global . bCoverFeedNoSheet ;

cycleTrigger ( CLK  :=  Global . bCycleTrigger ) ;
coverFrontEdge ( CLK  :=  Global . bScorerSheetDetected ) ;
coverBackEdge ( CLK  :=  Global . bScorerSheetDetected ) ;
camFrontEdge ( CLK  :=  Global . bScorerCam ) ;
camBackEdge ( CLK  :=  Global . bScorerCam ) ;

resetTimer ( IN  :=  fault ,  PT  :=  T#1S ) ;
flasher ( IN  :=  NOT  flasher . Q ,  PT  :=  T#250MS ) ;

IF  cycleTrigger . Q  THEN
    iCycleCount  :=  iCycleCount  +  1 ;
    shiftRegister  :=  SHL ( shiftRegister ,  1 ) ;
    shiftRegister . 0  :=  Global . bCoverSheetDetected ;
END_IF ;

CASE  state  OF
    StateType . S0_SHEET_NOT_EXPECTED :
        IF  first  THEN
            fault  :=  FALSE ;
        END_IF

        IF shiftRegister . 1  THEN
            nextState  :=  StateType . S1_SHEET_SEARCH ;
            ADSLOGSTR ( msgCtrlMask  :=  ADSLOG_MSGTYPE_STRING ,  msgFmtStr  := 'S0 -> S1' ,  strArg  :=  '' ) ;
        ELSIF coverFrontEdge . Q  AND  NOT  shiftRegister . 0  THEN
            nextState  :=  StateType . S8_FAULT ;
            ADSLOGSTR ( msgCtrlMask  :=  ADSLOG_MSGTYPE_STRING ,  msgFmtStr  := 'S0 -> S8' ,  strArg  :=  '' ) ;
        END_IF

    StateType . S1_SHEET_SEARCH :
        IF camFrontEdge . Q  THEN
            IF Global . bScorerSheetDetected  THEN
                nextState  :=  StateType . S2_BREAK_SEARCH ;
                ADSLOGSTR ( msgCtrlMask  :=  ADSLOG_MSGTYPE_STRING ,  msgFmtStr  :=  'S1 -> S2' ,  strArg  :=  '' ) ;
            ELSE
                nextState  :=  StateType . S8_FAULT ;
                ADSLOGSTR ( msgCtrlMask  :=  ADSLOG_MSGTYPE_STRING ,  msgFmtStr  :=  'S1 -> S8' ,  strArg  :=  '' ) ;
            END_IF
        END_IF

    StateType . S2_BREAK_SEARCH :
        IF camFrontEdge . Q  THEN
            IF NOT  Global . bScorerSheetDetected  THEN
                nextState  :=  StateType . S3_BREAK_COMPLETE ;
                ADSLOGSTR ( msgCtrlMask  :=  ADSLOG_MSGTYPE_STRING ,  msgFmtStr  :=  'S2 -> S3' ,  strArg  :=  '' ) ;
            ELSE
                nextState  :=  StateType . S8_FAULT ;
                ADSLOGSTR ( msgCtrlMask  :=  ADSLOG_MSGTYPE_STRING ,  msgFmtStr  :=  'S2 -> S8' ,  strArg  :=  '' ) ;
            END_IF
        END_IF

    StateType . S3_BREAK_COMPLETE :
        IF camBackEdge . Q  THEN
            IF NOT  Global . bScorerSheetDetected  THEN
                nextState  :=  StateType . S4_CYCLE_COMPLETE ;
                ADSLOGSTR ( msgCtrlMask  :=  ADSLOG_MSGTYPE_STRING ,  msgFmtStr  :=  'S3 -> S4' ,  strArg  :=  '' ) ;
            ELSE
                nextState  :=  StateType . S8_FAULT ;
                ADSLOGSTR ( msgCtrlMask  :=  ADSLOG_MSGTYPE_STRING ,  msgFmtStr  :=  'S3 -> S8' ,  strArg  :=  '' ) ;
            END_IF
        END_IF

    StateType . S4_CYCLE_COMPLETE :
        IF cycleTrigger . Q  THEN
            nextState  :=  StateType . S0_SHEET_NOT_EXPECTED ;
            ADSLOGSTR ( msgCtrlMask  :=  ADSLOG_MSGTYPE_STRING ,  msgFmtStr  := 'S4 -> S0' ,  strArg  :=  '' ) ;
        END_IF

    StateType . S8_FAULT :
        Global . bMisfeedAlarm  :=  TRUE ;
        fault  :=  TRUE ;
        IF  flasher . Q  THEN
            Global . bMisfeedAlarm2  :=  TRUE ;
        ELSE
            Global . bMisfeedAlarm2  :=  FALSE ;
        END_IF
        IF  resetTimer . Q  THEN
            Global . bMisfeedAlarm  :=  FALSE ;
            Global . bMisfeedAlarm2  :=  FALSE ;
            nextState  :=  StateType . S4_CYCLE_COMPLETE ;
            ADSLOGSTR ( msgCtrlMask  :=  ADSLOG_MSGTYPE_STRING ,  msgFmtStr  := 'S8 -> S4' ,  strArg  :=  '' ) ;
        END_IF
END_CASE

IF  state  <>  nextState  THEN
    first  :=  TRUE ;
    state  :=  nextState ;
    FOR  i  :=  32  TO  2  BY  - 1  DO
        Global . stateHistory [ i ]  :=  Global . stateHistory [ i  -  1 ] ;
    END_FOR
    Global . stateHistory [ 1 ]  :=  state ;
ELSE
    first  :=  FALSE ;
END_IF

```