Program File List

Name	Number	Type	Rungs	Debug	Bytes
[SYSTEM]	0	SYS	0	No	0
	1	SYS	0	No	0
CONTINUOUS	2	LADDER	7	No	322
2FAILSPLUS	252	LADDER	2	No	77
2FAILS5120	253	LADDER	2	No	74
2FAILS2560	254	LADDER	2	No	74
SYNMPULSES	255	LADDER	6	No	97

Model various multiple failed can timings; show their effect on the reject logic

1) Cans on a conveyor pass by an inspection station and continue on to a reject station
1.1) Cans that pass inspection continue on the conveyor past the reject station
1.2) Cans that fail inspection should be removed the conveyor by the pusher at the reject station
1.3) Camera(s) at the inspection station issue a pass or fail result for each can at time that can is at the inspection station
1.3.1) A fail result is modeled/emulated here via the [Stretched fail pulse] timer object in the routine LAD 255 SYNMPULSES
1.3.2) A pass result is not modeled in this test program program
2) On any scan cycle that detects the rising edge of the fail result,
2.1) Select the first of the three reject timer objects in Data File T104 that is not already selected by a previous fail event
2.1.1) Reject timer object REJECTn_TIMER is not selected when corresponding bit REJECTn is 0
2.1.2) Reject timer object REJECTn_TIMER is selected when corresponding bit REJECTn is 1
2.2) Start that reject timer object timing for 5120ms (5.12s) in a TON (Timer ON-delay) instruction
3) The reject timer object's increasing accumulation of time models the motion of the failed can
from the inspection station, along the conveyor, to the reject station
4) When a reject timer object expires (T104:x/DN bit will be 1),
4.1) The failed can, which selected that reject timer object, is assumed, in the model, to be
in front of the reject pusher solenoid.
4.2) Set the reject bit (TIME2REJECT) to 1 to trigger the solenoid and reject the failed can
3.2) Deselect that reject timer object (reset its reject bit to 0)

Rung 0000 calls the routine LAD 255 SYNMPULSES, which emulates particular timings of failed cans detected by the camera(s) at the inspection station; refer to the comments in routine SYNMPULSES for more detail.

0000

First Pass
S:1
15

FLL
Fill File
Source -1
Dest #B103:10
Length 7

SYNTH_MULTI_PULSES
JSR
Jump To Subroutine
SBR File Number U:255

Detect the rising edge (start) of a new failed can event

0001

FAIL_RESULT
B13:5
3

oneshot
B103:0
ONS
5

OS => One-Shot;
Failure rising edge
FAIL_OS
B103:0
0

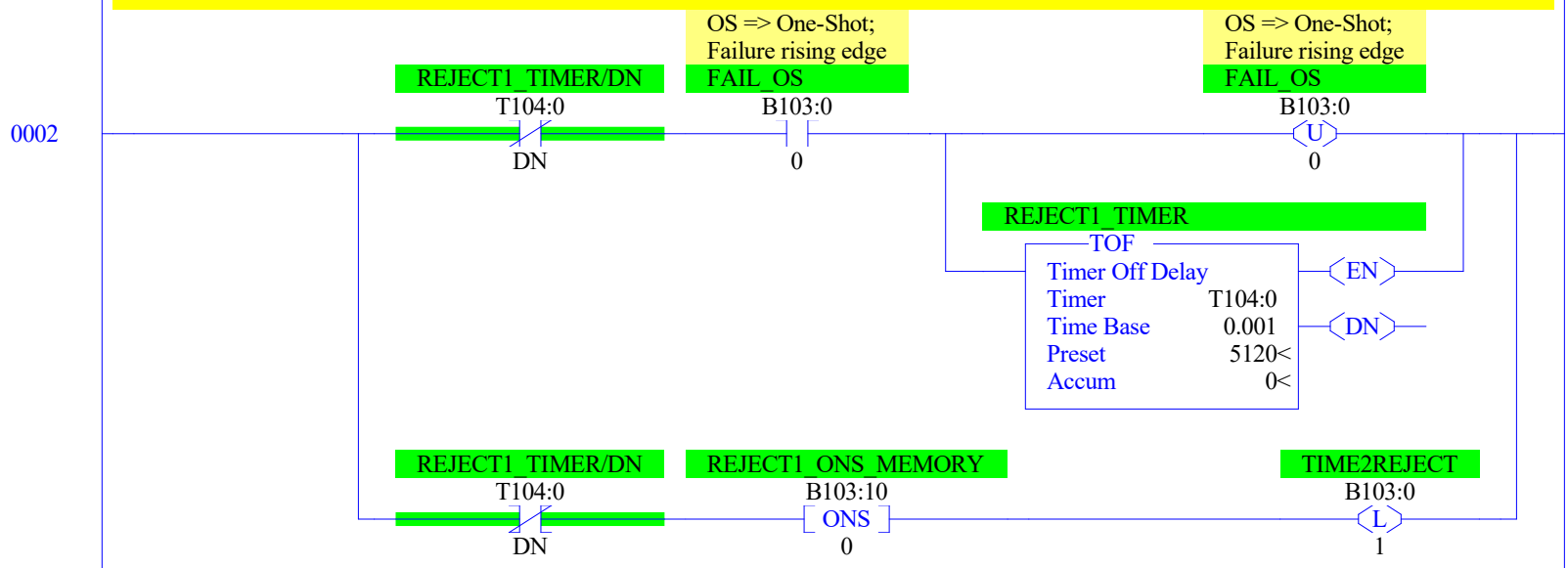
FAIL_COUNT
ADD
Add
Source A L9:0
0<
Source B 1
1<
Dest L9:0
0<

Multiple Overlapping Rejection Timers

Rungs 0002 controls the first reject timer object, REJECT1_TIMER T104:0 and one-shot memory bit B103:10/0

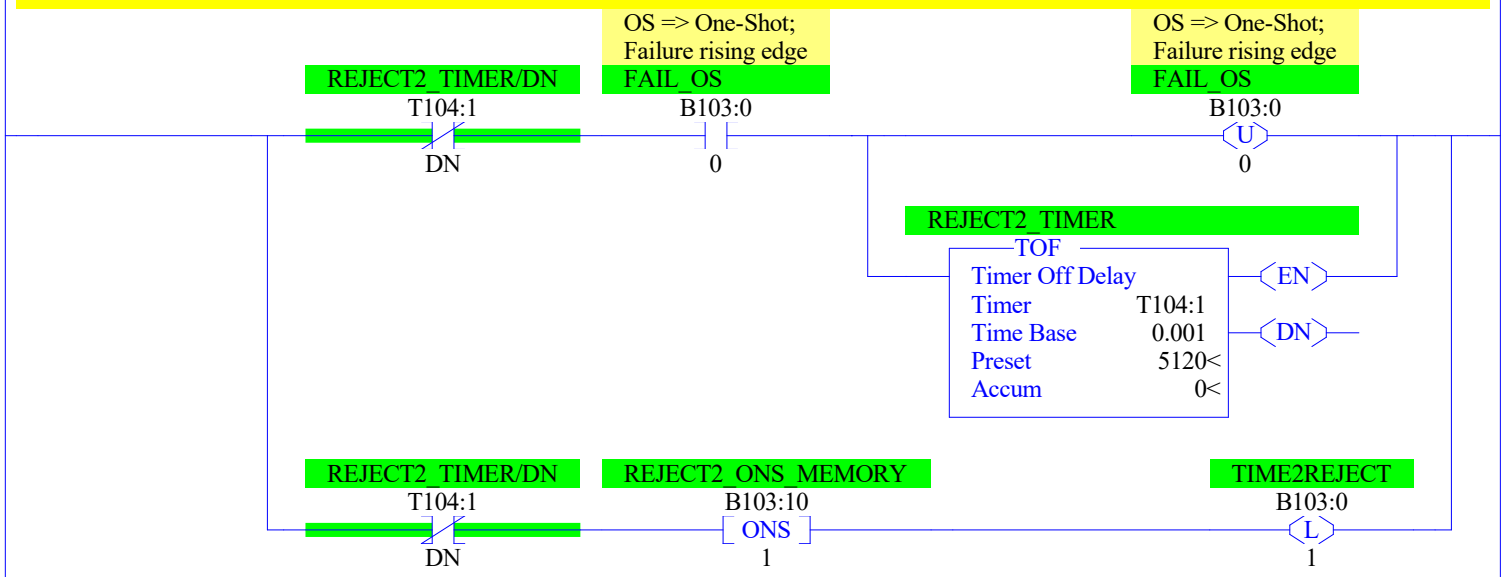
- On the top branch
  - On any scan cycle when the reject timer object IDX is neither already selected nor timing (value of T104:[IDX]/DN is 0, so XIO T104:[IDX]/DN evaluates TRUE)
    - If the failed can one-shot indicates the rising edge (beginning) of a failed can event , then
      - unlatch the failed can one-shot so no other reject timer objects will be selected on this scan cycle,
      - feed a TRUE rung into the Timer OFf-delay,
        - which will write a 1 as the value of T104:[IDX]/DN in preparation to start timing
          - which ensures the XIO T104:[IDX]/DN at the start of the top branch will not evaluate TRUE ...
            - EITHER on the next scan cycle, which will make the timer start timing,
            - OR on any subsequent scan cycles until the scan cycle ***AFTER*** the timer expires
  - On any scan cycle when the reject timer object IDX is already selected (value of T104:[IDX]/DN is 1, so XIO T104:[IDX]/DN evaluates FALSE)
    - ignore FAIL_OS
      - specifically, do not reset the value of FAIL_OS to 0,
        - so if that value is 1, the logic will continue "looking" for a subsequent unselected timer object T104:[IDX+1] where the value of T104:[IDX+1]/DN is 0
    - start or continue timing T104:[IDX], and
      - eventually the timer will expire, which will write a 0 to the value of T104:[IDX]/DN
        - indicating that a 5120ms delay has elapsed since a failed-can (FAIL_OS) event
          - *** N.B. see bottom branch description, next
        - also, this de-selects timer T104:[IDX], making it available on the next scan cycle
- On the bottom branch
  - Reevaluate XIO T104:[IDX]/DN, in case the timer expired during evaluation of the top branch
    - If the value of T104:[IDX]/DN transitioned from 1 to 0,
      - because the timer expired during evaluation of the top branch,,
      - then the one-shot ONS will evaluate to TRUE,
        - set the value of TIME2REJECT to 1
    - If the value of T104:[IDX]/DN is 0, but it was also 0 on the last scan cycle i.e. this is not a transition from 0 to 1,
      - then timer T104:0[IDX] is neither selected nor timing,
        - do nothing
    - If the value of T104:[IDX]/DN is 1,
      - then the timer either has just started or is timing and has not yet expired,
        - do nothing

* N.B. the timer object preset of 5.12s was chosen to make it easier to observe the behavior of reject logic in this demonstration program, and is, and is not meant to represent an actual conveyor line.



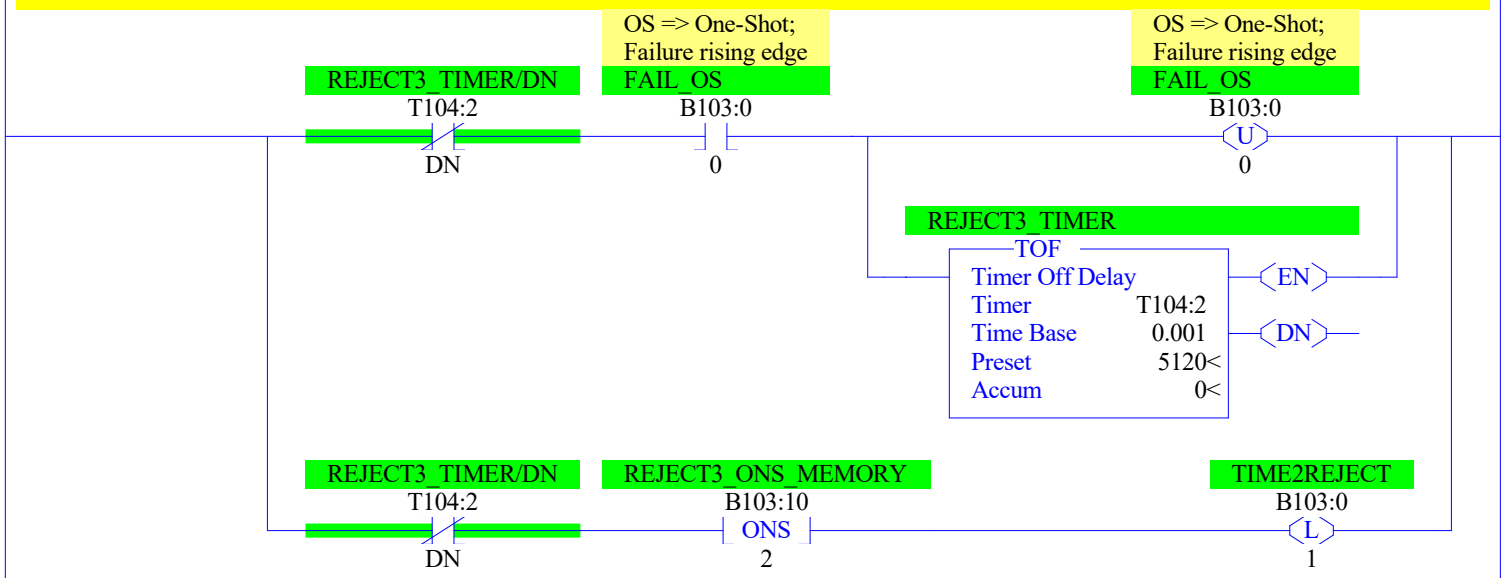
Second reject timer: T104:1 and B103:10/1

0003



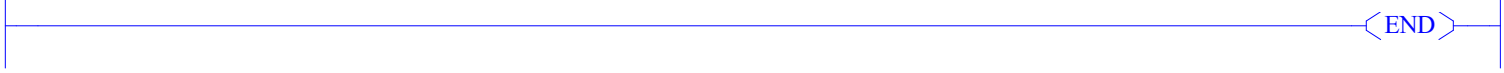
Third reject timer: T104:2 and B103:10/2

0004



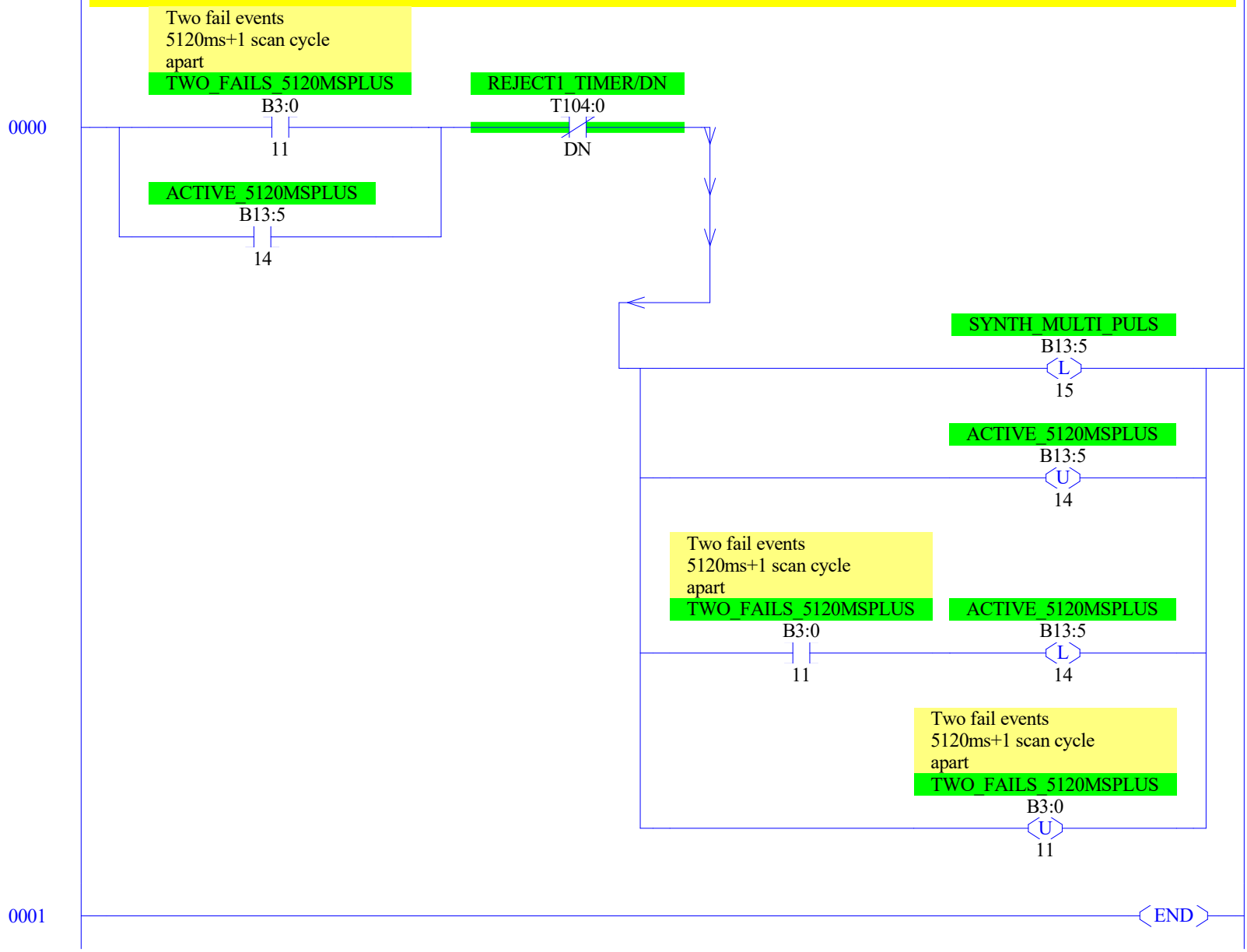
0005

0006



Toggling bit TWO_FAILS_5120MS_PLUS B3:0/11 will trigger two failed can events 5.12s, plus one scan cycle, apart, which means the second failed can event will occur on the scan cycle immediately after the reject timer object has expired from the first failed can event.

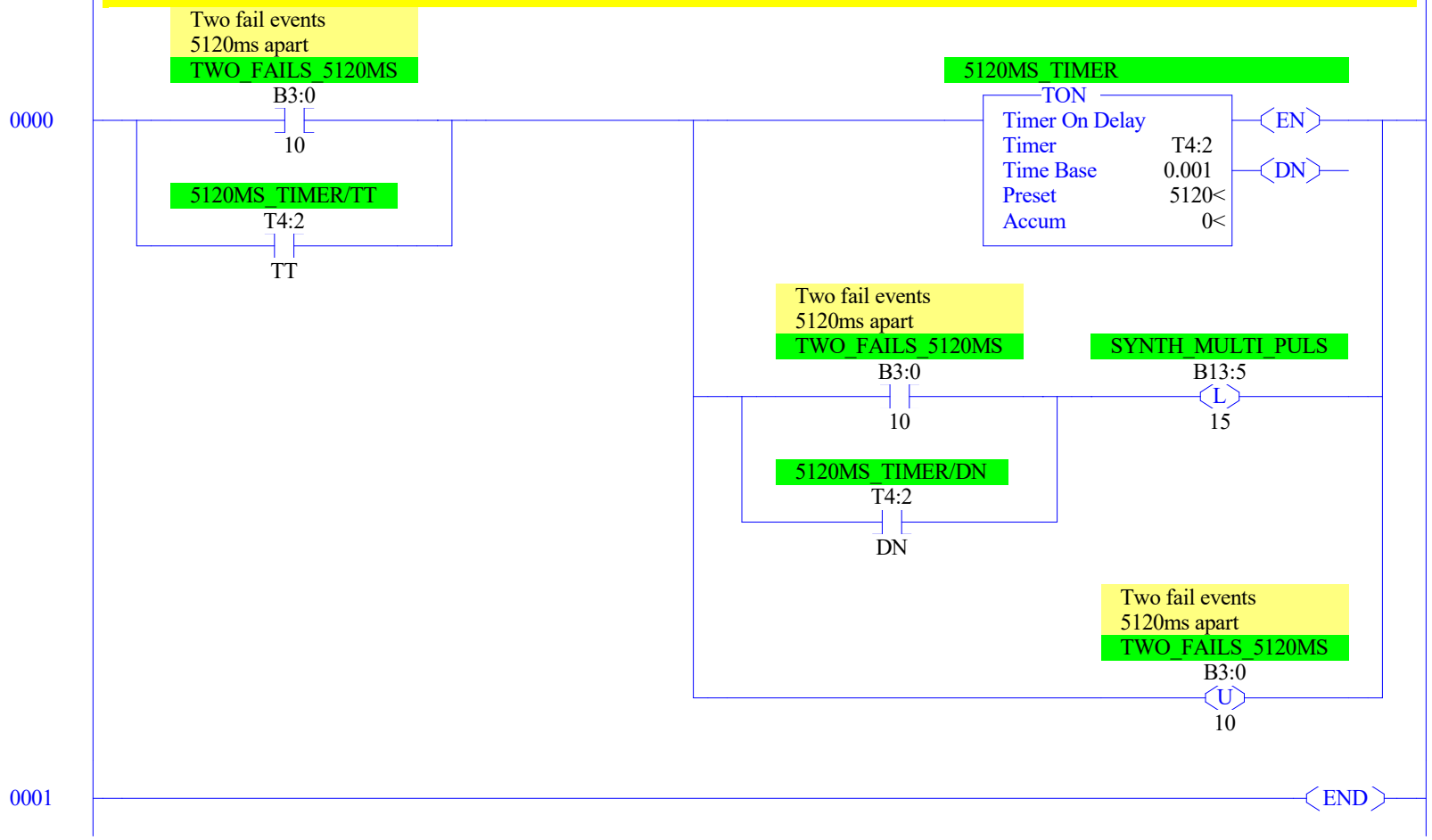
For more detail, refer to the comments in routine SYNMPULSES on the rung of the JSR call of this routine.



Toggling bit TWO_FAILS_5120MS B3:0/10 will trigger two failed can events 5.12s apart, which means the second failed can event will occur* on the same scan cycle that the reject timer object will expire from the first failed can event.

* that is the nominal case, but because the timing is done here in parallel with the reject logic timing, the two timer objects' /DN events may not occur on the same scan cycle.

For more detail, refer to the comments in routine SYNMPULSES on the rung of the JSR call of this routine.



Toggling bit TWO_FAILS_2560MS B3:0/9 will trigger two failed can events 2.56s apart, which will cause the reject logic in the main routine LAD 2 CONTINUOUS, to select and start two overlapping reject timer objects, and finally issue two reject triggers, each 5.12s after its corresponding failed can event

For more detail, refer to the comments in routine SYNMPULSES on the rung of the JSR call of this routine.

0000

Two fail events 2560ms apart
TWO_FAILS_2560MS
B3:0
9

2560MS_TIMER/TT
T4:1
TT

2560MS_TIMER
TON
Timer On Delay
Timer T4:1
Time Base 0.001
Preset 2560<
Accum 0<
EN
DN

Two fail events 2560ms apart
TWO_FAILS_2560MS
B3:0
9

2560MS_TIMER/DN
T4:1
DN

SYNTH_MULTI_PULS
B13:5
L
15

Two fail events 2560ms apart
TWO_FAILS_2560MS
B3:0
U
9

0001

END

Generate synthetic failure pulse(s)

This routine LAD 255 SYNMPULSES allows modeling SYNthetic Multiple failed can event PULSES

Rungs 0000-0002 each call a routine (2FAIL_*) that models a pair of failed can events with various timings within the pair; all of those routines latch the SYNTH_MULTI_PULS bit's value to 1 twice, which triggers the pulse stretcher timer on Rung 0004, which in turn triggers a failed can event on Rung 0004.

==========

Rung 0000 can trigger two failed can events 2.56s apart, which will cause the reject logic in the main routine LAD 2 CONTINUOUS, to select and start two overlapping reject timer objects, and finally issue two reject triggers, each 5.12s after its corresponding failed can event

This triggers correct behavior from the reject logic in main routine CONTINUOUS

0000

2_FAILS_2560MS

JSR

Jump To Subroutine

SBR File Number U:254

Rung 0001 can trigger two failed can events 5.12s apart, which means the second failed can event will occur* on the same scan cycle that the reject timer object will expire from the first failed can event.

*** N.B. the statement that follows refers to an older version of the reject logic in the main routine (LAD 2 CONTINUOUS) that used TON instructions, not TOFs

This results in incorrect behavior from the reject logic in main routine CONTINUOUS: the reject timer object from the first failed can event will still be selected on the scan cycle when when this new second failed can event is "looking" to select a new reject timer object (e.g. Rung 0002 in routine CONTINUOUS), so the logic will skip that already-selected reject timer object. However, that already-selected reject timer object will also expire on the next rung on that same scan cycle, which expiry will unlatch the selected bit (e.g. Rung 0003 in routine CONTINUOUS; bit REJECT1), and that unlatched bit will prevent the next reject timer object from being selected on the following rung (e.g. Rung 0004 in routine CONTINOUUS; see the instruction XIC REJECT1).

* that is the nominal case, but because the timing is done here in parallel with the reject logic timing, the two timer objects' /DN events may not occur on the same scan cycle.

0001

2_FAILS_5120MS

JSR

Jump To Subroutine

SBR File Number U:253

Rung 0002 can trigger two failed can events 5.12s, plus one scan cycle, apart, which means the second failed can event will occur on the scan cycle immediately after the reject timer object will expired from the first failed can event.

*** N.B. the statement that follows refers to an older version of the reject logic in the main routine (LAD 2 CONTINUOUS) that used TON instructions, not TOFs

This results in incorrect behavior from the reject logic in main routine CONTINUOUS: the reject timer object from the first failed can event will be selected by this second failed can event, because that reject timer objects selected bit (e.g. REJECT1) will have been unlatched after the timer expired (e.g. Rung 0003 in main routine CONTINUOUS). So on this next scan cycle when the second failed can event triggers, the feed rung into the TON instruction will be true, and since it was also true on the previous scan cycle i.e. when it expired from the first failed can event, the reject timer object will still be enabled as well as expired, so it will immediately issue a second reject trigger by latching the value of TIMER2REJECT to 1 (e.g. Rung 0003 in main routine CONTINUOUS)..

0002

2_FAILS_5120MS_PLUS

JSR
Jump To Subroutine
SBR File Number U:252

Rungs 0003 and 0004 stretch a synthetic failed can event (TEST_FAIL_INPUT or SYNTH_MULTI_PULS) for half a second. This is longer than would be acceptable in an conveyor-reject system, but it is done here in the demo test program to make it easier to observe the event.

TEST_FAIL_INPUT is meant to be manually triggered (e.g. in RSLogix 5000) to synthetically emulate a single failed can event.

SYNTH_MULTI_PULS is the bit triggered twice by each of the routines 2_FAILS_* called above; each of those routines has an internal bit that can be triggered manually to trigger the two latches of SYNTH_MULTI_PULS at the timing apropos whichever 2_FAILS_* internal bit is manually triggered.

0003

Single fail event
TEST_FAIL_INPUT
B3:0
8

SYNTH_MULTI_PULS
B13:5
15

TOF
Timer Off Delay
Timer T4:0
Time Base 0.001
Preset 500<
Accum 500<
EN
DN

Single fail event
TEST_FAIL_INPUT
B3:0
U
8

SYNTH_MULTI_PULS
B13:5
U
15

0004

Stretched fail pulse
T4:0
DN

FAIL_RESULT
B13:5
3

0005

END